AI-DRIVEN PREDICTIVE WELLNESS OF MECHANICAL SYSTEMS: ASSESSMENT OF TECHNICAL, ENVIRONMENTAL, AND ECONOMIC PERFORMANCE

by

Wo Jae Lee

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

School of Environmental and Ecological Engineering

West Lafayette, Indiana

May 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. John W. Sutherland, Chair

School of Environmental and Ecological Engineering

Dr. Fu Zhao

School of Mechanical Engineering

Dr. Martin Jun

School of Mechanical Engineering

Dr. Bruno Ribeiro

Department of Computer Science

Approved by:
Dr. John A. Howarter

ACKNOWLEDGMENTS

It would not have been possible to complete this doctoral dissertation without the help and support of the great people around me. First of all, I would like to acknowledge the unconditional support and love of my family, who is always on my side and root for my decisions. Without their support, this work would not have existed today.

I would like to express the deepest appreciation to my Ph.D. dissertation committee chair, Prof. John W. Sutherland, who has served as my academic advisor for the last five years. Prof. Sutherland consistently encourages me to think of a big picture and emphasizes the importance of details for the perfection and completeness of research. He continually conveyed a spirit of continuous improvement in regard to research and scholarship. Without his guidance and persistent support, I would not have been completed my Ph.D. I would like to extend my sincere appreciation and heartfelt gratitude to my Ph.D. dissertation committee members, Prof. Fu Zhao, Prof. Martin BG Jun, and Prof. Bruno Ribeiro, for their advice and encouragement.

I would also like to thank my previous and current colleagues at Laboratory for Sustainable Manufacturing (LSM) for providing me with valuable feedback on my research and making unforgettable memories in West Lafayette. I also wish to express sincere gratitude to Dr. Karim Helwani for his valuable guidance and advice during my internship at Amazon Web Services (AWS).

Lastly, I would like to acknowledge the financial support of Purdue University, particularly in the award of a research assistantship through Indiana Next Generation Manufacturing Competitiveness Center (In-MAC) and Wabash Heartland Innovation Network (WHIN) that provided the necessary financial support for this research. The funded projects allowed me to work with knowledgeable and intelligent researchers and challenge real-world problems in manufacturing plants.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

One way to reduce the lifecycle cost and environmental impact of a product in a circular economy is to extend its lifespan by either creating longer-lasting products or managing the product properly during its use stage. Life extension of a product is envisioned to help better utilize raw materials efficiently and slow the rate of resource depletion. In the case of manufacturing equipment (e.g., an electric motor on a machine tool), securing reliable service life as well as the life extension are important for consistent production and operational excellence in a factory. However, manufacturing equipment is often utilized without a planned maintenance approach. Such a strategy frequently results in unplanned downtime, owing to unexpected failures. Scheduled maintenance replaces components frequently to avoid unexpected equipment stoppages, but increases the time associated with machine non-operation and maintenance cost.

Recently, the emergence of Industry 4.0 and smart systems is leading to increasing attention to predictive maintenance (PdM) strategies that can decrease the cost of downtime and increase the availability (utilization rate) of manufacturing equipment. PdM also has the potential to foster sustainable practices in manufacturing by maximizing the useful lives of components. In addition, advances in sensor technology (e.g., lower fabrication cost) enable greater use of sensors in a factory, which in turn is producing greater and more diverse sets of data. Widespread use of wireless sensor networks (WSNs) and plug-and-play interfaces for the data collection on product/equipment states are allowing predictive maintenance on a much greater scale. Through advances in computing, big data analysis is faster/improved and has allowed maintenance to transition from run-to-failure to statistical inference-based or machine learning prediction methods.

Moreover, maintenance practice in a factory is evolving from equipment "health management" to equipment "wellness" by establishing an integrated and collaborative manufacturing system that responds in real-time to changing conditions in a factory. The equipment wellness is an active process of becoming aware of the health condition and of making choices that achieve the full potential of the equipment. In order to enable this, a large amount of machine condition data obtained from sensors needs to be analyzed to diagnose the current health condition and predict future behavior (e.g., remaining useful life). If a fault is detected during this diagnosis, a root cause of a fault must be identified to extend equipment life and prevent problem reoccurrence.

However, it is challenging to build a model capturing a relationship between multi-sensor signals and mechanical failures, considering the dynamic manufacturing environment and the complex mechanical system in equipment. Another key challenge is to obtain usable machine condition data to validate a method.

A goal of the proposed work is to develop a systematic tool for maintenance in manufacturing plants using emerging technologies (e.g., AI, Smart Sensor, and IoT). The proposed method will facilitate decision-making that supports equipment maintenance by rapidly detecting a worn component and estimating remaining useful life. In order to diagnose and prognose a health condition of equipment, several data-driven models that describe the relationships between proxy measures (i.e., sensor signals) and machine health conditions are developed and validated through the experiment for several different manufacturing-oriented cases (e.g., cutting tool, gear, and bearing). To enhance the robustness and the prediction capability of the data-driven models, signal processing is conducted to preprocess the raw signals using domain knowledge. Through this process, useful features from the large dataset are extracted and selected, thus increasing computational efficiency in model training. To make a decision using the processed signals, a customized deep learning architecture for each case is designed to effectively and efficiently learn the relationship between the processed signals and the model's outputs (e.g., health indicators). Ultimately, the method developed through this research helps to avoid catastrophic mechanical failures, products with unacceptable quality, defective products in the manufacturing process as well as to extend equipment service life.

To summarize, in this dissertation, the assessment of technical, environmental and economic performance of the AI-driven method for the wellness of mechanical systems is conducted. The proposed methods are applied to (1) quantify the level of tool wear in a machining process, (2) detect different faults from a power transmission mini-motor testbed (CNN), (3) detect a fault in a motor operated under various rotation speeds, and (4) to predict the time to failure of rotating machinery. Also, the effectiveness of maintenance in the use stage is examined from an environmental and economic perspective using a power efficiency loss as a metric for decision making between repair and replacement.

1. INTRODUCTION

1.1 Economic and Environmental Impact of Product Life Extension

One of the United Nations Sustainable Development Goals is to ensure sustainable consumption and production patterns. According to the 2019 progress report, worldwide material consumption has significantly increased, and imminent action is urged to prevent over-extraction of resources and increase resource efficiency [1]. Life cycle engineering (LCE) has emerged as a key concept to evaluate and enhance industrial sustainability, e.g., improved resource and energy efficiency, by taking into account economic, environmental, and social impacts across the product life cycle from raw material extraction, manufacturing, distribution, use, to end of life [2]. One way to reduce the environmental impact of a product in a circular economy is to extend its lifespan by either creating longer-lasting products or managing the products properly during their use stage [3]. Life extension of a product is envisioned to help better utilize raw materials efficiently and slow the rate of resource depletion [4].

Different products have different environmental impact profiles across their life cycle stages. As an example, the environmental burden of an IC-engine based automobile is dominated by the use stage, and a desktop computer is dominated by the materials processing stage. Product life extension is often viewed as an environmental benefit, in that it obviates the need for early product replacement with its associated environmental impacts. However, in some cases, rapid rate of change in a product may mean that a poorly performing product (during use), from an environmental perspective, is replaced with a better performing product. If such is the case, an early product failure may be a net environmental benefit.

Due to the fact that products have different environmental impact profiles across their life cycle stages (e.g., diesel engine vs. cell phone), they may have different strategies for life extension. Also, the pace of technological evolution (e.g., consumer electronic vs. home appliances) and a functionality change may affect the strategies. Thus, each product needs to be evaluated to determine whether a life extension is desirable for that product. For a given product, if life extension in the use stage is desirable, the useful life of a product can be prolonged through maintenance (e.g., predictive maintenance) [5]. To evaluate the maintenance effectiveness in the product use stage, let’s investigate several categories of products.

Expensive products such as aircraft engines, medical equipment, and wind turbines are often designed to have a long lifespan, and they are carefully managed during the use stage through continuous maintenance [6], i.e., intervention at prescribed intervals. Presumably, the logic that underpins such a maintenance strategy is that it is economically desirable to invest in some maintenance because of the high annualized product capital cost. The annualized capital cost is far larger than the annual maintenance cost. On the other hand, for products with a relatively small annualized capital cost, e.g., kitchen cabinets, microwaves, smartphones, and computers, the cost of maintenance may be prohibitive relative to the annualized capital cost (in addition, the efficacy of maintenance on life extension may be minimal). As a result, maintenance is not normally undertaken and such products are simply replaced when they fail. For such low cost durable goods, a better strategy may be to design and manufacture the products to be more durable and reliable.

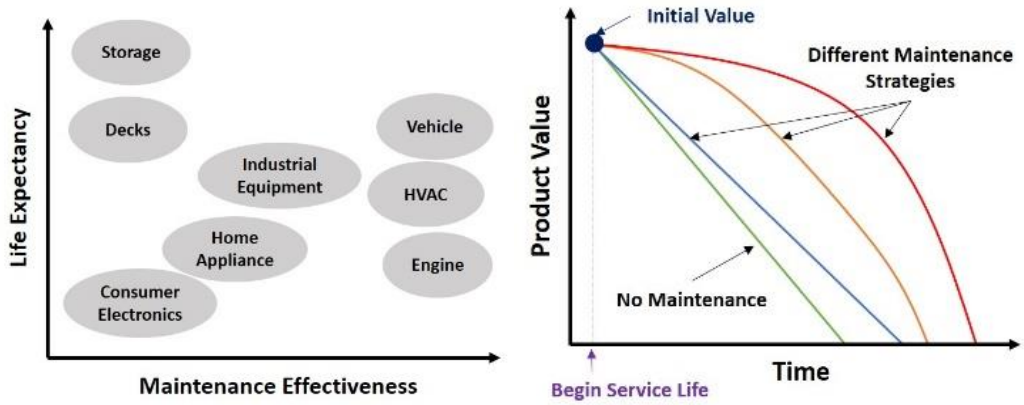
An increase in product life presents an opportunity to decrease material consumption and diminish the rate of depletion of natural resources. For several different product categories, the literature reports the average life expectancy and benefits of maintenance (shown in Table 1.1) [7, 8, 9]. As shown in the table, powered products (e.g., vehicles and HVAC systems) have relatively long life expectancy, but their service life and performance are highly dependent on how well maintenance tasks are performed (i.e., high maintenance effectiveness) during their service life. For products such as cabinets, home appliances, and decks, maintenance may not significantly increase their service life.

Figure 1.1a compares the maintenance effectiveness and life expectancy for different categories of products. Figure 1.1b displays how product value changes for different maintenance scenarios. The figure shows that the value of a product degrades over time as it is used (no investment in maintenance). If a maintenance strategy is utilized, the value still monotonically decreases, but at a slower rate. Different maintenance strategies affect the rate of value degradation.

To this point, our discussion of maintenance has solely focused on life extension and economics. But, of course, maintenance activities affect the life cycle, and thus, there are environmental implications. This is especially the case for products whose use stage has the greatest environmental impact [3]. Considering both environmental and economic perspectives, using a maintenance strategy that maximizes product life may not always be desirable. For example, for some products such as an engine, replacing an old one with a new model (that is more energy efficient) may reduce the fuel cost and the environmental burden. Moreover, maintenance costs may increase toward product end of life. Thus, the pace of technological advancement may affect the selection of a maintenance strategy. This suggests that maintenance strategy should consider the environmental impact across the life cycle stages as well as technological advancement of a new replacement product.

Table 1.1 Life expectancies of various products and maintenance effectiveness [7, 8, 9].

Product		Life Expectancy	Maintenance Effectiveness
Category	Name		
Home Appliances	Gas Range	15 yrs.	Medium
	Dryer	13 yrs.	
	Refrigerator	9 yrs.	
	Dishwasher	9 yrs.	
	Microwave Ovens	9 yrs.	
Storage	Kitchen Cabinets	50 yrs.	Low
	Garage Cabinet	100+ yrs.	
Decks	Wooden Decks	20 yrs.	Low
HVAC	Furnace	10-15 yrs.	High
	Heat Pumps	16 yrs.	
	Air Conditioner	10-15 yrs.	
	Electric water heater	10 yrs.	
Consumer Electronics	Mobile Phones	3-7 yrs.	Low
	Fax Machines	3-8 yrs.	
	Personal Computer	2-8 yrs.	
	Keyboard	3-6 yrs.	
	Mouse	3-6 yrs.	
Vehicles	Motor Vehicles	10 yrs.	High
	Watercraft	20 yrs.	
	Aircraft	30 yrs.	
Industrial Equipment	Milling Machines	20 yrs.	Medium
	Lathes	20 yrs.	
	Industrial Motor	10 yrs.	
Engines	Turbine / Jet Engines	15 yrs.	High
	Engine Instrument	15 yrs.	
	Reciprocating engine	7 yrs.	



(a) (b)

Figure 1.1 (a) Life Expectancy vs. Maintenance Effectiveness and (b) Product value vs. Time.

As a proxy for environmental impact, energy consumption is often employed. Table 1.2 presents the fraction of the total energy associated with different products for the cradle-to-gate and post cradle-to-gate portions of the life cycle. This table offers insights into which part of the life cycle the environmental impact is largest (i.e., material extraction, processing, and manufacturing vs. use and end of life). For a product whose impact is mostly associated with cradle-to-gate, an extension in product life (enabled by maintenance) delays product replacement with its concomitant environmental burden – this would generally be viewed as positive from an environmental standpoint. On the other hand, a product with a large impact post cradle-to-gate needs a carefully thought-out maintenance strategy; the strategy needs to consider potential environmental impacts and also the advisability of life extension given potential environmental benefits of replacement with a more technologically advanced product.

Based on the literature, it may be concluded that product life extension via maintenance is desirable for industrial equipment (e.g., generator), automobiles, HVAC systems, and engines. Figure 1.2 illustrates the role of maintenance in equipment life extension; the scenarios with and without maintenance in use phase. In the next section, the recent developments related to different maintenance strategies will be presented. In particular, an application of machine learning to industrial equipment is discussed.

Table 1.2 Energy consumed by products across life cycle stages.

Product	% Cradle-to-gate	% Post Cradle-to-gate	Reference
Roof Tile	83%	17%	[10]
I-Beam	97%	3%	[10]
Power Pole Cross Arms	80%	20%	[10]
Aircraft Hinge Fitting	99%	1%	[10]
Building	50.5%	49.5%	[11]
Generator	1.05%	98.95%	[12]
Computer	66%	34%	[13]
Mobile Phone	57%	43%	[14]
Automobiles	11%	89%	[15]
Air Conditioner	4%	96%	[16]

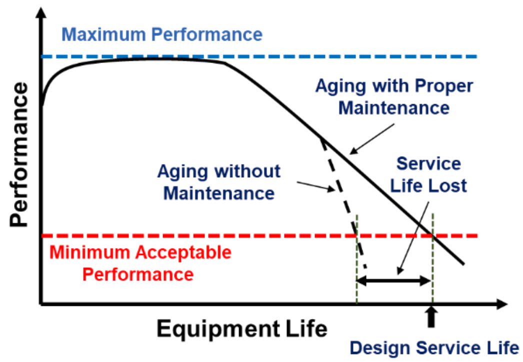


Figure 1.2 Role of maintenance in equipment life extension.

1.2 Equipment Maintenance Strategies

In the 21st century, smart manufacturing (Industry 4.0) is empowered by integration of cyber- and physical-systems with the evolution of computing infrastructures; artificial intelligence, big data, data analytics, cloud computing, IoT platform, etc. The integration of the systems with the help of ICT enables to construct an integrative and collaborative system that responds in real time to meet changing conditions in the factory, supply network, and customer demand. Smart manufacturing not only seeks to transform a manually operated factory into a highly automated plant, but also enables responses in real-time to changing conditions in manufacturing equipment, factory, supply chain network, and customer demand [17]. Smart manufacturing requires an ability to collect data (observation), process the data to secure critical knowledge (evaluation), find meaning in the knowledge (diagnosis), and formulate and implement appropriate manufacturing interventions (decision and implementation). These steps are required for closing the loop on any process control activity [18].

Smart manufacturing is defined as "fully integrated, collaborative systems" that respond in real-time to meet changing conditions in the factory, supply network, and customer demand. Figure 1.3 displays a graphical view of the smart manufacturing system. Well known benefits of smart manufacturing are: 1) improved maintenance (e.g., less unplanned downtime and extended equipment life, 2) real-time process and system control (e.g., optimal performance), 3) asset management (e.g., better utilization of resources), 4) equipment coordination (e.g., improved quality, productivity, and reduced costs), and 5) better forecasts (e.g., less work in progress and improved supply chain performance).

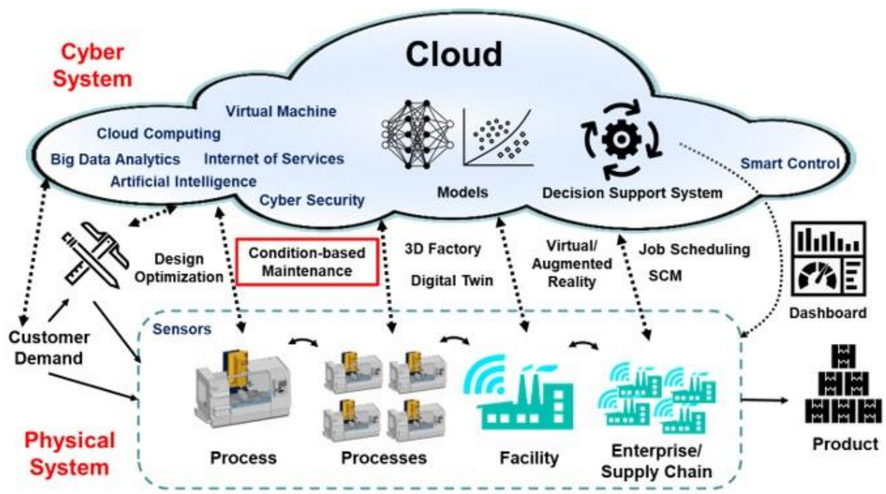


Figure 1.3. Overview of Smart Manufacturing.

In the complex manufacturing field where many elements (e.g., human and tangible and intangible resources) interact with each other [19], a large amount of data is collected and accumulated during manufacturing operations. A computing infrastructure informed by the processed manufacturing data can be controlled by pre-trained AI algorithms. To extract useful information from manufacturing data, AI techniques have been widely used. The techniques infuse intelligence into the systems to automatically learn and adapt to the changing environment using historical experience through training [20]. In addition, the ability to handle high-dimensional data, reduce complexity, improve existing knowledge, and identify relevant process relations are highlighted to demonstrate the applicability of the techniques in the manufacturing industry [21].

These abilities enable to forecast of the topic of the manufacturer's interest to possibly reduce variation in their production line and improve productivity and product quality. Therefore, the future behavior of the manufacturing system can be approximated by applying AI algorithms to the system, and this created knowledge may help decision making.

Extracted meaningful knowledge provides insights to make a better decision, which can assist the transformation toward sustainable practices in the manufacturing industry (e.g., reducing waste [22], increasing energy and resource efficiency [23], and predictive maintenance [24]). To achieve improved industrial sustainability using the smart manufacturing platform, one approach is to develop a communication tool between machinery and reliability/maintenance engineers to optimize machinery maintenance tasks. In the manufacturing plant, optimal maintenance strategies are necessary to ensure system reliability, reduce cost, avoid downtime, and maximize the useful life of a component [24]. According to the recent article, unplanned downtime caused by a poor maintenance strategy reduces a plant's overall productive capacity by up to 20 percent and costs around $50 Billion each year [25].

Figure 1.4 summarizes the different maintenance strategies and shows the pie chart describing the percentage by each strategy in the US manufacturing sector. The earliest maintenance strategy is the breakdown or unplanned maintenance (run to failure), in which no maintenance will occur until a machine breakdown happens [26]. In this situation, the utilization of a machine component may be increased to some extent, but unplanned downtime is unavoidable. Preventative maintenance, a more widely used strategy in the industry, inspects and maintains the components with periodic intervals to prevent unexpected machine breakages. However, the regular inspection/maintenance practice can incur long suspension time and high maintenance cost. Because of these pros and cons, a maintenance engineer often confronts with the tradeoff situation: they need to choose between maximizing the useful life of a component (unplanned maintenance) and maximizing uptime (preventive maintenance) [25].

While unplanned and preventive maintenances have the tradeoff scenario, predictive maintenance (PdM) is a promising technique that has an ability to break the tradeoff by maximizing the useful life of a component and uptime simultaneously. It is designed to monitor the condition of in-service equipment, and then predict when equipment will fail. It means that the future behavior/condition of machine components can be approximated, which will help to optimize maintenance tasks (e.g., prognostic health monitoring). Accordingly, the machine downtime and maintenance cost can be reduced significantly while making the maintenance frequency as low as possible.

More advanced maintenance techniques are prognostics and health management (PHM) and equipment wellness. PHM has appeared as an intelligent solution to increase availability of manufacturing systems and optimize maintenance planning by estimating the remaining useful life (RUL) or time to failure (TTF) of a component/equipment [27]. As shown in Figure 1.5, one common approach to estimate RUL is to (1) construct a health indicator using the information obtained from sensors, (2) apply the time series historical health indicator data to a regression model for forecasting, and (3) estimate the remaining time until the forecasting projection crosses the predefined threshold. General procedure including sensors, data acquisition (DAQ), raw data pre-processing, fault detection (mechanical diagnosis), and RUL/TTF prediction (i.e., mechanical prognosis) of PHM is illustrated in Figure 1.6. Sondalini [28] introduced the concept of equipment wellness by adopting W. Edwards Deming Philosophy for best reliability practices in an equipment maintenance. Under the equipment wellness strategy, a process of identifying a root cause of a problem in equipment is additionally considered to prevent reoccurring problems. It is more than being free from failures. It is a dynamic process prescribing the solution for better and extended service life.

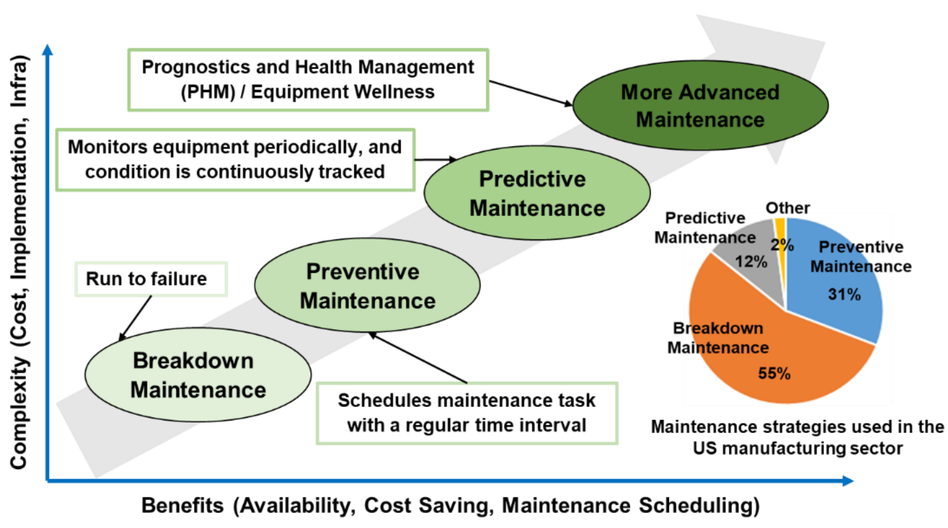


Figure 1.4 Maintenance Strategies; statistics in the pie chart are from [29].

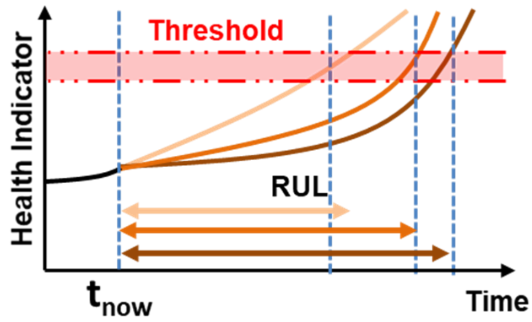


Figure 1.5 Remaining useful life (RUL) prediction based on machine health indicator.

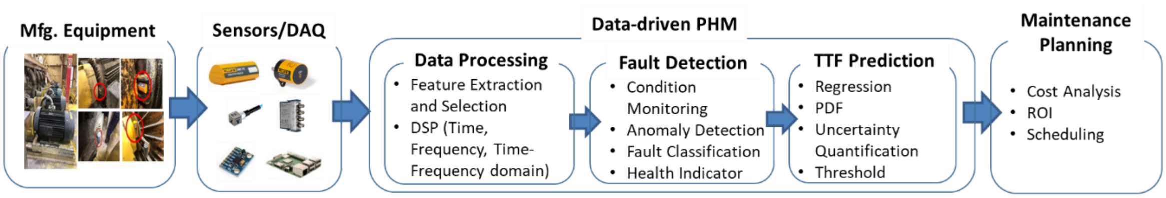


Figure 1.6 General procedure of prognostics and health management (PHM).

To summarize, the purpose of the maintenance is maximizing the availability of manufacturing systems to increase productivity while reducing maintenance costs by (1) optimizing maintenance tasks and (2) fixing potential defects before catastrophic equipment failures occur, i.e., prevent unplanned downtime. As shown in Figure 1.7, AI-driven maintenance enables to optimize the maintenance schedule by quantifying the remaining life of each component, and consequently, reduce maintenance cost by simultaneously decreasing machine downtime and repair cost. In the equipment wellness, adopted by W. Edwards Deming philosophy, there is no optimal cost (or minimum cost), but instead continue to put efforts to reduce the cost by identifying and removing the cause of problems (i.e., process control, not product control).

In this research, customized AI-driven models are developed for several machine systems, and the models are applied to a large-scale data for predicting the systems’ failure events, which may enable to extend the product life by conducting timely maintenance. Before going into details of each model, the recent developments related to predictive maintenance, machine learning, and PHM are presented in the next section.

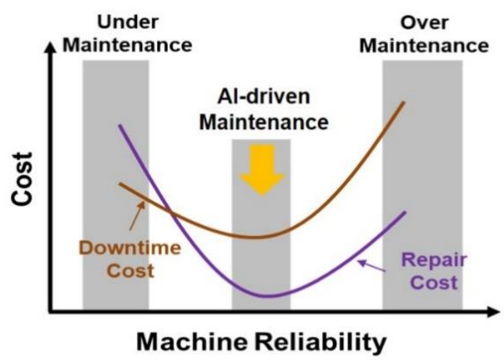


Figure 1.7. Cost vs. machine reliability.

2. LITERATURE REVIEW[1]

2.1 Predictive Maintenance of Machine Tool Systems[2]

2.1.1 Predictive Maintenance of Cutting Tool

In a milling process, a rotating cutting tool removes materials from a workpiece to obtain the desired shape. Over the machining time, a geometry of the tool changes as a result of the interaction between the tool and workpiece. In the process, a material is deformed plastically, and energy is expended in overcoming friction between the tool and workpiece [30]; a tool gradually wears due to the generation of heat and stress during the process. Consequently, it will degrade the performance of the cutting tool, which will affect a surface finish. A surface finish is considered as a critical measure of the product quality.

To ensure the product quality, the condition of the cutting tool is necessarily monitored and controlled. Failure to monitor the condition of cutting tool could generate a poor-quality product, which will turn out to be a scrap. Therefore, a PdM of cutting tool in the machining processes cannot only inform when a tool needs to be replaced (when the length of tool wear reaches its wear limit), but also enable to estimate the remaining useful life (RUL) of the tool.

The conditions of the cutting tool can be described by the lengths of wear on the different faces of the cutting tool as shown in Figure 2.1. Since a common practice to define the condition of the tool is measuring the abrasive wear length on a flank face of cutting tool [31], a flank wear limit is used as a metric to define the condition (normal, warning, and failure) of cutting tool in the simulation, which will be presented in Chapters 4 and 5.

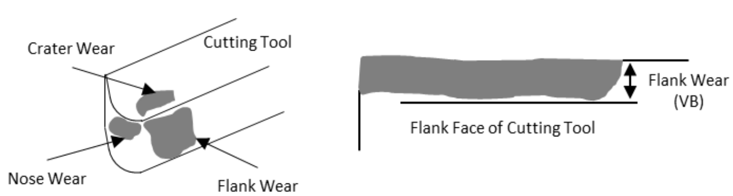


Figure 2.1 Types of cutting tool wear.

[1] More literature review can be found in each chapters.
[2] This work is a modified version of the paper published in Procedia Manufacturing [42].

2.1.2 Predictive Maintenance of Spindle Motor

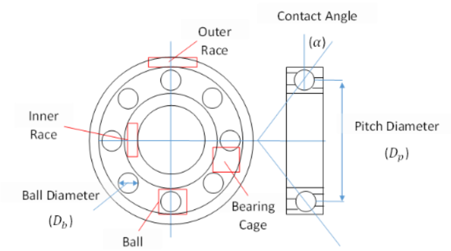
A spindle is a rotating mechanical element, and an important component in manufacturing because it directly affects the quality and productivity of manufacturing processes [31, 32]. Since the power is transmitted to machine tools through spindles, static and dynamic forces are constantly applied in the rolling elements. Continuously applied forces gradually wear the components (e.g., bearing, rotor, and shaft), and it could result in a mechanical breakage at the extreme cases.

Once a spindle is damaged, replacing the parts and calibrating accuracies such as tool runout are difficult tasks. Spare components can be stocked to be replaced during maintenance schedule. However, it is difficult to know the current condition of a component (especially, in slight wear condition) and to predict the remaining useful life of a component [33]. Therefore, a PdM of the spindle is a valuable method to optimize the maintenance jobs ahead while maintaining the process quality and productivity through preventing unexpected downtime.

Table 2.1 Parameters used in calculations of the characteristic frequencies.

Parameter	Pitch diameter	Ball diameter	Contact angle	Spindle speed
Description	D_p	D_b	α	f_s

Among many possible defects in rotating components, bearing defects are the main cause for the spindle damage [32]. For a PdM of spindles, therefore, piezo-electric force measurement sensors [34] and accelerometers [35] are used to measure the vibration due to the geometric changes of a ball bearing's inner and outer race. For the local defects of a ball bearing, characteristic frequencies can be calculated mathematically using the geometry of a rolling element bearing as shown in Table 2.1 and Figure 2.2 [36]. In Figure 2.2, the four different kinds of faults and their corresponding characteristic frequencies are explained: Inner race fault (f_{IR}), Outer race fault (f_{OR}), Rolling element fault (f_{ball}), and Bearing cage fault (f_{cage}). So, the spindle conditions can be evaluated using the variation of these four frequencies. In addition to the time and frequency domain analyses, AI techniques such as ANN (Artificial Neural Network), fuzzy logic, and Bayesian classification were used for finding the bearing faults in spindles [33].



$$f_{IR} = f_s \frac{N_b}{2}(1+ \frac{D_b}{D_p} cos\alpha)$$

$$f_{OR} = f_s \frac{N_b}{2}(1- \frac{D_b}{D_p} cos\alpha)$$

$$f_{ball} = f_s \frac{D_p}{2D_b}(1- \frac{D_b^2}{D_p^2} \cos^2 \alpha)$$

$$f_{cage} = f_s \frac{1}{2}(1- \frac{D_b}{D_p} cos\alpha)$$

Figure 2.2 The geometry of rolling element bearing and derivation of characteristic frequencies (modified from [36]).

2.2 Machine Condition Monitoring Research: Application of Machine Learning and Deep Learning on Machine Health Management

The conditions of the in-service machine tools may be translated by signals obtained from externally mounted sensors (e.g., accelerometer, microphone, dynamometer, and thermometer). To extract meaningful information from raw analog signals, first, they are necessarily processed to filter out unwanted frequency spectrums. Next step is to extract features from the processed signal to generate condition-related information as well as to compress data. Features can be extracted in either time domain or frequency domain depending on the tool characteristics (for example, Fourier transform of vibration signal from a spindle indicates a local defect of ball bearings clearly than the signal in the time domain).

In the field of machine maintenance research (e.g., machine condition monitoring), several different methodologies have been used to help with decision making and enhancing system reliability. Condition monitoring methods are often classified into three categories: 1) a physical model, 2) a knowledge-based model, and 3) a data-driven model [37]. A physical model-based methodology normally shows good success at reflecting the condition of the monitored system because the model is built based on accurate mathematical relations tied to physical processes. However, establishing an accurate physical model is challenging for complex manufacturing systems. Also, a physical model cannot generally be updated with on-line measurement data, which limits the model’s flexibility [38]. A knowledge-based methodology, such as an expert system, solves a specific domain problem using expert knowledge and heuristic rules. In this methodology, an accurate physical model is not required, but translating domain knowledge into rules (e.g., IF conditions) is difficult, and the model may not cope well with new situations. Lastly, a data-driven model estimates model parameters to fit the model using input and output data. This method is based on statistical learning theory, and the model automatically learns a relationship between input and output data (supervised learning) during the training phase. However, the method often requires a large amount of machine condition data for model training and testing.

Among the methods, data-driven models (e.g., artificial neural networks [39] and random forest [40], kernel principal component analysis [41]) have received a great deal of attention by researchers due to increasing availability of open-source data and advances in computing infrastructure (e.g., GPUs). A data-driven model, which is based on statistical learning techniques, can handle various types of data and discover hidden connections in large-scale data. Thus, this method may be a useful tool to identify the health condition of manufacturing equipment using sensor signals in real-time. This may enable condition-driven maintenance practice, or predictive maintenance (PdM) [42].

The data-driven techniques (i.e., AI models) can be divided into supervised, unsupervised, and reinforcement learning. Supervised learning trains extracted features with their corresponding labels. For example, if the features from spindle monitoring are tied to normal or fault state of the spindle, supervised learning algorithms can be used. It includes regression models, support vector machine (SVM), decision tree, ANN, etc. Unsupervised learning has no labels for each dataset but generates estimation models. K-means clustering and principal component analysis (PCA) are in the category of the unsupervised learning. Reinforced learning model learns itself from rewards and penalties, then the policy is generated to achieve the goal. After an AI model is trained, the result can be estimated from the model. For example, once a machine health model (predictive model) is trained from processed accelerometer signals and conditions of a machine (labels), its health condition can be estimated from current accelerometer signals.

For a data-driven methodology, once an initial set of raw sensor signals is obtained, it may be too large to be handled. To reduce the large-scale data without compromising its original character, the raw signals are generally preprocessed first through feature engineering, in which features (i.e., useful information) may be extracted using statistical measures (e.g., mean, RMS, standard deviation, kurtosis, and skewness) in time, frequency, or time-frequency domains [43]. In this stage, features must be carefully extracted because the performance of a data-driven model is largely dependent on the extracted features. Feature engineering may include the selection of some features, and then evaluation of the features to see if they adequately represent the large-scale data. This not only requires expert knowledge of the original dataset to decide which features should be included or excluded, but it is also a laborious process [44]. Thus, it is desirable to develop an automatic feature learning method to analyze sensor signal(s) without the necessity of human intervention [45].

Recently, deep learning methods have been successfully applied in various areas (e.g., computer vision and natural language processing). A convolutional neural network (CNN), a popular deep learning algorithm, is known as a state-of-the-art technique for processing and analyzing large datasets where the input data are often 2D images [46]. In CNN, a network architecture is generally designed to learn internal representations that are abstracted from the input data (e.g., image) by stacking multiple hierarchical layered structures [47]. One benefit of the CNN when processing machinery vibration signals is its ability to learn non-linear representations of the input data (e.g., acceleration) using the hierarchical structure [45]. This approach may make the CNN algorithm a useful tool for machine fault diagnosis since an indicator relating to the machine faults could be non-linearly correlated to the signals and their covariates. In addition, CNN requires little data preprocessing efforts because the algorithm is able to automatically learn the features from input data during the training phase—this is also called representation learning. This method makes it possible to select features without knowledge of past data and without intensive human effort.

There are many different ways to construct a CNN architecture; they differ in terms of how the multiple hierarchical layered structures are stacked. Different architectures have been explored with the aim of achieving either higher prediction accuracy and/or computational efficiency. One early CNN architecture, called LeNet, was proposed by LeCun [48]. This architecture consisted of eight layers and worked well for handwritten character recognition. To solve more complex image classification problems beyond character recognition, deeper networks (more layers) have been developed. However, with deeper networks, the training of the networks becomes more challenging (i.e., it becomes more difficult to optimize learnable parameters). Moreover, as a network is made deeper and deeper, its accuracy will improve, then plateau, and ultimately degrade [49]. To overcome the accuracy degradation problem, new ideas on CNN architecture have been proposed (e.g., inception module [50] and residual module [51]). Several novel architectures were popularized through image classification–related competitions (e.g., ImageNet Challenge). Some of these architectures are competitive with humans in terms of image classification. Deep learning applications have been mostly concentrated on image classification, speech recognition, and natural language processing. However, few studies are available where deep learning has been applied to machine condition monitoring.

In classical machine learning (ML) methods, features are extracted and selected from initial large-scale datasets first through feature engineering, and then used for training and testing a ML algorithm. Yu et al. [51] proposed a hidden Markov model (HMM) for machine health monitoring using features extracted from dynamic principal component analysis (PCA). Wu et al. [52] monitored the condition of an additive manufacturing process using an acoustic emission signal. They used PCA to reduce the amount of data needed to train the HMM. Pezzani et al. [53] proposed a support vector machine (SVM) to monitor the condition of a rotor bar in an induction motor. They extracted features from the motor current signal using statistical measures. Bhat et al. [54] also used an SVM to classify the condition of a cutting tool using images of the machined surface. They extracted and selected features through a gray-level co-occurrence matrix and Fisher discriminant analysis, respectively. Kane et al. [55] used statistical measures as an input data to an artificial neural network (ANN) for fault detection in a gearbox.

Among several data-driven models applied for machine condition monitoring, an ANN is one of the most attractive models due to its ability to manage large-scale data and its ease of deployment [37]. However, an ANN often requires data preprocessing (i.e., feature engineering), and how this preprocessing is done will affect ANN performance. Therefore, a method incorporating automatic feature learning (e.g., CNN) may be desired for processing large-scale data.

One specific type of ANN is the CNN, which allows features to be automatically learned during the training phase. Several efforts have been undertaken to build a machine condition monitoring system using CNN, and the recognition power of CNN has been actively researched and compared with classical ML algorithms (e.g., support vector machine, random forest, and ANN) [12, 23, 24]. Ince et al. [58] proposed a shallow CNN architecture for detecting a motor fault, and the method predicted the fault with an accuracy of 97.4%. In the study, the output class was limited to two conditions: healthy and not healthy (a fault has occurred). Jing et al. [59] and Chen et al. [57] introduced various gear faults in a gearbox testbed to collect the acceleration signal under different health conditions. In both studies, several different CNN architectures were applied to classify the health condition, and the classification accuracies were compared. Eren et al. [60] studied bearing fault diagnosis using the Case Western Reserve University Bearing Datasets. The study showed the effectiveness of the CNN method without feature extraction or selection processes. Janssens et al. [61] proposed an automatic bearing fault detection method using convolutional neural networks (CNN). In the study, different types of bearing faults (e.g., outer-raceway fault and rotor imbalance) were detected using acceleration signals obtained for a 25 Hz rotational speed. Jing et al. [56] also used a CNN for condition monitoring of gearboxes. They compared model prediction accuracies using both automatically learned features and manually extracted features. A number of CNN network configurations (e.g., various filter sizes, numbers of filters, and numbers of convolutional layers) were tested. Cacciola et al. [62] studied a neural network-based monitoring system to identify different root causes of mechanical imbalance problems in a rotor. Jia et al. [63] showed an improved performance of deep neural networks compared to shallow neural networks for the diagnosis of the bearing and planetary gearboxes using an auto-encoder for data preprocessing. The DL-based monitoring approach was reported to be superior to classical machine learning techniques (e.g., Support Vector Machine (SVM) and random forest) [56]. Khan and Yairi [64] reported various DL methods and their applications to a system health monitoring. The use of different deep learning architectures in recent published papers are summarized in Figure 2.3. Also, they concluded that there is a growing interest in applying DL methods in the engineering community, but many limitations still exist such as design, selection, and implementation of DL methods. An extensive review of and reference to ML and DL applications in machine condition monitoring research may be found in Peng et al. [37] and Kusiak [65].

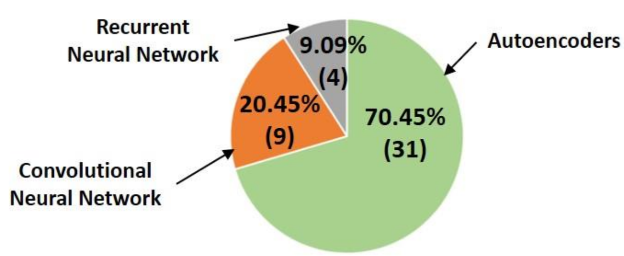


Figure 2.3. The use of different deep learning architectures in recent published papers [64].

3. RESEARCH GAP AND RESEARCH OBJECTIVES

3.1 Research Gap

ML methods have been extensively applied in machine condition monitoring research. DL applications are presently a very active area for research; however, there are limited instances where DL has been applied to machine condition monitoring. Also, many limitations still exist such as design, selection, and implementation of DL methods [64]. Based on the literature review, the following gaps have been found:

- Not sufficient study on DL applications to mechanical diagnosis and prognosis: still lacking if compared to other fields such as speech recognition and image classification
- Fault detection in time-varying conditions: limited studies considering changing operational conditions in a data-driven model (e.g., previous works on condition monitoring mainly focused on detecting a fault under constant rotational speed [66])
- Study on robust deep learning model: variability in a model's performance from data obtained from different operating settings is not well reported.
- Monitoring multiple mechanical components in the equipment. Most studies are limited to diagnose and prognose the health condition of one mechanical component.
- Lack of cost-benefit analysis: traditional maintenance vs. deep learning based maintenance
- Evaluation of Product life extension through AI-driven maintenance from an environmental point of view

Based on the research gap defined above, the research objective will be discussed in the next section.

3.2 Research Objectives

As is evident from the literature review, there have been growing interests in applying deep learning in machine health management. However, for real-world applications, the limitations discussed in the previous section should be addressed. The objective of the proposed work is to develop a systematic tool for maintenance in manufacturing plant using emerging technologies to support decision-making in real-time in the presence of changing conditions in manufacturing activities. To achieve this objective, the proposed efforts will focus on completing the following research tasks:

- **Collecting useable machine condition data from various machine tool systems.** Training machine learning models require a sufficient amount of data. To build a non-intrusive condition monitoring system, in which proxy measures (e.g., vibration and acoustic emission) are processed to diagnose and prognose machine conditions, a lab-scale testbed is designed and constructed to obtain "useable data" and to validate the proposed method. In the testbed, a fault condition needs to be configured to obtain "machine failure data" (Figure 3.1 shows an overview of machine condition data collection for the proposed research).
- **Modeling deep learning architecture to diagnose and prognose machine health conditions using proxy measures.** Various deep learning architectures (e.g., CNN and LSTM) and training methods have been proposed. This task will focus on developing a method to analyze machine condition data from the preprocessing method (e.g., feature engineering) to deep learning applications. The output of the proposed model could be a current condition or remaining useful life of a monitored machine.
- **Developing an abnormality detection system through the fusion of multi-sensor signals in a machining process.** A method to extract and evaluate meaningful values from multi-sensor signals is required to successfully monitor the condition of the machine tool during the process. This method has the goal of improving the reliability of sensor information by enabling better predictive performance and diminishing the effects of noise. This task will focus on detecting a targeted fault (i.e., abnormalities) during a machining process through effectively integrating signals collected from in situ multi-sensors.
- **Developing a speed invariant deep learning model.** Variability in a model's performance from data obtained from different operating settings is not well studied. For real world application, a deep learning model ideally detects a targeted fault under changing operational conditions. Thus, this task will focus on establishing a fault detection model whose accuracy is invariant to changes in the RPM.
- **Developing a time series forecasting model and quantifying uncertainty in the model's prediction.** To forecast machine health condition using time-varying data (e.g., trend and seasonality), a trained model may be retrained. In the proposed model, as a new observation

becomes available in time-series data, the model parameters will be updated. Then, the updated model will predict health indicators of the next multiple time steps. Here, multiple features will be extracted from raw-signals, and the extracted features and current health condition will be used in the model to forecast the future health indicators with confidence intervals.

- **Studying the effectiveness of maintenance in product life extension, and estimate the environmental and economic impacts**. The environmental and economic benefits of predictive maintenance have not been reported. Also, the product life extension through maintenance may not always be desirable. Therefore, this task will focus on investigating the effectiveness of the product life extension through predictive maintenance and estimating the environmental and economic impacts in industrial equipment maintenance.

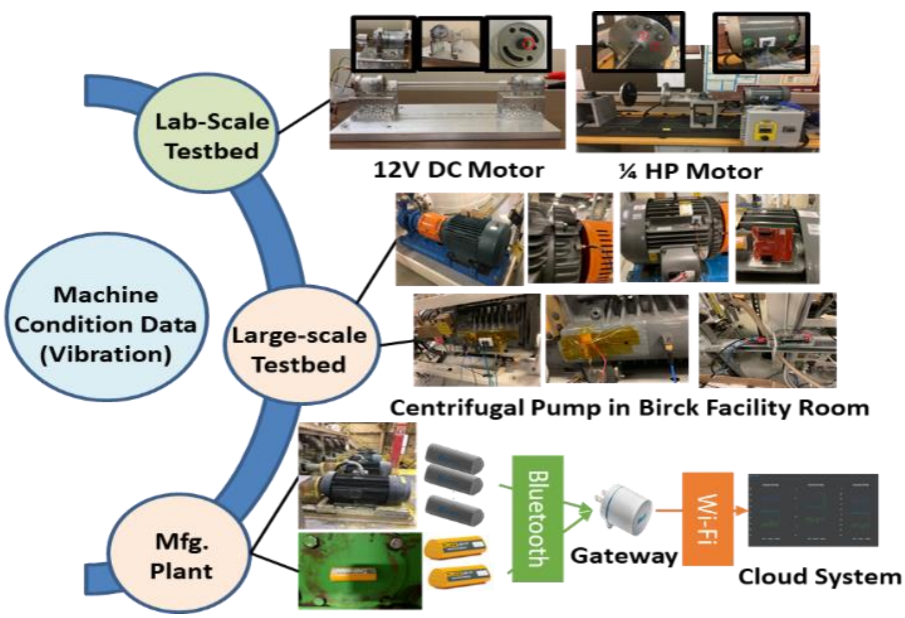


Figure 3.1. Collection of machine condition data for the proposed research.

4. DEVELOPMENT OF AN INTELLIGENT TOOL CONDITION MONITORING SYSTEM TO IDENTIFY MANUFACTURING TRADEOFFS AND OPTIMAL MACHINING CONDITIONS[3]

Reprinted with permission from Lee, W. J., Mendis, G. P., and Sutherland, J. W., 2019, "Development of an Intelligent Tool Condition Monitoring System to Identify Manufacturing Tradeoffs and Optimal Machining Conditions," *16th Global Conference on Sustainable Manufacturing*.

Abstract

Smart manufacturing has leveraged the evolution of a sensor-based and data-driven platform to improve manufacturing outcomes. As a result of increased use of sensors and networked machines in manufacturing operations, artificial intelligence techniques play a key role to derive meaningful value from big data infrastructure. These techniques can inform decision making and can enable the implementation of more sustainable practices in the manufacturing industry. In machining processes, a considerable amount of waste (scrap) is generated as a result of failure to monitor a tool condition. Therefore, an intelligent tool condition monitoring system is developed in this paper to identify sustainability-related manufacturing tradeoffs and a set of optimal machining conditions by monitoring the status of the machine tool. An evolutionary algorithm-based multi-objective optimization is used to find the optimal operating conditions, and the solutions are visualized using a Pareto optimal front.

Keywords: Smart and Sustainable Manufacturing; Artificial Intelligence; Evolutionary Strategies; Tool Condition.

4.1 Introduction

Smart manufacturing is enabled by developments in big data analytics, smart sensors, cloud computing, Internet of Things (IoT) platforms, and artificial intelligence, and has received significant attention [67]. Smart manufacturing can minimize cost, optimize labor use, and increase product quality and productivity in manufacturing operations through integration of cyber- and

[3] This work is a slightly modified version of the paper published in Procedia Manufacturing [148].

physical- systems. The integration of these systems is enabled by advanced computing infrastructure, informed by a large dataset collected during manufacturing, and is controlled using trained and selected machine learning techniques. Smart manufacturing extracts meaningful value from large datasets to advance existing analysis capabilities and provide new competencies [22]. Smart manufacturing can improve efficiency and reduce waste from the production process, which also increases the sustainability of the manufacturing enterprise. These smart manufacturing methods can enable better decision making and more sustainable practices in the manufacturing industry (e.g., waste reduction [22], better energy and resource efficiency [68], and higher product quality [69]).

To achieve "zero-waste" in the manufacturing process, this study focuses on avoiding/minimizing the waste (scrap) in the machining process, which is a widely used manufacturing process. In machining processes, surface integrity is a critical measure of product quality, which, if not properly monitored and controlled, can lead to defective parts and unnecessary waste. Surface integrity is profoundly affected by the condition of the cutting tool, which is difficult to detect during operation. Failure to monitor the condition of the tool can generate a considerable amount of waste, poor quality parts, and economic losses. Therefore, an intelligent tool condition monitoring system is required to ensure product quality and to minimize the amount of scrap in a highly-automated environment (e.g., CNC machining center).

To build an intelligent tool condition monitoring system, various sensing technologies (e.g., vibration, acoustic emission, force, and power) are incorporated in the manufacturing process to acquire information about the condition of the machine tool [70]. With the availability of sensor signals, data-driven models can be developed using artificial intelligence techniques (artificial neural networks [71], support vector machines [31], fuzzy systems [72], and random forest methods [73]) to monitor and predict the condition of the tool.

Although artificial intelligence techniques have been successfully applied to machine tool condition monitoring, the effects of deterioration in machine tool performance on the productivity of the production line have not been well studied. Conventional machining models are based on the assumption that the performance of the machine tool does not change over the course of the process [74]. However, from a quality perspective, the capability of the machine tool deteriorates over the operation as the cutting tool wears. Therefore, optimal machining parameters should be identified as a function of the condition of the tool by analyzing the tradeoffs between profit, product quality, and productivity.

In this paper, an intelligent tool condition monitoring system is developed not only to predict tool condition, but also to identify a set of optimal machining conditions. First, a data-driven model is developed. To obtain training and testing datasets, three types of sensors (current, vibration, and acoustic emission) are employed during the milling process under various cutting operations. Then, the collected raw signals go through signal processing, feature generation, and feature extraction/selection to be transformed into more useful signals [75]. After processing the dataset, a support vector machine is used to classify the condition of the tool. In order to find the optimal machining conditions as a function of the tool's condition, a multi-objective optimization is performed using evolutionary strategies, and the tradeoffs are investigated using a Pareto optimal front.

4.2 Development of an Intelligent Tool Condition Monitoring System

In the monitoring system, a support vector machine (SVM) and evolutionary strategies (ES) are employed to monitor the condition of the machine tool and to identify the optimal cutting conditions, respectively. The SVM is based on statistical learning theory, which is a nonparametric dependency estimation using a given dataset [76]. The ES was inspired by the idea of natural section and is used for global parameter optimization. Flank wear, which is abrasive wear on the flank face of the machine tool, is useful to quantify the condition of the tool [31]. A detailed schematic diagram of the tool condition monitoring system is shown in Figure 4.1.

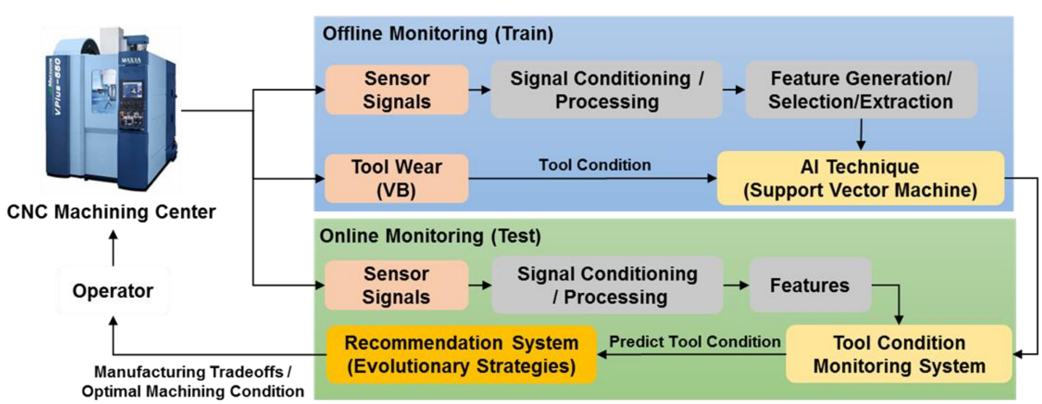


Figure 4.1 Development of an intelligent tool condition monitoring system.

The monitoring system is developed in two stages, offline monitoring (Training) and online monitoring (Testing). In the offline monitoring stage, the data-driven model (SVM) is trained using the processed sensor signals (input) and tool wear (output). In the online monitoring stage, a real-time multi-sensor dataset obtained during the machining process is applied to the trained data-driven model to predict tool wear. Given the condition of the machine tool, the ES based, multi-objective optimization algorithm optimizes several machining conditions (process parameters) for a set of evaluation parameters- workpiece quality, cost, and productivity – which can be adjusted based on a particular manufacturer's requirements.

4.2.1 Milling Experimental Dataset

To demonstrate the capabilities of the proposed monitoring system, a milling dataset was obtained under various operation conditions, as displayed in Table 4.1 [46, 47]. In the experiment, the milling operations were conducted to investigate the amount of flank wear (VB) after each cutting run. The tests were performed at a cutting speed of 200 m/min (=826 rev/min), two depths of cut (0.75 mm and 1.5 mm), and two feeds (0.25 mm/rev and 0.5 mm/rev). A cast iron workpiece with the dimensions of 483 mm (l) by 178 mm (w) by 51 mm (h) was machined.

Table 4.1 Milling experiment conditions.

Case	Depth of Cut (mm)	Feed (mm/rev)	Case	Depth of Cut (mm)	Feed (mm/rev)
Case 1 (++)	1.5	0.5	Case 3 (--)	0.75	0.25
Case 2 (+-)	1.5	0.25	Case 4 (-+)	0.75	0.5

During the milling experiment, five different signals, (1) the AC spindle motor current, (2) the table vibration (VBtable), (3) the spindle vibration (VBspindle), (4) the acoustic emission at the table (AEtable), and (5) the acoustic emission at the spindle (AEspindle), were collected. The vibration and acoustic emission sensors are mounted on the table and spindle, as found in [13]; other experimental details can also be found in that document as well.

Several useful features can be generated, extracted, and selected from the observed signals to describe more appropriately the tool condition under different machining operations [75].

4.2.2 Feature Generation / Extraction / Selection

Features of the dataset can be identified using "descriptors," representing central tendency or dispersion of the collected data. In this paper, 14 descriptors are employed to generate features from each run as shown in Table 4.2.

To determine which descriptors show a meaningful difference in performance as the tool degrades for each cutting run, the signals from Case 3 were tested, the meaningful descriptors were identified, and analysis was performed on all four cases using the meaningful Case 3 descriptors. Several descriptors display distinguishable behaviors as the number of runs increases (higher run means higher tool degradation), so 13 (AC), 8 (VBtable), 8 (AEtalbe), and 4 (AEspindle) features were extracted from entire descriptors.

The original signals from each selected descriptor may still have noise, and the dimensionality of the feature set (33) is too large to be analyzed. Therefore, principal component analysis (PCA) is used to extract new effective features (i.e., principal components) from the original features [79], while preserving global information. PCA is a mathematical procedure for compressing multi-variable data (e.g., data from multi-sensor) by mapping the data onto new axes, which are called principal components. The principal components are constructed by developing linear relationships between the original variables, as shown in [80].

Table 4.2 Descriptors used to generate features for analysis (x is observed signal and n is the number of samples).

Descriptor	Equation	Descriptor	Equation	Descriptor	Equation
1. Arithmetic Mean (AM)	$\frac{1}{n}\sum_{i=1}^{n} x_i$	6. Range (R)	$x_{\max} - x_{\min}$	11. Skewness (Skew)	$\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3/\sigma^3$
2. 50th Percentile (Med)	x_{k+1} for odd or $\frac{x_k + x_{k+1}}{2}$ for even	7. Variance (Var)	$\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n-1}$	12. Crest Factor (Cr)	$\frac{\max\lvert x_i\rvert}{\text{RMS}}$
3. Trimmed Mean (Trim)	$\frac{x_{r+1}+x_{r+2}+\ldots+x_{n-r-1}+x_{n-r}}{n-2r}$	8. Standard Deviation (Std)	$\sqrt{\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n-1}}$	13. Impulse Factor (Imp)	$\frac{\max\lvert x_i\rvert}{\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert}$
4. Interquartile Range (IQR)	$Q_3 - Q_1$	9. Root Mean Square (RMS)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}$	14. Margin Factor (Mar)	$\frac{\max\lvert x_i\rvert}{(\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert^{1/2})^2}$
5. Mean Absolute Deviation (MAD)	$\frac{1}{n}\sum_{i=1}^{n}\lvert x_i-\tilde{x}\rvert$	10. Kurtosis (Kur)	$\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4/\sigma^4$		

The goal of PCA is to find a new set of axes which maximizes the variance of the transformed data. The principal components can be found after eigen decomposition of the covariance matrix, and the eigenvector of the covariance matrix which has the largest variance is called first principal component, having a second largest variance is called second principal component, and so on. Here, PCA is applied to the extracted feature to compress the dataset without significant loss of information. The output (score matrix) from PCA is directly used for input to the data-driven model (SVM).

4.2.3 Support Vector Machine

The SVM technique is a state-of-the-art learning method used to solve multi-class classification problems [81]. The method generates several hyperplanes to maximize the margin between different classes [31]. In the method, an optimal hyperplane is sought to separate the different classes most effectively. The features close to the hyperplane, called support vectors (sv), are identified and are used to find optimal hyperplanes that maximize the margin between the different classes [82]. The training of the algorithm is performed by solving a linearly constrained quadratic optimization problem.

In this paper, a non-linear SVM is employed that incorporates a non-linear function (kernel function). The non-linear SVM transforms the original input space to a higher-dimensional feature space using a kernel function, where the dataset is linearly separable. A kernel function is defined as a dot product of two feature vectors. In this paper, the three kernel functions, linear ($K(x_i, x_k)= x_i^T x_k$), polynomial ($K(x_i, x_k)= (x_i^T x_k+1)^d$), and Gaussian ($K(x_i, x_k)= exp(-||x_i - x_k||^2 / 2\sigma^2)$) are tested ($d$ is a coefficient). Further discussion of the method can be found in [82].

The SVM was implemented in MATLAB R2017a software. For classification of the tool wear conditions, three-status (tool wear below 0.2 mm, 0.2 mm – 0.4 mm, and above 0.4 mm), four-status (tool wear below 0.10 mm, 0.10 mm – 0.3 mm, 0.3 mm – 0.45mm, above 0.45 mm), and five-status (tool wear below 0.15 mm, 0.15 mm – 0.30 mm, 0.30 mm – 0.45 mm, 0.45 mm – 0.6 mm, above 0.6 mm) classifications are used to examine sensitivity. The input dataset consists of two principal components (the first and second principal components) and two process parameters (depth of cut and feed), and k-fold cross validation (k=10) is run for training and testing. Different kernel functions are tested to identify the best-fitted kernel function, and the best results for three-, four-, and five-status classifications are presented in Figure 4.2 using a confusion matrix

(Wear 1, 2, 3, 4, and 5 in the figure indicate the tool statuses mentioned above). The cubic kernel (polynomial kernel with d=3) and linear kernel show the best performance in classifying three-status and four- and five-status classifications, respectively.

A confusion matrix is known to be an effective tool to visualize the performance of a classification technique. In the matrix, each row and column display true and predicted values. The prediction accuracies and the errors from misprediction are shown in boxes with the green horizontal stripes and orange vertical stripes, respectively. By displaying the accuracies and errors together in the matrix, the confusion matrix can show which classifier is confused when a prediction is performed. In Figure 4.2, as the number of classifiers increases, the accuracies decrease, as would be expected. All mispredictions in the figure are found at adjacent classifiers, which indicates that all predicted values have not significantly deviated from the true values.

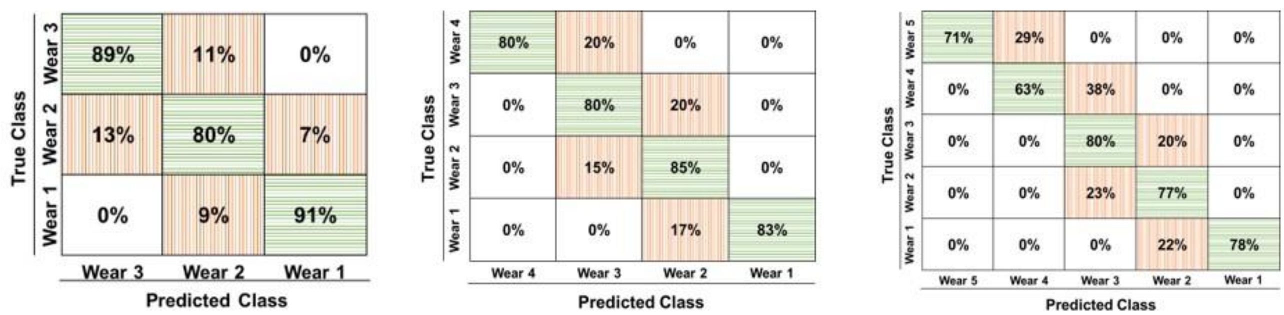


Figure 4.2 Confusion matrices for three-status with cubic kernel (left), four-status with linear kernel (middle), and five-status with linear kernel (right).

4.3 Multi-Objective Optimization to Identify Manufacturing Tradeoffs and Optimal Machining Condition

4.3.1 Evolutionary Strategies

A real-world problem may have multiple local and global optimum values. A local optima search method is based on a neighborhood search method, where a solution is steadily improved within the neighborhood, however the solution may not attain a global optimum value. The most successful global optimization algorithm is based on a stochastic search method (or evolutionary method), which enables the escape from local optimums [83]. ES, a widely used evolutionary method, imitates the behavior of organic evolution procedures to seek the optimum values [84]. The major working schemes of ES are population, recombination, mutation, and selection (elitism and evaluation). In the algorithm, as shown in Table 4.3, the candidate solutions are described by a parental set (μ) and an offspring set (λ). The random individuals (first parental set) are generated first, and the best individuals (first offspring set) are selected based on a selection algorithm (e.g., fitness function ($f(\boldsymbol{x}_i^{'})$)). Then, the best individuals (new parental set) produce a new offspring set, and so on. This process iterates until the termination condition is reached (j is greater than or equal to generation).

Table 4.3 (μ+ λ)-ES Algorithm.

Evolutionary strategies algorithm
1: **initialize** $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_\mu$
2: **while** j < generation
3: **for** i = 1 **to** λ **do**
4: select ρ parents: $\boldsymbol{x}_i^{\,p} \rightarrow \boldsymbol{x}_i$
5: recombination: $\boldsymbol{x}_i \rightarrow \boldsymbol{x}_i^{r}$
6: mutate $\boldsymbol{x}_i^{r} \rightarrow \boldsymbol{x}_i^{m}$
7 elitism $\boldsymbol{x}_i^{\,p} \cup \boldsymbol{x}_i^{m} \rightarrow \boldsymbol{x}_i^{'}$
8: evaluate $f(\boldsymbol{x}_i^{'})$
9: **end for**
10: **select** μ parent from $\{\boldsymbol{x}_1^{'}, \boldsymbol{x}_2^{'}, \ldots, \boldsymbol{x}_\lambda^{'}\} \rightarrow \{\boldsymbol{x}_1^{'}, \boldsymbol{x}_2^{'}, \ldots, \boldsymbol{x}_\mu^{'}\}$
11: j=j+1
12: **end while**

In order to generate the offspring population (lines between 3 and 9 in Table 4.3), the selected parent group iterates through recombination, mutation, and selection. The genetic information is mixed (discrete recombination) and randomness ($N(\xi, \sigma^2)$) is added in the recombination and mutation stages, respectively. In the selection process, the best solution from the union of the parental population and the offspring population (μ+ λ) is chosen using a fitness function ($f(\boldsymbol{x}_i^{'})$) [85].

As described above, a stochastic search method is a powerful technique to discover the global optimum in a real-world problem. Therefore, the (μ+ λ)-ES optimization technique is employed to find the optimal set of cutting conditions for a given set of objective functions.

4.3.2 Objective Functions

Objective functions are used to translate real-world behaviors into optimizable mathematical forms. Three main considerations in manufacturing are cost, quality, and production time. To satisfy a manufacturer's needs, the proper machining conditions have to be identified according to certain tradeoffs between the considerations. In the milling process, since condition of the machine tool changes over machining time, different cutting conditions may be required after each operation. Therefore, in this paper, the tool condition is incorporated as a part of the objective functions to search for an optimal set of machining conditions as the tool's capability changes. This approach can also minimize the potential waste generated due to poor product quality. In this section, three objective functions representing profit, quality, and productivity are formulated to conduct multi-objective optimization. The first objective, profit, is formulated as follows:

$$\text{Profit} = \text{Revenue Rate - Machine Operating Cost - Quality Cost - Tool Cost.} \tag{4.1}$$

Profit can be expressed as manufactured workpiece price ($) per unit machine time (min) with the assumption that sufficient demand exists for the workpiece and the process has an infinite horizon [74].

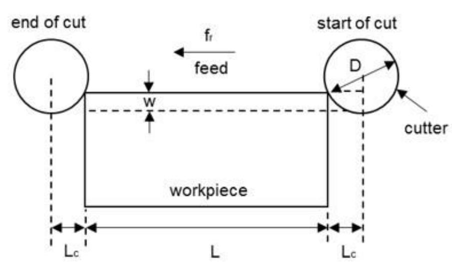


Figure 4.3 Top view of face milling operation.

Using the geometry shown in Fig 3, milling time ($t_{milling}$) and revenue rate (R) to manufacture a workpiece can be expressed as

$$t_{milling} = (L + 2L_C)/f_r, \tag{4.2}$$

$$L_c = \sqrt{w(D-w)}, \tag{4.3}$$

$$\text{R} = p/(t_{milling} + t_{handling}), \tag{4.4}$$

where f_r is the feed rate of the workpiece (mm/min), w is the width of cut (mm), D is the diameter of the cutter (mm), p is the revenue per workpiece ($), $t_{handling}$ is the loading/unloading time for the workpiece (min), $t_{milling}$ is the milling time (min), L is the length of the workpiece to be machined (mm), and L_c is the extent of the cutter's first contact with the workpiece (mm).

Machining operation cost can be divided into direct labor cost and machine cost, and can be written as

$$C_{labour} = (C_{handling} t_{handling} + C_{milling} t_{milling}) \big/ (t_{milling} + t_{handling}), \tag{4.5}$$

$$C_{machine} = C_z t_{milling} \big/ (t_{milling} + t_{handling}), \tag{4.6}$$

where $C_{handling}$ is the direct labor cost of loading/unloading (handling) per unit time ($/min), $C_{milling}$ is the direct labor cost of manning the machine per unit time ($/min), and C_z is the cost of a machine per unit in-cut time ($/min).

Traditionally, the quality cost is often ignored if a quality metric of a manufactured product is between the lower specific limit (LSL) and the upper specific limit (USL). However, the Taguchi proposed a simple quadratic equation to quantify the cost of quality loss [86]. In the equation, the loss ($) depends on the deviation from a target value. In this study, the cost of quality loss due to surface roughness (R_a) is incorporated in the profit model using Taguchi's quality loss function using a continuous assumption. Then, the cost of quality loss per unit machine time can be mathematically expressed as

$$C_{quality} = C_r (R_a - R_0)^2 \big/ (t_{milling} + t_{handling}) \text{ (if } \mathrm{R_a} > \mathrm{R_0}\text{, and 0 otherwise)}, \tag{4.7}$$

where C_r is the quality cost per unit^2 deviation ($/μm^2), R_0 is the target surface roughness (μm), and R_a is the arithmetic average surface roughness (μm) [74].

To estimate R_a as functions of feed and tool wear, the R_a equation from [87] is modified as follows (in the equation, Ra grows as feed and tool wear increase)

$$R_a = 0.0321 f^2 \big/ (r_w - \mathrm{VB}), \tag{4.8}$$

where f is the feed rate (mm/rev) (=f_r / spindle speed), r_w is the cutter nose radius (mm), and VB is the tool wear (mm).

In the machining process, a machine tool needs to be replaced when machining time of the tool exceeds the tool life. Assuming the tool wear is continuously monitored, a user-defined tool wear limit and the amount the tool wears at jth and (j-1)th operations may be used to calculate the tool cost per unit machine time. The approach provides a more nuanced metric of performance decrease than a constant tool depreciation value would allow. Instead, the tool cost per unit machine time depends on the amount of tool wear growth between adjacent cutting runs. Mathematically, this can be written as

$$C_{tool} = C_s \frac{\Delta VB_{j,j-1}}{VB_{limit}} \times \frac{1}{t_{milling} + t_{handling}}, \tag{4.9}$$

where C_s is the tool purchase/set-up cost ($), VB_{limit} is the tool wear limit (mm), $\Delta VB_{j,j-1}$ is the tool wear growth between jth and (j-1)th operation, and j is an integer.

Using the equations described above, the three objective functions are presented as follows (here, productivity and quality are represented by cycle time and surface roughness, respectively)

$$\text{Max: Profit } (= \text{R} - \text{C}_{labor} - \text{C}_{machine} - \text{C}_{quality} - \text{C}_{tool}), \tag{4.10}$$

$$\text{Min: Cycle Time } (= t_{milling} + t_{handling}), \tag{4.11}$$

$$\text{Min: Surface Roughness } (= R_a). \tag{4.12}$$

4.3.3 Identifying the Tradeoffs using a Pareto Optimal Front

In this section, the (μ+ λ)-ES algorithm is performed to identify the tradeoffs using the three objective functions developed in the previous section. The two machining process parameters (spindle speed and feed rate) are used as variables to define an optimal set of machining conditions. Machining characteristics used in the simulation are displayed in Table 4.4 (maximum and target surface roughness, cutter nose radius, labor cost, and quality cost are adopted from [74]). The three objective functions are subject to the constraints described in Table 4.4.

Table 4.4 Machining characteristics and constraints.

Machining Characteristics								Constraints	
L	483 mm	$C_{handling}$	0.5 $/min	r_w	0.8 mm	V_{min}	800 rev/min	Spindle Speed	$V_{min} \leq V \leq V_{max}$
$t_{handling}$	3.63 min	$C_{milling}$	0.5 $/min	$R_{\max}$	10 μm	V_{max}	1200 rev/min	Feed Rate	$f_{r\,\min} \leq f_r \leq f_{r\,\max}$
P	50 $	C_z	2 $	C_s	100 $	$f_{r\,\min}$	206.5 mm/min	Surface Finish	$R_a \leq R_{\max}$
D	70 mm	C_r	0.75 $/μm^2	VB_{limit}	0.6 mm	$f_{r\,\max}$	413 mm/min	Profit	$0 \leq$ Profit
W	40 mm	R_0	2.5 μm	$\Delta VB_{j,j-1}$	0.07 mm			Tool Wear	$VB \leq VB_{\mathrm{limit}}$

With the given machining characteristics and constraints, a multi-objective optimization is conducted to identify the tradeoffs between the objectives. The size of the parent and offspring population is set to 50, and the evolutionary process is iterated for 100 generations. Here, the two parameters (size and generation) are carefully selected to show Pareto optimal front line clearly in figure and to obtain exact global solutions, respectively. To visualize the tradeoffs at different tool conditions, Pareto optimal fronts are plotted at tool wear values of 0.15 mm, 0.3 mm, and 0.5 mm. The Pareto optimal front helps to quantify the tradeoffs among the objectives to compromise between parameters [88]. Therefore, multiple Pareto optimal solutions are likely to help manufacturers to select the machining process parameters that suit their preferences.

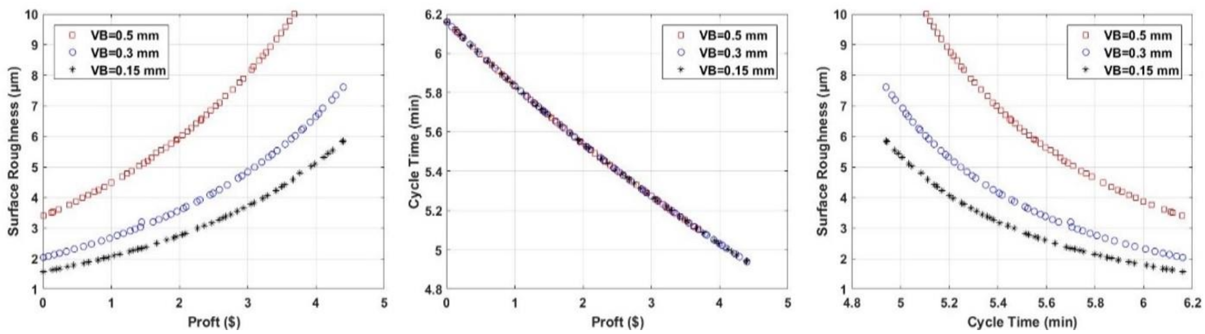
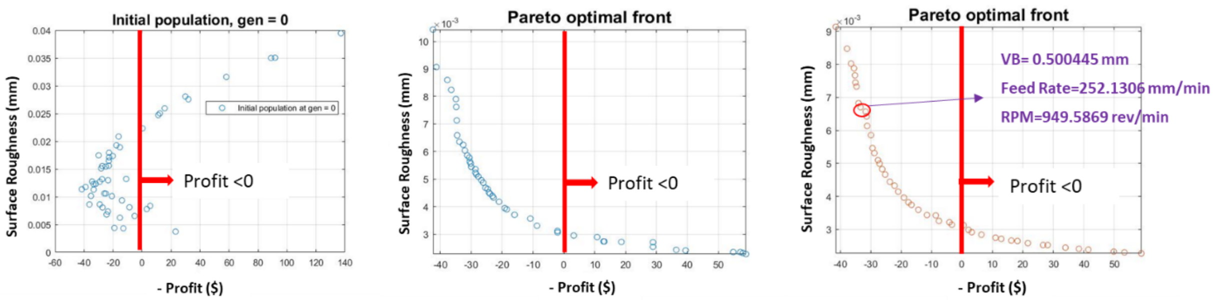


Figure 4.4 Three tradeoffs; profit vs. surface roughness (left), profit vs. cycle time (middle), and cycle time vs. surface roughness (right).

In Figure 4.4, the three tradeoffs between pairs of objectives are presented, and each individual symbol in the figures indicates a different combination of the process parameters. The profit is maximized as the surface roughness increases and cycle time decreases. However, customer requirements may set alternative upper bounds on the allowable surface roughness of the product, and the manufacturer must decide on an acceptable safety margin for their process, which can further bound this optimization. Although the quality cost is incorporated into the profit model and is directly related to surface roughness, other cost factors contribute more significantly to the overall cost of the product. Also, since the surface roughness is directly proportional to the square of the feed rate, the higher feed rates (shorter cycle time) cause poor surface quality. For a constant surface roughness, the profit decrease and the cycle time increases as the amount of tool wear increases. Therefore, these relationships enable a manufacturer to select the best combination of machining process parameter for their circumstances. By choosing an optimal set of parameters, the throughput and process of the manufacturing process can be optimized while minimizing new tool use and decreasing/eliminating poor quality products, thus creating a more sustainable manufacturing process.

As an example, two objective functions, -profit and surface roughness, are minimized. The constraints for variables are set to as follow: 0.2 mm <*VB*< 0.7 mm, 206.55 mm/min < Feed Rate < 413 mm/min, and 800 rev/min < RPM (Spindle Speed) < 1000 rev/min. Pareto optimal fronts at initial generation, 50th generation, and 100th generation are shown in Figure 4.5. At 100th generation, the best combination having minimum cycle time is decided as VB= 0.500445 mm, Feed Rate=252.1306 mm/min, and RPM=949.5869 rev/min. In Figure 4.6, three objective functions, -profit, surface roughness, and cycle time are minimized with fixed tool wear (= 0.3mm), and Pareto optimal front is shown. For this, the constraints for variables are set to as follow: 206.55 mm/min < Feed Rate < 413 mm/min and 800 rev/min < RPM (Spindle Speed) < 1000 rev/min.



(a) (b) (c)

Figure 4.5 Pareto optimal front; (a) initial generation, (b) 50th generation, (c) 100th generation.

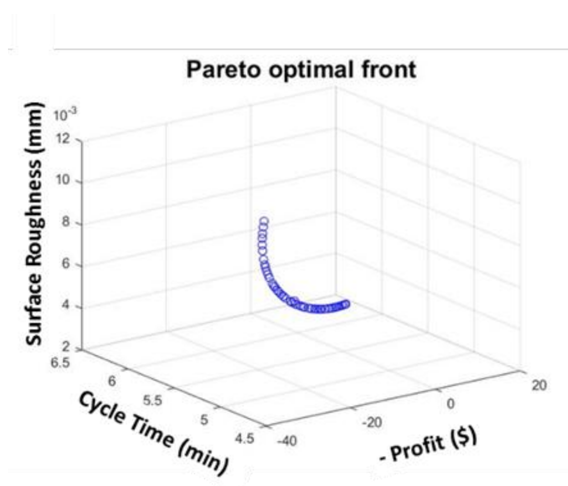


Figure 4.6 Pareto optimal front (100th generation).

4.4 Conclusion

In this paper, an intelligent tool condition monitoring system is proposed not only to monitor the condition of a machine tool, but also to identify an optimal set of machining conditions as a function of tool wear by optimizing tradeoffs between different objectives - profit, quality, and productivity. Since a tool’s performance changes over the machining time, tool condition information is incorporated in the multi-objective optimization technique to identify tradeoffs. The proposed monitoring system is expected to recommend a proper degree of tool utilization by maximizing a manufacturer’s needs. The recommend values enable better decision making, which can also help to reduce the amount of the scrap by controlling product quality.

5. MONITORING OF A MACHINING PROCESS USING KERNEL PRINCIPAL COMPONENT ANALYSIS AND KERNEL DENSITY ESTIMATION[4]

Reprinted with permission from Lee, W. J., Mendis, G., Triebe., M., and Sutherland, J. W., 2020, "Monitoring of a Machining Process using Multi Level Kernel Principal Component Analysis and Kernel Density Estimation," *Journal of Intelligent Manufacturing*

Abstract

Tool wear is one of the consequences of a machining process. Excessive tool wear can lead to poor surface finish, and result in a defective product. It can also lead to premature tool failure, and may result in process downtime and damaged components. With this in mind, it has long been desired to monitor tool wear/tool condition. Kernel principal component analysis (KPCA) is proposed as an effective and efficient method for monitoring the tool condition in a machining process. The KPCA-based method may be used to identify faults (abnormalities) in a process through the fusion of multi-sensor signals. The method employs a control chart monitoring approach that uses Hotelling's T2-statistic and Q-statistic to identify the faults in conjunction with control limits, which are computed by kernel density estimation (KDE). KDE is a non-parametric technique to approximate a probability density function (PDF). Four performance metrics, abnormality detection rate (ADR), false detection rate (FDR), detection delay (DD), and prediction accuracy (PA), are employed to test the reliability of the monitoring system and are used to compare the KPCA-based method with PCA-based method. Application of the proposed monitoring system to experimental data shows that the KPCA based method can effectively monitor the tool wear.

Keywords: Kernel Principal Component Analysis, Control Chart, Machining Process, Tool Condition Monitoring.

[4] This work was published in Journal of Intelligent Manufacturing [41]. The permission (license number: 5024361053735) is obtained to from Springer Nature include the paper in this thesis.

5.1 Introduction

Machining processes such as milling, turning, and drilling are widely used in industry. For machining processes, the surface finish is a critical measure of product quality. When problems occur in a machining operation, the surface may be damaged and a defective product may be produced. The machined part surface finish can be influenced by a number of factors, e.g., tool geometry, feed rate, cutting speed, and tool wear. In production, the gradual degradation, chipping, or even catastrophic failure of a cutting tool can all lead to poor surface texture. Since such problems are to be avoided, it would be desirable to monitor the surface finish of products generated over time. However, in practice, in-process surface finish measurement is very challenging, as is measuring tool wear in-process. Since neither surface finish nor tool wear can easily be monitored in-process, we generally look for proxy measures, i.e., more easily measurable quantities that are correlated to surface finish and tool wear.

With the advent of automated production, e.g., NC machining, automated tool condition monitoring has received considerable attention [89]. With an automated tool condition monitoring system, hardware and software are used to monitor the process and make decisions about the tool condition [90]. As with humans who use their senses to acquire a variety of information on the state of a machining process, an automated tool condition monitoring system may also use an assortment of sensors to learn about the process and the tool condition. Sensor signals have been used in a variety of different ways to estimate the condition of a tool, e.g., artificial neural network [36, 37], support vector machine [38], fuzzy system [92], and random forest [73].

Single sensor systems have been broadly employed to monitor tool condition (e.g., vibration sensors [93], acoustic emission sensors [94], and current sensors [95]). However, the sensitivity and noise rejection of the signals can vary with machining conditions [75]. The limitations of single sensor monitoring systems are well explained by Abellan-Nebot and Romero Subirón [75]. Because of the limitations, it is often assumed that obtaining information from multiple sensors is preferred to having data from a single sensor when endeavoring to predict tool condition [79]. However, the effective use and integration of data from multiple sensors to create the best possible description of the state of the process is challenging. These challenges include: (1) integrating the data from different sources (e.g., accelerometer and acoustic emission sensor), (2) integrating data of varying reliability (e.g., the data from a sensor is in error), and (3) synthesizing the data to estimate the state of the process (e.g., estimating the amount of flank wear using accelerometer and acoustic emission data). The integration of data from multiple sensors is often termed multi-sensor data fusion. This process has the goal of improving the reliability of sensor information [71] by enabling better predictive performance and diminishing the effects of noise. Multiple sensors can complement each other when monitoring tool condition. For example, an acoustic emission sensor may precisely estimate the state of the process while an accelerometer may not, or vice versa [96]. Methods to extract and evaluate meaningful values from multi-sensor signals in order to successfully monitor the condition of the machine tool are an on-going area of research [89].

The fusion of multi-sensor data requires an algorithm to process and synthesize the data. As a way to extract valuable information from multiple sensor signals, principal component analysis (PCA) has often been used [48, 49]. PCA is a mathematical procedure for reducing the dimensionality of data from multiple sensors by mapping the data onto new axes, which are called principal components [99]. The principal components are constructed by developing linear relationships among the original variables. Therefore, PCA plays an important role in extracting useful features/information from multi-sensor data [100]. However, PCA performs poorly in effectively addressing non-linear behaviors among sensor signals [101]. It is to be noted that dynamic data collected from sensors monitoring a machining process, e.g., accelerometer signals, may be related to one another by non-linear relations.

Kernel principal component analysis (KPCA) is an expanded form of PCA that can handle non-linear relationships among variables [102]. It transforms the observed data to a higher dimensional space using a non-linear function (kernel function). This method allows for the separation of normal and abnormal data obtained from manufacturing signals (that may have non-linear relationships), so it has the potential to identify whether a manufacturing process is under a good or bad (fault) condition.

In KPCA, after mapping the observed data into a new coordinate system, it is necessary to discriminate between normal and abnormal conditions. This is conventionally performed using a set of system-specific criteria. When the data exceeds certain limits, the data is identified as being abnormal (a fault is present) or “out-of-control” and an operator can work to address the fault. One way to identify faults during manufacturing operations is to develop a control chart and compare process data to the control chart limits.

Hotelling’s T^2-statistic and Q-statistic can be used to identify process abnormalities when working with multi-dimensional data that is time-varying [98]. Hotelling’s T^2-statistic and the Q-

statistic represent the Mahalanobis distance (variation in model subspaces) and the Euclidean distance (variation in residual subspaces), respectively [103]. Driven by the need to establish an automated tool condition monitoring system that can handle multi-sensor signals and monitor tool wear in real-time, in this paper, control charts for Hotelling's T^2-statistic and for the Q-statistic will be employed to monitor faults in the machining process and identify when tool changes should occur.

Control limits for Hotelling's T^2-statistic and for the Q-statistic are normally calculated using the F distribution and the χ^2 distribution with the assumption that the original data is independent and follows a multivariate Gaussian distribution [49, 55, 56]. For sensor signals obtained during milling experiments the data are likely highly auto-correlated, and thus, principal component scores are unlikely to follow the assumptions either. Thus, this assumption may result in less precise estimates of probabilities density function. With this in mind, a data-driven, and non-parametric approach, i.e., KDE, is used in this paper.

Tool wear is a critical factor influencing product quality, thus the wear should be controlled within a certain limit in a machining process. However, it is difficult to know tool condition during the process because the process is a nonlinear time-variant system [31]. Therefore, an effective on-line tool condition monitoring system is necessary to monitor tool failure in a non-intrusive environment. In the past, PCA has been extensively used to extract valuable information from multiple sensor signals while KPCA has been employed less. Accordingly, a difference in performance between PCA and KPCA has not been well reported. To apply a KPCA and control charts method in a machining process, control limits should be implemented. These control limits are based on a chosen risk level and should be properly selected. However, there has been a lack of research investigating tradeoffs of monitoring performance as control limits change. Thus, the uniqueness of this paper is as follows:

- a tool condition monitoring system is developed by applying the fundamental idea of KPCA with control charts to monitors tool wear during a machining process using proxy measures.
- in order to evaluate the proposed methods, four performance metrics, are studied, quantified, and their tradeoffs discussed.
- an empirical probability density function is used to find a control limit for a control chart.
- an optimal control limit is found for the method in terms of performance metrics.

This paper is organized as follows: first, an example for a non-linear behavior among sensor signals is briefly introduced and the advantages of KPCA over PCA are noted. Second, KPCA is discussed and it is shown how control charts may be developed from KPCA-modified data. Next, a method to analyze tool wear is developed, and the abnormal state is defined. Using experimental data collected from multiple sensors, a KPCA- and Hotelling's T^2 and Q control charts-based monitoring system is developed in two phases. In Phase 1, the model is trained using normal state data (the tool wear is below the wear limit) from a face milling operation, and control limits are calculated. In Phase 2, data that may or may not contain faults is visualized using the two control charts. These control charts can be used to detect abnormalities (faults) using the control limits (determined in Phase 1). To compare the performances of KPCA and PCA, the four performance metrics are evaluated, and an optimal control limit is investigated using the experimental data. Also, several KPCA models were trained using the different datasets. Each model is trained to monitor the tool wear during its predefined wear range. The training dataset for each model is decided by the predefined wear limits in this paper.

5.2 A Non-linear Behavior Example

When a sensor is deployed in a manufacturing process, the goal is often to acquire knowledge about some state variable (e.g., flank wear) that is not directly observable via the sensor signal. To achieve this goal, techniques have to be developed for analyzing sensor signals and for making decisions about the state (or condition).

Several approaches have been explored to integrate multi-sensor data to develop the best possible description of the state of the system [75]. One widely used method to integrate the data from multi-sensor systems is PCA [100]. PCA is a linear dimensionality reduction technique that removes the correlation among signal features. PCA is often used for high dimensional data to identify similarities and differences in data clustering behavior by extracting features, i.e., corrupted sensor data can be used as an input to extract features in PCA. This technique can simplify the complexity of the data without compromising the information in the original data.

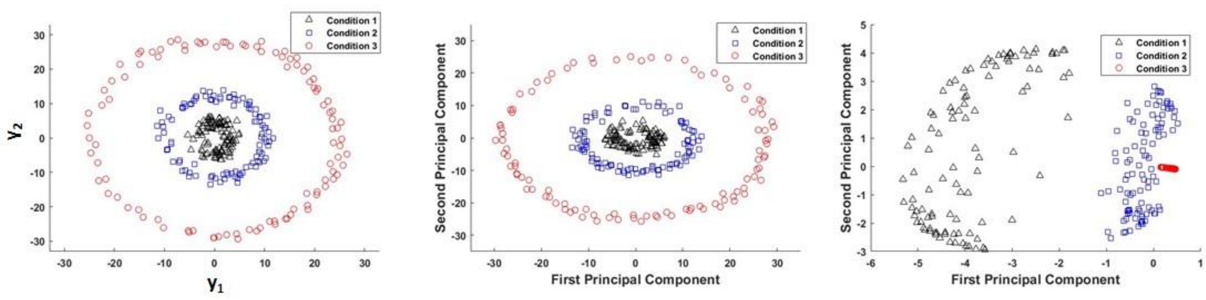
However, PCA may not be sufficient to describe all behaviors. Lever et al. [101] cautioned that PCA may fail to find non-linear data patterns. Several researchers have also expressed concerns about applying PCA to a non-linear signal due to its linear property [106]. To illustrate the problem when applying PCA to data obtained from a manufacturing process, consider a machining process for which it is desired to develop estimates of the condition of cutting tool (e.g., flank wear). Let us assume that a process is instrumented with two sensors (that provide signals y_1 and y_2) and each observed signal follows a different sinusoidal function as follows

$$y_1 = A\sin(t) + \varepsilon_1, \tag{5.1}$$

$$y_2 = B\cos(t + \pi) + \varepsilon_2, \tag{5.2}$$

where ε_1 and ε_2 are independent noise variables and the parameters (A, B) are (3, 5), (10, 12), and (25, 28) for Conditions 1, 2, and 3, respectively. Here, the Conditions 1, 2, and 3 are assumed to be low, medium, and high wear conditions. The noise variables are considered to reflect many small system factors that collectively produce a Gaussian pattern of variation (ε_1 and ε_2 are assumed N $(0,1^2)$). As is evident, the parameters A and B (the signal amplitudes) increase as the cutting tool condition shifts from low to high wear.

Values of y_1 and y_2 were generated for t = [0, 0.1, 0.2, …, 9.9, 10] for all three conditions (1, 2, and 3). As shown in Figure 5.1a, the y_1 and y_2 signals appear to form ellipses for each condition. As expected, transforming the data using a linear PCA transformation (Figure 5.1b) does not lead to the ability to discriminate the data from the different tool conditions. The result of applying KPCA to the three sets of y_1 and y_2 data is shown in Figure 5.1c. As is evident, the data from the three conditions are now clustered into groups that are linearly separated. Details regarding the KPCA method are provided in the next section.



(a) (b) (c)

Figure 5.1. Two sensor signals (a) in original space, (b) following PCA transformation, and (c) following KPCA transformation.

5.3 Kernel Principal Component Analysis (KPCA)

As a milling process progresses, the behavior of sensor data collected from the process is likely to change due to tool wear. In order to construct a monitoring system for process faults using KPCA, two phases need to be conducted. In Phase 1, the model is trained using data from the process when it is operating in-control (data from normal conditions). After training the model, control limits for the normal behavior of a milling process can be calculated. In Phase 2, data collected under both normal (in-control) and abnormal conditions (out-of-control) may be compared to the control limits and the process performance evaluated. Comparison of the data from Phase 2 with the control limit allows for abnormal behavior to be detected. The overall performance of the monitoring system can be assessed using such performance metrics as abnormality detection rate, false detection rate, and detection delay.

KPCA transforms the observed data into a higher dimensional space using a non-linear function (kernel function) and generates principal components using a method known as the "kernel trick." The transformed space is sometimes referred to as the "feature space." This technique is beneficial in situations where a linear transformation of the data in the original space does not provide discriminatory power. By mapping the original data using a kernel function, the transformed data can be separated in the newly mapped space. This enables the extraction of relevant features from the data, and can improve the predictive capacity of the method.

KPCA takes a normalized observed data matrix $X \in \mathbb{R}^{m \times n}$ with m rows (observations) and n columns (variables), and transforms it into the feature space using a non-linear mapping function $\phi(\boldsymbol{x})$. This can be mathematically written as $\boldsymbol{X} \in \mathbb{R}^{m \times n} \to \phi(\boldsymbol{X}) \in \mathbb{R}^{F}$ (F: feature space). To find new axes, a typical PCA technique can be applied in the feature space, but this is computationally expensive (Wang 2012). Instead, the kernel trick method is applied [108]. Using this method, a number of principal components can be extracted, up to m, while simultaneously decreasing computation time. In the kernel method, $\boldsymbol{\kappa}(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ and $\boldsymbol{\kappa}$ are called the kernel function and the kernel matrix, respectively.

When mapping the original data, the feature space can be accessed by a kernel function, which enables the linear separation of the data to identify a behavior in the feature space. Therefore, an appropriate kernel function should be selected depending on the type of data. The most widely used kernel functions are the polynomial kernel, the Gaussian kernel, and the hyperbolic tangent (*tanh*) kernel. When a kernel function is chosen, the kernel matrix can be directly computed from the training data. From this, the behavior of the feature space data may be identified when the behavior of the original data changes with time.

In Phase 1, a score matrix is created. The raw sensor data is transformed by KPCA to produce data in the feature space – the data in the feature space is termed the score matrix. The detailed procedures to calculate the score matrix are presented below.

5.3.1 Phase 1: Offline Training

To calculate the score matrix of the training data in Phase 1, a coefficient matrix (eigenvector), $\boldsymbol{\alpha}$, whose corresponding eigenvalues are organized in descending order must be obtained. This can be done by applying eigen decomposition to the kernel matrix. To do this, a covariance matrix ($\boldsymbol{C}_F$), must be determined first:

$$\boldsymbol{C}_F = \frac{1}{m}\sum_{i=1}^{m} \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i)^T. \tag{5.3}$$

Then, the eigenvalues (λ) and eigenvectors (e) that satisfy the following equation must be found

$$\boldsymbol{C}_F\boldsymbol{e}_p = \lambda_p\boldsymbol{e}_p, \tag{5.4}$$

where p represents the p^{th} dimension (p=1, 2, …, m) in the feature space. Using Eqs. (5.3) and (5.4), we have

$$\frac{1}{m}\left[\sum_{i=1}^{m}\phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i)^T\right]\boldsymbol{e}_p = \frac{1}{m}\sum_{i=1}^{m}\left(\phi(\boldsymbol{x}_i)\cdot\boldsymbol{e}_p\right)\phi(\boldsymbol{x}_i) = \lambda_p\boldsymbol{e}_p, \tag{5.5}$$

and the eigenvectors can be stated as a linear combination of the m mapped data values as follows [109]

$$\boldsymbol{e}_p = \frac{1}{m\lambda_p}\sum_{i=1}^{m}(\phi(\boldsymbol{x}_i)\cdot\boldsymbol{e}_p)\phi(\boldsymbol{x}_i) = \sum_{j=1}^{m}\alpha_{p,j}\phi(\boldsymbol{x}_j), \tag{5.6}$$

where $\boldsymbol{\alpha}$ is a coefficient matrix. By inserting $\boldsymbol{e}_p$, obtained from Eq. (5.6), into Eq. (5.5), Eq. (5.5) can be rewritten as

$$\frac{1}{m}\sum_{i=1}^{m}\phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i)^T\sum_{j=1}^{m}\boldsymbol{\alpha}_{p,j}\phi(\boldsymbol{x}_j) = \lambda_p\sum_{j=1}^{m}\boldsymbol{\alpha}_{p,j}\phi(\boldsymbol{x}_j). \tag{5.7}$$

Using the definition of the kernel function, $\boldsymbol{\kappa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$, and multiplying by $\phi(\boldsymbol{x}_k)^T$ on both sides of Eq. (5.7), the equation can be rewritten as follows:

$$\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{\kappa}(\boldsymbol{x}_k, \boldsymbol{x}_i)\sum_{j=1}^{m}\alpha_{p,j}\boldsymbol{\kappa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \lambda_p \sum_{j=1}^{m}\alpha_{p,j}\kappa(\boldsymbol{x}_k, \boldsymbol{x}_j), \tag{5.8}$$

where k =1, 2 …, m. When converting Eq. (5.8) into matrix notation [104],

$$\boldsymbol{\kappa}^2\boldsymbol{\alpha} = \lambda m \boldsymbol{\kappa}\boldsymbol{\alpha}, \tag{5.9}$$

where $\kappa \in \mathbb{R}^{m\times m}$.

To shift the mean of the mapped data from the n sensors, $\phi(\boldsymbol{x}_i)$ to zero, Eq. (5.10) may be applied. Equation (5.11) serves to center the kernel matrix:

$$\tilde{\phi}(\boldsymbol{x}_{i\ or\ j}) = \phi(\boldsymbol{x}_{i\ or\ j}) - \frac{1}{n}\sum_{i=1}^{m}\phi(\boldsymbol{x}_{i\ or\ j}), \tag{5.10}$$

$$\tilde{\kappa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tilde{\phi}(\boldsymbol{x}_i)^T \tilde{\phi}(\boldsymbol{x}_j), \tag{5.11}$$

where $\tilde{\boldsymbol{\kappa}}$ is centered kernel matrix. Equation (5.11) can be rewritten in matrix form as follows:

$$\tilde{\boldsymbol{\kappa}} = \boldsymbol{\kappa} - \mathbf{1}_m\boldsymbol{\kappa} - \boldsymbol{\kappa}\mathbf{1}_m + \mathbf{1}_m\boldsymbol{\kappa}\mathbf{1}_m, \tag{5.12}$$

where $\mathbf{1}_m$ is an $m\times m$ matrix in which each element is equal to $1/m$. By replacing $\boldsymbol{\kappa}$ with $\tilde{\boldsymbol{\kappa}}$ in Eq. (5.9), the solution of the eigen problem given by the following equation,

$$\tilde{\boldsymbol{\kappa}}\boldsymbol{\alpha} = \lambda m \boldsymbol{\alpha}, \tag{5.13}$$

yields the eigenvectors $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_{m\text{-}1}, \boldsymbol{\alpha}_m$ and the corresponding eigenvalues in descending order, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{m-1} \geq \lambda_m$. To normalize the eigenvectors, Eq. (5.14) may be applied [109]:

$$\boldsymbol{e}_P^T\boldsymbol{e}_p = \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_{p,i}\alpha_{p,j}\phi(\boldsymbol{x}_i)^T\phi(\boldsymbol{x}_j) = \lambda_P\boldsymbol{\alpha}_P^T\boldsymbol{\alpha}_p = 1. \tag{5.14}$$

Using the training data, $\boldsymbol{x}_j$ where j=1,2,…, m, the score matrix, i.e., the data in the principal component space, can be calculated using:

$$z_{j,p} = \sum_{i=1}^{m}\alpha_{p,i}\tilde{\boldsymbol{\kappa}}(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{5.15}$$

The score matrix, $\boldsymbol{z}$, will be used to calculate Hotelling's T^2-statistic and Q-statistic. If the model is trained with data obtained from normal operating conditions, the next step is to monitor a process using the developed model. In Phase 2, the model may be applied to the data that may contain abnormal conditions.

5.3.2 Phase 2: Online Monitoring

Once the trained model from Phase 1 is established, we can shift to Phase 2. For *n*-dimensional normalized test data, $\boldsymbol{x}_t \in \mathbb{R}^n$, the test kernel vector $\boldsymbol{K}_t \in \mathbb{R}^{1\times m}$ can be formulated with the training data used in Phase 1 as follows:

$$\left[\boldsymbol{K}_t\right]_i = \boldsymbol{\kappa}_t(\boldsymbol{x}_i, \boldsymbol{x}_t), \tag{5.16}$$

where $\boldsymbol{x}_i \in \mathbb{R}^n$, i=[1, 2, … , m], t=[1, 2, … , k], and k is the number of test data values. Then, the centered test kernel vector can be calculated:

$$\tilde{\boldsymbol{K}}_t = \boldsymbol{K}_t - \mathbf{1}_t\boldsymbol{\kappa} - \mathbf{K}_t\mathbf{1}_m + \mathbf{1}_t\boldsymbol{\kappa}\mathbf{1}_m, \tag{5.17}$$

where $\mathbf{1}_t = 1/m[1,\ 1,\ ..,\ 1] \in \mathbb{R}^{1\times m}$. Using the centered test kernel matrix, the score matrix for the test data is

$$\mathrm{Z}_{t,p} = \sum_{i=1}^{m} \alpha_{p,i}\tilde{\boldsymbol{K}}_t(\boldsymbol{x}_i, \boldsymbol{x}_t). \tag{5.18}$$

The data (newly mapped into principal component axes) is described by the score matrix. The next step is to select the number of principal components (p).

To ensure the method accurately represents the original data, a suitable number of principal components must be selected to ensure that there is no loss of information relative to the original data. Since the goal of PCA/KPCA is to reduce the dimensionality of the data, using all principal components is not desired. The cumulative explained variance (CEV) is used to select an appropriate number of principal components. As seen below, CEV describes the amount of variation of the selected principal components relative to the total variation associated with all principal components.

$$\text{CEV (\%)} = \sum_{i=1}^{p}\lambda_i \Big/ \sum_{i=1}^{n}\lambda_i \times 100. \tag{5.19}$$

The number of principal components considered in the model, p, may be increased until the CEV suitably describes all the information content. Usually, the number of principal components is chosen to ensure that the CEV exceeds 90% [110].

With an appropriate number of principal components determined, control charts can be generated. Control charts are often used in industry to identify and act upon abnormalities that occur during a manufacturing process. The application of KPCA to control charts will be discussed in the next section.

5.4 Control Charts for Abnormality Detection

After mapping the observed data into the new dimensional space, it is desired to distinguish abnormal behaviors as they arise relative to normal behavior. This can be achieved through the use of a control chart, which organizes information in a simple visual manner to determine when a process is in- or out-of-control. In this paper, control charts for Hotelling's T^2-statistic and for the Q-statistic are used to detect abnormalities during the milling process. Hotelling's T^2-statistic employs the Mahalanobis distance (variation in model subspaces) while the Q-statistic utilizes the Euclidean distance (variation in residual subspaces) [103].

5.4.1 Hotelling's T^2-statistic and the Q-statistic

Hotelling's T^2-statistic explains the variation of observed data and can be calculated as follows:

$$T_j^2 = \left[z_{j,1}, \cdots, z_{j,p} \right] \mathbf{\Omega}^{-1} \left[z_{j,1}, \cdots, z_{j,p} \right]^T , \tag{5.20}$$

where z is a score matrix, Ω is a diagonal matrix of the eigenvalues corresponding to the selected principal components, p is the number of selected principal components, j =[1, 2, … , m], and m is the number of samples. In Eq. (5.20), z can be replaced by the values for Z that were calculated in Eq. (5.18).

The T^2-statistic is able to distinguish abnormal from normal data using the degree of variation from the trained model. However, if the model is not fit appropriately, it may fail to detect abnormal data, especially when the difference between normal and abnormal data is relatively small. To compensate for this weakness, the Q-statistic may be used [111]. The Q-statistic is the difference between the sum of the squared variations in the entire feature space and the sum of the squared variations in the principal component space. An inappropriate selection of the number of principal components (p) may cause a large error in the Q-statistic. A large error can also arise if there is a structural change in the data. Mathematically, the Q-statistic can be written as [104]:

$$Q_j = \sum_{i=i}^{m} z_{j,i}^2 - \sum_{i=i}^{p} z_{j,i}^2. \tag{5.21}$$

Again, Z can be used in place of z. For better understanding, Mahalanobis (T^2) and Euclidean (Q) distances are graphically represented in Figure 5.2 using original space (X-Y) and two principal component space.

Tracking the T^2- and Q-statistics over time are complementary methods to detect abnormalities. For T^2 and Q control charts that have been developed based on "in control" behavior, an abnormal event can be detected when either a T^2 or Q value exceeds the control limits. This will be discussed in the next section.

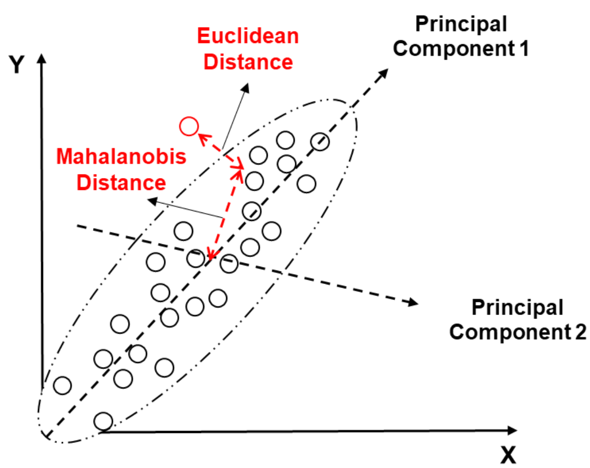


Figure 5.2. Mahalanobis distance (T^2) and Euclidean distance (Q).

5.4.2 Determination of Control Limits Using a Kernel Density Estimation

Given the m values for the T^2- and Q-statistics from Eq. (5.20) and Eq. (5.21), control limits can be established for a process behaving normally. Then, as additional data are subsequently collected, abnormal behavior can be detected. In control charts for T^2-statistic and Q-statistic, only upper control limits are used. Often, the control limits for T^2 and Q control charts are computed using F and $\chi 2$ distributions, respectively, with the assumption that the original data is independent and follows a multivariate Gaussian distribution (Ketelaere et al. 2015). Since such an assumption may not be true for our application, we establish control limits using KDE. KDE is a data-driven, and non-parametric approach to approximate a PDF. KDE creates a PDF of a random variable x (x is T^2 or Q) to smoothly fit the data. Given a random sample $(x_1, x_2, \ldots, x_w)$, KDE creates the PDF using the following relation:

$$PDF(x) = \frac{1}{wh}\sum_{i=1}^{w} K\left(\frac{x - x_i}{h}\right), \tag{5.22}$$

where K is a kernel function, h is the bandwidth (a smoothing parameter), and w is the number of samples. Silverman [112] provides some examples of widely-used kernel functions and bandwidths,

$$K(x) = \frac{\exp(-x^2 / 2)}{\sqrt{2\pi}}, \tag{5.23}$$

$$h = \left(\frac{4\hat{\sigma}^5}{3w}\right)^{0.2}, \tag{5.24}$$

where $\hat{\sigma}$ is the estimated standard deviation of x (σ).

For a given risk level (1-β), and T^2-statistic and Q-statistic values, the upper control limit c_l of the control charts can be defined as follows:

$$P(x < c_l) = \int_{-\infty}^{c_l} PDF(T^2 or\ Q)\ dx = \beta. \tag{5.25}$$

The risk level, 1-β, is the probability of a point falling beyond the upper control limit due to chance when the process is "in control."

A control limit is a key factor discriminating in control and out of control. Improper selection of the control limit can lead to a large false positive (Type 1) and false negative (Type 2) error. Therefore, in this paper, various control limits between 90.0% and 99.9% are investigated to find an optimal limit (alarming threshold, i.e., control limit, is represented by a percentage using β in this paper). Once limits have been established for the T^2 and Q control charts, attention shifts to using the charts for the purpose of process monitoring. During this process monitoring phase, we are interested in how well the control charts perform, e.g., how rapidly abnormal behavior is detected.

5.4.3 Performance of the Monitoring System

To ensure the reliability of the monitoring system, its performance must be studied. Abnormality detection rate (ADR), false detection rate (FDR), detection delay (DD), and prediction accuracy (PA) have been employed to quantify the performance of the monitoring system. ADR quantifies the rate of abnormality detection after the process has entered an abnormal state. FDR measures the rate of false detection before the abnormal state. These two metrics work in tandem to identify whether the KPCA analysis has scaled the data to an appropriate signal-to-noise level, so that the monitoring system is able to discriminate between normal and abnormal data. While an occasional outlier will always occur (Type I error), these metrics can help identify when too many outliers are detected. DD is the time delay between when a process enters an abnormal state and the abnormal state is detected. If the DD is excessive, the abnormality is not suitably addressed by the operator in a timely manner. These three metrics (ADR, FDR, and DD) are written as

$$\text{ADR}(\%) = \frac{number\ of\ data\ detected\ as\ abnormal}{number\ of\ actual\ abnormal\ data} \times 100, \tag{5.26}$$

$$\text{FDR}(\%) = \frac{number\ of\ false\ detection}{number\ of\ actual\ normal\ data} \times 100, \tag{5.27}$$

$$\text{DD} = predicted\ first\ abnormal\ sample\ number - actual\ first\ abnormal\ sample\ number. \tag{5.28}$$

PA can be simply calculated by dividing the sum of true positive and true negative by total number of total observation.

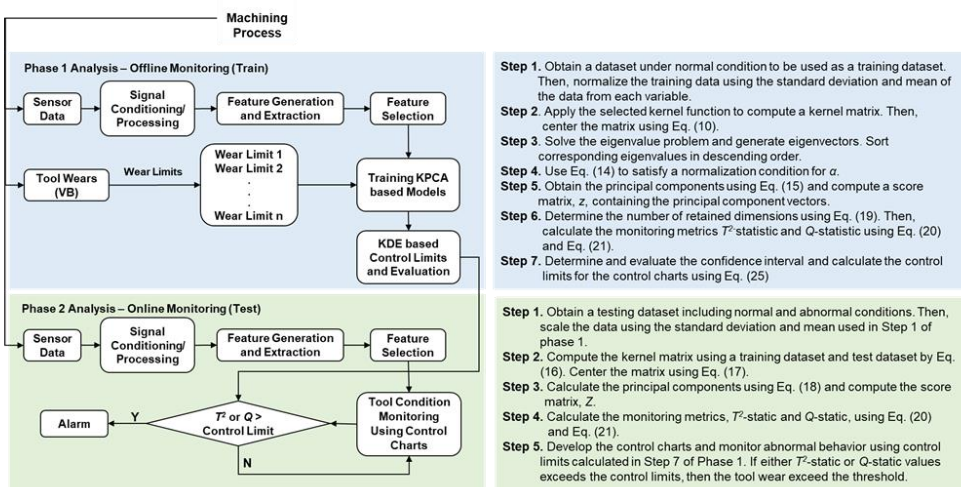


Figure 5.3 Procedure for development and use of the monitoring system (in the figure the chapter numbers are omitted in the equation numbers).

After all the metrics are obtained, the monitoring system can be applied to real data obtained from a process. The detailed procedure to establish the system is described in Figure 5.3. As the milling process progresses, the behavior of observed sensor data is likely to change due to increasing amounts of tool wear. Therefore, abnormality must be defined to monitor the condition of the machine tool.

5.5 Monitoring and Detecting Abnormalities in a Milling Process

An abnormality can be defined as "cutting tool failure or faults," which can be caused by breakage, rapid dulling, or gradual wearing of the tool. Tool breakage is caused by excessive forces and can be catastrophic to the process and perhaps the machine tool. Rapid tool dulling is caused by excessive heat generation and intense stress during the cutting process. Gradual wear is caused by normal use of the tool (e.g., heat generation and stress). Gradual wear is defined as a steady increase in wear length on the flank and rake faces of the cutting tool. For this paper, we define cutting tool failure as when the measured flank wear exceeds a specified limit. Flank wear is a widely used metric to determine the remaining life of a cutting tool. However, the flank wear limit of a given tool may vary with manufacturer, brand, and type. According to ISO 3685-1977 [112], tool dullness is related to the average height of wear on the flank face, and the wear is recommended to be less than 0.3 mm to ensure product quality [97]. Several researchers have used 0.2 mm as a tool wear limit in experiments [64, 65]. Based on the previous studies and the ISO standard, in this paper, several wear limits (W1=0.15 mm, W2=0.2 mm, W3=0.3 mm) are employed during the study of the monitoring system's performance.

Following the training period that establishes the control charts limits during normal behavior, the system enters the monitoring phase. If the amount of the observed tool wear exceeds the wear limit, it is identified as an abnormality (fault) by the method.

A model is trained with data obtained while the cutting tool wear is under the wear limit. This results in transformed data in the principal component space that is in a "normal state." Because three wear limits are defined in this paper, three KPCA models are trained using their corresponding training dataset, i.e., the data collected while a cutting tool is under the corresponding wear limit. After training, the models can identify performance (high tool wear) that does not fit with this normal state. When data does not fit with the normal state behavior, the process is said to be in an “abnormal state.” The proposed methodology can be used to test the automated tool condition monitoring system using experimental data.

5.6 Experiment Set Up

Milling processes see widespread use in industry. During a milling process, the material is deformed plastically in the shear zone. The deformation requires force, induces vibrations, and creates other tangible evidence of its occurrence. This evidence may be acquired by sensors (e.g., accelerometer, force dynamometer, and microphone). As the tool wears, more energy is required to achieve the deformation. This change can be related to the tool condition because it is reflected in the power (current), vibration, acoustic emission, etc. during the cutting process. As a result, data acquired by sensors can be used to inform models used to monitor the milling process.

Data from milling experiments available at the NASA repository was used to test the proposed tool condition monitoring system [77]. The experiments contain data that represent both normal (below wear limit) and abnormal (above wear limit) states, which enables the development of an algorithm to separate the two states. Some of the key details of the experiments are noted below; others may be found in the cited work.

In the experiments, milling operations were conducted on a Matsuura machining center MC-510V under various cutting conditions (Table 5.1) to investigate tool wear. A 70 mm face mill with six TiC/TiC-N/TiN coated KC 710 tool inserts was used to cut the workpiece. The milling experiments were performed at a cutting speed of 200 m/min (826 rev/min), two axial depths of cut (0.75 mm and 1.5 mm), and two feeds (0.25 mm/rev and 0.5 mm/rev). Two workpiece materials (cast iron and J45 stainless steel) were used, with initial dimensions of 483 mm (l) by 178 mm (w) by 51 mm (h). The height of the workpiece was reduced by the axial depth of cut during the course of three parallel passes of the face mill over the workpiece. Following each pass (or run), the flank wear (VB) was measured with a microscope and recorded. Thus, the more a tool is used, the more the tool wears. Once the wear became very large (i.e., far larger than the wear limit), the experiment was stopped. The number of runs was dependent on the process parameters and workpiece material hardness.

During the milling operation, a current converter (OMRON, K3TB-A1015), a current sensor (FLEXCORE, CTA 213), an acoustic emissions sensor (PHYSICAL ACOUSTIC GROUP, WD 925), and an accelerometer (ENDEVCO, 7201- 50) were used to collect data. The acceleration/vibration and the acoustic emission sensors were mounted on the table and spindle. Six different data streams were acquired: (1) AC spindle motor current, (2) DC spindle motor current, (3) table acceleration (VBtable), (4) spindle acceleration (VBspindle), (5) acoustic emission at table (AEtable), and (6) acoustic emission at spindle (AEspinlde). The data was digitized and fed through an RMS meter to produce 9,000 data points for each run.

Table 5.1. Experiment condition.

Experiment Number	Depth of Cut (mm)	Feed (mm/rev)	Material
1	1.5	0.5	Cast Iron
2	0.75	0.5	Cast Iron
3	0.75	0.25	Cast Iron
4	1.5	0.25	Cast Iron
5	0.75	0.25	Stainless Steel
6	0.75	0.5	Stainless Steel
7	1.5	0.25	Stainless Steel
8	1.5	0.5	Stainless Steel

Each run contained data from the non-cutting portion of the operation, before the tool made contact with the workpiece, and after the face mill stopped cutting. These data points were not included in the analysis (only data during the cutting portion of the operation was considered). For each run, 175 RMS data were uniformly sampled from the cutting portion of the 9,000 data set (approximately one-second interval between data points). The 175 data point sets for multiple runs were appended together to provide data for an entire experiment, and they were divided into a training and a testing dataset.

In the next section, the proposed monitoring system will be tested using the milling data to show how well the combination of the KPCA algorithm and control charts are able to monitor the abnormal behavior during the milling process.

5.7 Application of the KPCA and Control Charts Methods to the Experiment Data

In order to implement the KPCA based monitoring system, MATLAB software was used. To compare the performance of KPCA with PCA, a polynomial kernel ($\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^T \boldsymbol{x}_j + r)^d$) was selected with the kernel function parameters set to r=1 and d=3; the polynomial kernel was found to most effectively separate the given data. As has been noted, three wear limits (W1, W2, and W3) were used to study the monitoring performance. But, here, let's focus on one wear limit (0.2

mm) first to visualize control chart in this section, i.e., a 0.2 mm wear limit was used to differentiate between a “sharp” tool and a dull tool (a single KPCA model). Also, a control limit, 99%, was selected to look for statistical signals. Data collected with a sharp tool should exhibit normal behavior, while data taken with a dull tool may demonstrate abnormal behavior. Abnormal behavior is determined by alarming thresholds, i.e., T^2 and Q control limits. These thresholds are computed by KDE using Eq. (5.22) and are shown in the control charts using horizontal dashed lines. To indicate the point at which the sharp tool becomes dull (i.e., the first sample for which the wear limit of 0.2 mm is exceeded), the abnormality starting point (ASP) is marked in the control charts using a vertical dotted line. Accordingly, a vertical line in the control charts indicates a ground truth.

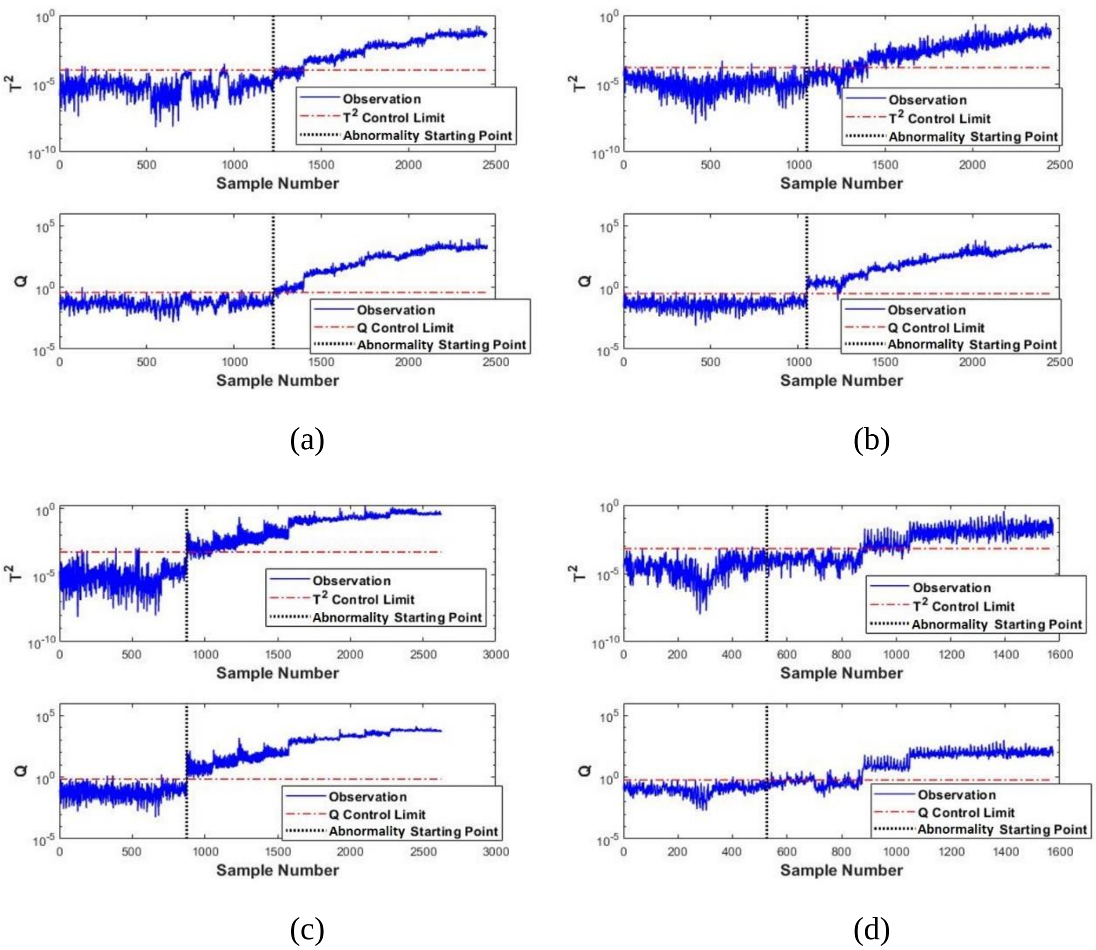


Figure 5.4 Control charts for T^2 and Q statistics; (a) Experiment 2, (b) Experiment 3, (c) Experiment 5, and (d) Experiment 6.

5.7.1 Development of Control Charts Using the Milling Data

KPCA is used to convert raw multi-sensor data into the principal component space. Control charts for T^2 and Q may then be established to monitor the process during the cutting operation; tool wear behavior can be related to the behavior on the control charts. The data for experiments 2, 3, 5, and 6 in Table 5.1 were selected to be examined. Control charts for these data with 99% control limit (horizontal dashed line) are shown in Figure 5.4.

To train the models, 1225, 1050, 875, and 525 data points out of 2450, 2450, 2625, and 1575 were used, respectively, for Figure 5.4a-d. These data points were chosen because the tool wear is below the wear limit. To achieve a 90% CEV, 22 principal components were used, and control limits were calculated using the KDE as discussed in section 5.4.2. After the data points listed above, the tool enters the dull state. Thus, beginning with samples 1226, 1051, 876, and 526 (for Figure 5.4a, Figure 5.4b, Figure 5.4c, and Figure 5.4d), we begin the process monitoring phase to look for statistical signals. Because of our knowledge of the manner in which the experiments were performed, we also happen to know that the process is truly in an abnormal state (dull tool).

As shown in Figure 5.4, T^2 and Q values were plotted relative to the control limits. Before the ASPs, the T^2 and Q values were computed using the data collected when a sharp tool was employed in the operations. Therefore, the values are expected to be below their respective control limits. Starting from ASPs, it is expected that we will have occasional T^2 and Q values below their respective control limits since a dull cutting tool is being used in the operation. In general, the control charts in Figure 5.4 show that the Q control charts captured the abnormalities faster than the T^2 control charts. However, as mentioned earlier, these two control charts have a complementary relationship. If either a T^2 or Q value exceeds its control limits, then this signal indicates that the process has entered an abnormal state.

To evaluate the performance of the monitoring system, the four performance metrics DD, ADR, FDR, and PA were employed. For Figure 5.4a, Figure 5.4b, and Figure 5.4c, the DDs were 0. This means the control charts signaled the presence of an abnormality as soon as the tool entered the dull state. Since the feed rate was 206.5 mm/min (= 826 rev/min * 0.25 mm/rev) in Experiment 3 and 5 (Figure 5.4b and Figure 5.4c), the cutting time for one run was approximately 2.3 minutes (workpiece length: 483 mm). Therefore, the interval between two data points was approximately 0.8 seconds (0.4 seconds when feed is 0.5 mm/rev). For Figure 5.4d, delays of 2 points were observed from the time at which a dull tool began to be used, which corresponds to a small time period.

The ADRs were as 98.45%, 99.71%, and 100% for Figure 5.4a, Figure 5.4b, and Figure 5.4c, respectively. This means that the KPCA based control charts are very sensitive to detecting the use of dull tool; as the tool becomes duller values on the control charts are even further beyond the control limits (i.e., the models can capture most of the out-of-control points after the process has entered an abnormal state). In Figure 5.4d, the ADR value was small relative to the other cases. However, although several true abnormal data points were not evaluated as "out of control" by the method, the first alarm was indicated with only a small detection delay; this signal is important information to an operator.

The FDRs were computed as 0.98%, 1.14%, 0.91%, and 1.14% for Figure 5.4a, Figure 5.4b, Figure 5.4c, and Figure 5.4d, respectively. Since the 99% ($\beta = 0.99$) control limits were established using the KDE method, nominally 1% of the training data are expected to be above the control limit before the process enters the abnormal state. As expected, in all tests, the number of false alarms were in that range. To decrease the number of false alarms, looser control limits (increased β value) may be used, but this would make the charts less sensitive when it is desired to quickly detect signals associated with a dull tool.

The performance metrics for all experiments are described in Table 5.2. In the table, KPCA method is compared with PCA method to investigate the effectiveness of KPCA in terms of the performance metrics (to ensure that the CEV exceeds 90% in both PCA and KPCA, 4 and 22 principal components are used in PCA and KPCA, respectively). Overall, KPCA outperforms PCA: the average values of the ADR, FDR, DD, and PA for KPCA were 93.90% 1.04%, 3, and 95.77%, and for PCA were 86.08%, 1.09%, 25, and 91.03%, respectively.

To summarize, the multi-sensor signals collected during the milling operations were processed by the KPCA method, and the KPCA-modified data were used to calculate T^2 and Q values to describe the behavior of the tool wear in the control charts. To differentiate a sharp (normal) and a dull (abnormal) tool in a control chart, a 99% control limit was computed using the KDE, and the T^2 and Q values were plotted relative to the control limits. The performance of the monitoring method was evaluated by the three performance metrics (ADR, FDR, DD, perdition accuracy). Across the entire experiment, the first abnormality (alarm) is identified with very few (or zero) delays after the process has entered the abnormal state.

Table 5.2 Performance metrics for all experiments using KPCA and PCA.

Experiment Number	Method	ADR	FDR	DD	PA
1	KPCA	100%	0.91%	0	99.73%
	PCA	98.00%	1.26%	0	98.22%
2	KPCA	98.45%	0.98%	0	98.73%
	PCA	85.80%	1.14%	148	92.33%
3	KPCA	99.71%	1.14%	0	99.35%
	PCA	99.29%	1.05%	0	99.14%
4	KPCA	98.38%	1.14%	0	98.57%
	PCA	90.86%	1.14%	0	94.06%
5	KPCA	100%	0.91%	0	99.70%
	PCA	95.66%	1.03%	0	96.76%
6	KPCA	74.95%	1.14%	2	82.92%
	PCA	62.76%	0.95%	17	74.86%
7	KPCA	86.37%	0.97%	20	91.88%
	PCA	76.04%	1.26%	20	85.91%
8	KPCA	93.33%	1.14%	0	95.31%
	PCA	80.19%	0.91%	14	86.94%

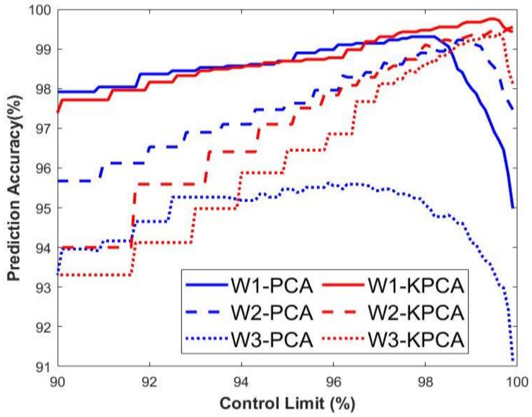
However, as noted above, since a control limit is a crucial factor in discriminating normal and abnormal, the improper selection of control limit may make PCA better than KPCA. Thus, an optimal control limit should be identified to enhance the monitoring system. In the next section, control limits between 90.0% and 99.9% are investigated to find an optimal control limit in terms of four performance metrics.

5.7.2 Enhancing Monitoring Performance Using an Optimal Control Limit

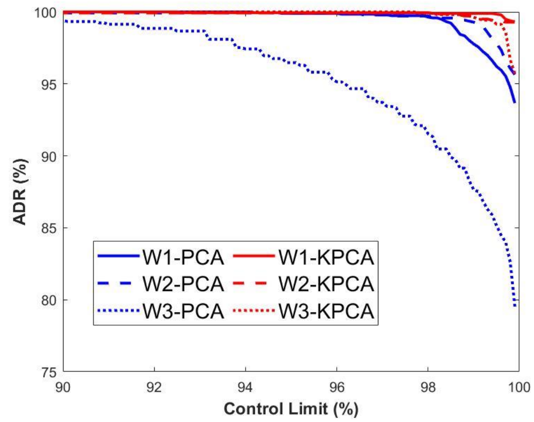
In this section, we investigated the change in the performance metrics as a control limit varies from 90.0% to 99.9%. Also, three different wear limits (W1, W2, and W3) were considered to see whether the KPCA method is always outperforming the PCA method. Data from Experiment 3 in Table 5.1 were selected to compare the two methods, and the three performance metrics (PA, ADR, and FDR) are plotted in Figure 5.5; KPCA clearly outperforms PCA for DD. For PA, as shown in Figure 5.5, the maximum values for W1, W2, and W3 (99.31%, 99.22%, and 95.63% for PCA and 99.76% 99.55%, and 99.39% for KPCA) were observed when β is 0.982, 0.987, and 0.959 for PCA and 0.995, 0.999, and 0.996 for KPCA. PCA show a higher PA when a control limit is lower in the range while KPCA becomes better as the control limit becomes higher in the range. In Figure 5.5b and Figure 5.5c, ADR and FDR are evaluated. KPCA shows a better performance in ADR while PCA performs better in FDR. In ADR, all values decrease as control limits, i.e., increasing β value, since the charts become less sensitive as mentioned above. On the other hand, the number of false detections, i.e., FDR, decrease as the control limit increases. PCA performed better for FDR in overall, but no distinct difference was observed between PCA and KPCA as the control limit increases. As shown in Figure 5.5, there are tradeoffs among the performance metrics. Therefore, the control limit should be selected carefully to enhance the monitoring capability in the machining process. The comparison of the performance metrics for all experimental data when β =0.98, 0.99, and 0.999 are summarized in Table 5.3.

As shown in Figure 5.5, KPCA does not always outperform PCA, but higher performance can be achieved when a proper control limit is selected. This may be due to the ability of KPCA to handle non-linear behavior (e.g., acceleration and acoustic emission) by mapping data to a higher dimensional space where it can be linearly clustered. Thus, PCA may work well for sensor data showing linear behavior. In the application of this study, i.e., analyzing sensor data collected from a milling process, KPCA showed a better performance for all sensor data obtained from various cutting conditions in terms of the performance metrics.

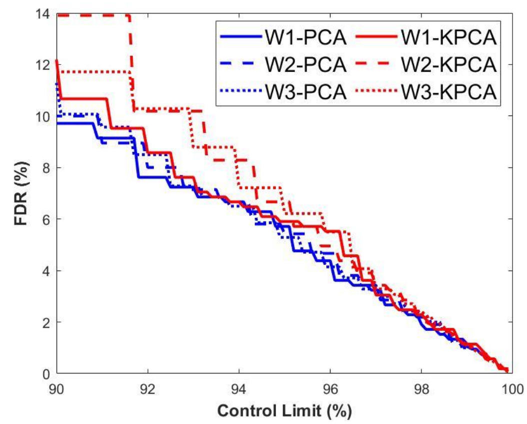
The data from Experiment 3 were also applied to several popular classification techniques. These include Decision Tree (DT), K-Nearest Neighbors (KNN), Naive Bayes Classifier (NBC), and Quadratic Discriminant Analysis (QDA). To show the competitiveness of the proposed method, one of the performance metrics, PA, is compared among the classification techniques for three wear limits (PA is a key performance metric for classification algorithms). This can be seen in Figure 5.6. All methods show high accuracies (more than 90%), but on average a KPCA- and Hotelling's T^2 and Q control charts-based monitoring system has the highest accuracy (99.41%), followed by DT (99.03%), KNN (98.07%), QDA (98.07%), NBC (97.57%), and PCA (96.41%).



(a) PA



(b) ADR



(c) FDR

Figure 5.5 Comparisons of the performance metrics for three wear limits against various control limits (Experiment 3).

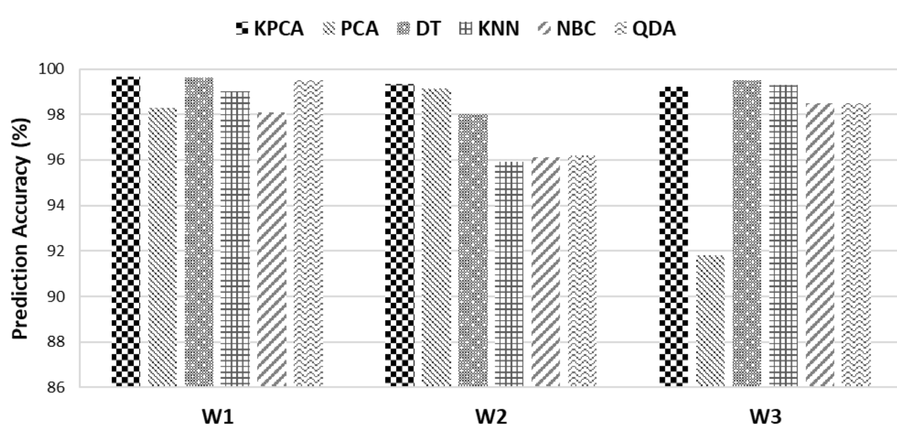


Figure 5.6 Comparison of prediction accuracy for other classification techniques among three wear limits (W1, W2, and W3).

Table 5.3. Comparison of the performance metrics among the three wear limits and three control limits.

Experiment Number	Method	Wear Limit	β = 0.98				β = 0.99				β =0.999			
			ADR	**FDR**	**DD**	**PA**	**ADR**	**FDR**	**DD**	**PA**	**ADR**	**FDR**	**DD**	**PA**
1	KPCA	W1	100.00%	2.14%	0	99.50%	100.00%	1.00%	0	99.76%	100.00%	0.14%	0	99.97%
		W2	100.00%	2.17%	0	99.36%	100.00%	0.91%	0	`99.73%	100.00%	0.00%	0	100.00%
		W3	92.17%	100.00%	0	54.22%	87.71%	100.00%	0	51.60%	73.43%	0.08%	4	84.34%
	PCA	W1	100.00%	2.29%	0	99.46%	100.00%	1.14%	0	99.73%	98.29%	0.14%	0	98.66%
		W2	99.76%	2.17%	0	99.19%	98.00%	1.26%	0	98.22%	91.57%	0.11%	0	94.02%
		W3	81.03%	100.00%	0	47.66%	70.23%	0.98%	0	82.08%	45.43%	0.08%	40	67.87%
2	KPCA	W1	98.86%	2.19%	0	98.41%	97.36%	1.33%	0	97.92%	93.29%	0.10%	2	96.12%
		W2	99.27%	2.45%	0	98.41%	98.45%	0.98%	0	98.73%	90.45%	0.08%	5	95.18%
		W3	100.00%	2.43%	0	98.61%	100.00%	1.21%	0	99.31%	99.71%	0.07%	1	99.84%
	PCA	W1	99.21%	2.00%	6	98.69%	98.64%	1.14%	179	98.73%	93.86%	0.10%	180	96.45%
		W2	86.61%	2.20%	65	92.20%	85.80%	1.14%	148	92.33%	84.65%	0.08%	178	92.29%
		W3	88.57%	2.29%	3	93.80%	85.43%	1.00%	3	93.18%	79.05%	0.14%	37	90.94%
3	KPCA	W1	99.95%	2.29%	0	99.47%	99.90%	1.14%	0	99.67%	99.32%	0.19%	0	99.43%
		W2	99.93%	2.00%	0	99.10%	99.71%	1.14%	0	99.35%	99.29%	0.10%	0	99.55%
		W3	99.90%	2.43%	0	98.57%	99.62%	1.07%	0	99.22%	95.71%	0.07%	0	98.12%
	PCA	W1	99.64%	1.90%	0	99.31%	97.77%	1.14%	0	98.00%	93.66%	0.19%	0	94.98%
		W2	99.71%	2.19%	0	98.90%	99.29%	1.05%	0	99.14%	95.64%	0.10%	1	97.47%
		W3	91.52%	2.29%	1	95.06%	87.62%	1.00%	1	94.12%	79.52%	0.14%	1	91.14%
4	KPCA	W1	98.86%	2.29%	1	98.51%	98.69%	1.14%	1	98.74%	94.61%	0.00%	2	96.23%
		W2	99.05%	2.14%	0	98.57%	98.38%	1.14%	0	98.57%	95.33%	0.00%	0	97.20%
		W3	96.46%	2.51%	0	96.97%	94.74%	1.26%	0	96.74%	85.26%	0.00%	0	92.63%
	PCA	W1	96.73%	1.14%	2	97.37%	93.88%	0.76%	2	95.49%	40.98%	0.19%	17	58.63%
		W2	93.81%	1.71%	0	95.60%	90.86%	1.14%	0	94.06%	21.71%	0.14%	3	52.97%
		W3	92.80%	2.40%	0	95.20%	83.20%	0.80%	0	91.20%	25.26%	0.11%	4	62.57%
5	KPCA	W1	99.58%	2.00%	0	99.16%	99.27%	1.00%	0	99.20%	95.95%	0.00%	0	97.03%
		W2	100.00%	2.06%	0	99.31%	100.00%	0.91%	0	99.70%	98.91%	0.00%	0	99.28%
		W3	93.27%	2.00%	0	95.16%	85.97%	0.95%	0	91.20%	81.78%	0.00%	0	89.07%

Table 5.3 continued

		W1	95.38%	1.57%	1	96.19%	92.62%	1.14%	1	94.29%	42.86%	0.00%	248	58.10%
	PCA	W2	99.94%	4.34%	0	98.51%	95.66%	1.03%	0	96.76%	18.91%	0.00%	360	45.94%
		W3	83.56%	1.33%	9	89.60%	76.95%	1.33%	9	85.64%	9.14%	0.00%	1085	45.49%
6		W1	100.00%	2.57%	0	99.43%	100.00%	1.14%	0	99.75%	99.92%	0.00%	0	99.94%
	KPCA	W2	76.86%	1.90%	2	83.94%	74.95%	1.14%	2	82.92%	69.52%	0.19%	2	79.62%
		W3	99.57%	2.40%	0	98.48%	98.71%	1.03%	2	98.86%	92.43%	0.11%	2	96.57%
		W1	99.43%	2.29%	0	99.05%	98.04%	0.86%	0	98.29%	93.47%	0.00%	0	94.92%
	PCA	W2	67.62%	1.90%	17	77.78%	62.76%	0.95%	17	74.86%	53.71%	0.00%	343	69.14%
		W3	87.86%	1.94%	5	93.52%	82.71%	0.91%	7	91.81%	61.57%	0.23%	9	82.79%
7		W1	81.37%	2.14%	9	87.11%	78.93%	1.21%	36	85.84%	73.07%	0.00%	42	82.43%
	KPCA	W2	88.62%	2.23%	10	92.60%	86.37%	0.97%	20	91.88%	76.75%	0.00%	20	86.86%
		W3	96.00%	100.00%	0	45.91%	90.86%	1.05%	0	95.08%	72.78%	0.00%	1	86.98%
		W1	79.54%	2.43%	10	85.81%	74.55%	1.07%	22	83.03%	57.87%	0.07%	42	72.50%
	PCA	W2	82.37%	2.23%	20	89.07%	76.04%	1.26%	20	85.91%	60.70%	0.11%	21	77.74%
		W3	81.40%	100.00%	1	38.93%	74.86%	1.05%	19	87.43%	55.32%	0.10%	19	78.58%
8		W1	98.11%	2.00%	0	98.08%	97.60%	1.00%	0	98.00%	95.54%	0.00%	2	96.82%
	KPCA	W2	95.11%	2.06%	0	96.12%	93.33%	1.14%	0	95.31%	82.86%	0.11%	4	88.94%
		W3	93.47%	2.29%	0	95.59%	88.98%	0.98%	1	94.00%	67.76%	0.08%	21	83.84%
		W1	91.14%	2.29%	2	93.02%	87.94%	1.14%	2	91.06%	66.91%	0.14%	3	76.33%
	PCA	W2	87.37%	2.17%	8	91.10%	80.19%	0.91%	14	86.94%	18.79%	0.11%	14	47.76%
		W3	83.02%	1.96%	0	90.53%	73.63%	1.14%	12	86.24%	13.22%	0.08%	126	56.57%

5.8 Summary

This paper has presented a KPCA- KDE- and Hotelling's T^2 and Q control charts-based process monitoring method for the purpose of rapidly detecting a worn cutting tool, and thus avoiding catastrophic tool failures, products with unacceptable surface finish, and defective product. The proposed method converts raw multi-sensor data into principal component space, and the KPCA-modified data are used to calculate T^2 and Q values to develop control charts. In this manner, the behavior of the tool wear can be related to the behaviors of the T^2 and Q values on the control chart (i.e., a signal is produced on the charts as an out-of-control value when the wear exceeds the wear limit). Optimal control limits for each case were investigated to enhance monitoring capability in terms of the performance metrics.

The proposed monitoring system was applied to the multi-sensor signals collected during milling operations under various cutting conditions to show how effectively the combination of the KPCA- and KDE-algorithm and control charts is able to monitor tool wear during the process. Also, several KPCA models were used for monitoring of multiple wear limits in the process. The results of the application show that the first abnormality (alarm for high tool wear) was identified with very few or zero delays after the process has entered the abnormal state for all cases. The effectiveness of the proposed method was described with the help of three performance metrics (FDR, ADR, and DD, PA), and compared with the PCA-based method. The result of the tests also demonstrates that the proposed method can effectively integrate multi-sensor information and synthesize the data to estimate the state of the process. In other words, the behavior of the tool wear can be effectively related to the behavior of the T^2 and Q values on the control charts. Future work may investigate application of the proposed method to other manufacturing processes and physical systems (e.g., motor and gear box monitoring).

6. LEARNING VIA ACCELERATION SPECTROGRAMS OF A DC MOTOR SYSTEM WITH APPLICATION TO CONDITION MONITORING[5]

Reprinted with permission from Lee, W. J., Wu, H., Huang, A., and Sutherland, J. W., 2020, "Learning via Acceleration Spectrograms of DC Motor System with Application to Condition Monitoring," *The International Journal of Advanced Manufacturing Technology*.

Abstract

In a highly automated manufacturing plant, the reliability of manufacturing equipment is critical for normal operation. A sudden machine breakdown can bring unexpected downtime, shorter lifespan of equipment, and lower operational efficiency. Breakdowns can lead to defective parts and consume extra energy—issues that are undesirable from an environmental standpoint—and also erode productivity and increase costs. To improve machine tool reliability, a machine may be continuously monitored to track its health condition. Monitoring a machine often provides large amounts of data that must be processed to distill useful information. Electric motors are found in many pieces of common manufacturing equipment. Deep learning methods can be combined with data collected on motors, e.g., acceleration time-frequency data, to identify motor condition. In this paper, three state-of-the-art deep learning architectures are evaluated for their ability to effectively monitor motor condition. Experiments are performed on a lab-scale motor test bed to secure condition data for several common motor faults. Tri-axial acceleration data are collected and converted into 2D images (spectrograms) using the power spectral density function. Some of these experiments are used to tune the deep learning algorithms, and others are used to test the proposed monitoring methods. The relative performances of the architectures are assessed, and it is demonstrated that the use of time-frequency images within a deep learning context can efficiently handle large amounts of data and effectively monitor the motor condition.

Keywords: Deep Learning, Motor Condition Monitoring, Convolutional Neural Network, Motor Test Bed.

---

[5] This work was published in The International Journal of Advanced Manufacturing Technology [160]. The permission (license number: 5024241246953) is obtained from Springer Nature to include the paper in this thesis.

6.1 Introduction

In the twenty-first century, advances in technologies (e.g., artificial intelligence, Internet of Things, big data, and smart sensors) are enabling the emergence of smart manufacturing (Industry 4.0). Smart manufacturing not only seeks to transform a manually operated factory into a highly automated plant, but also enables responses in real time to changing conditions in manufacturing equipment, factory, supply chain network, and customer demand [115]. Smart manufacturing requires an ability to collect data (observation), process the data to secure critical knowledge (evaluation), find meaning in the knowledge (diagnosis), and formulate and implement appropriate manufacturing interventions (decision and implementation). These steps are required for closing the loop on any process control activity [18].

One potential application of smart manufacturing is intelligent machine maintenance, which has as its first step machine monitoring. When monitoring a machine, often proxy measures are collected and analyzed (e.g., acceleration) since direct observation of the state of the bearings, for example, is not generally possible. Information obtained from these proxy measures may then be used to estimate the health of the manufacturing equipment. In a highly automated plant, the health of the manufacturing equipment (e.g., motors) is critical for long equipment life and safety. In a smart factory, maintenance should not only enhance machine safety, improve product quality, and decrease maintenance cost, but also prevent unforeseen equipment downtime. An intelligent maintenance system can also extend the useful life of equipment as shown in Figure 6.1.

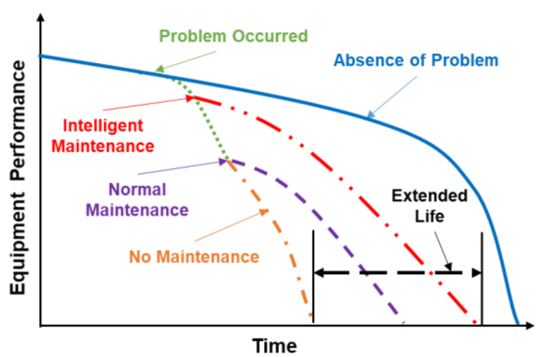


Figure 6.1 Role of maintenance in equipment life extension.

In the field of machine maintenance research, several different methodologies have been used to help with decision making and enhancing system reliability (e.g., avoiding unplanned equipment downtime). Peng et al. [37] classified these efforts into three categories: physical model-based methodologies, knowledge-based methodologies, and data-driven methodologies. Among these three methodologies, data-driven approaches (e.g., machine learning) have gained recent attention owing to rapid development in computing processing power (e.g., GPU processing). A data-driven model, which is based on statistical learning techniques, can handle various types of data and discover hidden connections in large-scale data. Thus, this method may be a useful tool to identify the health condition of manufacturing equipment using sensor signals in real time. This may enable condition-driven maintenance practice, or predictive maintenance (PdM) [116].

For a data-driven methodology, once an initial set of raw sensor signals is obtained, it may be too large to be handled. To reduce the large-scale data without compromising its original character, the raw signals are generally preprocessed first through feature engineering, in which features (i.e., useful information) may be extracted using statistical measures (e.g., mean, RMS, standard deviation, kurtosis, and skewness) in time, frequency, or time-frequency domains [117]. In this stage, features must be carefully extracted because the performance of a data-driven model is largely dependent on the extracted features. Feature engineering may include the selection of some features, and then evaluation of the features to see if they adequately represent the large-scale data. This not only requires expert knowledge of the original dataset to decide which features should be included or excluded, but it is also a laborious process [44]. Thus, it is desirable to develop an automatic feature learning method to analyze sensor signal(s) without the necessity of human intervention [45].

Recently, deep learning methods have been successfully applied in various areas (e.g., computer vision and natural language processing). A convolutional neural network (CNN), a popular deep learning algorithm, is known as a state-of-the-art technique for processing and analyzing large dataset where the input data are often 2D images [38]. In CNN, a network architecture is generally designed to learn internal representations that are abstracted from the input data (e.g., image) by stacking multiple hierarchical layered structures [47]. One benefit of the CNN when processing machinery vibration signals is its ability to learn non-linear representations of the input data (e.g., acceleration) using the hierarchical structure [45]. This approach may make the CNN algorithm a useful tool for machine fault diagnosis since an indicator relating to the machine faults could be non-linearly correlated to the signals and their covariates. In addition, the CNN requires little data preprocessing efforts because the algorithm is able to automatically learn the features from input data during the training phase—this is also called representation learning. This method makes it possible to select features without knowledge of past data and without intensive human efforts.

There are many different ways to construct a CNN architecture; they differ in terms of how the multiple hierarchical layered structures are stacked. Different architectures have been explored with the aim of achieving either higher prediction accuracy and/or computational efficiency. One early CNN architecture, called LeNet, was proposed by LeCun [48] This architecture consisted of eight layers and worked well for handwritten character recognition. To solve more complex image classification problems beyond character recognition, deeper networks (more layers) have been developed. However, with deeper networks, the training of the networks becomes more challenging (i.e., it becomes more difficult to optimize learnable parameters). Moreover, as a network is made deeper and deeper, its accuracy will improve, then plateau, and ultimately degrade [49].

To overcome the accuracy degradation problem, new ideas on CNN architecture have been proposed (e.g., inception module [50] and residual module [51]). Several novel architectures were popularized through image classification–related competitions (e.g., ImageNet Challenge). Some of these architectures are competitive with humans in terms of image classification. Deep learning applications have been mostly concentrated on image classification, speech recognition, and natural language processing. However, few studies are available where deep learning has been applied to machine condition monitoring. These will be described in Section 6.2.

In this paper, the application of CNNs to machine condition monitoring is explored. To experimentally validate the method, a lab-scale test bed was constructed, which enables the introduction of different motor faults. The test bed allows for tri-axial acceleration signals for various conditions to be collected. To test robustness of the model, several trials of the same experiment (containing eight tests) were conducted. Acceleration data are transformed into time-frequency images (spectrograms) using the power spectral density function. During the training phase, these images are used to tune the CNNs. To compare the performance of different CNN architectures, one simple architecture and three state-of-art architectures (GoogLeNet, AlexNet, and ResNet50), which are known to be the most powerful for image classification, are used. After training the models, the performances of the CNN architectures are tested/compared using spectrograms from different experimental trials. These performances are also compared with the results from several classical machine learning methods.

6.2 Machine/deep Learning Applications

In classical machine learning (ML) methods, features are extracted and selected from large-scale initial datasets first through feature engineering, and then used for training and testing a ML algorithm. Yu et al. [118] proposed a hidden Markov model (HMM) for machine health monitoring using features extracted from dynamic principal component analysis (PCA).Wu et al. [52] monitored the condition of an additive manufacturing process using an acoustic emission signal. They used PCA to reduce the amount of data needed to train the HMM. Pezzani et al. [53] proposed a support vector machine (SVM) to monitor the condition of a rotor bar in an induction motor. They extracted features from the motor current signal using statistical measures. Bhat et al. [54] also used an SVM to classify the condition of a cutting tool using images of the machined surface. They extracted and selected features through a gray-level co-occurrence matrix and Fisher discriminant analysis, respectively. Kane et al. [55] used statistical measures as an input data to an artificial neural network (ANN) for fault detection in a gearbox.

Among several data-driven models applied for machine condition monitoring, an ANN is one of the most attractive models due to its ability to manage large-scale data and its ease of deployment [37]. However, an ANN often requires data preprocessing (i.e., feature engineering), and how this preprocessing is done will affect ANN performance. Therefore, a method incorporating automatic feature learning (e.g., CNN) may be desired for processing large-scale data. Figure 2 shows the comparison of classical ML and deep learning (DL) methods.

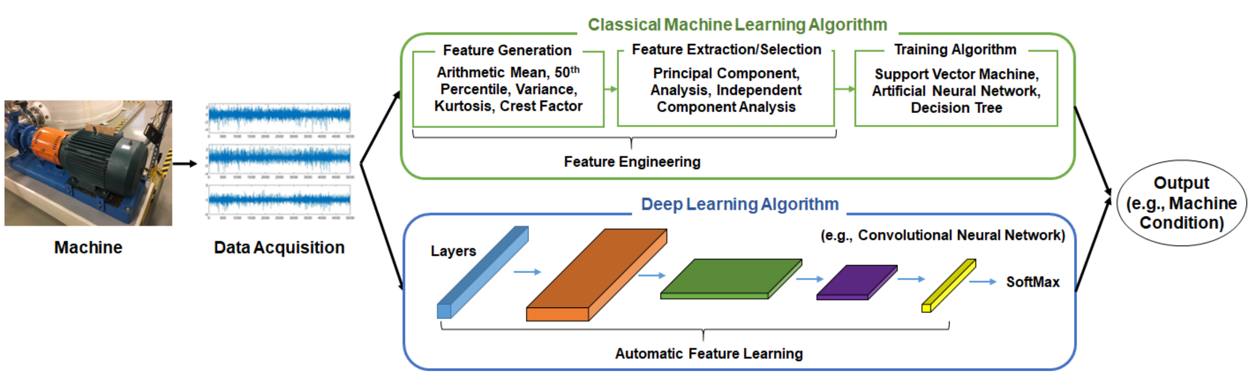


Figure 6.2 Comparison of classical machine learning and deep learning.

One specific type of ANN is the CNN, which allows features to be automatically learned during the training phase. Several efforts have been undertaken to build a machine condition monitoring system using CNN, and the recognition power of CNN has been actively researched and compared with classical ML algorithms (e.g., support vector machine, random forest, and ANN) [12, 23, 24]. Ince et al. [58] proposed a shallow CNN architecture for detecting a motor fault, and the method predicted the fault with an accuracy of 97.4%. In the study, the output class was limited to two conditions: healthy and not healthy (a fault has occurred). Jing et al. [56] and Chen et al. [57] introduced various gear faults in a gearbox test bed to collect the acceleration signal under different health conditions. In both studies, several different CNN architectures were applied to classify the health condition, and the classification accuracies were compared. Eren et al. [60] studied bearing fault diagnosis using the Case Western Reserve University Bearing Datasets. The study showed the effectiveness of the CNN method without feature extraction or selection processes. Extensive review of and reference to ML applications in machine condition monitoring research may be found in Peng et al. [37].

As is evident, ML methods have been extensively applied in machine condition monitoring research. DL applications are presently a very active area for research; however, there are limited instances where DL has been applied to machine condition monitoring. Therefore, a goal of this paper is to apply DL to condition monitoring. Other key contributions are as follows:

- A lab-scale test bed, in which single-fault and combined faults can be configured, is designed and constructed to consider both single- and multiple-fault scenarios.
- Instead of 1D CNN, which has been widely applied for the DL applications in the condition monitoring study, 2D CNN is used to classify a motor condition (most of the state-of-art architectures are based on 2D CNN).
- Relatively shallow networks have been generally used for machine condition monitoring. In this study, both shallow and deep networks are tested and compared.
- Four experimental trials are conducted at different times to collect training and testing datasets. Three different training types are employed to train CNNs, and each model is evaluated using a dataset obtained from different experimental trials. Also, the performance of CNNs is compared with classical ML methods.

We believe that this is the first machine condition monitoring study that considers both a single fault and combined multiple faults using state-of-art 2D CNN architectures.

6.3 A Proposed Methodology for Motor Condition Monitoring

6.3.1 Time-frequency Analysis: Power Spectral Density

In a time-frequency analysis, a signal is shown simultaneously in both the time and frequency domains. To visualize the time-frequency analysis as a 2D graphical image, a spectrogram is often used. A spectrogram describes the strength (e.g., amplitude or power) of a certain waveform over time at various frequencies existing in the waveform. Sometimes, to compute the frequency domain representation of a given signal, the fast Fourier transform (FFT) is used. The FFT is an efficient algorithm to compute the discrete Fourier transform (DFT) of a waveform. The FFT is well-suited to situations where a limited number of "dominant" frequency components exist in the signal being analyzed [119].

For cases where there are many important frequency components in the signal being analyzed, the power spectral density (PSD) is often used. Such may be the case when monitoring motor vibrations. A motor has numerous rolling elements (e.g., gears, bearings, and rotors) moving during operation. The collective effects of numerous factors from inside and outside of a motor may result in "noise" added to the system. With the PSD, the corrupted vibration signal can be characterized by computing the power contents of the signal over the frequency domain (i.e., power is the square of the FFT's magnitude in frequency domain). A nonparametric estimate of PSD, $P(f)$, with window function $h[n]$ can be calculated as

$$P(f) = \frac{\Delta t}{N}\left|\sum_{n=0}^{N-1} h[n]x[n]e^{-i2\pi fn}\right|^2, \; -\frac{1}{2\Delta t} < f \leq \frac{1}{2\Delta t} \tag{6.1}$$

where Δt is the sampling interval.

In order to calculate the time-dependent power spectrum of a given signal, the following steps should be conducted: (1) divide the signal into overlapping consecutive segments, (2) apply a selected window function to each segment, and then (3) compute the short-time Fourier transform (unit: g) and convert into power terms (unit: g^2/Hz) using Eq. (6.1). Then, the transformed consecutive segments (i.e., multiple vectors) are combined to form a matrix by indexing the row and the column with frequency and time, respectively. Thus, each element has a time-frequency representation. To transform the PSD matrix into a 2D visible image (spectrogram), each element in the matrix is represented by a color scale, which corresponds to the magnitude of a power at a certain time-frequency. Once 2D spectrograms are generated, they can be used in a 2D CNN for the image classification tasks.

6.3.2 Convolutional Neural Network

A CNN, inspired by a visual system's structure, becomes a favorite type of neural network for image processing tasks in the computer vision community. Like an ANN, CNN is a sequence of layers containing learnable weight and bias, and each layer receives some inputs from a previous layer and performs a given task to transfer new information to a next layer. Each layer has a different task, and the task is performed automatically within the algorithm.

Four main layers used in a CNN are an input layer, a convolution layer, a pooling layer, and a fully connected layer. As shown in Figure 6.3, a typical deep learning architecture can be constructed by alternating and stacking multiple convolutional layers and pooling layers between input layer and fully connected layer. To explain the role of each layer, consider processing a 2D color image with the size of $N_h^{[0]} \times N_w^{[0]}$ as follows.

An input layer is the first layer of a CNN and takes the raw pixel values of the image containing three color channels (R, G, B). The information of the input layer with the size of $N_h^{[0]} \times N_w^{[0]} \times 3$ is passed to the next layer for feature extraction process.

A convolution layer (Conv), assuming *l*th convolution layer where *l*=[0, n], is connected with each unit in a previous layer through a filter bank (assume the size of the filter bank used for *l*th convolution layer is $f_{con}^{[l]} \times f_{con}^{[l]} \times N_{f,con}^{[l]}$ where $f_{con}^{[l]}$ is a filter size used for *l*th convolution layer and $N_{f,con}^{[l]}$ is the number of filters in the filter bank). At this layer, the filters containing learnable weights convolve with respective small regions in the previous layer, which are called input feature maps. Here, the size and the number of the regions are decided by filter size ($f^{[l]}$), stride ($s^{[l]}$), and padding ($p^{[l]}$). Then, the weighted sum passes through an activation function to generate output feature maps with the size of $N_{h,con}^{[l]} \times N_{w,con}^{[l]} \times N_{f,con}^{[l]}$, where $N_{h/w,con}^{[l]} = (N_{h/w,con}^{[l-1]} + 2p_{con}^{[l]} - f_{con}^{[l]}) / s_{con}^{[l]} + 1$. Several nonlinear activation functions have been widely used in deep learning algorithms (e.g., sigmoid, tanh, and rectified linear unit (ReLU)), and ReLU has been proven to be the most efficient in many deep learning studies recently [120]. ReLU is a half-wave rectifier and switches all negative values in the layer to 0 (i.e., ReLU(x)= max (0, x)). Using the ReLU function, each output map can be written as:

$$\mathrm{X}_j^{[l]} = ReLU\left(\sum_{i \in M_j} W_{\mathrm{ij}}^{[l]} * X_i^{[l-1]} + b_j^{[l]} \right) \tag{6.2}$$

where l represents a lth convolution layer, M_j is a selection of input feature maps, W is a filter containing learnable weigh with, * is a convolution operation, and b is a bias weigh matrix.

A pooling layer (Pool), also called subsampling, is normally placed after a convolution layer, and it performs a down-sampling task: reducing the dimension of the input feature maps using a filter with the size of $f_{pool}^{[l]} \times f_{pool}^{[l]} \times N_{f,pool}^{[l]}$, where $f_{pool}^{[l]}$ is a filter size used for lth pooling layer, and $N_{f,pool}^{[l]}$ is the number of filter in the filter bank [121]. The down-sampling may lead to a loss of information, but it enables less computational time and avoids over-fitting (i.e., if a convolution layer performs feature construction, a pooling layer conducts feature extraction or selection process) [45]. Empirically, the maximum pooling method significantly outperforms other pooling methods, and it is commonly used as the down-sampling method in CNNs [122]. The max pooling operation is done by simply applying a max filter to non-overlapping regions in the input feature map, so there is no learnable weight in the filters. After passing through the filters, the output feature maps with the size of $N_{h,pool}^{[l]} \times N_{w,pool}^{[l]} \times N_{f,pool}^{[l]}$, where $N_{h/w,pool}^{[l]} = (N_{h/w,pool}^{[l-1]} + 2p_{pool}^{[l]} - f_{pool}^{[l]}) / s_{pool}^{[l]} + 1$ are generated.

A fully connected layer (FC) (e.g., multilayer perceptron (MLP)) is placed after multiple stacks of convolution layers and pooling layers. The FC takes the final output feature maps as a form of a feature vector (v_f) and performs a classification task. MLP is a feedforward neural network, and output of the network can be written as (assuming single layer):

$$c = f(w_c v_f + b_c) \tag{6.3}$$

where w_c is a weight vector, b_c is a bias vector, and f is a nonlinear activation function. Once an output vector from the FC is computed, a SoftMax classifier will identify the best possible class (e.g., motor condition) among the possible target classes (i.e., a SoftMax only can be used for the problem whose target outputs are discrete values). A SoftMax transforms the output vector into a probability distribution over predicted output classes, so each class can be interpreted by probabilities.

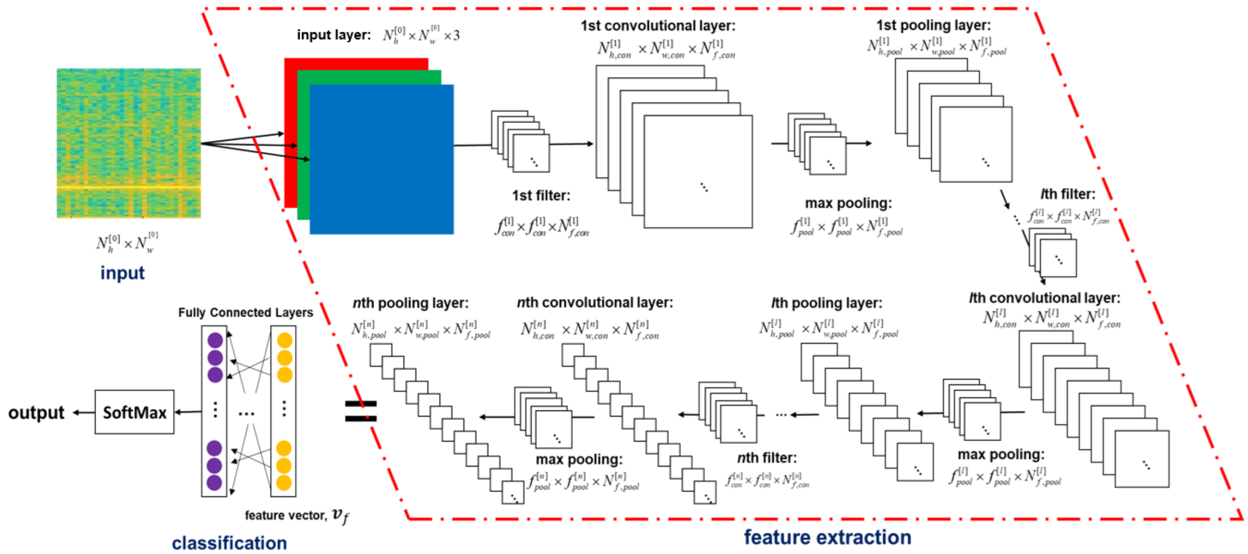


Figure 6.3 Typical CNN architecture.

Once a deep learning architecture has been constructed, a backpropagation algorithm is normally implemented using a training dataset as described in [123]. During the training phase, the learnable weights ($W_{ij}^{[l]}$, w_c) and bias ($b_j^{[l]}$, b_c) are updated, and the training will continue until satisfying specified termination conditions (e.g., a specified number of iterations).

To solve more complex image classification problems, several attempts have been undertaken to build a deeper network (more layers) to increase the performance of the model. Therefore, in addition to a simple CNN architecture ([Input]-[Conv]-[MaxPool]-[FC]-[SoftMax]-[Output]), three powerful CNN architectures, which are GoogLeNet, Alex Net, and RestNet50, are trained using acceleration time-frequency images. For a better understanding, the highlights of each network are explained as follows.

GoogLeNet, consisting of 22 layers, is the first network to introduce the concept of the "inception modules" that can (1) perform different sizes of convolutions using parallel filters on the input feature maps and (2) concatenate all filter outputs together for the next layer [50]. The graphical view of the inception module is shown in Figure 6.4. The parallel filters in the building block (inception modules) show a significant improvement in computation and memory efficiency. On the other hand, Alex Net has a relatively simple architecture with the eight layers containing convolutional layers, max pooling layers, and fully connected layers [124]. Alex Net is the first network to attempt to use the ReLU activation function, which showed more efficient training performances than conventional activation functions (e.g., sigmoid and *Tanh*). ResNet50, also called residual neural network, is the most recently proposed architecture among three networks and has a relatively deeper network compared to GooLeNet and AlexNet [51]. A main idea of ResNet50 is to solve an accuracy degradation problem in a deep network using shortcut connections (skipping one or more layers), as well as residual learning. Within the network, there are multiple stacks of similar building blocks (50 layers), which are called residual modules. The graphical view of the residual module is shown in Figure 6.5. More details regarding the network architectures can be found in the cited references.

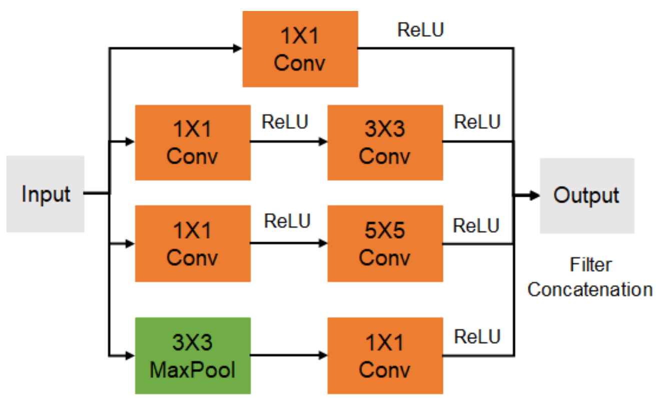


Figure 6.4 Inception module

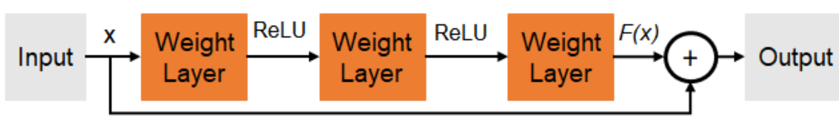


Figure 6.5 Residual module.

In the next section, several motor faults, which will be further introduced in the motor test bed during the experiment, are described.

6.4 Motor Fault Scenarios

Acceleration signals for different condition settings were collected: idle (off), normal, single fault (gear defect, misalignment, and looseness), and a combination of two fault settings were introduced in the test bed. A brief overview of each fault is described as follows.

6.4.1 Gear Defects

A gear is one of the most common mechanical components in machinery equipment. Its role is to transmit the mechanical power among machine components. A normal gear generally produces low-frequency vibrations, but lots of high-frequency vibrations can be generated when a gear has a defect [125]. Assuming that there is a defect such as a broken gear tooth, it not only generates higher amplitudes of vibration but also induces high-frequency vibrations once per rotation. The generated high-frequency oscillation repeatedly impacts a machine during the operation [126]. Ultimately, the gear defect can reduce the lifespan of machine components, decrease a machine’s efficiency, and cause unplanned downtime.

6.4.2 Shaft Misalignment

Accurate and precise shaft alignment is vital for reliable power transmission from a motor to driven equipment. Misalignment occurs when shaft centerlines of two connected machines (e.g., motor and generator) are not aligned with each other (i.e., the centerline is offset). The misalignment can be caused by poor maintenance tasks, improper installation, insufficient bolt tightening, shaft failure, etc. Along with gear defects, misalignment is also one of the main causes of high vibrations in a motor. Unlike gear defects, misalignment is not an observable physical failure, but the vibration caused by an incorrect alignment can result in rapid wear on the mechanical components (e.g., bearing gear, shaft, and rotor) and temperature increase; thus, ultimately leading to a premature machine failure [127] (i.e., misalignment is a root cause of many mechanical failures in motors).

6.4.3 Mechanical Looseness

Mechanical looseness is a common problem in rotating machinery, commonly caused by mechanical damages, loose bolts, excessive clearance between components, incorrect fit between components, etc. [128]. Mechanical looseness slowly grows in a rotating element as the mechanical components get worn down. Therefore, a maintenance engineer should tighten the equipment and replace supplementary components before they become severely worn down. Like shaft misalignment, mechanical looseness also induces the high vibrations in rotating machinery and can lead to a premature machine failure through the rapid wear [125].

To collect the acceleration signals for different motor condition settings, a lab-scale motor test bed was set up. Details on the experimental setup and data acquisition method are explained in the next section.

6.5 Experiment Setup and Conduct of the Experiment

A motor has many different states during an operation. A state can be changed from normal to abnormal as a result of internal (e.g., gear defect) and external (e.g., looseness and misalignment) faults. Also, in some cases, a combination of different faults occurs. When a motor condition turns into an undesirable state resulting from existing fault(s), tangible evidence of its occurrence may be created. This evidence may be acquired by a sensor (e.g., accelerometer), and then, the associated fault may be identified by processing the signals.

To collect and store tri-axial acceleration signals for different condition settings, a lab-scale motor test bed, in which a single fault or combined faults can be configured, was constructed as shown in Figure 6.6. In the test bed, two 12-V DC motors (Actobotics, no. 638350) were used, each equipped with a 100:1 ratio gearbox. One motor (power motor) was driven by a 12-V DC power supply (Eventek, KPS305D), and the other one (load motor) served as the load (e.g., motor-generator setting). The output shafts of the two motors were connected by a stainless steel precision shaft (8 mm (d) × 200 mm (l)) through the couplers (4 to 8 mm), and each motor was tightly fixed by a bore bottom tapped clamping mount (25-mm bore). Each clamping mount was placed on an aluminum channel (76.2 mm (l) × 33.53 mm (w) × 38.1 mm (h)) and fixed in place by screws. The two channels supporting the clapping mounts were placed on the Aluminum 6061 solid flat plate (Stoner Metals, AB2383) with dimensions of 355.6 mm (l) × 152.4 mm (w) × 12.7 mm (h). To rivet the channels on the plate, the plate was machined in the CNC machining center; drill the holes at the precise locations to accept the connectors so that the centerlines of the two motors' output shafts are designed to be parallel-aligned. The accelerometer was attached to the mounting clamp, which made direct contact with the motor housing as shown in Figure 6.6; motor housing was not flat enough to attach the sensor using an adhesive.

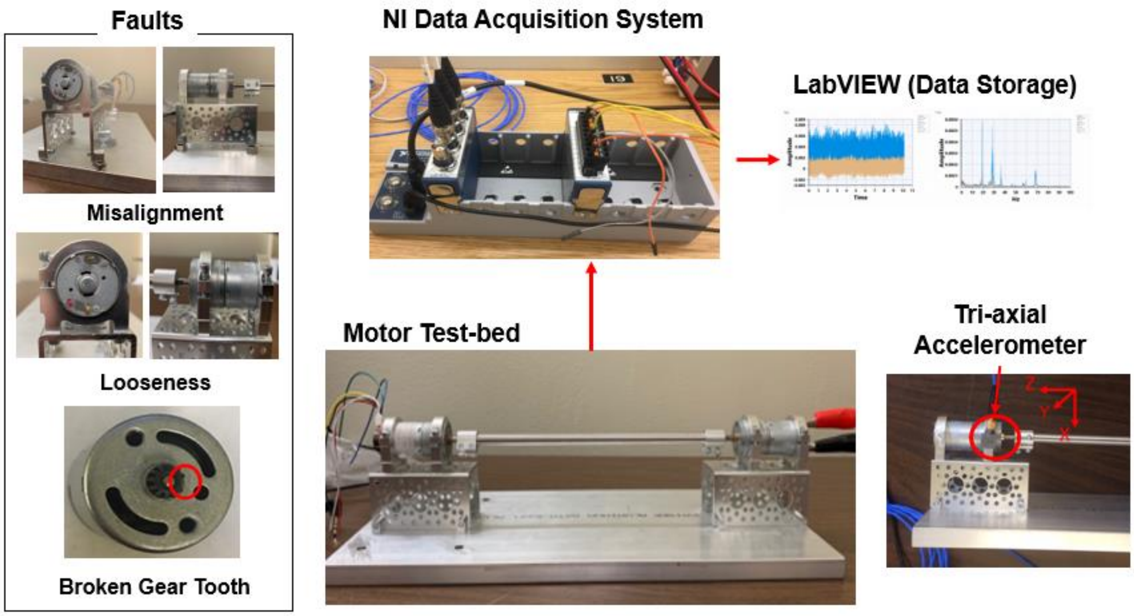


Figure 6.6 Lab-scale motor test bed.

Table 6.1 Experiment design.

Test Number	Experiment Trial 1	Experiment Trial 2	Experiment Trial 3	Experiment Trial 4
Test 1	Idle	Looseness	Broken Gear Tooth and Looseness	Broken Gear Tooth and Misalignment
Test 2	Normal	Broken Gear Tooth and Misalignment	Broken Gear Tooth and Misalignment	Normal
Test 3	Broken Gear Tooth	Looseness and Misalignment	Looseness	Broken Gear Tooth
Test 4	Looseness	Normal	Misalignment	Looseness and Misalignment
Test 5	Misalignment	Misalignment	Idle	Misalignment
Test 6	Looseness and Misalignment	Broken Gear Tooth and Looseness	Looseness and Misalignment	Idle
Test 7	Broken Gear Tooth and Looseness	Idle	Normal	Broken Gear Tooth and Looseness
Test 8	Broken Gear Tooth and Misalignment	Broken Gear Tooth	Broken Gear Tooth	Looseness

As shown in Table 6.1 and Figure 6.6a single fault (i.e., either broken gear tooth, misalignment, or looseness) and two combined faults (i.e., either looseness and misalignment, broken gear tooth and looseness, or broken gear tooth and misalignment) were planned; three combined faults were excluded because the motor did not work properly in that setting. The gear, which is directly connected to the motor shaft, was physically damaged to break the gear tooth.

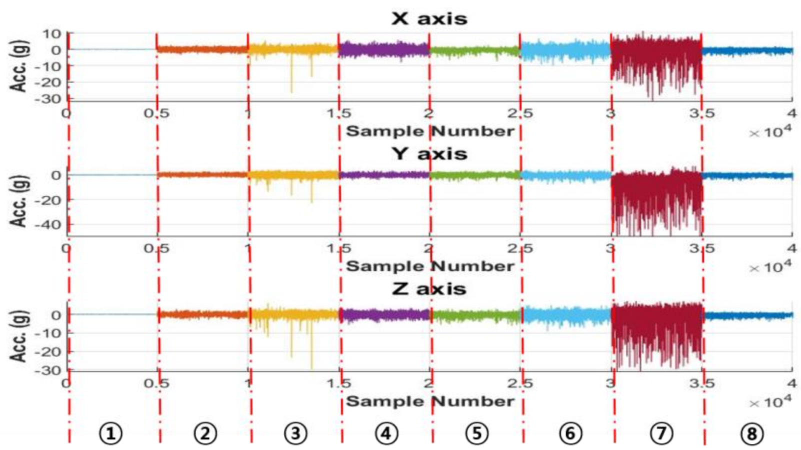
The centerlines of the two motor shafts were misaligned by adding the structural components, M6 hex nuts, between the channel and the plate on the left side of the channel. This results in the shift of the power motor centerline to 8◦ in the clockwise direction relative to the right side of the channel. The screws tightening the power motor were loosened by five screw turns to make some clearance between the components. In addition to the single and combined fault settings, idle (motor off) and normal (without any faults) condition settings were also included, which represent the motor non-working and normal working conditions, respectively.

The data acquisition was enabled by the tri-axial accelerometer (PCB PIEZOTRONICS, J356A45) and the national instrument compact data acquisition system (NI cDAQ), including cDAQ-9178 CompactDAQ chassis and NI-9234 C Series Sound and Vibration Input Module. When initiating data collection under a steady-state operating condition, the machinery vibration was acquired by the accelerometer (transducer) in a form of electrical signal, and the electrical signal was converted into the acceleration (unit: g) using the sensitivity (mV/g). The observed signals were visualized and stored in a desktop using the LabVIEW software. The sampling frequency for all three axis channels was set to $Fs = 5\ kHz$, and digital data were sampled 100 times every 5 s using time delay function in the LabVIEW. Here, one sampling trial includes 15,000 data, so sampling 100 times equal to 1,500,000 points (one test). Once the data acquisition setting is complete, different condition settings were introduced in the test bed to collect the respective tri-axial acceleration signals.

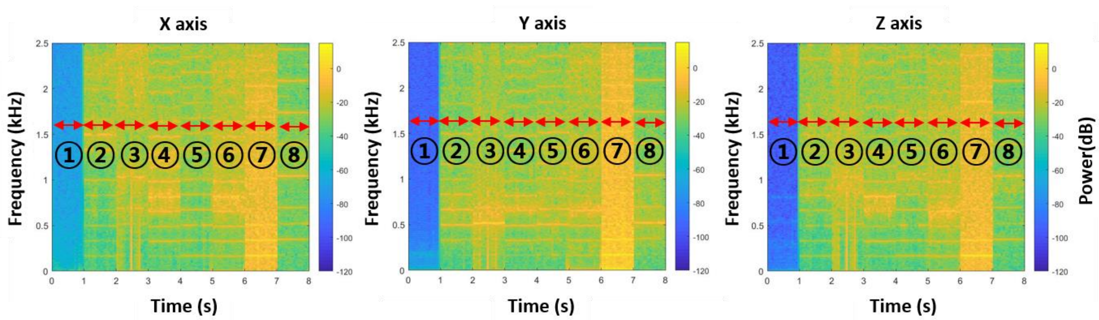
Four experimental trials were performed as shown in Table 6.1; the test sequence in each experimental trial was randomly changed. In each experimental trial, eight tests were conducted and acceleration signals under the respective settings were collected and stored as explained before. Each condition setting was introduced in the test bed one by one. After obtaining all signals from four experimental trials, the signals were transformed into the time-frequency images (i.e., spectrograms) to be used in the 2D CNNs.

6.6 Spectrogram Preparation

After tri-axial acceleration, signals for different condition settings were collected and stored; the stored data were transformed into spectrograms to be used for the training and testing datasets. In Figure 6.7, an overview of one sampling trial from each condition setting is shown in (a) time domain and (b) time-frequency domain. One sampling trial corresponds to 5000 data points collected during 1 s for each axis. Here, one sampling trial is randomly chosen from experimental trial 1, and the numbers from ① to ⑧ shown in the figures correspond to the condition settings from test 1 to test 8 in experiment trial 1. For the transformation from the time to the time-frequency domain, as described in Section 6.3.1, the power spectral density was used to calculate the time-dependent power spectrum of the collocated signals. The frequency band limit was set to [0, Fs/2] (whole Nyquist range), and the Kaiser window function was selected, which converges to 0 at the beginning and end points.



(a)



(b)

Figure 6.7 Overview of the tri-axial acceleration signals for eight condition settings; (a) time-domain and (b) time-frequency domain.

As shown in Figure 6.7, certain cases show recognizable differences in the time domain and the time-frequency domain among the conditions (e.g., ①, ③, and ⑦), but other cases do not. The most significant axis shifted as the motor condition changed in terms of RMS features in the time-domain (e.g., except for the idle condition, x- and y-directions show the highest values, respectively, in ②, ④, and ⑤ and in ③, ⑤, ⑦, and ⑧). In Figure 6.7a, the data are observed in both positive and negative directions; thus, it indicates that both positive and negative forces were acted on the acceleration sensor. But, for some cases (e.g., ⑦), higher accelerations were observed in the negative direction. This is possibly due to the fault settings in the test bed (e.g., structural components were added between the channel and the plate on the "left side" only).

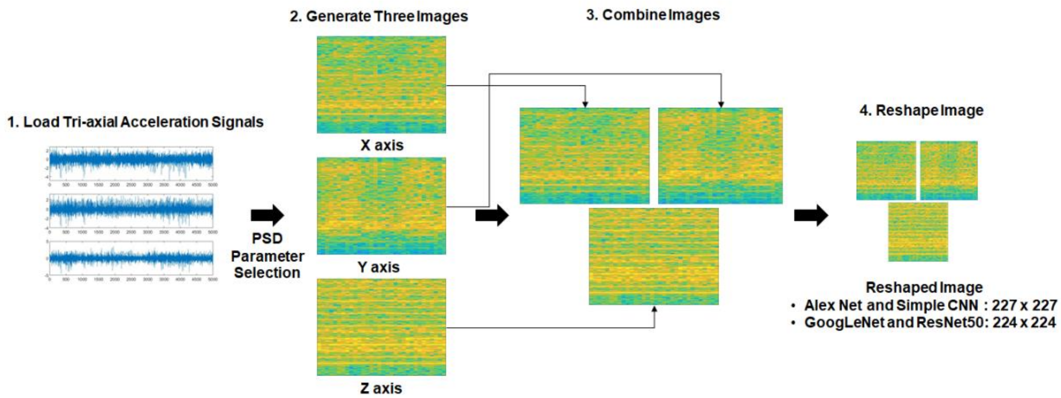


Figure 6.8 Generating one combined image using one collection (i.e., 5000 × 3 acceleration singles).

Figure 6.8 describes the process of combining the three axis information into one image. In the process, only color-scaled images were saved. Redundant information such as tick and tick label were removed. Also, the size of the images was reshaped using a bilinear interpolation scaling method to be compatible with the input layers of the CNNs being used. The scaling method generates new pixels using interpolation when scaling up an image while weighted averages of pixel values in the nearest neighborhood are used for output when shrinking an image.

In the next section, four different CNN architectures were trained and tested to show how effectively the method can extract features automatically from time-frequency images and how well the method recognizes a pattern to classify a motor condition.

6.7 Training and Use of the CNNs

To train the models, transfer learning was performed using the fine-tune pre-trained CNN architectures (GoogLeNet, Alex Net, and ResNet50), which are available as open sources. Transfer learning is a method of learning a new task through a transfer of knowledge that has already been learnt [129], i.e., transfer learning is often used for a new collection of data in deep learning application. Usually, transfer learning for a fine-tune pre-trained network is much faster than training the network with randomly initialized learnable weight. This is because a pre-trained network has already learned features for a wide range of images; a pre-trained network was already trained by over a million images. Therefore, transfer learning was performed to learn a new task (motor condition classification) using the spectrograms generated in Section 6.6.

Training was performed on a PC platform (Precision 5820 Tower) equipped with Intel Xeon with 32 GB RAM and GeForce GTX 1080 TI with 11GB GDDR5X and was implemented in MATLAB. Training parameters for all CNN architectures were set to the same (iterative optimization algorithm=stochastic gradient decent, learning rate=0.01, and the maximum number of epochs=30). To evaluate performance of the CNN, the training and testing dataset are divided as shown in Table 6.2; each testing dataset was obtained from different experimental trials with the corresponding training dataset. Also, three training types were employed to investigate how CNN performance changes as the number of training dataset increases.

Table 6.2 Training and testing dataset.

Test Number	Details
Test 1	• Training Dataset: Experiment Trial 1 • Testing Dataset: (a) Experiment Trial 2, (b) Experiment Trial 3, and (c) Experiment Trial 4
Test 2	• Training Dataset: Experiment Trials 1 and 2 • Testing Dataset: (b) Experiment Trial 3 and (c) Experiment Trial 4
Test 3	• Training Dataset: Experiment Trials 1, 2, and 3 • Testing Dataset: (c) Experiment Trial 4

The four CNN architectures were trained through transfer learning with the help of GPU, and prediction accuracies of each case are summarized in Table 6.3. Before explaining the overall performance, one case from each network architecture was selected to investigate the prediction accuracies.

Four confusion matrices for GoogLeNet, Alex Net, Renet50, and Simple CNN, which were trained with the training type I and tested with the data obtained from (c) experimental trial 4 (in Table 6.2), are shown in Figure 6.9. Confusion matrix is a popular tool to visualize the performance of a classification algorithm. In the matrix, each row and column display predicted class and true class (each class is numbered the same as the numbering in Figure 6.7. The diagonal and off-diagonal cells in the boxes with the green horizontal stripes and orange vertical stripes indicate the number of correctly classified observations and incorrectly classified observations, respectively. The column on the far right (white boxes) shows the positive predictive value (PPV) and false discovery rate (FDR). The row at the bottom (white boxes) presents the true positive rate (TPR) and false negative rate (FNR). PPV, FDR, TPR, and FNR can be calculated using the Eqs (6.4)-(6.7). The values in the green box are the prediction accuracy (i.e., the global percentage of true positive) and the prediction error. By displaying the accuracies and errors together in the plot, the confusion matrix can show where the classification method is confused when a prediction is performed.

$$\mathrm{PPV}(\%) = \frac{\text{true positive}}{\text{true positive + false postive}} \times 100, \tag{6.4}$$

$$\mathrm{FDR}(\%) = 1 - \mathrm{PPV}, \tag{6.5}$$

$$\mathrm{TPR}(\%) = \frac{\text{true positive}}{\text{true positive + false negative}} \times 100, \tag{6.6}$$

$$\mathrm{FNR}(\%) = 1 - \mathrm{TPR}. \tag{6.7}$$

Table 6.3 Prediction accuracies using four CNN architectures.

CNN Architecture	**Training Type**	**(a)**	**(b)**	**(c)**
GoogLeNet	I	88.5%	88.8%	96.8%
	II		93.8%	98.3%
	III			99.9%
Alex Net	I	88.3%	87.6%	95.1%
	II		99.6%	99.8%
	III			99.9%
ResNet50	I	89.9%	90.1%	94.4%
	II		99.4%	99.0%
	III			100.0%
Simple CNN	I	72.4%	76.3%	81.1%
	II		98.4%	100.0%
	III			98.1%

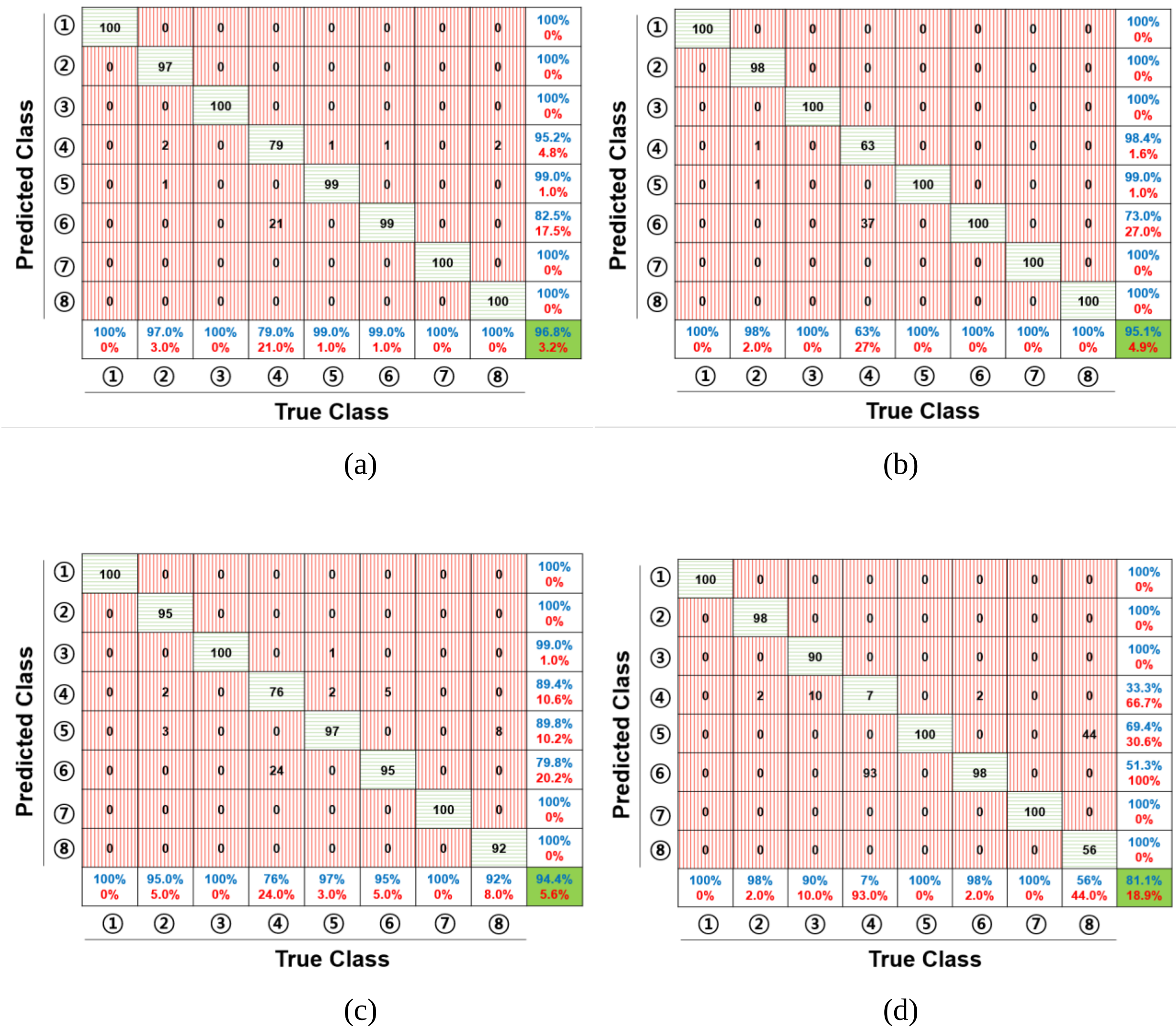


Figure 6.9 Confusion matrices for (a) GooLeNet, (b) Alex Net, (c) ResNet50, and (d) simple CNN.

The prediction accuracies for GoogLeNet, Alex Net, ResNet50, and simple CNN are 96.8%, 95.1%, 94.4%, and 81.1%, respectively, as shown in the green box in Figure 6.9. When the minimum number of image was used for training, three state-of-art CNN architectures outperform the simple CNN architecture for the classification of the acceleration time-frequency images. GoogLeNet, Alex Net, and ResNet50 predicted the most class correctly with some confusions mainly between ④ (Looseness) and ⑥ (Looseness and Misalignment). Simple CNN shows the significant confusions (1) between ④ (Misalignment) and ⑥ (Looseness + Misalignment) and (2) between ⑤ (Misalignment) and ⑧ (Broken Gear Tooth and Looseness).

Next, average prediction accuracies (i.e., the average of global percentage of true positive) of CNNs are investigated to show how the number of training data affects the performance. As clearly shown in Table 6.3, as the number of training data increases, i.e., from training type I to training type III, the accuracy shows an increasing trend. In Figure 6.10, the average accuracies computed using different training types are plotted. The average accuracies of GoogLeNet, Alex Net, and Resnet50 rose steadily as the number of training data increases, but simple CNN showed the slight drop in training type III. Also, it was found that the average accuracy of the simple CNN was far below that of GoogLeNet, Alex Net, and ResNet50 in training type I. For the computation time, as shown in Figure 6.10, the training time of Resnet50, which has a relatively deeper network, took the longest time, while the training time of Simple CNN, which has a relatively shallow network, took the shortest time.

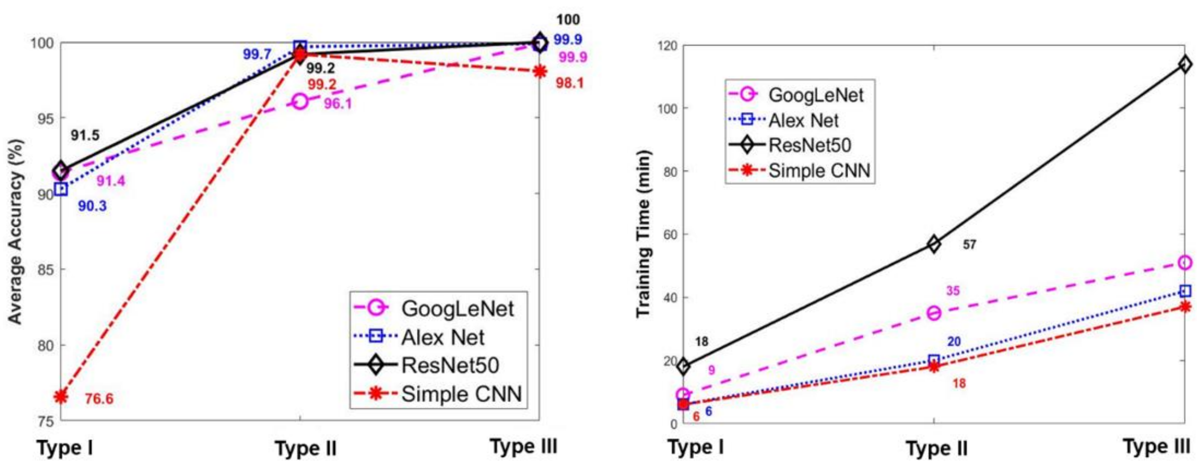


Figure 6.10 Average prediction accuracies (left) and training time (right) of four CNN architectures using different training types.

In this section, the prediction accuracy of each CNN was evaluated using the testing dataset, which were obtained from different experimental trials with the training dataset, i.e., training and testing data were collected at different time, i.e., adequacy of the trained CNN models is tested. Accordingly, the evaluation of the CNN architectures with the testing dataset may explain not only the robustness of a CNN model being used but also how well the key features can be extracted from the acceleration time-frequency images under different environmental noise levels.

6.8 Comparing the CNNs with Classical ML Algorithms

In this section, the same datasets used in the previous section were applied to several classical ML algorithms: (1) artificial neural network (ANN), (2) cubic support vector machine (SVM), (3) decision tree (DT), (4) *k*-nearest neighbors (KNN), and (5) quadratic discriminant analysis (QDA). As illustrated in Section 6.2, a classical ML algorithm commonly incorporates a featuring engineering process to extract and select useful information from a large-scale dataset. Therefore, before training the models, a featuring engineering was performed as follows: (1) features were extracted using statistical measures (RMS, mean, kurtosis, standard deviation, and skewness) from the tri-axial dataset, i.e., 15 features, and (2) features were selected using PCA with the explained variance of 90%.

Table 6.4 Prediction accuracies using classical ML algorithms.

CNN Architecture	Training Type	(a)	(b)	(c)
ANN	I	64.5%	67.9%	54.3%
	II		90.3%	86.6%
	III			74.0%
SVM	I	79.1%	66.3%	66.3%
	II		91.7%	78.8%
	III			82.1%
Decision Tree	I	76.6%	72.3%	57.4%
	II		79.5%	63.6%
	III			77.3%
KNN	I	79.1%	74.1%	60.5%
	II		87.8%	75.4%
	III			82.5%
QDA	I	71.3%	71.4%	60.1%
	II		81.4%	74.3%
	III			76.3%

The selected features were applied to the classical pattern recognition algorithms using three training types defined in Table 6.2. The algorithms were trained and tested in the same way as conducted in the previous section, and the prediction accuracies are summarized in Table 6.4. To compare the classification performance between the classical pattern recognition algorithms and the four CNN architectures, the average prediction accuracies were plotted in Figure 6.11. As shown in the Figure 6.11, four CNNs outperform the classical pattern recognition algorithms in all training types. In terms of global accuracy (average of the accuracy values from types I, II, and III), ResNet50 showed the highest prediction accuracy followed by Alex Net, GoogLeNet, simple CNN, SVM, KNN, ANN, QDA, and DT.

To sum up, the recognition power of CNN over classical pattern recognition algorithms was experimentally validated. From the experimental results, the application of the time-frequency images to CNN can effectively classify the motor conditions.

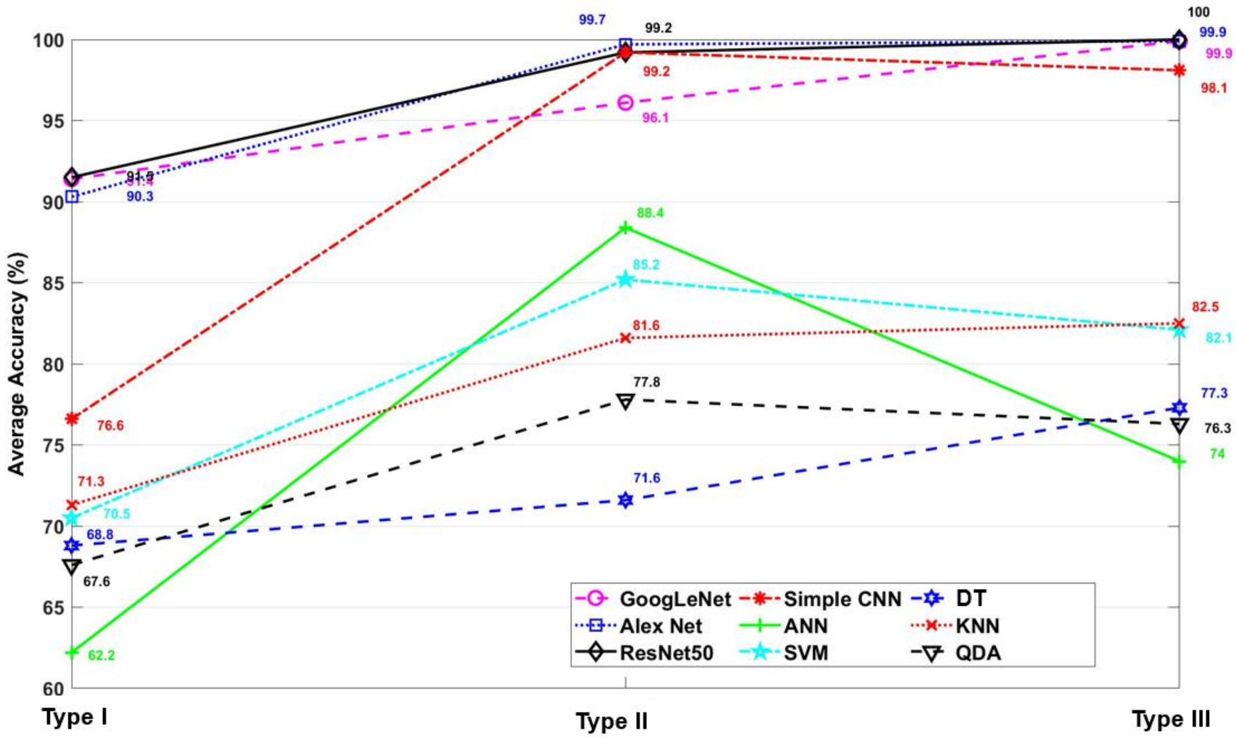


Figure 6.11 Average prediction accuracies of the classical ML algorithms and the four CNN architectures.

6.9 Conclusion

In this paper, an application of acceleration time-frequency images (spectrograms) to CNNs, which is commonly used for real-life object classification, was explored to classify various motor conditions. To experimentally validate the recognition power of the method, a lab-scale test bed, in which a single fault or a combination of different faults settings can be configured, was set up to collect tri-axial acceleration signals for various condition settings. And then, four different 2D CNN architectures were tested to estimate the motor conditions using the acceleration time-frequency images, which were generated from power spectrum density. From the study, following conclusions were drawn:

- The application of the experiment data to the CNNs showed that all motor conditions, which were introduced in the test bed, was classified with high prediction accuracy, especially in GoogLeNet, Alex Net, and ResNet50.
- Some confusions were observed in simple CNN architecture when classifying a single fault and a combination of faults, but the prediction accuracy could be improved as increasing the number of training data. In terms of the global prediction accuracy, ResNet50 showed the best performance followed by Alex Net, GoogLeNet, and simple CNN
- This study experimentally proved that the 2DCNNis able to classify a combination of two faults as well as a single fault, which was difficult in classical ML algorithms.

The CNN methods were able to identify a cause of variation in acceleration signals obtained from a rotating machinery (i.e., a model learns patterns generated by a different source of fault(s) using a sequence of layers containing learnable weights). Thus, the method can play as a bridge to connect a large-scale machinery data and machine health condition. Ultimately, it could be a promising solution for any types of condition monitoring problems.

7. DEVELOPMENT OF A SPEED INVARIANT DEEP LEARNING MODEL WITH APPLICATION TO CONDITION MONITORING OF ROTATING MACHINERY[6]

Reprinted with Permission from Lee, W. J., Xia, K., Denton, N., Ribeiro, B., Sutherland, J. W., 2020 "A Development of Speed Invariant Deep Learning Model with Application to Condition Monitoring of Rotating Machinery," *Journal of Intelligent Manufacturing*

Abstract

The application of cutting-edge technologies such as AI, smart sensors, and IoT in factories is revolutionizing the manufacturing industry. This emerging trend, so called smart manufacturing, is a collection of various technologies that support decision-making in real-time in the presence of changing conditions in manufacturing activities; this may advance manufacturing competitiveness and sustainability. As a factory becomes highly automated, physical asset management comes to be a critical part of an operational life-cycle. Maintenance is one area where the collection of technologies may be applied to enhance operational reliability using a machine condition monitoring system. Data driven models have been extensively applied to machine condition data to build a fault detection system. Most existing studies on fault detection were developed under a fixed set of operating conditions and tested with data obtained from that set of conditions. Therefore, variability in a model's performance from data obtained from different operating settings is not well reported. There have been limited studies considering changing operational conditions in a data-driven model. For practical applications, a model must identify a targeted fault under variable operational conditions. With this in mind, the goal of this paper to study shaft rotational speed invariance via a deep learning method, which can detect a mechanical imbalance, i.e., targeted fault, under varying speed settings. To study the speed invariance, experimental data obtained from a motor test-bed are processed, and time-series data and time-frequency data are applied to long short-term memory and convolutional neural network, respectively, to evaluate their performance.

[6] This work was published in Journal of Intelligent Manufacturing [175]. The permission (license number: 5024361053735) is obtained to from Springer Nature include the paper in this thesis.

Keywords: maintenance, long short-term memory, convolutional neural network, machine condition monitoring, mechanical imbalance.

7.1 Introduction

Starting with the industrial revolution in the 18th century, the manufacturing industry has experienced many radical changes such as mass production and system automation. Manufacturing is poised to be changed again with emerging technologies such as artificial intelligence (AI), smart sensors, and Internet of Things (IoT) that seek to establish an integrated and collaborative manufacturing system that responds in real time to changing conditions in the factory. This new trend, i.e., smart manufacturing, is leading the next revolution in the manufacturing industry, and it may enable sustainable growth in the manufacturing sector through the improvement of various manufacturing performance measures such as energy efficiency, quality, and productivity [130]. As one example of a national effort to capitalize on new technologies, the U.S. Department of Energy's Advanced Manufacturing Office (AMO) launched the Clean Energy Smart Manufacturing Innovation Institute (CESMII) to advance the country's manufacturing competitiveness and reduce its environmental impact (CESMII). The institute supports research and development of technologies that can collect, share, and process the huge amount of data obtainable from manufacturing activities in real-time.

One application of smart manufacturing is an intelligent maintenance system [41]. A goal of the maintenance is maximizing the availability of manufacturing systems to increase productivity while reducing maintenance cost by 1) optimizing maintenance tasks and 2) fixing potential defects before catastrophic equipment failures occur, i.e., prevent unplanned downtime. To enable this, the condition of equipment needs to be continuously monitored without interruption (non-intrusive monitoring), and future behavior must be predicted (e.g., prognostic health management) [132]. With the present proliferation of sensing and communication technologies available in a production line, extensive machine condition data may be collected in many factories. The condition data are normally proxy measures (e.g., vibration, acoustic emission, and temperature). Thus, a method is required to extract meaningful information, e.g., health condition, from large-scale condition data available from operating equipment.

Condition monitoring methods are often classified into three categories: 1) a physical model, 2) a knowledge-based model, and 3) a data driven model [37]. A physical model-based methodology normally shows good success at reflecting the condition of the monitored system because the model is built based on accurate mathematical relations tied to physical processes. However, establishing an accurate physical model is challenging for complex manufacturing systems. Also, a physical model cannot generally be updated with on-line measurement data, which limits the model's flexibility [38]. A knowledge-based methodology, such as an expert system, solves a specific domain problem using expert knowledge and heuristic rules. In this methodology, an accurate physical model is not required, but translating domain knowledge into rules (e.g., IF conditions) is difficult and the model may not cope well with new situations. Lastly, a data-driven model estimates model parameters to fit the model using input and output data. This method is based on statistical learning theory, and the model automatically learns a relationship between input and output data (supervised learning) during the training phase. However, the method often requires a large amount of machine condition data for model training and testing.

Among the methods, data-driven models (e.g., artificial neural networks [39], random forest [40] , and kernel principal component analysis [41]) have received a great deal of attention by researchers due to increasing availability of open source data and advances in computing infrastructure (e.g., GPUs). Recently, deep learning (DL) methods, which originated from artificial neural networks (ANN), are being applied extensively to machine condition datasets for health condition monitoring research. Janssens et al. [61] proposed an automatic bearing fault detection method using convolutional neural networks (CNN). In the study, different types of bearing faults (e.g., outer-raceway fault and rotor imbalance) were detected using acceleration signals obtained for a 25 Hz rotational speed. Jing et al. [59] also used a CNN for condition monitoring of gearboxes. They compared model prediction accuracies using both automatically learned features and manually extracted features. A number of CNN network configurations (e.g., various filter sizes, numbers of filters, and numbers of convolutional layers) were tested. Cacciola et al. [62] studied a neural network-based monitoring system to identify different root causes of mechanical imbalance problems in a rotor. Jia et al. [63] showed an improved performance of deep neural networks compared to shallow neural networks for the diagnosis of the bearing and planetary gearboxes using an auto-encoder for data preprocessing. The DL-based monitoring approach was reported to be superior to classical machine learning techniques (e.g., Support Vector Machine (SVM) and random forest) [59]. Khan and Yairi [64] summarized various DL methods and their applications to a system health monitoring. They concluded that there is a growing interest in applying DL methods in the engineering community, but many limitations still exist such as design, selection, and implementation of DL methods.

As is evident from the literature review, DL is an evolving and growing area for machine condition monitoring research, and its ability to predict conditions offers substantial promise (some people may argue that DL applications to mechanical diagnosis and prognosis are still lacking when compared to other fields such as speech recognition and image classification). One may think that, because training a DL model is computationally expensive, it may not be suitable for the manufacturing applications. However, recently, there have been significant advancements in the DL research field to overcome shortcomings by reducing connectivity in networks (e.g., CNN) and developing an efficient training method (e.g., Adam optimizer). One attractive advantage of DL is reducing the amount of effort for feature engineering by learning non-linear representations in a large amount of dataset using multiple hierarchical layered structures. This may enable the model to predict a targeted fault, in which an indicator relating to a target fault is non-linearly correlated to a machine health condition. Such a model may pose the ability to detect and locate a fault in sophisticated manufacturing equipment.

Several types of popular network architectures (e.g., CNN and recurrent neural network) and their variant were widely applied on the machine condition data, and their performances on the machine fault diagnosis were evaluated and reported in the cited paper. However, work related to a model's response to data obtained from operating conditions that differ from the training data has not been extensively examined, i.e., a trained model may work well only for data obtained from a certain operating setting. Therefore, there is a lack of studies focusing on a performance variation of a deep learning model, which has already been tuned with the data obtained from a certain operational condition, to the data collected from the different operational conditions. Although DL is known to be a powerful tool to automatically learn and discover representations needed for classification from large-scale datasets (called representation learning), it may be difficult for a DL model to detect a targeted fault when analyzing data collected under previously unseen operating settings. Because machine operating settings can change during the manufacturing process, a model's performance should be invariant to a variable operating environment (e.g., variable rotational speed) while monitoring a system. Park et al. [66] argued that previous works on condition monitoring mainly focused on detecting a fault under constant rotational speed although many real-world applications run under variable speed. Accordingly, DL applications need to be further studied and tested with data obtained from different operating conditions (e.g., different rotational speeds (RPM)) as well as using various types of machine condition data (e.g., acceleration and acoustic emission). Ultimately, a method that is invariant to changes in rotational speed (RPM) must be considered for the practical applications. The main contributions of this paper are: 1) the idea on the property of RPM invariance is discussed, 2) a DL based mechanical imbalance monitoring system is proposed, 3) an improved long short-term memory (LSTM) model is developed using an attention mechanism, 4) performance variations of deep learning models (CNN and LSTM) to the data obtained from the different operational conditions are examined and compared using experimental data, and 5) the effectiveness of the proposed method (Scaled and Smoothed TS-LSTM with Attention) is demonstrated.

The paper is organized as follow. Frist, the property of RPM invariance is mathematically explained, which defines detection accuracy invariance to varying RPM. Then, data preprocessing methods, which will be combined with deep learning models, are proposed. For DL architectures, LSTM and CNN are employed to detect a targeted fault, and their basic theories and the customized architectures are explained. To experimentally study the RPM invariance in a deep learning model, sets of experiments were conducted using a motor testbed. During the experiment, machine condition data were collected using a triaxial accelerometer under various RPMs at certain mechanical imbalance levels. Then, raw signals are processed to extract features and the features are applied to evaluate DL models' performances under both constant RPM and varying RPM conditions. Performance variations in DL models are reported using the data obtained from previously unseen RPM settings during the training phase (i.e., test a model with the data obtained at different rotational speeds). All data collected from the experiments reported on herein will be available via the Purdue Laboratory for Sustainable Manufacturing (LSM).

7.2 Invariance to Changing Rotational Speed in Fault Detection

The goal is to establish a fault detection model whose accuracy is invariant to changes in the RPM (we will refer to this as 'RPM invariance'). This will be accomplished by predicting a targeted fault condition using a proxy measure, e.g., vibration, in a motor system that runs at previously unseen RPMs. Given motor vibration data points (either raw or processed) $\boldsymbol{x} \in \Omega$, the function of interest is $f : \Omega \to \{1, 2, ..., N\}$ which maps the data points to the corresponding fault condition $y = f(\boldsymbol{x})$ when N conditions are defined. The shape of the sample space, Ω, varies depending on the format of the data. The function, f, is approximated using a data driven model (e.g., neural network model), $\hat{f}_\theta$, parameterized by θ to make a prediction of y, $\hat{y} = \hat{f}_\theta(\boldsymbol{x})$. The rotational speed of a motor can be described as a function of the data collected from that motor, defined as $r : \Omega \to \mathbb{R}^+$ such that $r(\boldsymbol{x})$ is the RPM of data points $\boldsymbol{x} \in \Omega$. Then, the notion of RPM invariance can be defined as follows. Given $\boldsymbol{x} \in \Omega$ and $\alpha \in \mathbb{R}^+$, let $s : \Omega \times \mathbb{R}^+ \to \Omega$ such that $s(x,\alpha)=x'$, where $f(\boldsymbol{x}') = f(\boldsymbol{x})$, $r(\boldsymbol{x}') = \alpha r(\boldsymbol{x})$, and α is the ratio of desired (testing) RPM to current (training) RPM. Here, $\boldsymbol{x}'$ is an RPM transformation of $\boldsymbol{x}$ by α. Then, the property of RPM invariance for $\hat{f}_\theta$ is

$$\hat{f}_\theta(s(\boldsymbol{x},\alpha)) = \hat{f}_\theta(\boldsymbol{x}) \tag{7.1}$$

for all $\boldsymbol{x}$ and α. This property means that changing the RPM of the data should not affect the prediction of the model. To achieve this, this paper focuses on the details of r and f, and finding a procedure for determining them.

7.3 Data Preprocessing and Deep Learning Models

Once the data acquisition plan (e.g., sensor type, sampling rate, and data acquisition interval) is decided, a sensor can be mounted on manufacturing equipment, and raw sensor signals may be collected for a certain machine health condition. Then, the collected signals, i.e., machine condition data, can be processed to generate features, which may better represent the machine health condition. In case of vibration signals, a popular measure for condition monitoring of rotating elements, features from the time, frequency, and time-frequency domain data are often used for deep learning (DL) applications [66]. Also, in order to analyze a non-stationary vibration signal, order analysis or order-tracking method were often used to extract vibration data related to the rotational speeds. However, during the experiment in this study, a range of rotational speed was not very wide, and a speed was not increased continuously (i.e., increased from 300 to 380 in 20 RPM increments). Thus, features from order analysis may not be useful. Instead, other data preprocessing methods, which will be described in this section, are employed in this paper.

In this section, two data preprocessing methods and two DL architectures are introduced to study the RPM invariance in a DL model. In “Scaling and Smoothing of Time-series Data Obtained from Different RPM Settings" section, a method which may have the properties of RPM invariance in a LSTM model is proposed first. A second data preprocessing method, i.e., continuous wavelet transform (CWT), which will be combined with CNN, is explained in "Extracting Time-Frequency Features Using Continuous Wavelet Transform" section. CWT is a technique to extract time-frequency features from vibration signals, and CWT has been often combined with CNN models [134]. Therefore, this method may be a good candidate to compare with the first approach.

In Section 7.3.2, the basic theory and proposed architecture of two deep learning methods, LSTM and CNN, are explained. The selection of a model is dependent on the type of data being analyzed. For data collected over time (e.g., time series data), a recurrent neural network architecture is often used, specifically a long short-term memory (LSTM) model [135]. For data arranged in a matrix such as time-frequency data (e.g., short time Fourier transform and wavelet transform), a convolutional neural network (CNN) model normally is used [45]. Therefore, in this paper, time-series data and time-frequency data are used to evaluate the LSTM-based model and the CNN-based model, respectively.

7.3.1 Data Preprocessing for Vibration Signal

Scaling and Smoothing of Time-series Data Obtained from Different RPM Settings

A change in the speed (RPM) of equipment with rotational elements almost always leads to changes in the frequency content of vibration sensor signals. For example, increasing the RPM may shift the dominant frequencies in the frequency domain to larger values. By the time scaling property of the Fourier transform, a scale in the frequency domain corresponds to an inverse scale in the time domain. Hence, if the test data RPM is different from the RPMs for the training data, then one may expect that a transformation of the test data would better match the vibration frequencies observed in the training data.

However, in practice, not all vibrations captured in the sensors are related to the RPM of the motor. Other factors like fluid flow, electrical components, and non-rotating elements all affect vibrations. Furthermore, high frequency noise tends to obscure the structural content in the data. To remedy this, a noise-reducing data transformation is implemented for raw data that may mimic data collected from other RPM settings. This procedure, visualized in Figure 7.1, involves: 1) scaling the time-domain data by the ratio α using a spline interpolation, 2) converting the data to the frequency domain using the discrete Fourier transform (DFT), 3) filtering out high frequency components using a low pass filter and removing less significant amplitudes, and 4) converting back to the time domain using the inverse DFT. The frequency removal acts as a smoothing procedure, removing some of the abrupt changes in the data, and is applied to all data used in the models regardless of whether the RPM needs to be changed.

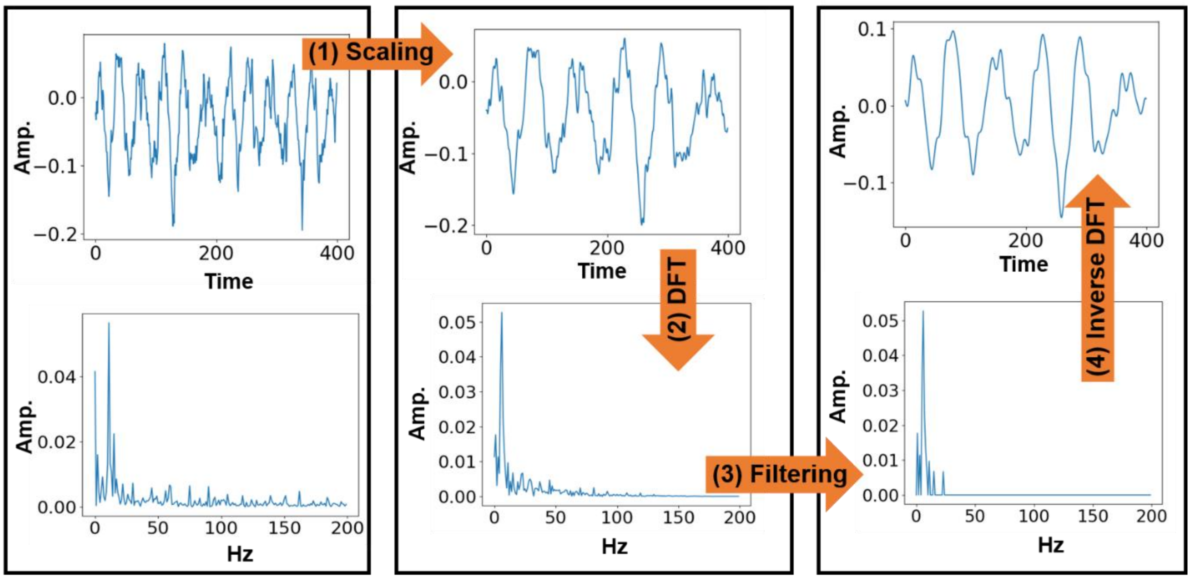


Figure 7.1 Raw data transformation procedure visualized in both the time and frequency domains.

Extracting Time-Frequency Features Using Continuous Wavelet Transform

Time-frequency analysis transforms a signal in the time domain, $x(t)$, to the time-frequency domain, in which various frequency components are present over time (e.g., short time Fourier transform). Unlike the short time Fourier transform which generates time-frequency representations in the fixed frequency resolution, a wavelet transform creates a frequency-dependent frequency resolution using a scalable window function called the mother wavelet (ψ) [45]. Given a wavelet function, $\psi(t) \in L^2(\mathbf{R})$, which has nonzero values only in certain range, the continuous wavelet transform is written as

$$\mathrm{W}_x(\tau, \mathrm{s}; \psi) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t) \psi^*(\frac{t-\tau}{s})\, dt \tag{7.2}$$

where τ, s, and $\psi^*(\bullet)$ are the translation parameter, scale parameter, and the complex conjugate of $\psi(\bullet)$, respectively. Here, the signal is convolved with a scaled wavelet, thus $W_x(\tau, s)$ represents the degree of correlation between the signal and the wavelet given τ and s [66]. Because the wavelet transform enables multi-scale analysis of a signal using the two variables, τ and s, it can effectively extract time-frequency features from nonstationary and transient signals [136]. In this paper, the Morlet wavelet is employed, which is mathematically expressed as

$$\psi(t) = e^{-t^2/2}\cos(5t). \tag{7.3}$$

Wavelet transforms have been extensively used to extract time-frequency features and combined with various machine learning techniques in condition monitoring research [88, 89]. However, the previous studies mainly focus on detecting a targeted fault for a constant speed condition. In the present work, however, as explained before, DL models will be trained with features extracted from CWT, and then evaluated using experimental data obtained from varying RPM settings and compared with the method explained in “Scaling and Smoothing of Time-series Data Obtained from Different RPM Settings.”

7.3.2 Deep Learning Models Using LSTM and CNN

Long Short-Term Memory (LSTM)

An LSTM follows a recurrent architecture, that is, outputs from one layer can serve as inputs for the same layer, allowing information to persist across entire sequences of inputs. A typical LSTM architecture is shown in Figure 7.2 [138].

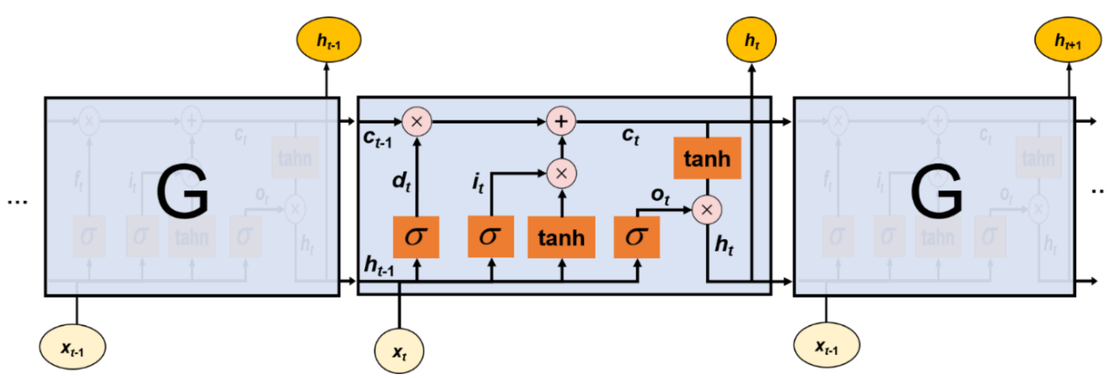


Figure 7.2 A typical LSTM architecture.

The LSTM can be distinguished from other recurrent neural networks by its use of gates. Specifically, with each pass through the recurrent layer of the LSTM, depicted as “G” in Figure 7.2, the following may be computed:

$$d_t = \sigma(W_f \times (h_{t-1} \oplus x_t) + b_f), \tag{7.4}$$

$$i_t = \sigma(W_i \times (h_{t-1} \oplus x_t) + b_i), \tag{7.5}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tanh(W_c \times (h_{t-1} \oplus x_t) + b_c), \tag{7.6}$$

$$o_t = \sigma(W_o \times (h_{t-1} \oplus x_t) + b_o), \tag{7.7}$$

$$h_t = o_t \cdot \tanh(c_t), \tag{7.8}$$

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{7.9}$$

Here, the operation symbols, $\times$, $\oplus$, and $\cdot$, represent the matrix multiplication, the concatenation, and the element-wise multiplication, respectively. σ and tanh represent the sigmoid function and the hyperbolic tangent, which has output values between 0 and 1 and between -1 and 1, respectively. *W* and *b* are learnable weights and biases.

A qualitative explanation can be provided for each of these gates. *d* is the forget gate, with values between 0 and 1 that determines how much of the previous state to retain. *i* is the input gate, with values between 0 and 1 that determines how much of the input to accept. *c* is the cell state, which can be described as the memory of the layer. It uses the forget and input gates to determine how much information to retain and change between iterations. *o* is the output gate, which has values between 0 and 1 that determines how much of the cell state to pass to the output. Finally, *h* is the hidden state, which is passed as the output to the next layer and is also passed back into the same layer for the next iteration. It is simply the cell state filtered by the output gate.

The architecture used in this paper combines one of these LSTM layers with the attention mechanism [138] and a fully connected layer. After passing the data through the LSTM layer, the vector of hidden states, *h*, is passed through an attention mechanism described by the following equations.

$$q = \tanh(W_h \times h + b_h), \tag{7.10}$$

$$\beta = \mathrm{softmax}(W_q \times q + b_q), \tag{7.11}$$

$$h_{att} = \sum_{i=1}^{m} \beta_i h_i. \tag{7.12}$$

Once again W and b are learnable weights and biases. The softmax activation function is a function that normalizes a vector so that all its values sum to 1. Qualitatively speaking, q represents a learned embedding for each of the hidden states, β is a normalized weight vector that assigns an importance value to each of the hidden states, and h_{att} is an average of the m hidden states weighted by β. Finally, h_{att} is passed through a fully connected layer for the final prediction.

The attention mechanism in the LSTM model can be used to overcome limitations of long sequential data by determining how much "attention" should be paid to each time step in the hidden state. A typical LSTM model uses only the hidden state information from the final time step, often causing information from earlier iterations to be forgotten. However, attention uses information from all time steps of the hidden state, prioritizing the ones that are most important for classification, so important information in the past is not lost.

For time series data, the input is three stacked time series – vibration data from the X, Y, and Z directions, i.e., processed data from a triaxial accelerometer. Each with 400 time-steps as shown in Figure 7.3 (this architecture is called TS-LSTM in this paper). The model 1) passes each of the 400 time-steps through the LSTM layer, 2) takes the hidden states (from h_1 to h_{400}) from the entire pass, 3) multiply with the outputs from the attention mechanism, and 4) feeds them through a 128-length fully-connected layer for classification. The fully-connected layer outputs a value between 0 and 1, which is rounded to produce the predicted targeted fault. The number in gray rectangular (e.g., 128 x 128) means there are 128 x 128 connection between layers.

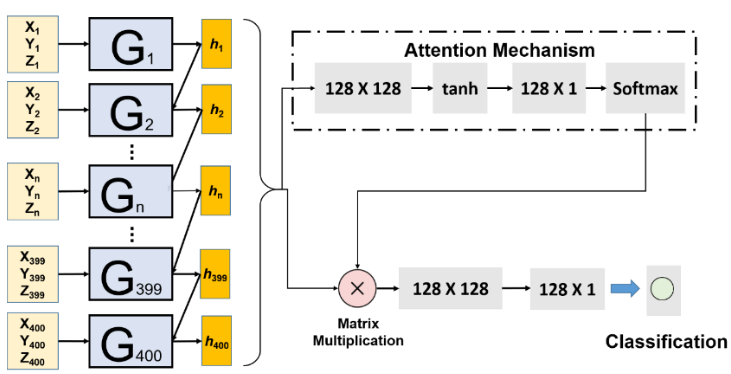


Figure 7.3 A proposed LSTM architecture for time-series data (TS-LSTM).

Convolutional Neural Network (CNN)

A CNN consists of alternating convolutional layers and pooling layers, followed by a fully connected layer. Convolutional layers use several filters, each mapping the input matrix to an output matrix. Filters take small regions of the input, multiply them by learned weights, and pass the result to the output. Formally, a convolutional layer convolves input X_{in} as described in Eq. (7.14), where W_k and b_k are the weights and bias of the k th filter, and g is the activation function, often the rectified linear unit (ReLU).

$$g(x) = \max(0, x), \tag{7.13}$$

$$X_{out,k} = g(X_{in} * W_k + b_k). \tag{7.14}$$

In the equation, * represents the convolution operator, where an output matrix is produced by applying the k th filter across all regions of the input. All outputs from all filters are stacked to produce the input for the next layer. This final output is often called a feature map since the values represent features of the original input.

Pooling layers reduce the dimensionality and spatial precision of the input by sub-sampling the input. Pooling locally combines each window of the input into a single value in the output. In this paper, max pooling (i.e., max filter) is used, so each value in the output of the pooling layer corresponds to the maximum value of a small region in the input.

Finally, after multiple convolutional and pooling layers, the resulting output is passed through a fully connected layer. Mathematically this can be written as:

$$X_{out} = g(W_j X_{in} + b_j). \tag{7.15}$$

Eq. (7.15) describes the effect of the fully connected layer on the input X_{in}, where W_j and b_j are the weights and bias respectively for the j th output. Here, g is an activation function, and ReLU is used as the activation function for all layers except for final layer, for which the sigmoid function is used.

In this paper, time-frequency data are applied to a CNN model; the proposed CNN architecture is shown in Figure 7.4 (called CWT-CNN). The figure is generated using NN-SVG (NN- SVG). The input is three matrices, representing the amplitudes in the X, Y, and Z directions across 400 time-steps and 311 frequencies. The output is a value between 0 and 1, which is rounded to produce the predicted targeted fault.

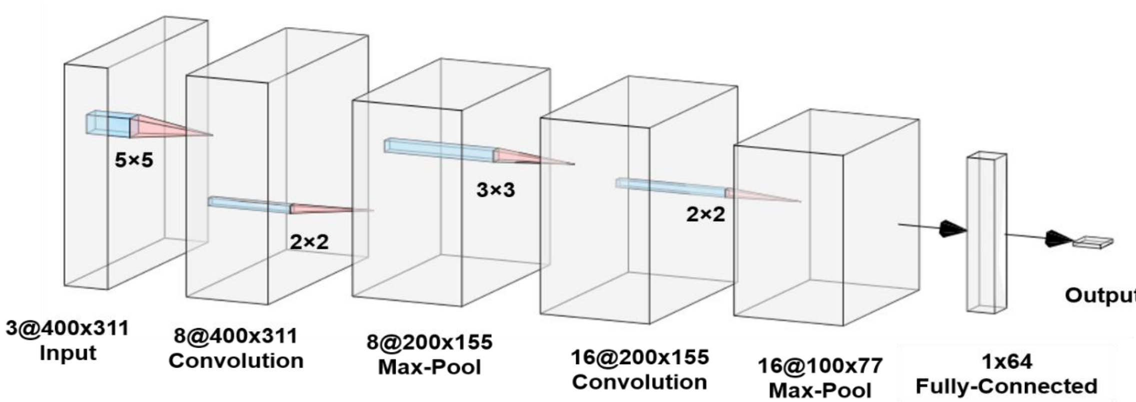
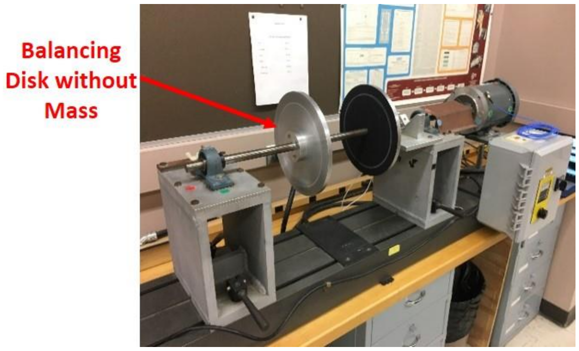


Figure 7.4 A proposed CNN architecture for time-frequency data (CWT-CNN).

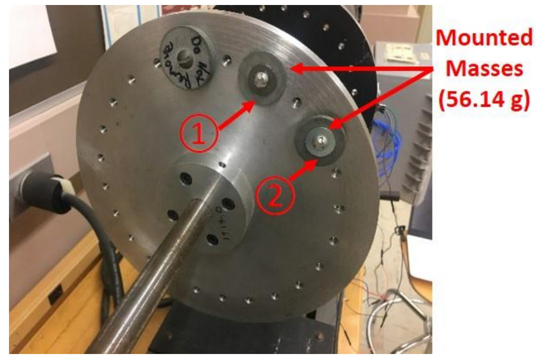
7.4 Mechanical Imbalance Experiment and Condition Data Acquisition

A mechanical imbalance (e.g., rotor unbalance) can be defined as an uneven distribution of mass/force about a rotating centerline. In a motor system, where power is transmitted from a motor to driven equipment, some level of mechanical imbalance is always present owing numerous factors, e.g., rotor wear/damage, debris buildup, manufacturing and assembly variation, and poor design [62]. When this imbalance becomes large, it may affect the performance of the motor system. A mechanically unbalanced motor system may experience rapid wear on mechanical components (e.g., bearings), and consequently lead to a shorter life span of manufacturing equipment. Failure of mechanical components can often be traced to system imbalance (and, in turn, the imbalance is often attributable to other causes), so it is generally prudent to detect an imbalance and take corrective actions as early as possible.

To collect acceleration signals for different imbalance conditions in a motor system, experiments were conducted using a motor testbed. Overall configuration of the testbed is shown in Figure 7.5a. The testbed is equipped with a ¼ horsepower motor with pulse width modulation variable speed DC drive. To induce different levels of mechanical imbalance in the testbed, two planar balancing disks with 24 equally spaced holes are mounted on a shaft between two bearing supports. To create an imbalance condition during some of the tests, two masses were mounted to the disk as shown in Figure 7.5b (the masses of ① and ② in Figure 7.5b are 27.06 g and 29.08 g, respectively). The photo tachometer (Extech , 461895) was used to measure motor speed during the experiment (Figure 7.5c). A triaxial accelerometer (PCB PIEZOTRONICS, J356A45) was attached using adhesive as shown in Figure 7.5d.

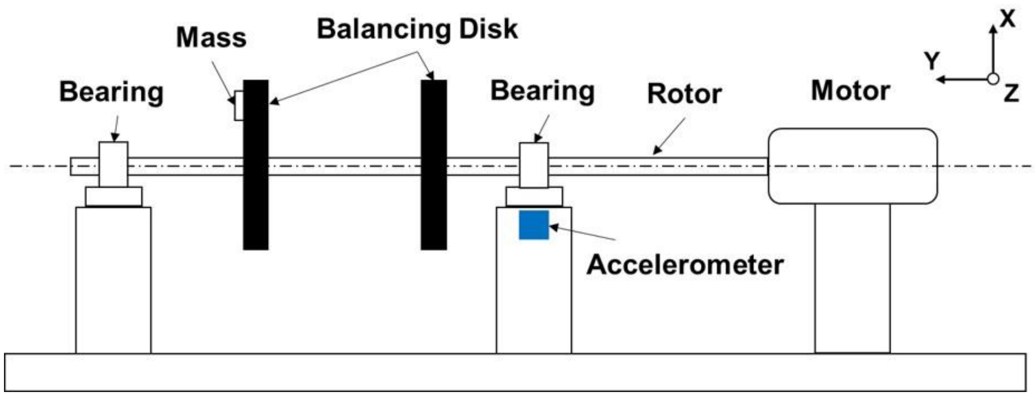


(a)



(b)

(c)



(d)

Figure 7.5 Motor testbed for mechanical imbalance experiment; (a) overall configuration, (b) balancing disk with mounted masses, (c) tachometer, and (d) schematic diagram.

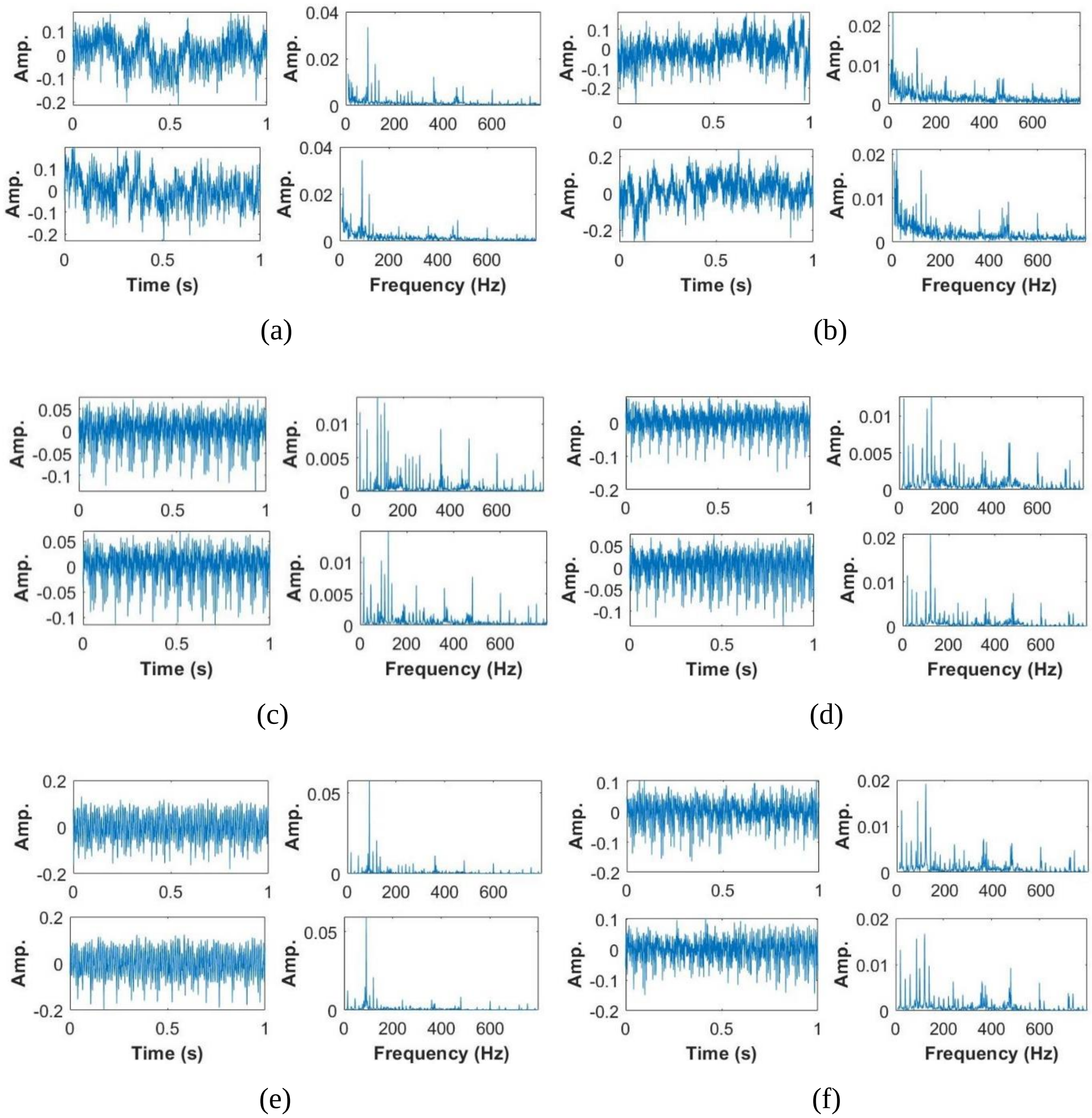


Figure 7.6 Typical triaxial acceleration signals and their spectra for different imbalance levels (top is the default setting and bottom is the mass-loaded setting) and RPMs: (a) X axis and RPM=300, (b) X axis and RPM=380, (c) Y axis and RPM=380, (d) Y axis and RPM=380, (e) Z axis and RPM=300, (f) Z axis and RPM=380.

In the experiment, two levels of the mechanical unbalance were introduced. The two levels are a “balance” or default condition (no masses on the disks) and imbalanced conditions (with the two masses added to the disks). During the experiment, the motor speed (RPM) was increased from 300 to 380 in 20 RPM increments, and triaxial acceleration signals were collected using a

National Instruments (NI) Compact Data Acquisition System that included chassis (NI, cDAQ-9178) and Sound and Vibration Input Module (NI, NI-9234). The data were collected under steady-state operating conditions for a total of ten data sets (two levels of imbalance and five rotational speeds, 300, 320, 340, 360, and 380 RPM). LabVIEW software was used to store the sensor signals in a PC, and the sampling rate for the X, Y, and Z channels was set to 3.2 kHz. The digital data were sampled 50 times at 10-second intervals (i.e., $3{,}200 \times 3 \times 50 = 480{,}000$ data for one set).

Figure 7.6 shows examples of the typical triaxial acceleration signals obtained from different imbalance and operating conditions and their corresponding spectra using FFT. Also, in order to numerically compare the differences in the tri axial acceleration signals, two features, root mean square (RMS) and Kurtosis, which were extracted from the time domain and frequency domain, are computed in Table 7.1. As expected, the longitudinal direction (Y axis) displays a smaller vibration than other directions in terms of RMS value during the operation because there was no significant movement in the longitudinal direction in the testbed. While the highest RMS values were observed in X axis, Z axis shows the greatest Kurtosis values in most cases which means that a heavier tail exists over the frequency distribution. As seen in the table, a distinguishable pattern can be found in each axis as rotational speed and load-setting change.

Despite noticeable differences among the acceleration signals shown in Figure 7.6, it may be hard to manually distinguish a mechanical imbalance condition by looking at the differences. With this in mind, the proposed methods are applied to the experimentally collected condition data to diagnose the imbalance condition in a motor system.

7.5 Application of Machine Condition Data to Deep Learning Models

In this section, the deep learning models described in Section 7.3.2 are trained and tested using the acceleration signals obtained from the experiment. As mentioned, the goal of paper is to develop a fault detection model whose accuracy is invariant to changes in the RPM. However, the models' performance at a constant RPM setting is evaluated first to show whether trained DL models are able to detect a targeted fault properly at the constant operating condition (Section 7.5.1). Subsequently, a model's performance to data obtained from operating conditions different from the training data is studied (Section 7.5.2).

Table 7.1 Comparison of tri-axial acceleration signals using RMS (time-domain) and Kurtosis (frequency-domain) features.

Axis	Feature	300 RPM Default Setting	300 RPM Mass-loaded Setting	380 RPM Default Setting	380 RPM Mass-loaded Setting
X	RMS	0.0609	0.0662	0.0621	0.0716
	Kurtosis	114.0695	68.8551	33.0453	25.7101
Y	RMS	0.0286	0.0258	0.0289	0.0304
	Kurtosis	59.7082	78.0904	41.52	91.8306
Z	RMS	0.0517	0.0518	0.0351	0.0352
	Kurtosis	400.8934	427.1034	64.1671	71.3961

7.5.1 Constant RPM Setting

Before training the models, raw acceleration signals obtained from the experiment are divided into training (70%) and testing dataset (30%), and processed as described in Section 7.3.1. Because a constant RPM setting is considered here, the noise-reducing data transformation, i.e., scaling and smoothing, which described in “Scaling and Smoothing of Time-series Data Obtained from Different RPM Settings,” is not included in this section. Instead, raw time-series data are used to train and test the TS-LSTM model. Time-frequency data are extracted using CWT, and frequencies between 0 and 400 Hz are used because the motor ran at low speeds during the experiment.

Once time-series and time-frequency are prepared, training and testing are conducted in a PC platform (Precision 5820 Tower). To implement the proposed method, for hardware, any standard PC with a decent Nvidia GPU (preferable) will be enough because the deep learning models used in this study are relatively shallow, i.e., there are not many learnable parameters compared to a model often used for image recognition. For software, Python 3 along with some packages (e.g., numpy, PyTorch, and torchvision) are used. The PC platform used for this study is equipped with Intel Xeon with 32 GB RAM and GeForce GTX 1080 TI with 11GB GDDR5X. The proposed deep learning architectures (TS-LSTM and CWT-CNN) are implemented through Pytorch deep learning framework.

Each model is trained with the Adam optimizer [140] over 500 epochs with a learning rate of 0.001 and a batch size of 16. The parameters were selected by a trial and error experiment using a technique called GridSearch, in which 1) sets of possible hyper-parameter values were taken, e.g., learning rates of 0.0001, 0.001, and 0.01 and batch size of 8, 16, 32, and 64, 2) every combination was tried once, and 3) the one that performed the best was chosen. To avoid an overfitting problem in the models, L2 regularization with lambda = 0.0002 is used, and models are trained to minimize the cross-entropy loss function (L),

$$L(y, \hat{f}_\theta(x)) = -(y\log(\hat{f}_\theta(x)) + (1-y)\log(1-\hat{f}_\theta(x)) + \lambda\|\theta\|^2, \tag{7.16}$$

where $\hat{f}_\theta(x)$ is the model output for input data, x, parameterized by weights θ, y is the true label of x, and λ is the L2 regularization weight. Each experiment include 25 trials and accuracies are reported with 95% confidence intervals.

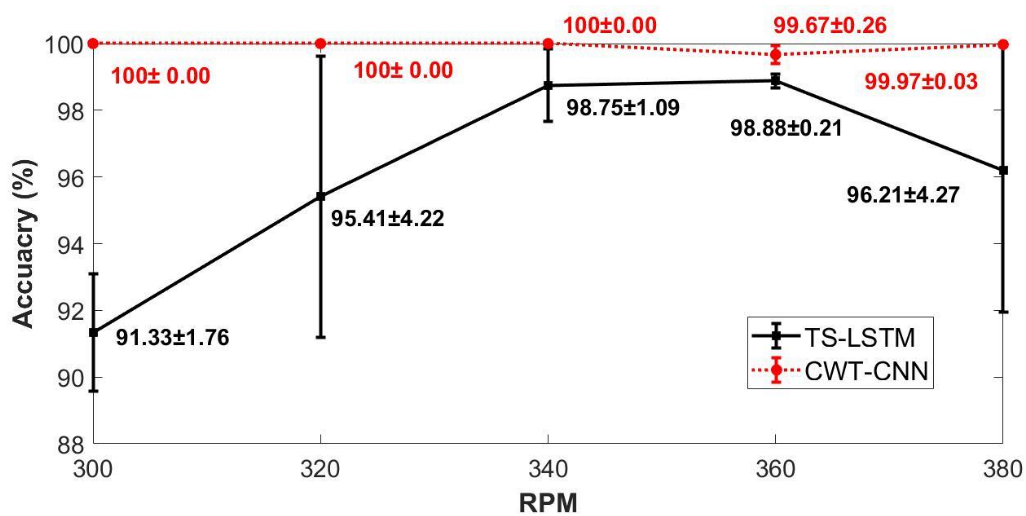


Figure 7.7 Prediction accuracies with 95% confidential intervals for constant RPM settings.

Here, for constant RPM, a model is trained and tested using data collected from same RPM. Thus, both TS-LSTM and CWT-CNN architectures are used to develop models for each RPM settings, i.e., a model trained with data obtained at the rotational speed of 300, 320, 340, 360, or 380 RPM is tested with the data obtained at the rotational speed of 300, 320, 340, 360, or 380 RPM, respectively. Figure 7.7 shows prediction accuracies with 95% confidential intervals for TS-LSTM and CWT-CNN models (each model was trained and tested 25 times). As shown in the figure, CWT-CNN models outperform TS-LSTM models for all RPM cases. This is expected because raw signals were used in TS-LSTM models. Normally, acceleration signals are acquired with redundant information, which may not relevant to a machine condition. Due to the high accuracy of the CWT-CNN models, one may assume a possibility of overfitting. However, there is not much overfitting because all accuracies displayed in Figure 7.7 are "test accuracies" (i.e., the test dataset was not used for training the models). Also, a poor test performance is normally observed in the case of overfitting. One possible reason for the high accuracy is because, on same-RPM training and testing, the differences between the machine condition data of the two states are so vast that it is fairly simple for the models to find a comfortable decision boundary.

In next section, the noise-reducing data transformation is implemented for the time-series data. A model is trained with data obtained from one RPM setting, and the trained model is evaluated using data obtained at previously unseen RPM settings to reflect varying RPM condition.

7.5.2 Varying RPM Setting

For the study of RPM invariance in a deep learning model, the LSTM and the CNN are trained with one dataset (i.e., data collected from one RPM setting), and tested with the other dataset (i.e., data collected at the other RPM setting). In this way, a model's performance variation with data from previously unseen RPM can be evaluated. Also, in order to demonstrate the effectiveness of the attention mechanism and the noise-reducing data transformation in the LSTM model, the performance of the LSTM model 1) without attention mechanism and 2) without noise-reducing data transformation are reported together.

Before training the models, the time series data go through the noise-reducing data transformation to extract features, which may have the property of RPM invariance in a LSTM model. To implement this, first, time-series data are scaled using α as described in "Scaling and Smoothing of Time-series Data Obtained from Different RPM Settings." Second, high frequency components are removed through a low pass filter because the motor ran at low RPMs. Less significant amplitudes, i.e., lower amplitudes, in the frequency domain are subsequently removed. Lastly, the filtered data in the frequency domain are converted back to the time domain using inverse DFT (per the procedure visualized in Figure 7.1), and they are used to train and test the LSTM architecture described in "Long Short-Term Memory (LSTM)" (we call this as "Scaled and Smoothed TS-LSTM"). Time-frequency data are obtained through CWT as described in "Extracting Time-Frequency Features Using Continuous Wavelet Transform" and Section 7.5.1, and the data are applied to CNN architecture explained in "Convolutional Neural Network (CNN)."

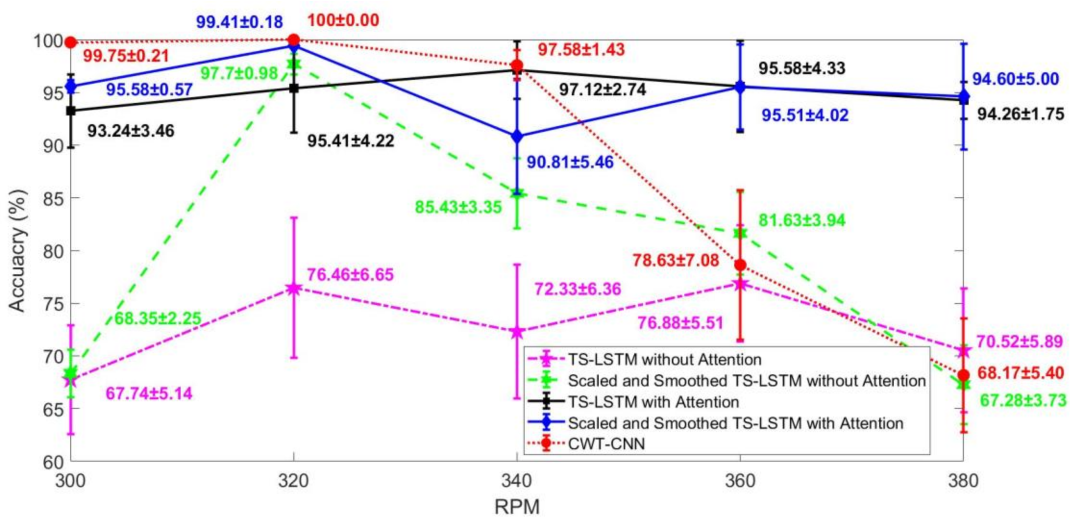
Trainings of the models were conducted on the same PC platform using the same parameters as described in Section 7.5.1. A model was trained using dataset from one RPM setting (training

RPM) and evaluated by the dataset (testing RPM), which were obtained from different RPM settings with the training dataset (i.e., training dataset and testing dataset were collected in different rotational speed settings). Here, five models, 1) TS-LSTM without Attention 2) Scaled and Smoothed TS-LSTM without Attention, 3) TS-LSTM with Attention, 4) Scaled and Smoothed TS-LSTM with Attention (proposed method), and 5) CWT-CNN were examined. Results of TS-LSTM and Scaled and Smoothed TS-LSTM can be used to demonstrate the effectiveness of the noise-reducing data transformation in the LSTM model. Similarly, result of the models with attention and without attention can be used to evaluate the effectiveness attention mechanism in the LSTM model.

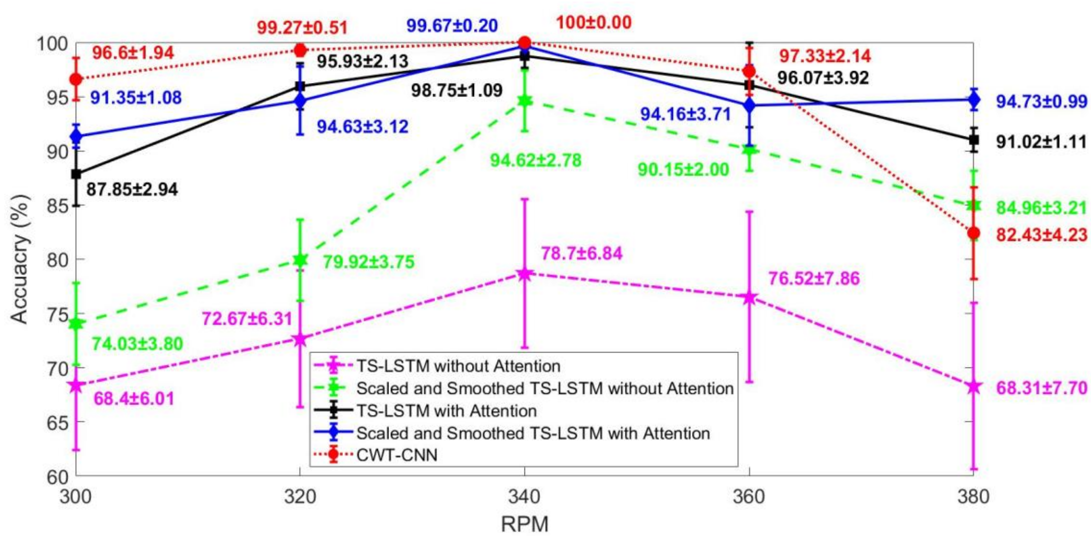
To examine the performance variation of the five models, the models trained with data obtained from rotational speeds of (a) 320 RPM and (b) 340 RPM are selected, and prediction accuracies are displayed in Figure 7.8 with 95% confidence intervals. In the figure, the performances of multiple models are plotted together to 1) graphically display the accuracy drift away as the new speed deviates from what has been tuned, and 2) compare the proposed method, i.e., Scaled and Smoothed TS-LSTM with Attention, with other typical methods. As described in Section 7.2, the goal of this study is to develop a method, which will be likely to have the property of RPM invariance using RPM transformation and deep learning model.

For CWT-CNN, in all experiments, the model's performance tends to drop significantly as the test data RPM differs from the training data RPM despite performing well for constant-RPM cases. This is expected as there was no RPM-invariant method implemented in the CNN model. Similarly, the LSTM models without attention mechanism also show this significant drop, but the model incorporating noise-reducing data transformation (i.e., Scaled and Smooth TS-LSTM) displays better performance than the model using raw data (i.e., TS-LSTM) in the most cases. The LSTM models, which have attention mechanism, do not show the significant drop in performance, especially in the Scaled and Smoothed TS-LSTM with Attention. The models with attention mechanism also can still often maintain accuracies above 90% from test data with significantly different RPMs. So, the effectiveness of the Attention mechanism in the LSTM model and the effectiveness of the scaling and smoothing of time-series method for the property of RPM invariance are demonstrated by comparing with other methods (or other combination).

Table 7.2, the performances of the five models for the experiments, which are not included in Figure 7.8, are summarized. In the table, the values in the boxes with grey color present the performance for constant RPM setting and the values in the boxes with white color show the performance for variable RPM setting. While the TS-LSTM with scaled and smoothed data does not outperform the TS-LSTM with raw data in every case, it performs better on average, especially in cases where the RPM difference is greater. This shows that the scaling and smoothing procedure on LSTMs does indeed provide significant benefits in varying RPM situations. Also, the prediction accuracies of the LSTM model are significantly improved by adding the attention mechanism in the model.



(a)



(b)

Figure 7.8 Models' performance variation when considering data obtained from previously unseen RPM settings; a model was trained with data from (a) 320 RPM and (b) 340 RPM settings.

Table 7.2 Performances of the models when considering data obtained from previously seen (italics shading) and unseen RPM settings.

Training RPM	Model	Testing RPM				
		300	320	340	360	380
300	TS-LSTM without Attention	98.66 ± 0.32%	91.20 ± 2.15%	74.55 ± 5.74%	76.47 ± 5.48%	67.33 ± 5.71%
	Scaled and Smoothed TS-LSTM without Attention	97.88 ± 0.47%	79.38 ± 3.04%	82.60 ± 3.38%	83.95 ± 2.73%	59.06 ± 3.18%
	TS-LSTM with Attention	91.33 ± 1.76%	77.66 ± 4.57%	64.75 ± 5.63%	65.87 ± 5.87%	58.67 ± 6.40%
	Scaled and Smoothed TS-LSTM with Attention	98.50 ± 0.43%	72.50 ± 3.51%	83.90 ± 3.85%	83.38 ± 3.92%	72.76 ± 4.65%
	CWT-CNN	100 ± 0.00%	65.17 ± 5.23%	58.17± 5.59%	51.50 ± 1.85%	50.63 ± 0.50%
360	TS-LSTM without Attention	77.18 ± 4.24%	85.25 ± 4.01%	86.37 ± 4.93%	81.93 ± 6.12%	81.12 ± 5.25%
	Scaled and Smoothed TS-LSTM without Attention	76.30 ± 4.17%	90.20 ± 3.02%	87.28 ± 3.77%	95.09 ± 2.06%	80.44 ± 4.71%
	TS-LSTM with Attention	84.25 ± 4.44%	91.06 ± 2.56%	97.93 ± 0.63%	98.88 ± 0.21%	89.37 ± 0.99%
	Scaled and Smoothed TS-LSTM with Attention	84.24 ± 4.83%	94.39 ± 1.39%	88.59 ± 4.99%	98.62 ± 0.35%	85.12 ± 6.04%
	CWT-CNN	96.6 ± 1.94%	99.27 ± 0.51%	97.33 ± 2.14%	99.67 ± 0.26%	84.43 ± 4.23%
380	TS-LSTM without Attention	52.91 ± 4.70%	55.38 ± 6.37%	72.15 ± 6.61%	64.29 ± 5.72%	68.26 ± 7.26%
	Scaled and Smoothed TS-LSTM without Attention	70.14 ± 3.01%	83.87 ± 3.17%	76.50 ± 4.50%	73.54 ± 5.92%	96.21 ± 2.06%
	TS-LSTM with Attention	75.66 ± 5.07%	83.32 ± 5.14%	85.84 ± 6.05%	90.37 ± 5.72%	96.21 ± 4.27%
	Scaled and Smoothed TS-LSTM with Attention	77.00 ± 3.58%	91.37 ± 4.87%	93.43 ± 2.59%	83.40 ± 4.55%	98.15 ± 2.70%
	CWT-CNN	61.05 ± 6.23%	87.75 ± 5.55%	96.93 ± 1.66%	99.83 ± 0.08%	99.97 ± 0.03%

7.6 Conclusion

This paper has proposed a DL-based method for condition monitoring of rotating machinery that is invariant to changes to rotational speed. To experimentally validate the RPM invariance in a deep learning model, sets of experiments were conducted to collect machine condition data (i.e., triaxial acceleration) at various RPMs. The condition data were processed to extract features, which may better represent the RPM invariance in a model, and were applied to train and test the proposed LSTM and CNN architectures. The RPM invariance for the models was examined by using the data obtained from previously unseen RPM settings.

Through the results of the DL experiment, a condition was well classified when tested on data with the same RPM as its training set. However, for all models, the prediction accuracies tended to degrade as the test data RPM began to differ. Overall, LSTM model showed smaller performance degradation to previously unseen RPMs than did the CNN model. Also, LSTM classifies the condition more effectively in terms of average accuracies. The Scaled and Smoothed TS-LSTM, in which raw signals are processed through the noise-reducing data transformation, is proven to be a better method than the TS-LSTM. From this, it is shown that the signal processing method can enhance the model's performance in the LSTM model. Also, the effectiveness of the attention mechanism in the LSTM model is demonstrated by comparing the performance of the models with and without attention mechanism.

In conclusion, this paper introduced RPM invariance, and it was tested through the proposed methods. Also, the models' uncertainties to varying speeds were quantified and compared. For real world application, a condition monitoring system must identify a targeted fault under variable operational conditions. Thus, a model's invariance to varying operating condition must be considered in the diagnosis and prognosis of machine health. As a future work, a method, which can detect a targeted fault under any speeds (i.e., when test RPM is unknown) will be studied.

8. TIME TO FAILURE PREDICTION OF ROTATING MACHINERY USING DYNAMIC FEATURE EXTRACTION AND GAUSSIAN PROCESS REGRESSION[7]

Abstract

Recent advances in sensor technology and computing power allow for the generation/utilization of larger and more diverse sets of data. These developments enable the creation of data-driven models that can support real-time decision making. Such a decision aid can allow for predictive maintenance (PdM) to be undertaken on a much greater scale in manufacturing plants, where the equipment to be maintained are often complex electro-mechanical systems (e.g., rotating machinery). PdM includes data-driven prognostics and health management (PHM), and seeks to prevent unexpected downtime, reduce maintenance cost, and extend equipment service life. To enhance the performance of a prognostic model, one key task is to collect high-quality data, and in the past this has often involved using a feature extraction method to get meaningful information from a large noisy dataset. However, such methods may not handle noisy data well (e.g., machine vibration) or address measurement errors adequately. Consequently, extracted features may not represent a degradation process suitably as a machine approaches a failure or fault. Also, effects of sensor types (e.g., piezoelectric- or Micro-Electro-Mechanical System-based) on the feature extraction and prediction model have not been much explored yet. To overcome this limitation, dynamic feature extraction is proposed to mitigate the effect of noisy statistical features in a monotonic trend by introducing a statistical penalty. Then, the features extracted through the method are used to construct a health indicator (HI). With the available historical HI values, a probabilistic regression model, i.e., Gaussian process regression, may be used to forecast the time to failure (TTF) of rotating machinery with uncertainty propagation. To validate the proposed method, acceleration data (from two types of accelerometers) were collected from rotating machinery for several run-to-failure cases. The proposed method is demonstrated to provide excellent forecasts of TTF for both accelerometer types.

Keywords: Time to Failure, Dynamic Feature Extraction, Gaussian Process, Uncertainty Quantification, Rotating Machinery.

---

[7] This work is submitted to Journal of Intelligent Manufacturing.

8.1 Introduction

Rotating elements are common critical components in pieces of equipment. A variety of engineering applications (e.g., wind turbines, electrical motors, and pumps) have such rotating elements. An unexpected failure in rotating machinery can result in catastrophic consequences such as reduced productivity and economic loss [93, 94]. One might expect that such failures might be avoided or better anticipated through equipment maintenance; however, Qin et al. [143] noted that nearly half of the maintenance expenditures were wasted due to their ineffectiveness. It appears clear that equipment maintenance is a critical task in ensuring equipment availability that must also consider time and cost.

Recently, the emergence of new technologies, e.g., Internet of Things (IoT), Artificial Intelligence (AI), and smart sensors, and the advancement in computing infrastructures, e.g., graphics processing unit (GPU), have enabled a shift from conventional maintenances philosophies, e.g., breakdown and preventive maintenances, to advanced maintenance approaches, e.g., predictive maintenance, in manufacturing plants [143]. Predictive maintenance, also called condition-based maintenance, generally consists of sensors, data acquisition, signal processing (data preprocessing), fault detection, and remaining useful life (RUL) prediction[8]. As for time to failure (TTF) prediction, data-driven prognostics and health management (PHM) has gained significant attention recently in the field of machine condition monitoring research [95, 96, 97]. In data-driven PHM, a model such as machine/deep learning is trained and validated with preprocessed sensor signals obtained from run-to-failure experiments, and the trained model is deployed to make TTF predictions so as to avoid machine breakdowns [98, 99].

In order to build a PHM system for a piece of manufacturing equipment using historical machine condition data, three steps are considered: (1) data preprocessing, (2) model development, and (3) model validation. The first stage, data preprocessing (or feature engineering), is conducted to extract key features from a large amount of data using statistical measures such as root mean square (RMS), kurtosis, and variance [148]. Ren et al. [149] extracted features from vibration signals in the time- and frequency-domains using statistical measures, and then, the extracted features were used to train and test a deep neural network for RUL predictions of bearings. Ali et

[8] When a machine failure occurs, many in the literature describe this as the "end of useful life." But, of course, in many cases the machine can be repaired to put it back into service life. For such a case, the authors prefer the phrase "time to failure (TTF)" over the frequently used "remaining useful life (RUL)." For elements that cannot be repaired, such as a bearing, RUL is certainly appropriate.

al. [150] fitted the extracted features to the Weibull distribution to enhance the stability of a data-driven model. Park et al. [66] used a wavelet transform to extract fault relevant time-frequency features from vibration signals, and these were used for detecting mechanical failures in a planetary gear system.

There was a study that incorporated an optimization technique to identify the best combination of features to reflect machine health. Qin et al. [143] constructed a health indicator (HI) by identifying an optimal degradation indicator using a genetic programming algorithm. The genetic algorithm randomly created multiple mathematical combinations of the extracted features, and an optimal combination was selected based on a pre-defined fitness function. Then, the optimal combination, which is a health indicator (HI), was used in a Wiener-process model to predict RUL of rotating machinery. Guo et al. [151] extracted time, frequency, and time-frequency features from vibration signals, and the most sensitive features were selected based on the evaluation metrics (monotonicity and correlation). The selected features were synthesized to construct an HI for the RUL prediction of bearings. Bektas et al. [144] employ neural networks as a filtering method in the data preprocessing stage. Overall, the effect of the data-preprocessing on the performance of a similarity-based RUL prediction algorithm was investigated. The importance of data-preprocessing on RUL prediction was also discussed. Li et al. [152] provide a comprehensive review of literature on PHM applied to rotating machinery.

As argued in the articles cited above, the performance of a prognosis model depends on how well features are extracted from the raw data. Performance can be improved if better methods are available to distill "good" features from noisy data to construct an HI [153]. Here, a “good” feature means that it is robust to noise in the data. And a "good" HI means it is monotonically related with a mechanical degradation process [151]. Accordingly, along with model development (e.g., designing model structure, hyper-parameter tuning, and optimization), development of an HI construction method, which exhibits a monotonically increasing value as a machine degrades and approaches a failure, is an active area of research [151]. Many recent HI studies have designed an HI by synthesizing multiple statistical features directly calculated by mathematical equations. Or, features, which are sensitive to a fault, were selected based on an evaluation metric (e.g., monotonicity). These approaches may work well for a given dataset; however, they may not be adaptive and applicable to new datasets (or other mechanical systems). A feature identified as optimal may not be optimal for other mechanical systems or for different operational settings. In the worst case, an "optimal" feature may not exist, or be unknown due to a lack of knowledge.

In the case of acceleration data collected from a piece of equipment, as in the case of machine condition monitoring, a signal may include errant data due to changes in environmental conditions (e.g., temperature), fluctuation in process settings (e.g., rotational speed), sensor instabilities, etc. [154]. Features obtained from such a noisy signal will then also be noisy. Furthermore, if normally a feature directly relates to the level of degradation, this relationship may not be evident with a noisy signal and noisy feature. Thus, even in the face of monotonically increasing degradation over time, a noisy feature may not present a clear monotonic trend. Therefore, in this paper, a new feature extraction method, called dynamic feature extraction, is proposed to mitigate the effect of noisy statistical features by adding a statistical penalty while maintaining a monotonic trend. This means that, rather than identifying or selecting good features, the feature observed at the current time is updated based on a probability density function (PDF). A PDF is constructed using kernel density estimation (KDE) with historical data within a certain time window, and the PDF is dynamically updated with new time windows over time. Features extracted through the proposed method are combined to construct an HI, and they are evaluated by the evaluation metrics (i.e., monotonicity and correlation) using several run-to-failure datasets obtained from the experiments in this study. Additionally, the proposed method is compared with several popular data smoothing techniques as well as typical feature extraction method.

To demonstrate whether the proposed feature extraction method effectively works for a TTF prediction of a rotating machinery, the method is combined with a regression model. With the available historical HI values, a failure threshold was defined and TTF of rotating machinery was estimated using a probabilistic regression model. In this study, GP (Gaussian process) regression is employed for the TTF prediction with the uncertainty quantification of the prediction. GP is a Bayesian machine learning method and a non-parametric regression model. It models unknown functions with available data and describes the functions using probability measures on the function space. GP regression is known to have numerous advantages such as non-parametric and adaptive hyper-parameter learning, and predictions interpreted with a probability distribution [66]. Thus, the method is appropriate for non-linear and time-varying data [155]. Also, unlike the artificial neural network model, which requires a lot of training data and extensive parameter tuning, GP works well with a small dataset, less depends on parameter tuning, and also it's computationally more affordable. These advantages make it more suitable for our application. In this study, GP regression is trained with historical HI values computed by dynamic feature extraction. Then, TTF predictions are made on several run-to-failure datasets in consort with uncertainty quantification (i.e., aleatory uncertainty and epistemic uncertainty).

To validate the proposed method, acceleration data was collected during several run-to-failure trials on in-service vacuum pumps. In our data collection, two types of accelerometers, (1) piezoelectric and (2) micro-electromechanical systems (MEMS) accelerometers, were deployed to collect data. The two types are distinguished by their sensing mechanism. In a piezoelectric accelerometer, a piezoelectric sensing element induces an electric charge proportional to the applied acceleration; for a MEMS accelerometer, movement of mass creates a capacitance change under an applied acceleration [156]. A piezoelectric accelerometer is the most common sensor used for vibration measurement/analysis (e.g., diagnostics of rotating machinery) due to its large frequency response range, stable sensitivity, and low noise, but such sensors are expensive. Meanwhile, MEMS accelerometers are low-cost and have been widely used for many applications (e.g., motion detection) due to their low cost. However, MEMS accelerometers are only suitable for low-frequency range of vibrations and have high noise and small measurement range. Thus, accelerometer type should be selected based on the application of interest (e.g., frequency range, sensitivity, measurement range, and budget). In the machine health monitoring research field, the effect of sensor types on the model's predictions has not been well reported yet although "how to collect data" will significantly contribute to the model's prediction. In this paper, the proposed method is not only validated using the experimental data obtained from the different types of sensors, but also the effects of the sensor type on the model's performance are investigated. The key contributions of the paper are as follows:

- A new feature extraction method (called dynamic feature extraction) is proposed to mitigate the effect of noisy statistical features by introducing a statistical penalty,
- GP regression predicts TTF with an uncertainty quantification, and the TTF is interpreted by a probability distribution,
- The proposed method is validated using experimental data collected during several run-to-failure trials on in-service vacuum pumps, and
- the effects of accelerometer type (piezoelectric and MEMS) on the model's predictive capability are investigated.

All data collected from the experiment described in this paper will be available via the Purdue Laboratory for Sustainable Manufacturing (LSM) [133].

8.2 Motivation

As has been noted, predictive maintenance of manufacturing equipment requires sensed information on the equipment state in order to assess its wellness and forecast upcoming failures. When a large amount of machine condition data (e.g., vibration) is collected, say at a high sampling rate for many sensors, one strategy for managing the data is to first extract features from the data. Common features that may be extracted include the root mean square (RMS), kurtosis, mean, variance, and statistical distance metrics. As mentioned before, it is important to extract "good" features from noisy data to construct an HI because the goodness of such an indicator may determine the performance of a data-driven prognostic model.

Almost always, the state of a machine will undergo increasing degradation over time. And, it is to be expected that metrics (e.g., vibration) that are proportional to the degree of degradation would likewise demonstrate monotonical behavior. However, owing to noise in sensor data, features extracted from sensor signals using statistical measures may not present a clear monotonic trend. Although the features may generally tend to increase or decrease over time, their values may fluctuate, or even display large, sudden departures from stable behavior (outliers). The magnitude of the fluctuations and the frequency of outliers/spikes may vary depending on the equipment being monitored and the sensors employed, but some unavoidable factors such as changing process parameters and measurement errors always exist. Although the details on the datasets used for this study will be explained in detail in Section 8.4, let us use a part of datasets to illustrate the problem and discuss our proposed method.

Let $\boldsymbol{X}_T = \{x_1,\, x_2,\, \dots ,\, x_n\} \in \mathbb{R}^n$ be a sample (in this case, a time series) collected at time $T \in \{1,\, 2,\, \dots ,\, t\}$, where n is the number of data points in one sample. Then, a set of features at time T, $\boldsymbol{G}_T = \{g_1,\, g_2,\, \dots ,\, g_k\} \in \mathbb{R}^k$ can be obtained using feature mapping functions, $\phi_k(\cdot)$, where k is the number of features extracted from one sample. For t samples collected over time, the sample data may be transformed into features, $G : \boldsymbol{X}_{1:t} \in \mathbb{R}^{n\times t} \rightarrow \phi_{1:k}(\boldsymbol{X}_{1:t}) \in \mathbb{R}^{k\times t}$. This process, when n=12,000, k=4, and t=1438, may be visualized in Figure 8.1 using data collected with a data collection rate within a sample, time duration of a sample, and time between samples of 12 kHz, 1 second, and 1 hour, respectively. As is evident from Figure 8.1, the data within a sample seem to follow a Gaussian distribution with zero mean. The distribution graphs in the lower left-hand portion of the figure also seems to show that the variance tends to increase over time (high amplitudes of vibration are observed as a machine failure approaches).

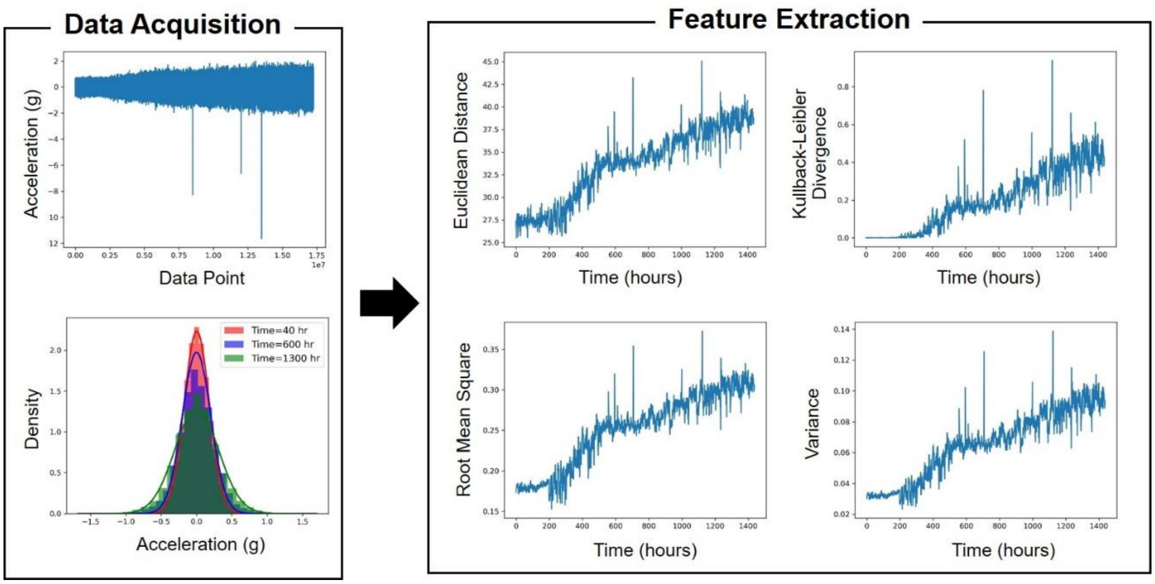


Figure 8.1 Extracting features from a run-to-failure dataset.

In Figure 8.1, the behavior of four features, Euclidean distance (ED), Kullback-Leibler divergence (KLD), root mean square (RMS), and variance (VAR), are displayed over time. These features were extracted using Eqs. (8.1)-(8.4) and will be the features evaluated in this study. ED is a distance measure quantifying a relative difference between two datasets. Thus, the relative differences (i.e., distance) from the sample, $\boldsymbol{X}_1 = \{x_1,\ x_2,\ \ldots,\ x_n\}$, obtained from an initial condition ($T = 1$) to the samples, $\boldsymbol{X}_T = \{x'_1,\ x'_2,\ \ldots,\ x'_n\}$, collected after the initial condition ($T = \{2,\ 3, \ldots,\ t\}$) are plotted in the figure. KLD (i.e., $\mathrm{KL}(p \,||\, q)$) computes a difference between two probability distributions for a random variable (see Eq. (8.2)). The divergence from the probability distribution for an initial condition, *p*, to the probability distribution for some subsequent condition, *q*, was also calculated. Assuming that two probability distributions are Gaussian with $p(x) \sim N(\mu_1, \sigma_1^2)$ and $q(x) \sim N(\mu_2, \sigma_2^2)$, where x is a random variable, the divergence equation can be rewritten as a function of the means and standard deviations of the probability distributions. RMS is the square root of the average of the squared amplitudes, thus it presents an averaged magnitude. Variance measures the statistical dispersion of a given sample.

$$f_{ED} = \sqrt{\sum_{i=1}^{n}(x_i' - x_i)^2}, \tag{8.1}$$

$$f_{KLD} = \sum p(x)\log\frac{p(x)}{q(x)} = \log\frac{\sigma_2}{\sigma_1} - \frac{1}{2} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} \text{(if } p \text{ and } q \text{ are Gaussian)}, \tag{8.2}$$

$$f_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}, \tag{8.3}$$

$$f_{VAR} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2. \tag{8.4}$$

As shown in Figure 8.1, all extracted features fluctuate over time and display many sudden departures from stable behavior (this behavior is also observed in other statistical measures which are not presented in this paper and the monotonicity of these features will be further discussed in Section 8.5.1). Consequently, this may result in the construction of poor HI, which may result in a poor predictive model, which provides inaccurate TTFs. To handle this problem, a simple and fast feature extraction method is proposed, which is discussed in the next section.

8.3 Methodology

8.3.1 Dynamic Feature Extraction

Kernel density estimation is a way of approximating an unknown probability density function (PDF) for a given dataset, and may be mathematically written as

$$PDF(x) = \frac{1}{Nh}\sum_{i=1}^{N} K(\frac{x - x_i}{h}), \tag{8.5}$$

where N is the number of data points in a given dataset, h is the bandwidth (also known as a smoothing parameter), and K is the selected kernel function. A probability density function can be constructed by (1) placing a selected kernel function, K, on each data point, (2) summing all generated kernel functions, and (3) dividing by Nh to create a valid probability function (so that the integral of the PDF equals to 1). Silverman [157] proposed a Gaussian kernel function and an optimal bandwidth as shown in Eqs. (8.6) and (8.7),

$$K(x) = \exp(-x^2/2)\big/\sqrt{2\pi}\,, \tag{8.6}$$

$$h = (4\hat{\sigma}^5/3N)^{0.2}\,, \tag{8.7}$$

where $\hat{\sigma}$ is the standard deviation of a given dataset. In the proposed method, the KDE method approximates a PDF using a certain length of historical features, and the PDF is updated as new data becomes available.

Table 8.1 Pseudo code of dynamic feature extraction algorithm.

Dynamic Feature Extraction Algorithm
Require: $\boldsymbol{\alpha}$ = [α_1, α_2, α_3]: penalty parameters, l: window size where $l \leq t$-1, m = number of samples drawn from PDF, n= number of datapoints in one sample, t = data collection time (hour)
Function ECDF(x)
1: dataset **=** [x_1, x_2, …, x_m]
2: **return (**number of observations **≤** x**)** ÷ m: empirical CDF of a given dataset: Eq. (8)
Function iECDF(u)
1: assert u >= 0 and u <= 1
2: x^* **=** argmin $ECDF(x) - u$: Brent’s method is used to find a unique solution $u \in [0, 1]$
3: **return** x^*
For T=1+l **to** t **do**
1: **input**: $\boldsymbol{D}_T$ = [$\boldsymbol{X}_{T-l}$, $\boldsymbol{X}_{T-l+1}$, …, $\boldsymbol{X}_{T-1}$, $\boldsymbol{X}_T$] $\in \mathbb{R}^{n\times l}$: historical sample data with window size of l at current time T
1: $\boldsymbol{Z}_T$ = [G_1, G_2, …, G_{l-1}, G_l]: feature extraction through $G: \boldsymbol{D}_T \in \mathbb{R}^{n\times l} \to \phi(\boldsymbol{X}_{T-l:T}) \in \mathbb{R}^l$
2: $PDF_T \leftarrow$ KDE($\boldsymbol{Z}_T$): construct a PDF using KDE (Eq. (5))
3: **for** v = 1 **to** m **do** $V_T^v \sim PDF_T$: draw m samples from PDF_T
4: dataset $\leftarrow \boldsymbol{V}_T$
5: **for** u =1 **to** l **do**
6: **if** Z_T^u < iECDF(0.5): $H_T^u \leftarrow Z_T^u$ +(iECDF(0.5) – Z_T^u) × α_1
7: **else if** iECDF(0.5) ≤ Z_T^u < iECDF(0.95): $H_T^u \leftarrow Z_T^u$ – (Z_T^u - iECDF(0.5)) × α_2
8: **else**: $H_T^u \leftarrow Z_T^u$ – (Z_T^u – iECDF(0.5)) × α_3
9: **end for**
10:**output**: $\boldsymbol{H}_T$: updated features at time T
End for
Output: $\boldsymbol{H}$ = [$\boldsymbol{H}_1$, $\boldsymbol{H}_2$ …, $\boldsymbol{H}_t$]: updated features for t hours

To calculate a new feature using a PDF approximated by KDE, first, several samples are drawn from a probability distribution. With the m samples (x_1, x_2, …, x_m), an empirical cumulative distribution function (ECDF) for a random variable x can be calculated as follows,

$$\mathrm{ECDF}(x) = \frac{\text{Number of Observation} \leq x}{m} = \frac{1}{m}\sum_{i=1}^{m} 1_{[x_i,\,\infty]}(x), \tag{8.8}$$

where $1_A(x)$ is the indicator function; $1_A(x) = 1$ if $x \in A$ or 0 if $x \notin A$ for a subset A of a set X. Once an ECDF is computed, an inverse ECDF (iECDF), which outputs $\hat{x} = \mathrm{ECDF}^{-1}(u) = \mathrm{iECDF}(u)$, can be obtained for any $u \in [0,\ 1]$. Then, a relative distance between the 50th percentile (u=0.5) and an originally extracted feature value is used to update the feature value (i.e., new feature). The method is designed to give a higher penalty to more extreme feature values (largely deviations from a distribution). In the method, the 50th percentile (median) is used rather than the mean, since the median is less sensitive to noise or outliers. In the iECDF, Brent's method (root-finding algorithm) is adopted to find an optimal solution, x^*, using:

$$x^* = \arg\min_x ECDF(x) - u. \tag{8.9}$$

The algorithm for dynamic feature extraction is presented in Table 8.1. As described in the pseudo code, a PDF is updated over time as new feature values (next time-step) become available, and the originally extracted features are updated using the relative distance mentioned above. In order to make the method more robust and to ensure monotonic behavior over time, different penalty parameters (α_1, α_2, α_3) are introduced, and they are adopted based on a relative distance and a direction between an input feature and 50th percentile value (iECDF(0.5)). For example, a higher penalty value is applied for features whose values are greater than or equal to iECDF(0.95) than for values between iECDF(0.5) and iECDF(0.95). This will help to mitigate any effect of extreme values on the features, and preserve a monotonic trend. To quantify the degree of monotonicity the evaluation metrics of Eqs. (8.10) and (8.11) may be used [151]:

$$\text{Monotonicity}=\left|\frac{\text{number of } d/dx>0}{h-1}-\frac{\text{number of } d/dx<0}{h-1}\right|, \tag{8.10}$$

$$\text{Corr} = \frac{\left|\sum_{t=1}^{h}(x_t-\bar{x})(L_t-\bar{L})\right|}{\sqrt{\sum_{t=1}^{h}(x_t-\bar{x})^2\sum_{t=1}^{h}(L_t-\bar{L})^2}}, \tag{8.11}$$

where h is the number of observations, Corr is a correlation between features, x, and observed time, L, which measures a degree of linear correlation. Monotonicity metric is used in the study because the useful life of a machine monotonically decreases as a function of time in a run-to-failure setting. Corr metric is used to measure the relationship between the features and the degradation process. These two metrics are commonly used for the feature selection and the evaluation of HI [103, 104, 110].

After calculating the features, the features are interpreted using principal component analysis (PCA) to obtain an HI. PCA is known as a dimensional reduction technique, in which observed data are mapped into new axes (eigenvectors from the PCA) [41]. In this paper, the value associated with the first principal component (i.e., the value projected on the first principle component) is used as the HI. In the next section, Gaussian process (GP) regression, which predicts a TTF along with an estimate of its uncertainty, is discussed.

8.3.2 Gaussian Process (GP) Regression

GP regression is a nonparametric Bayesian approach to develop a model for an unknown function, $f(\cdot)$, which represents the relationship between inputs and outputs. The method defines a probability distribution over a function space, and initially $f(\cdot)$ is assumed to follow a prior distribution, and any finite collection of observations (i.e., function values) follow a joint Gaussian distribution [159]. GP utilizes a mean function ($m(\cdot)$) and a covariance function ($k(\cdot,\cdot)$), which describe the central tendency and uncertainty of the function, $f(\cdot)$, respectively. Mathematically, GP is defined as

$$f(\cdot) \sim p(f(\cdot)) = \mathrm{GP}(m(\cdot),\ k(\cdot,\cdot)). \tag{8.12}$$

where $m(x) = \mathrm{E}[f(x)]$ and $k(x,x') = \mathrm{E}[(f(x) - m(x)(f(x') - m(x')]$. Based on the dataset, a different covariance fusion or a combination of multiple covariance functions can be used [155]. In this paper, the sum of four covariance functions (a squared exponential kernel, a linear kernel, a white noise kernel, and a rational quadratic kernel functions) are used, where

$$k_{SE}(x,x') = s^2 \exp\left[-\frac{(x-x')^2}{2l^2}\right], \tag{8.13}$$

$$k_{linear}(x,x') = (x - c)(x' - c), \tag{8.14}$$

$$k_{white}(x,x') = s^2 I, \tag{8.15}$$

$$k_{RQ}(x,x') = s^2(1 + \frac{(x-x')^2}{2\alpha l^2})^{-\alpha}, \tag{8.16}$$

$$k(x,x') = k_{SE}(x,x') + k_{linear}(x,x') + k_{white}(x,x') + k_{RQ}(x,x'), \tag{8.17}$$

and s, c, l, and α are hyper parameters, that can be "learned" through maximum likelihood estimation. The covariance functions were selected based on trial and error experiment using experimental data, which will be explained in next section.

Before seeing any data, Eq. (8.12) is a probabilistic measure of $f(\cdot)$ using prior beliefs about the function (for this, zero mean (m=0) is often used). Using a given dataset, D, a function ($f(\cdot)$) can be learned. Based on Bayes's rule, a posterior state of knowledge, $p(f(\cdot),\ D) \propto p(D \mid f(\cdot))P(f(\cdot)),$ can be calculated by the product of likelihood ($p(D \mid f(\cdot))$) and the prior estimate ($P(f(\cdot))$). More details are provided below.

Suppose there is a dataset, $D = \{\boldsymbol{X},\ \boldsymbol{Y}\}$, which includes observed input and output data, $\boldsymbol{X} = [x_1,\ x_2,\ ...,\ x_n]$ and $\boldsymbol{Y} = [y_1,\ \mathrm{y}_2,\ ...,\ \mathrm{y}_n]$ (D is a training dataset). Then, GP regression models the unknown function, $f(x_m)$, where $m \in \{1,\ 2,\ ...\ ,\ n\}$. To improve numerical stability, zero mean Gaussian noise is added; GP regression for a single observation is then:

$$y_m = f(x_m) + e_m,\ \text{where } e_m \overset{\text{i.i.d.}}{\sim} N(0,\ \sigma^2), \tag{8.18}$$

$$y_m \mid f(x_m) \sim N(f(x_m),\ \sigma^2), \tag{8.19}$$

where y is the noisy measurement of $f(x)$ and σ^2 is the variance of the noise. In the model, a GP models $f(x_m)$, which is a latent function. Then, for an arbitrary input dataset, $\boldsymbol{X}^* = [x_1^*,\ x_2^*,\ ...,\ x_p^*]$ (i.e., testing dataset), GP makes the following predictions: $\boldsymbol{f}^* = [f^*(x_1^*),\ f^*(x_2^*),\ ...,\ f^*(x_p^*)]$.

To derive a posterior estimate, $p(\boldsymbol{f}^* \mid D)$, by the definition of the GP, the joint probability density of $\boldsymbol{y}$ and $\boldsymbol{f}^*$ is defined as [158]:

$$p(y_{1:n}, f_{1:p}^* \mid x_{1:n}, x_{1:p}^*) = N(\begin{pmatrix} y_{1:n} \\ f_{1:p}^* \end{pmatrix} \Big| \begin{pmatrix} \boldsymbol{m}(x_{1:n}) \\ \boldsymbol{m}(x_{1:p}^*) \end{pmatrix}, \begin{pmatrix} \boldsymbol{K}(x_{1:n}, x_{1:n}) + \sigma_n^2 \mathbf{I} & \boldsymbol{K}(x_{1:n}, x_{1:p}^*) \\ \boldsymbol{K}(x_{1:p}^*, x_{1:n}) & \boldsymbol{K}(x_{1:p}^*, x_{1:p}^*) \end{pmatrix}) \tag{8.20}$$

which is a joint Gaussian distribution and $\mathbf{I}$ is the identity matrix. Then, using the sum and Bayes' rules, the predicted distribution (i.e., posterior predictive distribution) is as follows:

$$p(f_{1:p}^* \mid x_{1:p}^*, x_{1:n}, y_{1:n}) = N(f_{1:p}^* \mid \boldsymbol{m}_n(x_{1:p}^*),\ \boldsymbol{K}_n(x_{1:p}^*, x_{1:p}^*)). \tag{8.21}$$

Then, the posterior GP is [158]

$$f(\cdot) \mid x_{1:n}, f_{1:n} \sim \mathrm{GP}(m_n(\cdot), k_n(\cdot,\cdot)) \tag{8.22}$$

where m_n and k_n are posterior mean and covariance functions, and computed as

$$m_n(x) = m(x) + \boldsymbol{k}(x, x_{1:n})(\boldsymbol{K}(x_{1:n}, x_{1:n}) + \sigma^2\mathbf{I})^{-1}(y_{1:n} - \boldsymbol{m}(x_{1:n})), \tag{8.23}$$

$$k_n(x, x') = k(x, x') - \boldsymbol{k}(x, x_{1:n})(\boldsymbol{K}(x_{1:n}, x_{1:n}) + \sigma^2\mathbf{I})^{-1}\boldsymbol{k}^T(x, x_{1:n}). \tag{8.24}$$

where $\boldsymbol{k}$ is a $1\times n$ row vector and $\boldsymbol{K}$ is $n\times n$ matrix. As seen in Eqs. (8.22) and (8.23), the mean and covariance function depend on the observed data. Thus, the belief about the model is encoded after seeding with the training data. Then, the predictive distribution of the function at a test location, X^*, is

$$p(f(X^*)\,|\,D) = N(f(X^*)\,|\,m_n(X^*),\ \sigma_n^2(X^*)) \tag{8.25}$$

where $\sigma_n^2(X^*) = k_n(X^*, X^*)$ (i.e., predictive variance). Accordingly, the predictive distribution of the measurement, y^*, can be written as

$$p(y^*\,|\,X^*, D) = N(y\,|\,m_n(X^*),\ \sigma_n^2(X^*) + \sigma^2) \tag{8.26}$$

As illustrated above, the GP regression outputs the probability measures on the function space by (1) assigning a prior estimate first, and (2) updating with the training dataset. Therefore, both a mean prediction and an uncertainty for that prediction are provided. In this paper, the uncertainty is characterized using 2.5% and 97.5% quantiles (i.e., the lower and upper bounds). This is done for the epistemic uncertainty (predictive variance) and the full uncertainty (i.e., sum of predictive variance and noise variance, aleatory uncertainty). These uncertainty bounds are $m_n(X^*) \pm 1.96\sqrt{\sigma_n^2(X^*)}$ and $m_n(X^*) \pm 1.96\sqrt{\sigma_n^2(X^*) + \sigma^2}$, respectively. In the next section, details on the experiment and data acquisition setup for the two types of accelerometers (piezoelectric and MEMS) are discussed.

8.4 Experiment for Run-to-failure Data Collection

8.4.1 Data Acquisition Setup

Three vacuum pumps (Edwards, QDP80) in the Birck Nanotechnology Center at Purdue University (Indiana, USA) were selected to be monitored. Each pump is connected to a three-phase power supply with 30 amps and 208-220 VAC. Process cooling water at 0.3 GPM is employed to regulate the temperature in the pumps. The pumps run 24hrs/day, 7days/week, and 48-50 weeks per year. They are used for semiconductor furnaces, and they are stopped only for regular maintenance (i.e., preventive maintenance).

In Figure 8.2, the data acquisition setup on one pump is illustrated (identical setups for all pumps) as well as the hardware and software requirements of piezoelectric and MEMS accelerometers for the data collection. To measure the pumps' vibrations, three piezoelectric accelerometers (PCB PIEZOTRONICS, one J356A45 and two TLD352A56) and three MEMS accelerometers (Adafruit, ADXL345) were mounted on the pumps, i.e., two accelerometers (piezoelectric and MEMS) were attached to each of the three vacuum pumps. The direction of the measured acceleration is radial, which is normal to the axis of rotation of the pump shaft.

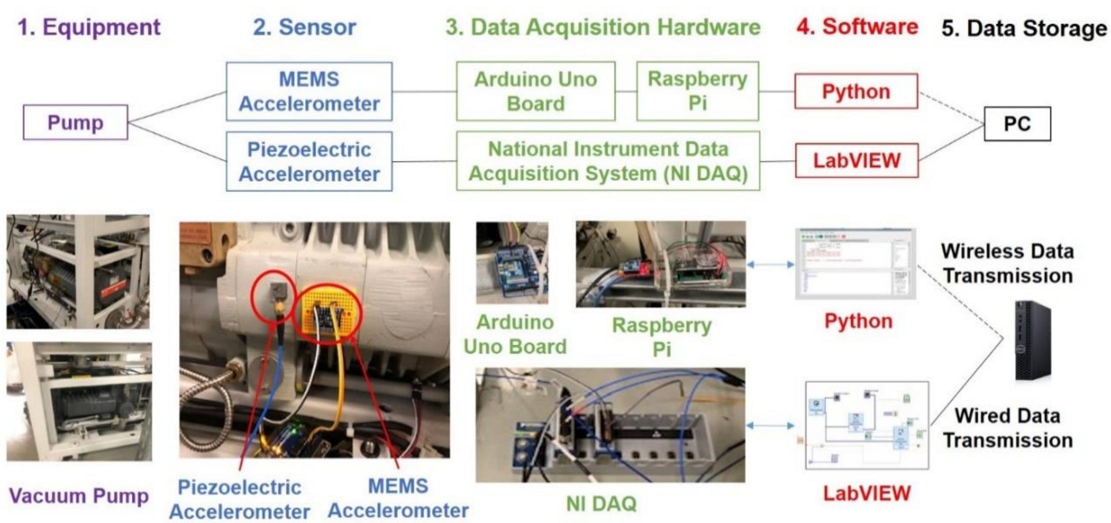


Figure 8.2 Data acquisition setups for piezoelectric and MEMS accelerometers.

For the piezoelectric accelerometers, the data acquisition was enabled by a National Instruments data acquisition system (NI DAQ) including cDAQ-9178 CompactDAQ chassis and NI-9234 C Series Sound and Vibration Input Module. Acceleration signals were stored in a PC using LabVIEW software with a sampling frequency of 12 kHz for each sensor. Digital data were collected for one second every hour.

For the MEMS accelerometers, Arduino Uno boards and a Raspberry Pi were used as a data acquisition hardware, whose costs are substantially lower than the piezo-based accelerometer. The MEMS accelerometers support I2C serial interface, thus the Arduino boards communicate with the MEMS accelerometers using the I2C protocol. The data collected from the boards were sent to the Raspberry Pi through serial communication with a sampling frequency of 545 Hz for each sensor. Digital data were obtained for one second every hour with the same sampling time and interval as for the piezoelectric accelerometers. Because the Raspberry Pi does not have enough storage for long-term data collection, all data collected from the MEMS accelerometers were uploaded to a personal cloud server using python programming language through a Wi-Fi connection. Then, the uploaded data were downloaded to a PC. The cost comparison between the piezoelectric and MEMS data acquisition setups is presented in Table 8.2.

While the three pumps have an identical operational and physical setup, their health conditions are not in the same at the beginning of the data collection since the lengths of utilization time are different (the most recently rebuilt/refurbished pump probably has a longer time left until a failure). The health condition of each pump at the beginning of the data collection were determined based on expert advice (facility maintenance engineer). A “good” estimate of the health condition of a vacuum pump is the color of the bearing grease, which is discussed in the next section.

8.4.2 Health Estimation of Vacuum Pumps

One easy way to estimate the health condition of a vacuum pump without dissembling the entire pump is by observing the color of the bearing grease (a perfluoropolyether (PFPE) lubricant). As shown in Figure 8.3a, the color can be observed by opening the bearing end-cover. Roughly speaking, the brightness of the grease color in the end cover cavity can be used to estimate the current health of the vacuum pump. Similar to gray scale, the grease color (originally white) becomes darker as more debris particles are generated from bearing wear (i.e., the grease becomes contaminated). Figure 8.3b shows different grease conditions, with indices ranging from 1 to 4 (a higher index number indicates a worse health condition), but the grease color associated with index 4 can be even darker in more extreme conditions. Here, the indices are defined based on the visual inspection.

Every six months, the bearing grease on the bearing covers is checked for quality and most of the grease is manually replaced at this time. Based on past experience, the longer the interval between major rebuilds correlates to a quicker progression of darkening between the six-month inspection periods. This could be explained by the shedding of contaminants and debris from mechanical wear between major service intervals. As the wear on the components increases and component shapes slightly change (fit between components deteriorates), the rate at which material sheds can increase.

Table 8.2 Cost comparison between piezoelectric and MEMS accelerometers setup (the number in parentheses is the quantity).

Category	Data acquisition setup	
	Piezoelectric accelerometers	MEMS accelerometers
Sensor	PCB sensor: $2,525 (3)	ADXL 345: $29.97 (3)
Data acquisition hardware	NI chassis: $1,407 (1)	Arduino board: $66 (3)
	NI module: $2,062 (1)	Raspberry Pi: $40.97 (1)
	Shielded Wire: $237 (3)	Jumper wire: $5.79 (1)
		Breadboard: $7.98 (3)
		32GB SD card: $8.29 (1)
Software	Free (university license)	Free
Total costs	**$6,231**	**$159**

Based on the grease colors in the end-cover cavity of each pumps, the initial health conditions (more specifically, the health conditions at the beginning of the data collection) of the three pumps were determined. Since the three pumps were purchased and rebuilt at different times, they have different index values. For example, pump 1, which was the most recently rebuilt, showed an almost-new condition, while pumps 2 and 3, which were rebuilt about six years ago, displayed higher indices. Next, acceleration signals obtained from the vacuum pumps are visualized and described.

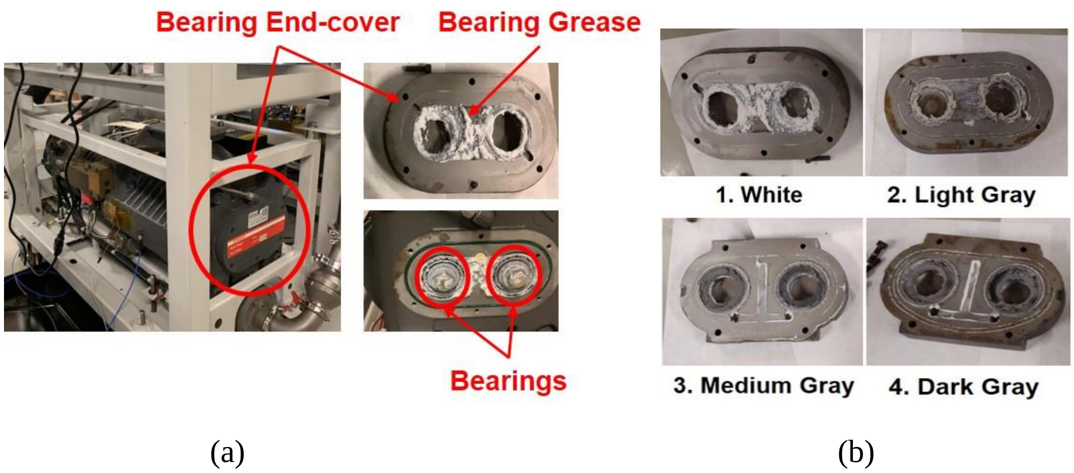
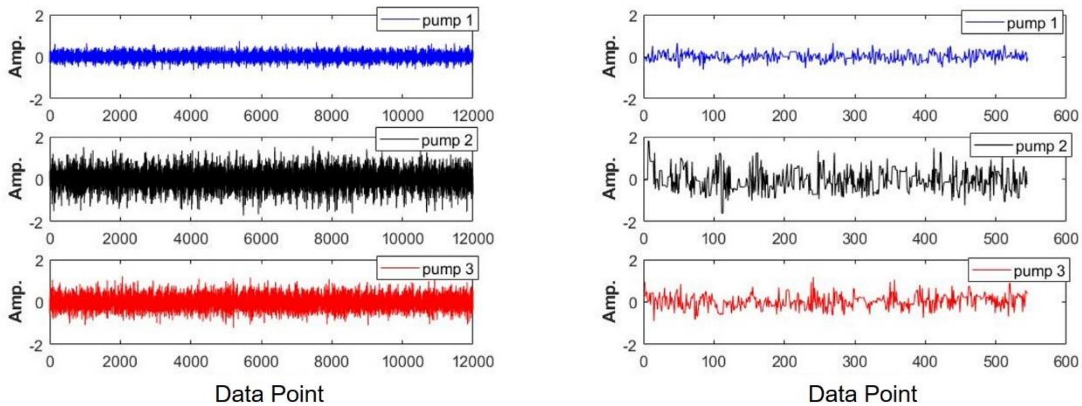


Figure 8.3 Bearing grease index to estimate the health condition of the pump; (a) bearing location and (b) grease indices.



(a) (b)

Figure 8.4 One sample collected from (a) piezoelectric accelerometer (12,000 data points) (b) MEMS accelerometer (545 data points).

8.4.3 Visualization of Experimental Data

Figure 8.4 visualizes one sample of data collected from each pump near the start of the testing process. As shown in the figure, relatively small amplitudes of the acceleration are observed in pump 1, whose initial condition was almost-new. On the other hand, pumps 2 and 3, which have been in service for several years (since being rebuilt), generate relatively higher amplitudes of acceleration during operation. While collecting data over the 10-month period of monitoring, pumps 2 and 3 experienced failures once and twice, respectively. After each failure, they were sent to a repair facility and refurbished. The refurbished pumps were monitored again once they were again installed at the facility using the same data acquisition setup. Figure 8.5 visualizes several selected function-to-failure[9] datasets collected over the 10-month period, with details for each dataset summarized in Table 8.3.

[9] In all, roughly 7000 hours of data were collected on each of the three pumps using the two types of accelerometers. Every hour a sample was collected from each accelerometer (sampling rates of 12,000 Hz for piezo and 545 Hz for MEMS). These data were all stored and could be reviewed as necessary. In practice, the pumps were operated using a run-to-failure approach. When a failure occurred, the data leading up to the failure was analyzed. The data records in Figure 8.5 do not show the entire "run-to-failure," but rather the portion of the signal in the days leading up to the failure, i.e., "function to failure."

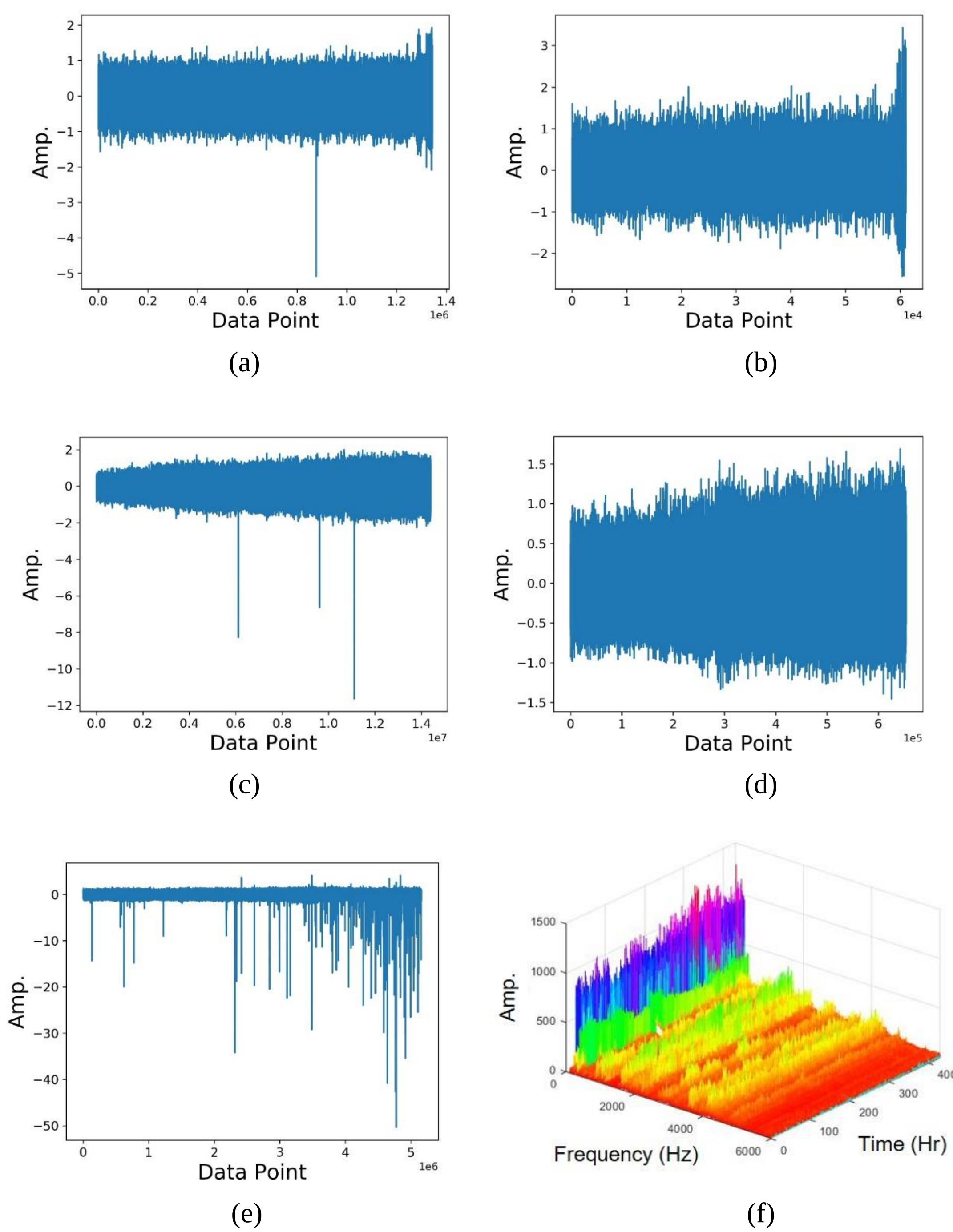


Figure 8.5 Visualization for run-to-failure data; (a) pump 2-piezoelectric (dataset #1), (b) pump 2-MEMS (dataset #2), (c) pump 3-piezoelectric (dataset #3), (d) pump 3-MEMS (dataset #4), (e) pump 3-piezoelectric (dataset #5), and (f) waterfall plot of dataset #5.

Table 8.3 Function-to-failure dataset.

Name	Sensor Type	Failure Type	Length (hrs.)	Number of data points
dataset #1	piezoelectric	rapid failure	112	1,344,000
dataset #2	MEMS	rapid failure	112	61,040
dataset #3	Piezoelectric	gradual failure	1,200	14,400,000
dataset #4	MEMS	gradual failure	1,200	654,000
dataset #5	Piezoelectric	gradual failure	430	5,160,000

Figure 8.5a (piezoelectric) and Figure 8.5b (MEMS) display the data collected for pump 2 over a 112 hour period. A sudden pattern change was observed in the acceleration signals for both sensor types several hours before the failure, indicating that a rapid failure was occurring in the pump. Unlike the preceding dataset, Figure 8.5c (piezoelectric) and Figure 8.5d (MEMS) present a gradual failure in pump 3, with data over 1200 hours shown. Figure 8.5c and Figure 8.5d were obtained just after pump 3 had been refurbished, and pump 3 was in a new condition at the beginning of data shown in the figures. Figure 8.5e (piezoelectric) exhibits another dataset (430 hours in length) that ultimately ends with a failure for pump 3.

Each sample over time was transformed into the frequency domain using a fast Fourier transform (FFT). Figure 8.5f shows a waterfall plot of the FFT for dataset #5. As the pump was close to the end of life, the eccentric acceleration signals were observed more frequently. The accumulation of the eccentric behavior of the pump may lead to a breakdown. In the waterfall plot (Figure 8.5f), the amplitudes for the different frequency components in the signal change over time; in some cases, these amplitudes increase as the pump approaches failure. Frequencies that display such behavior may be associated with certain rolling elements in the pump. The amplitudes associated with such frequencies may be good estimators of the health condition of the associated rolling element (e.g., bearing or rotor). This can be further investigated with the help of more advanced frequency analysis. Details regarding frequency analysis for multi fault detection in rotating machinery can be found in Lee et al. [160]. No usable function-to-failure data was obtained during the 10 months in pump 1 (failure did not occur). Therefore, the data visualized in Figure 8.5 (i.e., datasets #1-5) are used to validate the proposed method.

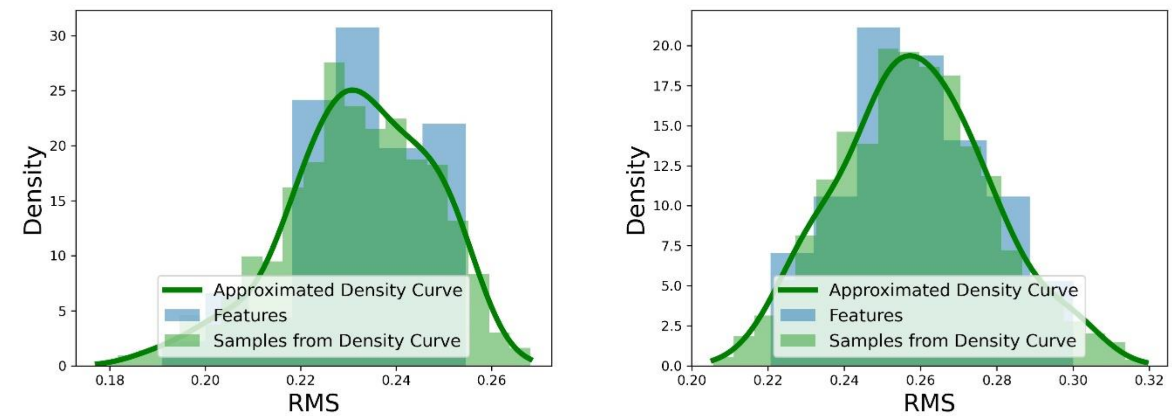
8.5 Result

In the section, the dynamic feature extraction method is applied to the experimental data (Section 8.5.1) and they are evaluated using the evaluation metric. The extracted featured are processed to obtain HI values, and the HI values are applied to a GP regression model for TTF forecasting (Section 8.5.2). To implement the proposed method, Python program language along with several packages (e.g., numpy, scipy, and Gpy) were used.

8.5.1 Updating Features Using Dynamic Feature Extraction

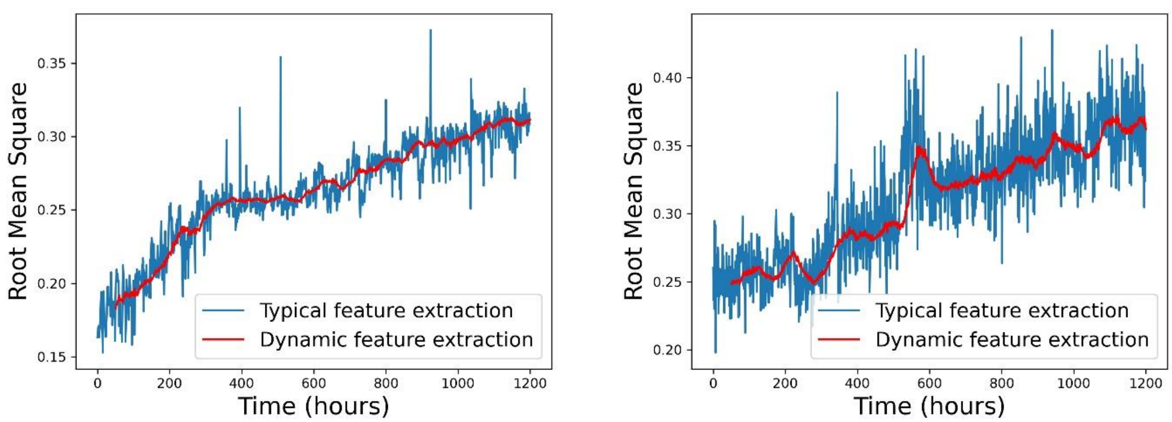
As has been noted, the five datasets noted in Table 8.3 will be employed to assess the proposed method. As a first step, let us compare the proposed feature extraction method with a typical feature extraction approach using the evaluation metric (monotonicity). As described in Section 8.3.1, a PDF at time T is constructed by KDE using historical feature data with the window length of l. In the method, following parameters are used: $\alpha 1=0.95$, $\alpha 2=0.80$, $\alpha 3=0.99$, l=20, m=1000, t= data collection time (Length in Table 8.3), n=12,000 for piezoelectric, and n=545 for MEMS. To visually show the results, dataset #3 and dataset #4 ((c) and (d) in Figure 8.5) are selected. As examples, Figure 8.6 shows the approximated density curves, the histograms of the RMS features, and the histograms of samples drawn from the approximated density curves (here, RMS features between 200 and 250 hours are used). As seen in the figure, the KDE adequately approximates the original distribution. From the approximated density curve, 1,000 samples are drawn, and they are used to calculate the ECDF. Then, the difference between the 50th percentile and an originally extracted feature value is used to compute a new feature as explained in section 8.3.1.

In Figure 8.7, the comparisons between the typical feature extraction and dynamic feature extraction are shown using the data obtained from piezoelectric (Figure 8.7a) and MEMS accelerometers (Figure 8.7b). In both cases, the dynamic feature extraction method effectively mitigates the effect of extreme values on the features and preserve a monotonic trend, which means that less fluctuations and less sudden spikes are observed while maintaining the desired overall increasing trend. A comparison of Figure 8.7a and Figure 8.7b shows that the features extracted from data obtained from the MEMS accelerometer have larger fluctuations than the features extracted from the data obtained from the piezoelectric accelerometer. This makes sense considering the difference in measurement capability between the two accelerometers; the piezoelectric accelerometer setup has (1) relatively low measurement noise/error and (2) internal signal conditioning equipped in NI DAQ, which is also a reason why the piezoelectric sensor is more expensive than the MEMS accelerometer.



(a) (b)

Figure 8.6 Approximated density curves (PDF) using KDE, histograms of RMS features, and histograms of samples drawn from the approximated density curves; (a) dataset #3 (piezoelectric) and (b) dataset #4 (MEMS).



(a) (b)

Figure 8.7 Comparison between typical feature extraction and dynamic feature extraction (proposed method); (a) dataset #3 (piezoelectric) and (b) dataset #4 (MEMS).

Table 8.4 Monotonicity and Corr of features extracted from different methods; the numbers in each box represent Monotonicity/Corr (bold numbers are the highest).

Name	Feature	Typical	MA	ES1	ES2	Dynamic (Our)
Dataset #3	ED	0.0676/0.9293	0.0375/0.9172	0.0218/0.9425	0.0542/0.9508	**0.1043**/**0.9697**
	KLD	0.3311/0.9042	0.1009/0.8921	0.2312/0.9137	0.3628/0.9200	**0.6180**/**0.9427**
	RMS	0.4862/0.9240	0.1426/0.9127	0.3668/0.9359	0.5079/0.9436	**0.8198**/**0.9629**
	VAR	0.7381/0.9356	0.2394/0.9232	0.6298/0.9500	0.7364/0.9594	**0.9633**/**0.9767**
Dataset #4	ED	0.0125/0.8055	0.0134/0.9069	0.0042/0.9260	0.0425/0.8173	**0.0492**/**0.9571**
	KLD	0.0125/0.8455	0.1022/0.9160	0.1526/0.9324	0.1927/0.8791	**0.3161**/**0.9535**
	RMS	0.0242/0.8419	0.1541/0.9165	0.1910/0.9340	0.2577/0.8871	**0.4529**/**0.9584**
	VAR	0.0525/0.8290	0.2479/0.9113	0.3094/0.9306	0.3778/0.8875	**0.6547**/**0.9603**
Dataset #5	ED	0.0701/0.4611	0.0024/0.7000	**0.1916**/0.7861	0.0514/0.7336	0.0891/**0.8769**
	KLD	0.0935/0.5677	0.1962/0.7819	0.0701/0.8455	0.1262/0.7879	**0.4249**/**0.9018**
	RMS	0.1168/0.4742	0.2577/0.7099	0.0748/0.7929	0.2103/0.7319	**0.5013**/**0.8812**
	VAR	0.1449/0.3641	0.3995/0.5937	0.1963/0.6940	0.2710/0.6197	**0.5420**/**0.8271**

To quantify the degree of monotonicity, the evaluation metrics, whose value is between 0 and 1, are calculated using Eqs. (8.10) and (8.11). The time-series of feature values are more monotonic as a metric is closer to 1. Datasets #3, #4, and #5, for which the gradual failures were observed during data collection, were used to compare the degree of monotonicity of the features. In addition to the typical feature extraction (let's say Typical) and the dynamic feature extraction (let's say Dynamic), three popular data smoothing techniques are also evaluated for comparative study, and they are as follow:

- Moving Average (MA): MA is a data smoothing technique, which computes an average over a specific time period (i.e., window size). Mathematically, a value at time t can be calculated as $\hat{x}_t = (x_{t-l} + x_{t-l+1} + ... + x_{t-1})/l$ where x is the feature and l is the window size.
- Exponential Smoothing 1 (ES1): exponential smoothing is a time series smoothing technique, which applies exponentially decreasing weights to the past observations. In the method, a higher weight is applied to a more recent observation. The method is called as the simple exponential smoothing, and mathematically, it can be written as $\hat{x}_{t|t-1} = \alpha x_{t-1} + \alpha(1-\alpha)x_{t-2} + \alpha(1-\alpha)^2 x_{t-3} + ...$, where α is the smoothing parameter.
- Exponential Smoothing 2 (ES2): the extended exponential smoothing technique, called Holt's method [161], is used, and it can be written as $\hat{x}_{t+u|t-1} = q_{t-1} + ub_{t-1}$ where $q_{t-1} = \alpha x_{t-1} + (1-\alpha)(q_{t-2} + b_{t-2})$ and $b_{t-1} = \beta^*(q_{t-1} - q_{t-2}) + (1-\beta^*)b_{t-2}$, and β^* is the trend smoothing parameter.

The parameters, β^*, α, and l, for the above three techniques are determined using grid search method, and they are set to 0.3, 0.2, and 10, respectively. In Table 8.4, the performance of each method is reported. As seen in the table, the proposed approach outperforms other methods in all cases. The average monotonicity value for each of the four features (ED, KLD, RMS, and VAR) increases by 0.4962, 0.3428, and 0.2830 and the average Corr value for each of the four features increases by 0.0517, 0.1269, and 0.4050 after applying the dynamic feature extraction method to datasets #3, #4, and #5. To compare all methods in a more intuitive way, the averages of monotonicity and Corr of the four features are visualized in Figure 8.8. As expected, all methods improved the degree of monotonicity, but the proposed method performs the best overall.

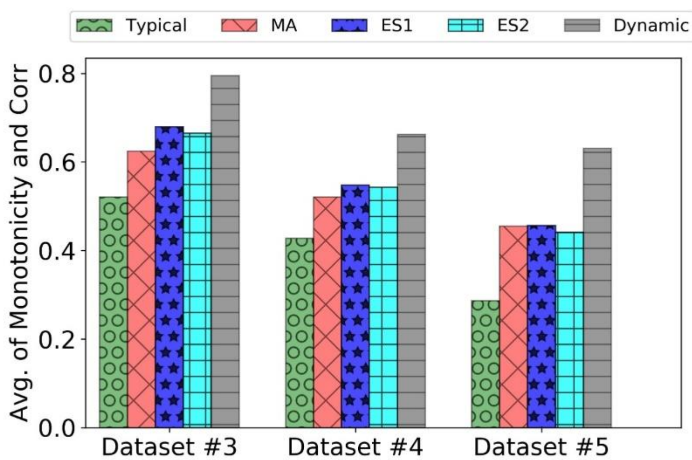


Figure 8.8 Monotonicity of features extracted using typical feature extraction (Typ) and dynamic feature extraction (Dyn).

All four features, ED, KLD, RMS, and VAR, are computed through dynamic feature extraction and they are scaled to between 0 and 1 to present in one figure as shown in Figure 8.9. All four features extracted from each dataset display similar trends as the health condition of a pump evolves toward breakdown. In the next section, the extracted features are interpreted using PCA to obtain an HI, and the HI is applied to GP regression for TTF predictions.

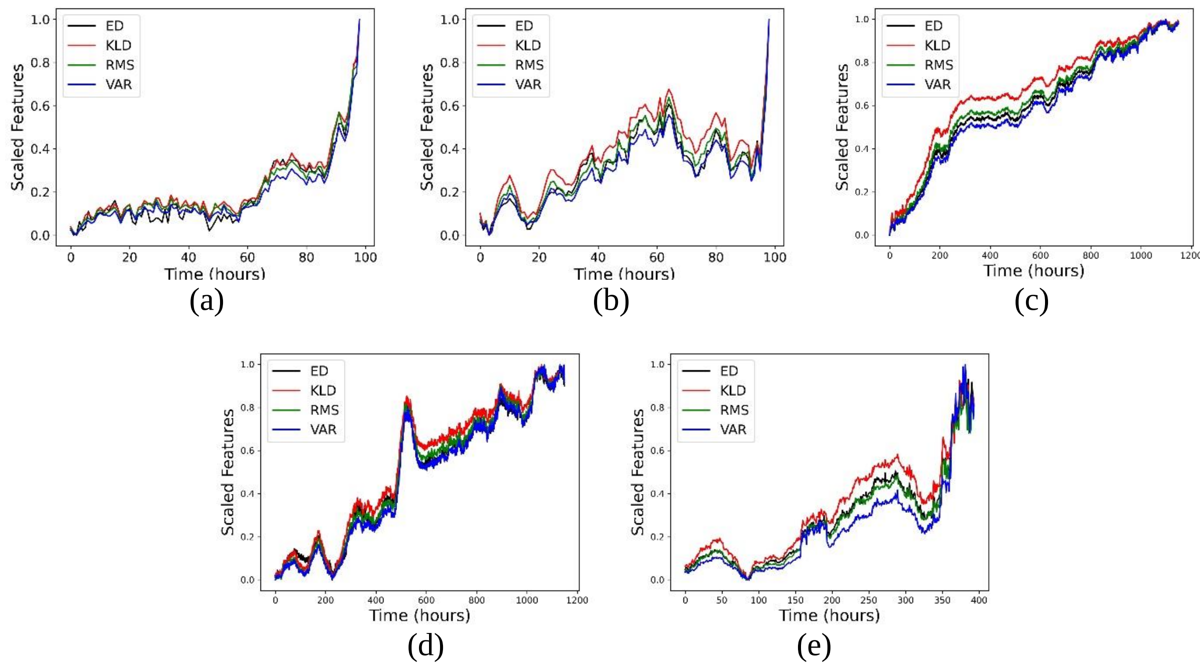


Figure 8.9 Application of dynamic feature extraction to (a) dataset #1, (b) dataset #2, (c) dataset #3, (d) dataset #4, and (e) dataset #5.

8.5.2 TTF Prediction and Uncertainty Quantification

The four extracted features are processed through PCA, and the value associated with the first principal component is used as the HI. The absolute value of the minimum HI value is added to all HI values to ensure that there are no negative HI values. In Figure 8.10, the HI values for each dataset are plotted. As seen in Figure 8.10, the health indicators at time = 0 differ greatly between datasets #1, #2, and #5 and datasets #3 and #4. This is because, as mentioned in Sections 8.4.2 and 8.4.3, the usage time of the pumps since last rebuild were very different at the outset of data collection. As has been stated, the bearing grease index at time=0 was 3 for datasets #1, #2, and #5 and 1 for datasets #3 and #4.

With the historical HI values, GP regression can be used to predict HI values of next time steps. To calculate TTF using the predicted HI values, a threshold (i.e., a HI value corresponding to a machine failure) must be determined first. Based on the HI values observed in Figure 8.10, the thresholds are set to 2 and 2.15 for the datasets collected from the piezoelectric accelerometer (datasets #1, #3 and #5) and for the datasets collected through the MEMS accelerometer (datasets #2 and #4), respectively. These particular HI thresholds were chosen to correspond to a case when a pump is on the verge of failure. The threshold for the MEMS accelerometer was set higher than the threshold for the piezoelectric accelerometer because of the difference in magnitudes between the two sensor types. For example, as shown in Figure 8.7, a larger average RMS is observed for dataset #3 (MEMS) relative to dataset #4 (piezo). Although the two sensors measured the acceleration of the pump at the same time, the difference in RMS magnitudes can be caused by several factors such as sensor mounting methods, mounted locations, and sensing mechanisms.

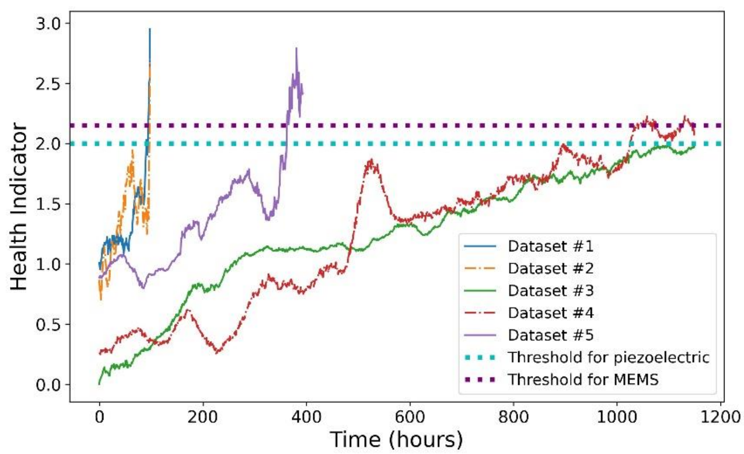


Figure 8.10 Construction of health indicator (HI).

For TTF predictions, datasets #3 and #4 were chosen to be analyzed and visualized due to the long period of the degradation observed during the data collection. To predict TTF, first, HI is predicted using a subset of historical data. As shown in Figure 8.11, the HI predictions are made by the GP regression after 550 hours and 800 hours. Here, the small zero mean Gaussian noise variance, σ^2 = 0.00025, is added to the predictions as a measurement error (i.e., aleatory uncertainty). The small variance is selected because noisy features are already filtered out in the previous stage (dynamic feature extraction). In the figure, training data is labeled as observations, and predictions are made with the uncertainty characterized using 2.5% and 97.5% quantiles (i.e., the lower and upper bounds), which means that 95% of future values would fall within this interval. To present TTF probabilistically, the probability distributions of TTF are shown at the corner of each figure. The probability distribution of TTF (here depicted with a histogram) are generated by i) drawing 1,000 samples from the posterior predictive distribution of a trained GP regression model, ii) calculating the difference between the time when the prediction is made (either 550 or 800) and the time when the predicted HI value reaches the threshold (for each sample), and iii) drawing a histogram using the values obtained from the differences (based on 1000 values). The full uncertainty in the figure indicates a sum of aleatory uncertainty and epistemic uncertainty.

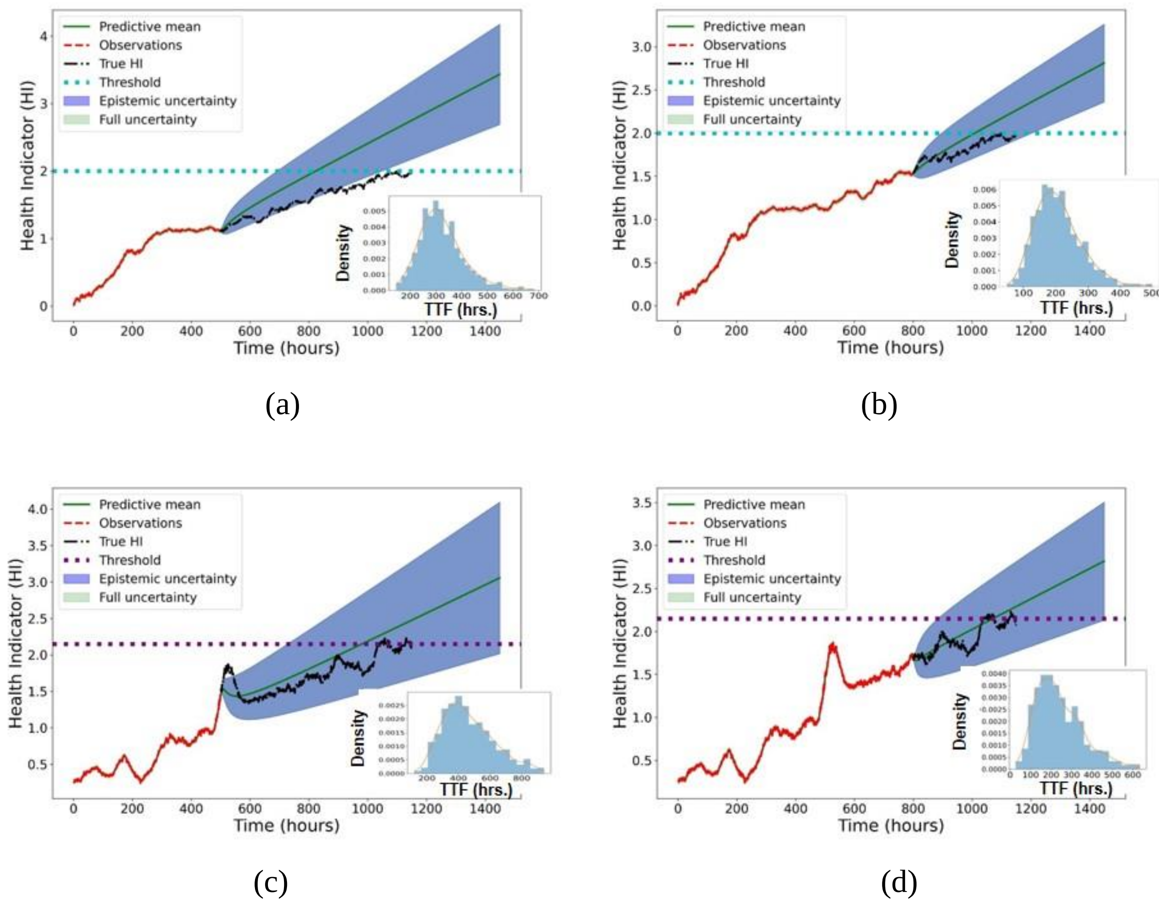
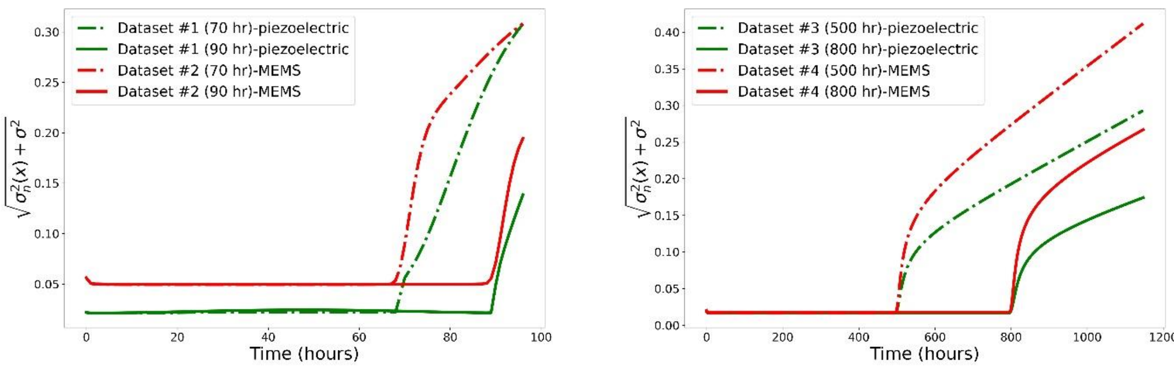


Figure 8.11 TTF predictions; (a) dataset #3-550hr (b) dataset #3-800hr (b) dataset #4-550hr (b) dataset #4-800hr.

The data used in Figure 8.11ab and Figure 8.11cd were collected by the piezoelectric accelerometer and the MEMS accelerometer, respectively. As shown in the figure, the larger uncertainties on the prediction are observed when the acceleration data collected by MEMS accelerometer are used in the model. Figure 8.12 presents more details on the uncertainty quantification. To compare the prediction uncertainties of different cases, datasets #1 and #2 (Figure 8.12a) and datasets #3 and #4 (Figure 8.12b) are chosen. As seen clearly, for all cases, the uncertainties start to grow after the end point of observation time. A larger uncertainty is observed when there is less training data (larger uncertainty for Figure 8.11a than for Figure 8.11b). Also, a GP regression model trained with features, which were extracted from data collected using MEMS accelerometers, shows larger uncertainty in the prediction than a model trained with features, which were extracted from data obtained using piezoelectric accelerometers.



(a) (b)

Figure 8.12 Uncertainty behavior using data obtained from piezoelectric and MEMS accelerometers.

8.6 Conclusion

In this paper, a new feature extraction method, dynamic feature extraction, is developed to reduce the effect of noisy statistical features. In our method, instead of selecting or searching for better/optimal features, we process the features to make them more useful (i.e., increasing the degree of monotonicity). Also, a GP regression model with a customized covariance function is combined with the feature extraction method to predict TTF of a vacuum pump. To experimentally validate the proposed method, function-to-failure datasets were collected from in-service vacuum pumps. The data were collected using two types of accelerometers: piezoelectric and MEMS accelerometers, to investigate the effect of sensor type on the model's predictions. In addition to the data preprocessing and model development/validation, details on the data acquisition setups for two different types of the accelerometer are also explained in this paper.

To validate the proposed feature extraction method, two evaluation metrics, monotonicity and Corr, are employed. After applying the dynamic feature extraction, the monotonicity and Corr of four features (ED, KLD, RMS, VAR), which are all features considered in this study, increase. Also, we empirically demonstrate the effectiveness of the proposed method by comparing it with three popular smoothing techniques. For the TTF prediction, as more historical HI data available, the predictions on HI become closer to the true HI, and the true HI falls within the upper and lower predictive limits as expected. For the uncertainty quantification, a model trained with features, which were extracted from data measured using MEMS accelerometers, shows larger uncertainty in the predictions than a model trained with features, which were extracted from data obtained using piezoelectric accelerometers.

To conclude, experimental data collected from in-service rotating machinery has demonstrated the effectiveness of the proposed method. A key feature of this work is that the data have been collected from actual in-service equipment supporting a semiconductor manufacturing facility, and there was no need to resort to artificially created faults. It is believed that the proposed method may be readily applied for TTF prediction of many types of manufacturing equipment.

9. ENVIRONMENTAL AND ECONOMIC CONSIDERATION IN THE MAINTENANCE OF INDUSTRIAL EQUIPMENT

9.1 Introduction

In the previous chapters, AI-driven methods have been intensively discussed. The previously discussed topics mainly focus on detecting a mechanical defect or forecasting a time to failure of equipment. In this chapter, the environmental and economic impacts are evaluated for the maintenance of industrial equipment. In Introduction (section 1), the effectiveness of maintenance for different products was discussed. As claimed, the product life extension through maintenance may not always be desirable from the environmental and economic perspectives. In this chapter, an electric motor is taken as an example to evaluate the equipment maintenance from the environmental and economic standpoints. Further, a case study including machine learning application and the estimations of economic loss and the environmental impact (CO_2) of different maintenance strategies (preventive, breakdown, and preventive maintenance) is discussed.

An electric motor is the most common industrial equipment in manufacturing plants, and it has broad application in many industries due to its functional diversity. Hence, the electric motors are the primary source of power consumption; it occupies a considerable fraction of total national power consumption. For example, in the U.S., nearly 75% of total industrial energy is consumed by electric motors [162]. Recently, there has been a growing concern about energy consumption in the industry. Several strategies were introduced to improve electric motors' efficiency, which will ultimately lead to reduction in the cost and CO_2 emission [161, 162]. One recommendation is to replace existing motors with high-efficiency motors, which can save more cost from the reduced energy usage than spending the cost of replacement. Li et al. [164] reported that 3-5 billion US dollars could be saved by adopting more efficient motors in the US industry. According to [165], for a 10-year operating period, the electricity cost accounts for 95% of lifetime motor cost, and the remaining 5% include purchase price, installation cost, maintenance cost, etc. As argued in Chapter 1.1, extending product life through maintenance may not always be necessary from environmental and economic perspectives. Replacing an old one with a new energy efficient model may reduce the fuel cost and the environmental burden. An old one may transmit power less efficiently than a new model from the motor to the driven equipment (i.e., requiring more input power as more powers are lost in the transmission). Another option is to properly maintain a motor to keep it close to original/optimal power efficiency over the use phase. Since the power efficiency of a motor can decrease over the operating hours due to progress of fault in a motor, regular maintenance such as oil/grease change and bearing inspection may help to prevent the efficiency loss; but it will incur additional costs (e.g., manpower) and equipment downtime. Also, it may require purchasing specialized machine monitoring tools (e.g., vibration analyzer) to identify a root cause (e.g., type of defect in a motor) of the efficiency loss.

Considering the lifetime motor cost, the power efficiency (the ratio of output power to input power) is the critical metric for decision-making, for example, repair vs. replacement and preventive maintenance vs. breakdown maintenance. Therefore, the economic loss and environmental impacts associated with a motor's efficiency loss may be helpful information for physical assets management in the production line [166]. In the next section, a mathematical model is discussed to present the economic loss and environmental impact as a function of the efficiency loss.

9.2 Efficiency Loss by Defect in Motor

As argued above, efficiency is an important metric for decision-making between repair and replacement. Based on the efficiency, the economic loss and environmental impacts (e.g., CO_2 emission) may be quantified. Generally, the efficiency of an electric motor can be defined as the ratio of output power to input power as follows,

$$\eta = \frac{P_{\text{output}}}{P_{\text{input}}} = \frac{P_{\text{output}}}{P_{\text{output}} + P_{\text{loss}}} = \frac{\tau \times \omega}{P_{\text{output}} + P_{\text{loss}}} \tag{9.1}$$

where P_{output}, P_{input}, P_{loss}, τ, and ω are the output power transmitted to driven equipment, the input power to a motor, the power loss due to various factors (e.g., friction, sound, and temperature increase), the torque of a shaft, and the rotational speed, respectively. Under a constant load setting, any fault(s) in a motor can increase P_{loss}, which will decrease η. For example, incorrect alignment (or misalignment) of the motor's shaft and driven equipment's shaft can generate excessive vibrations, heats, and noisy sounds. More details on the several types of mechanical defects and their impacts on the motor are explained in section 6.4. Lu et al. [167] presented a power flow and the definition of power losses in induction motors. For a better understanding of the motor's power efficiency, a diagram for power flow and power loss is shown in Figure 9.1. As seen in Figure 9.1, there are several types of power loss in the process of transforming electrical power to mechanical power. For the evaluation of efficiency, the input power and the out power are only considered as described in Eq. (9.1).

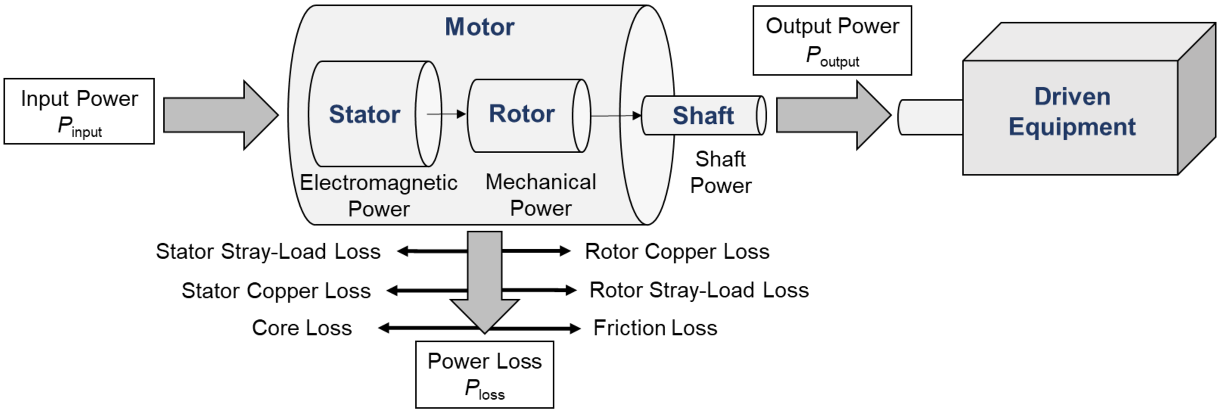


Figure 9.1 Power flow and power loss in electric motors.

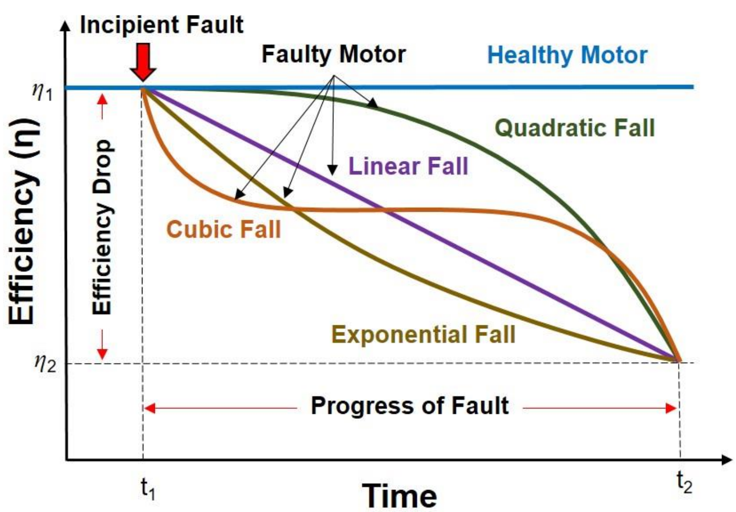
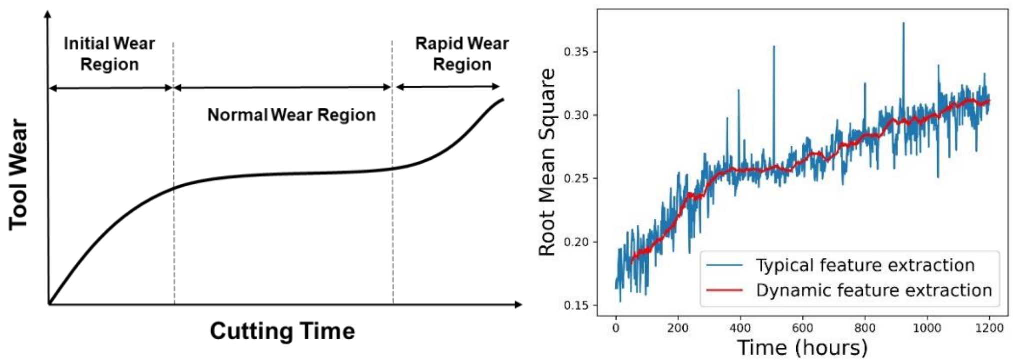


Figure 9.2 Efficiency erosion models (Linear, Quadratic, and Exponential) for continuous operation under constant load setting (adapted from [168]).



(a) (b)

Figure 9.3 (a) Typical growth pattern of tool wear (modified from [111]) and (b) average amplitude of vibration (i.e., RMS) over operating hour (figure from section 8.5.1).

As mentioned, η is likely to drop over operating hours due to a progress of fault in a motor (i.e., increasing severity of a fault). Therefore, when estimating the economic loss and environmental impacts caused by the efficiency loss due to a fault's progression, the dynamic of η (i.e., time dependent) should be considered. Assuming a constant load setting (i.e., constant P_{output}), η can be represented as follows

$$\eta(t) = \frac{P_{\text{output}}}{P_{\text{output}} + P_{\text{loss}}(t)}. \tag{9.2}$$

To describe a trend of efficiency loss over time, Singh et al. [168] proposed the three models using linear, quadratic, and exponential functions as shown in Figure 9.2. Those selected functions present different rates of efficiency drop over time, but they may not appropriately represent the efficiency drop caused by a mechanical degradation. As shown in Figure 9.3a, a tool wear grows through initial wear region, normal wear region, and rapid wear region over cutting time [111]; there is a slow growth region in the middle. Also, as shown in the time-to-failure experimental data (Figure 9.3b), the average amplitude of vibration (i.e., RMS) increases slowly between 400 hr and 600 hr, which may relate to the normal wear region in the tool wear growth graph. The increase in the average amplitude of vibration may correlate with the increase in $P_{loss}(t)$, so a motor's efficiency may decrease slowly in the middle of a motor's life. Since the efficiency drop caused by a progression of fault (e.g., progression of a mechanical wear) is mainly focused in this study, to reflect the growth pattern of tool wear in the efficiency drop, a cubic function is considered as well.

In Figure 9.2, for a healthy motor, the ratio of output power to input power is constant, so η_1, which is the efficiency observed from a healthy motor, is constant over time. If an incipient fault is developed at time t_1, η_1 starts to decrease over the time, $T = t_2 - t_1$, at a different rate depending on the severity and the type of defect in a motor. Mathematically, the linear, quadratic, and exponential, and cubic efficiency erosion in Figure 9.2 are express as

$$\eta_{linear}(t) = at + \eta_1 \qquad \text{where } a = \frac{\eta_2 - \eta_1}{T}, \tag{9.3}$$

$$\eta_{quadratic}(t) = \eta_1 + bt^2 \ \text{ where b} = \frac{\eta_2 - \eta_1}{T^2}, \tag{9.4}$$

$$\eta_{exponential}(t) = \eta_1 e^{ct} \ \text{ where c} = \frac{\ln \eta_2 - \ln \eta_1}{T}, \tag{9.5}$$

$$\eta_{cubic}(t) = d(t - \frac{T}{2})^3 + \frac{\eta_1 + \eta_2}{2} \ \text{ where } d = \frac{4(\eta_2 - \eta_1)}{T^3}, \tag{9.6}$$

where a, b, c, and d are negative values. To estimate P_{loss} over the advancement of a fault, the above models are adopted in this study. Unlike the method used in [168] in which P_{input} was presented as a function of time, in this study, P_{loss} is presented as a function of time. Let's consider the linear case first. If differentiating Eq. (9.2) on both sides, it is

$$\frac{d\eta(t)}{dt} = \frac{d}{dt}\left(\frac{P_{\text{output}}}{P_{\text{output}} + P_{\text{loss}}(t)}\right) = \frac{-P_{\text{output}}}{(P_{\text{output}} + P_{\text{loss}}(t))^2}\frac{d}{dt}(P_{\text{loss}}(t)). \tag{9.7}$$

Using Eq. (9.3), Eq. (9.7) can be further expressed as

$$a = \frac{-P_{\text{output}}}{(P_{\text{output}} + P_{\text{loss}}(t))^2}\frac{d}{dt}(P_{\text{loss}}(t)). \tag{9.8}$$

Let's p_{t_1} and p_{t_2} are the P_{loss} at t_1 and t_2, respectively, where $t_1 = 0$ and $t_2 = T$ (let's $t_1 = 0$ and $t_1 \le t \le t_2$ for clarity). After rearranging Eq. (9.8), integrating over time interval on both sides, and solving the equation as follow

$$\int_{p_{t_1}}^{p_{t_2}} \frac{dP_{\text{loss}}(t)}{(P_{\text{output}} + P_{\text{loss}}(t))^2} = \int_0^T -\frac{a}{P_{\text{output}}}dt, \tag{9.9}$$

$$-\frac{1}{(P_{\text{output}} + p_{t_2})} + \frac{1}{(P_{\text{output}} + p_{t_1})} = -\frac{aT}{P_{\text{output}}}. \tag{9.10}$$

Then, p_{t_2} and $P_{input\_t_2}$ (the input power at t_2) can be express as a function of time as follow

$$p_{t_2} = \frac{1}{\frac{aT}{P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}} - P_{\text{output}}, \tag{9.11}$$

$$P_{input\_t_2} = P_{\text{output}} + p_{t_2} = \frac{1}{\frac{aT}{P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}}. \tag{9.12}$$

Assuming the linear efficiency erosion, the additional power, ΔP_{linear}, required to transmit to the driven equipment compared to a healthy motor at $t=T$ is

$$\Delta P_{linear} = P_{input\_t_2} - P_{input\_t_1} = \frac{1}{\frac{aT}{P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}} - (P_{\text{output}} + p_{t_1}) = \frac{1}{\frac{aT}{P_{\text{output}}} + \frac{1}{P_{input\_t_1}}} - P_{input\_t_1}. \tag{9.13}$$

For the quadratic and exponential efficiency erosions, ΔP can be calculated similarly as done in the linear case; inserting Eq. (9.4), Eq. (9.5), and Eq. (9.6) in Eq. (9.7) for the consideration of quadratic, exponential, and cubic erosions, respectively. Then, $\Delta P_{quadratic}$, $\Delta P_{exponential}$, and ΔP_{cubic} are expressed as follow

$$\Delta P_{quadratic} = \frac{1}{\frac{bT^2}{P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}} - (P_{\text{output}} + p_{t_1}) = \frac{1}{\frac{bT^2}{P_{\text{output}}} + \frac{1}{P_{input\_t_1}}} - P_{input\_t_1}, \tag{9.14}$$

$$\Delta P_{exponential} = \frac{1}{\frac{\eta_1(e^{cT}-1)}{P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}} - (P_{\text{output}} + p_{t_1}) = \frac{1}{\frac{\eta_1(e^{cT}-1)}{P_{\text{output}}} + \frac{1}{P_{input\_t_1}}} - P_{input\_t_1}, \tag{9.15}$$

$$\Delta P_{cubic} = \frac{1}{\frac{dT^3}{4P_{\text{output}}} + \frac{1}{(P_{\text{output}} + p_{t_1})}} - (P_{\text{output}} + p_{t_1}) = \frac{1}{\frac{dT^3}{4P_{\text{output}}} + \frac{1}{P_{input\_t_1}}} - P_{input\_t_1}. \tag{9.16}$$

To estimate the economic loss (\$) and environmental impact ($kgCO_2$) incurred by the efficiency loss due to an advancement of a fault in a motor over time, additional energy consumed by a faulty motor can be used. This can be achieved by integrating $\Delta P(t)$ over the time between t_1 and t_2, as follow

$$\Delta E = \int_{t_1}^{t_2} \Delta P(t)dt. \tag{9.17}$$

To be more specific, ΔE is the additional amount of energy (kWh) consumed over time, $T=t_2-t_1$, in a faulty motor compared to a healthy motor. To present ΔP as a function of t, i.e., $\Delta P(t)$, simply replacing T (constant value) with t (variable) in Eq. (9.13), Eq. (9.14) and Eq.(9.15) for the linear, quadratic, and exponential fall cases. For the cubic fall case, Eq. (9.16) can be rewritten as

$$\Delta P_{cubic}(t)=\frac{1}{\frac{d}{P_{\text{output}}}(t^3-\frac{3}{2}Tt^2+\frac{3}{4}T^2t)+\frac{1}{P_{input\_t_1}}}-P_{input\_t_1}. \tag{9.18}$$

Then, the additional amount of energy for the linear, quadratic, exponential, and cubic efficiency fall scenarios can be calculated by inserting Eq. (9.13), Eq. (9.14), Eq.(9.15), and (9.18) into Eq. (9.17), respectively, and they are expressed as follow

$$\Delta E_{linear}=\int_{t_1}^{t_2}\Delta P_{linear}(t)dt=\frac{P_{\text{output}}}{a}\ln(\frac{aP_{input\_t_1}}{P_{\text{output}}}T+1)-P_{input\_t_1}T, \tag{9.19}$$

$$\Delta E_{quadratic}=\int_{t_1}^{t_2}\Delta P_{quadratic}(t)dt=\frac{1}{\sqrt{{-4b}/{P_{\text{output}}P_{input\_t_1}}}}\ln\left|\frac{{2bT}/{P_{\text{output}}}-\sqrt{{-4b}/{P_{\text{output}}P_{input\_t_1}}}}{{2bT}/{P_{\text{output}}}+\sqrt{{-4b}/{P_{\text{output}}P_{input\_t_1}}}}\right|-P_{input\_t_1}T, \tag{9.20}$$

$$\Delta E_{exponential}=\int_{t_1}^{t_2}\Delta P_{exponential}dt=-\frac{P_{output}}{\eta_1 c}(e^{-cT}-1)-P_{input\_t_1}T, \tag{9.21}$$

$$\begin{aligned}\Delta E_{cubic}=\int_{t_1}^{t_2}\Delta P_{cubic}(t)dt=&\frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^2}T^2+2\sqrt[3]{\alpha}\sqrt[3]{\beta}T+4\sqrt[3]{\beta^2}\right|}{4}\right)}{6\sqrt[3]{a}\beta^{\frac{2}{3}}}-\\
&\frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^2}T^2-2\sqrt[3]{\alpha}\sqrt[3]{\beta}T+4\sqrt[3]{\beta^2}\right|}{4}\right)}{6\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}+\frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^{-1}}\right|\left|\sqrt[3]{\alpha}T+2\sqrt[3]{\beta}\right|}{2}\right)}{3\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}-\frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^{-1}}\right|\left|\sqrt[3]{\alpha}T-2\sqrt[3]{\beta}\right|}{2}\right)}{3\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}\\
&+\frac{\arctan\left(\frac{\sqrt{3}\sqrt[3]{\alpha}\sqrt[3]{\beta^2}T+\sqrt{3}\beta}{3\beta}\right)}{\sqrt{3}\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}+\frac{\arctan\left(\frac{\sqrt{3}\sqrt[3]{\alpha}\sqrt[3]{b^2}T-\sqrt{3}\beta}{3\beta}\right)}{\sqrt{3}\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}-P_{input_t_1}T,\end{aligned} \tag{9.22}$$

where α and β in Eq. (9.22) are $\frac{d}{P_{output}}$ and $\frac{T^3\alpha}{8}+\frac{1}{P_{input\_t_1}}$. The detailed calculation to obtain ΔE_{cubic} (Eq. (9.22)) is available in Appendix.

Lastly, the additional cost and the environmental impact (i.e., CO_2 emission) associated with the additional energy consumed, ΔE, can be calculated as follow

$$\Delta C_{loss} = \Delta E \times C_{energy}, \tag{9.23}$$

$$\Delta EI_{CO_2} = \Delta E \times EF_{CO_2}, \tag{9.24}$$

where C_{energy} (\$/kWh) is the electricity cost per kWh and EF_{CO2} ($kgCO_2$/kWh) is the carbon dioxide emission factor.

In the next section, a case study including the application of machine learning to industrial equipment will be presented. Also, the economic loss and environmental impacts are estimated based on the available information and the equations presented above.

9.3 Case Study: Maintenance of Vacuum Pumps

A vacuum pump is a widely used device in vehicles, robot arms, HVAC, manufacturing equipment, etc. The function of the vacuum pump is to transfer air into or out of a certain volume by inducing a pressure difference between two regions in a pumping chamber (e.g., atmospheric and ultimate vacuum). Inside the pump, there is an enclosed electric motor. A motor driven-shaft is connected to driven machine for power transmission (we specifically focus on the motor in a pump). During normal pump operation, proper maintenance (e.g., oil change) is recommended to be conducted as suggested by the manufacturer. This also may enable the life extension of many mechanical (e.g., gearbox and bearings) components in a pump.

In order to apply a machine learning method in a maintenance task, first, enough machine condition data (i.e., sensor signals) needs to be collected and processed to train and validate a model. Then, the trained model can be deployed to monitor the equipment condition and predict its future failure. Direct measurement of machine condition (e.g., healthy) is difficult. Often, vibration data is used as a proxy for machine state. Three piezoelectric accelerometers (PCB PIEZOTRONICS, TLD352A56) were mounted on three vacuum pumps (Edwards, QDP80) in the Birck Nanotechnology Center (Indiana, USA). Each pump is connected to a three-phase power supply with 30 amps and 208-220 VAC, and 0.3 GPM process cooling water is employed to regulate the temperature in the pumps. The pumps run 24hrs/day, 7days/week, and 48-50 weeks per year. They are used for furnace vacuum equipment, and they only are stopped for regular maintenance. In Figure 9.4, the data acquisition setup on one pump is illustrated (the pumps have identical setups). Data acquisition was enabled by a National Instruments compact data acquisition system. Acceleration signals were stored in a PC using LabVIEW software with a sampling frequency of 12 kHz for each pump. Digital data were collected for one second every hour.

While the three pumps have an identical operational and physical setup, their health condition may be different since their ages are different. In other words, the one rebuilt most recently might have a longer remaining useful life. To build a data-driven machine learning model, especially for supervised-learning, labels of corresponding acceleration signals are required. Here, labels can be the health condition of the equipment such as bad or good. One "good" estimation of the health condition of a vacuum pump may be based on the color of the bearing grease. Health condition labels will be described in the next section.

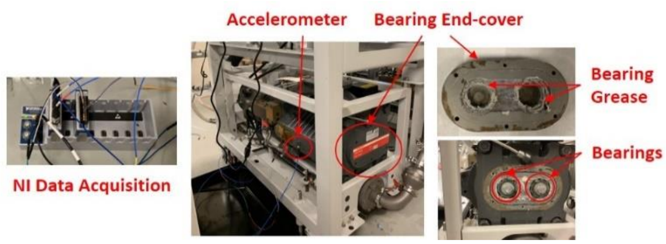


Figure 9.4 Vacuum pump monitoring using an accelerometer.

9.3.1 Bearing Health Indicator

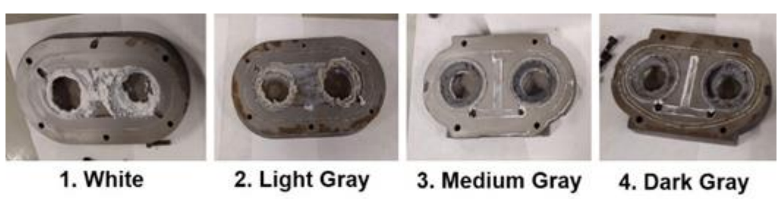


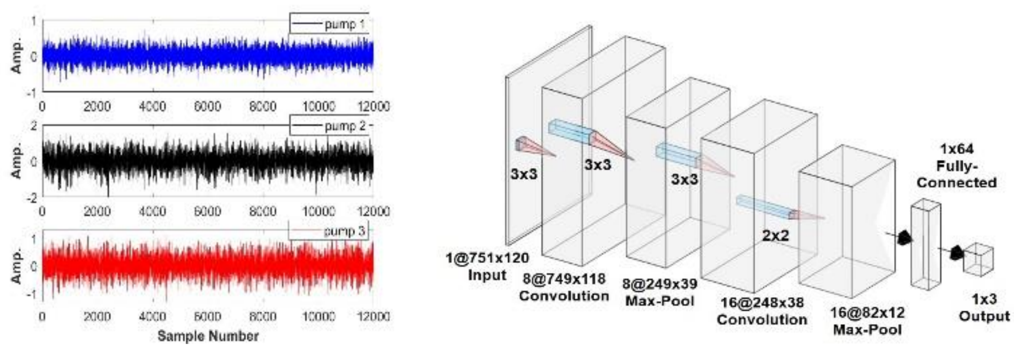
Figure 9.5 Bearing grease index to estimate equipment health condition.

One easy way to estimate the health condition of the vacuum pumps without dissembling the entire body is by observing the color of the bearing grease. As shown in Figure 9.4, the color can be observed by opening the bearing end-cover. The brightness of the grease color in the end cover cavity is used as a health indicator (perfluoropolyether (PFPE) is used as a bearing grease). Similar to grayscale, the grease color (originally white) becomes darker as more debris particles are generated from bearing wear and the grease is contaminated. Figure 9.5 shows the index numbers from 1 to 4 (the number increases as failure approaches), but it can be even darker in more extreme conditions.

Table 9.1 Current health condition of pumps and purchase rebuild history.

Pump #	Grease Index	Purchased Date	Recent Rebuild Date
1	1	October 2015	Jul 2017/ Aug 2019
2	3	September 2011	Sep 2015/ Jul 2018
3	4	September 2007	Oct 2009/ Dec 2013

Based on the grease color in the end-cover cavity of each pump, the current conditions of three pumps were determined as illustrated in Table 9.1. As mentioned, since the three pumps were purchased and rebuilt at different times, they all showed clearly different index values. As expected, pump 1, which was recently rebuilt, shows an almost-new as condition, while pump 3, which was rebuilt about six years ago, displays the highest index indicating that it is close to failure (observations made in August 2019). Next, acceleration signals obtained from the vacuum pumps and a machine learning application will be discussed.

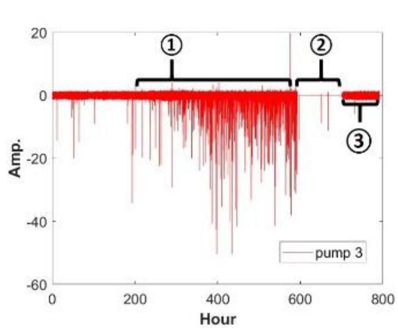


(a) (b)

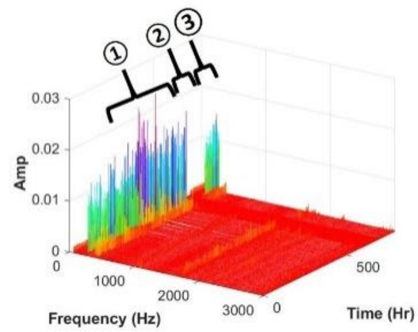
Figure 9.6 (a) raw acceleration signals observed from pumps (pumps 1, 2, and 3 have the bearing grease indexes of 1, 3, and 4) and (b) CNN architecture.

9.3.2 Machine Learning Application

Once proxy measures are obtained from sensors, they are often pre-processed to eliminate redundant information and noise before applying machine learning [160]. Signal processing (e.g., digital filters) and feature engineering (e.g., extracting features from time, frequency, time-frequency domains) are common methods used in pre-processing [169]. In Fig. 4a, raw signals obtained from one sampling trial are plotted. As expected, the amplitude of the acceleration increased as the time since last rebuild increased.



(a)



(b)

Figure 9.7 Long-term data collection; (a) time domain and (b) time-frequency domain (waterfall plot). ①, ②, and ③ in the figures present the incipient failure, pump stop (maintenance), and reoperation.

In order to classify the pump conditions using the collected acceleration signals, a convolutional neural network (CNN) was employed. CNN is a popular method to classify 2-D data types [160]. Therefore, 1-D acceleration signals in the time domain were converted into the time-frequency domain (i.e., a spectrogram). A spectrogram that displays the power spectral density presents the power content of a signal at various frequencies over time, viz.,

$$p(f) = \frac{\Delta t}{N} \left| \sum_{n=0}^{N-1} h(n) x(n) e^{-i 2\pi fn} \right|^2, \; -\frac{1}{2\Delta t} < f < \frac{1}{2\Delta t} \tag{9.25}$$

where Δt is a sampling frequency. In the analysis, the Hamming window function is employed, and the time-frequency features of each sampling trial are computed. Then, the processed data are used to train and test the CNN model shown in Figure 9.6b. For the CNN architecture, two convolutional and two max-pooling layers followed by a fully connected layer are used between the input and output layer. The data were obtained for 150 hrs (i.e., the number of data points is 12k*150*3), and is divided into 70% and 30% for training and testing the model, respectively. The model was implemented using Keras deep learning framework, and the training and testing were conducted on a PC equipped with GeForce GTX 1080 TI with 11GB GDDR5X.

After 10 epochs, both training and testing accuracy reached 100%, which means that the trained model was able to classify all the testing data correctly. This is possible because there are distinct differences among the signals, and unwanted signals were filtered out by the power spectral density function. Thus, the representations in the acceleration data for each pump's condition were learned using the CNN. Figure 9.7 displays the long term data collection (~ one month) for pump 3, including incipient failure and pump stop time. When temperature became extreme, the internal control system automatically shut down the pump. In the figure, abnormal values can be clearly observed in both the time and waterfall plots, and this data could be used for prognostic modelling to estimate the remaining time until breakdown.

9.3.3 Estimation of Economic Loss and Environmental Impact

As seen in Figure 9.7, the large amplitudes of vibration were observed before the breakdown. Also, in Figure 8.7, the average amplitude of vibration (i.e., Root Mean Square) tends to increase over the operation hours while a failure was gradually being progressed; this may relate to the increase of P_{loss}, and consequently, result in the efficiency loss. Based on the information we described above, the economic loss and environmental impact caused by the efficiency loss are quantified in this section.

According to the equipment manual [170], the capacity of the motor (i.e., rated power/load) is 6kW. An electric motor is designed to operate at between 50% to 100% of the rated load, and the maximum efficiency is usually achieved when a load is 75% of the rated load [171]. Under this condition, the range of load that can be used in a motor used in this case study is between 3 kW (4.023hp) and 6 kW (8.046 hp). Also, it may show the maximum efficiency when a load is 4.5 kW (6.035 hp), which is 75% of the rated load.

For the case study, several assumptions are made in this study. A motor is assumed to be operated under a constant load (i.e., constant operating condition). Also, P_{output} is assumed to be 75% of the rated power, which means that a motor operates under optimal efficiency. To calculate an additional amount of energy (kWh) consumed over operating hours in a faulty motor compared to a healthy motor (i.e., ΔE), the efficiency of a healthy motor (i.e., flawless motor) is set to 0.9. The efficiency of the healthy motor is assumed to decrease over operating hours, and three operating hours, 4,032, 8,064, and 24,192, are evaluated for an assumed efficiency drop. These three operating hours correspond to 6-month, 1-year, and 3-year operating periods (running the equipment 48-week per year). As an example, in Figure 9.8, the 0.3 efficiency drop (from 0.9 to 0.6) over a 1-year operating period is presented for the different efficiency erosion scenarios. To calculate the economic loss and environmental impact associated with an additional amount of energy, the electricity cost per kWh and the emission factor are set to as C_{energy} = 0.0666 ($/kWh) [172] and EF_{CO2} =1.0024 (kgCO_2/kWh) [173].

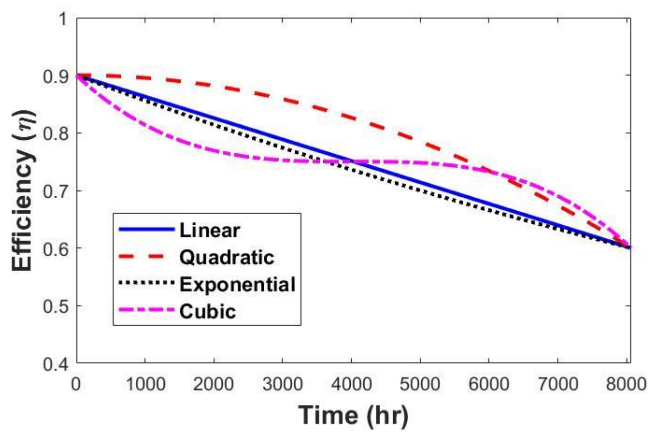
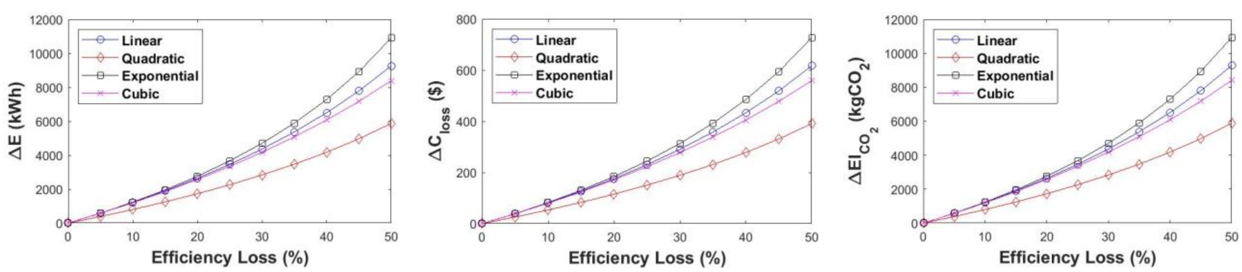


Figure 9.8 Efficiency erosion models (Linear, Quadratic, Exponential, and Cubic) for 1-year operating period (8,064 operating hours).

With the assumption defined above, ΔE, ΔC_{loss}, and ΔEI_{CO2} are calculated under the four efficiency erosion scenarios using Eqs. (9.19), (9.20), (9.21), and (9.22), respectively. Over the operating hours, the motor's efficiency can drop at a different rate depending on a type of fault in a motor; the different efficiency loss fall trends may reflect these different rates. Figure 9.9 displays the additional amount of energy, economic loss, and CO_2 emission for a 6-month operating period under the different efficiency fall scenarios. The figure shows the quantities against different efficiency losses from 0 to 50%. As seen in the figure, the more efficiency drops, the more electricity is used in all cases, and consequently, the more economic loss and CO_2 emission. Among the four scenarios, the additional amounts are the highest when the exponential erosion is assumed, followed by linear, cubic, and quadratic erosions.

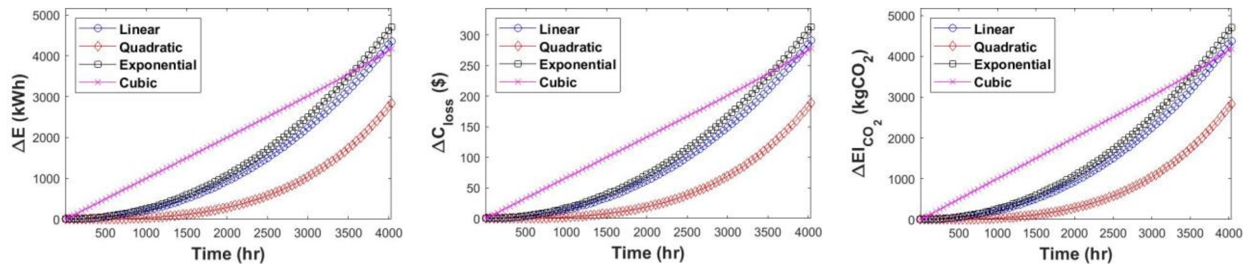
In Figure 9.10, with the assumption that the efficiency decreases by 30% in 4,032 hours (6-month operating period), the additional energy consumption, economic loss, and CO_2 emission are plotted over the operating hours. As shown in the figure, in the early stage, which might be related to an early mechanical degradation stage, the increases are not much significant except for cubic erosion scenario, in which a rapid efficiency drop is assumed in the early stage. However, in all other cases, ΔE, ΔC_{loss}, and ΔEI_{CO2} increase sharply after the early stage (after around 200~400 hours), so early maintenance may be beneficial in this situation.

The motor considered herein has relatively small rated power, so the economic loss may be fairly insignificant for the efficiency drop (if considering labor cost for maintenance). Even after a 30% drop of efficiency, as shown in Figure 9.9b, the economic losses for a six-month operating period year are \$291, \$189, \$313, and \$278 for the linear, the quadratic, the exponential, and the cubic respectively. Normally, in a manufacturing plant, a lot of motors with high mechanical output ratings are also used. Therefore, the additional amounts of energy, economic loss, and CO_2 emission for different rated power cases are also investigated in Figure 9.11. In the figure, the input power (75% of rated power) ranges between 1 kW and 200 kW. Also, the efficiency is assumed to decrease by 30% for the six-month operating period (4,032 hours), as assumed in Figure 9.10. As shown in Figure 9.11, the additional amounts of energy, cost, and emission tend to increase linearly over the input power. The higher the motor's capacity, the better it may be to conduct maintenance in the early fault stage through predictive or preventive maintenance.



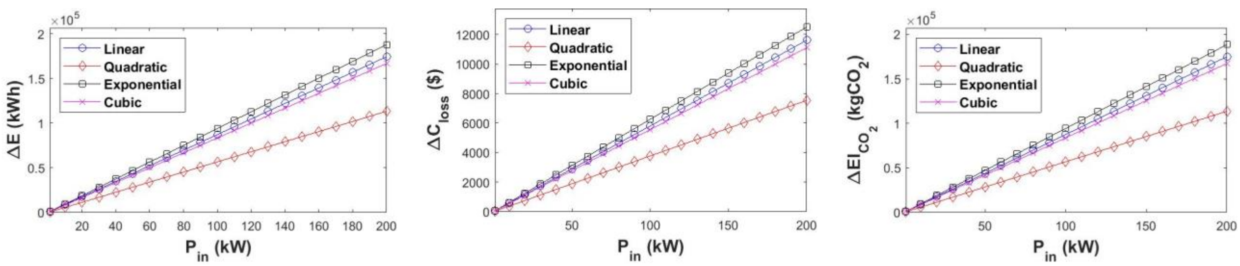
(a) (b) (c)

Figure 9.9 Additional amount of (a) energy, (b) economic loss, and CO_2 emission for different efficiency losses in a faulty motor compared to a healthy motor under four efficiency erosion scenarios; 6-month operating period (4,032 operating hours).



(a) (b) (c)

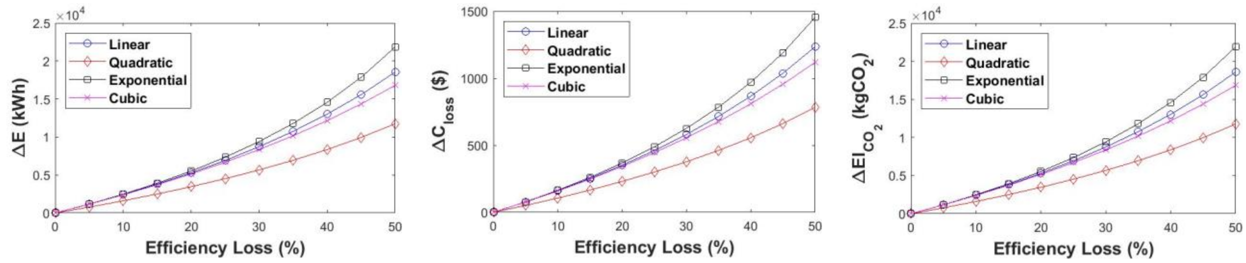
Figure 9.10 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission when efficiency drops from 90% to 60% over 6-month operating hours (4,032 hours) under four efficiency erosion scenarios.



(a) (b) (c)

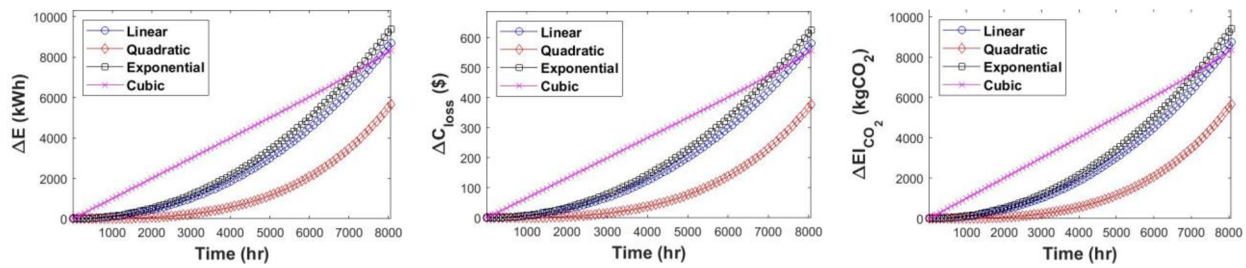
Figure 9.11 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission for different rated loads/input powers under four efficiency erosion scenarios; efficiency drop by 30% over 6-month operating hours (4,032 hours).

As mentioned, several different operating periods are also investigated. As done for the six-month operating period (8,064 hours) in Figure 9.9, Figure 9.10, and Figure 9.11, the same analyses are performed in Figure 9.12, Figure 9.13, and Figure 9.14 for the 1-year operating period (8,064 hours), and in Figure 9.15, Figure 9.16, and Figure 9.17 for the 3-year operating period (24,192 hours). The results show clearly that the longer the operating period, the higher ΔE, ΔCloss, and ΔEICO2; ΔE, ΔCloss, and ΔEICO2 linearly increase as a function of operating hour.



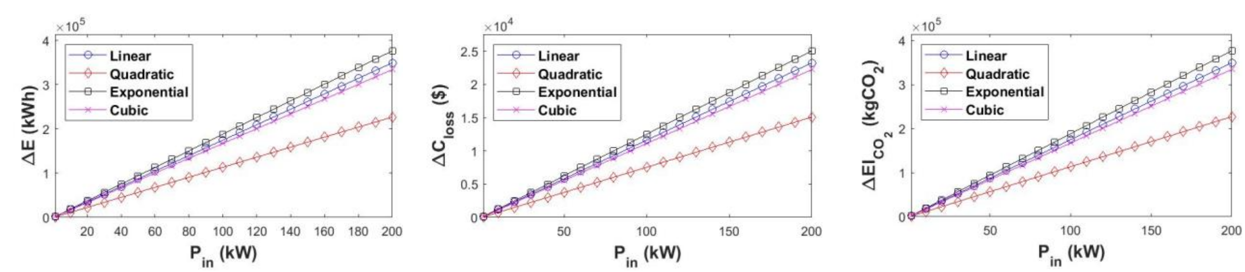
(a) (b) (c)

Figure 9.12 Additional amount of (a) energy, (b) economic loss, and CO_2 emission for different efficiency losses in a faulty motor compared to a healthy motor under four efficiency erosion scenarios; 1-year operating period (8,064 operating hours).



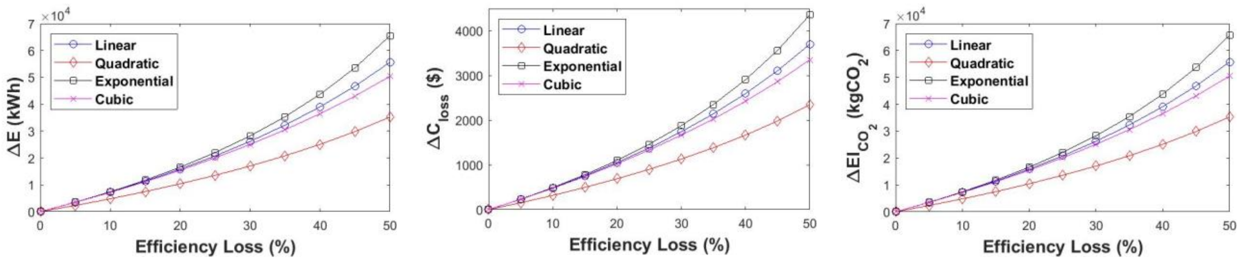
(a) (b) (c)

Figure 9.13 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission when efficiency drops from 90% to 60% over 1-year operating period (8,064 hours) under four efficiency erosion scenarios.



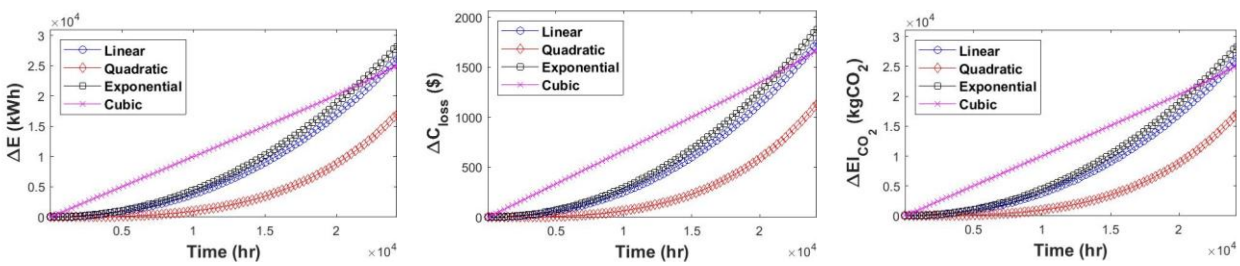
(a) (b) (c)

Figure 9.14 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission for different rated loads/input powers under four efficiency erosion scenarios; efficiency drop by 30% over 1-year operating period (8,064 hours).



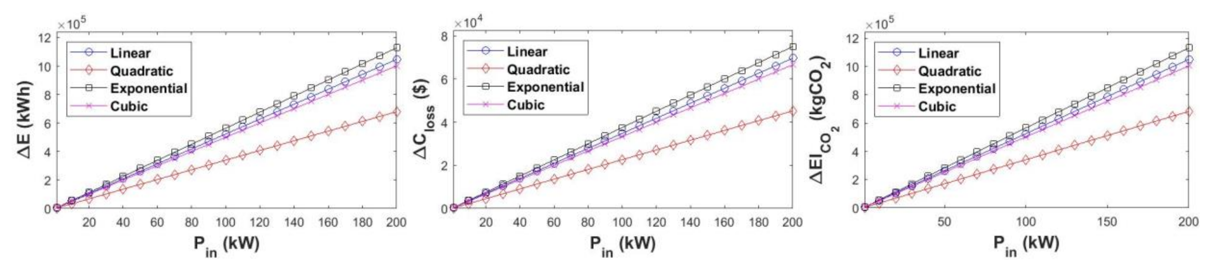
(a) (b) (c)

Figure 9.15 Additional amount of (a) energy, (b) economic loss, and CO_2 emission for different efficiency losses in a faulty motor compared to a healthy motor under four efficiency erosion scenarios; 3-year operating period (24,192 hours).



(a) (b) (c)

Figure 9.16 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission when efficiency drops from 90% to 60% over 3-year operating period (24,192 hours) under four efficiency erosion scenarios.



(a) (b) (c)

Figure 9.17 Increases in (a) energy consumption, (b) economic loss, and (c) CO_2 emission for different rated loads/input powers under four efficiency erosion scenarios; efficiency drop by 30% over 3-year operating period (24,192 hours).

Table 9.2 Maintenance procedure of the vacuum pumps.

Interval	Maintenance Task
Every Month	• Inspect oil levels/conditions and change/top off as necessary. • Perform leak rate test. • Log temperatures of pump, blower, and process cooling water in/out.
Every Six Month	• Inspect and re-pack bearing caps with grease. • Check oil levels on blower coupling cover and shaft seal as well as pump reservoir. • Inspect filter trap and replace if necessary. • Inspect dead leg for oil or particle contamination. • Inspect coolant reservoir and investigate if levels change. • Inspect cooling water hoses for deterioration.
Every three Year	• Send pump out for refurbishment if breakdown • Install and use a spare pump during refurbishment.

From the equipment maintenance schedule provided by a facility maintenance engineer, preventive maintenance has been conducted every certain period of time. Table 9.2 summarizes the maintenance tasks of the vacuum pumps. As seen in the table, several maintenance works are conducted every one-month, six-month, and three-year. If a pump becomes breakdown or a failure happens regularly, it is sent out for refurbishment (normally every three years). Based on the given information in Table 9.2, for the 3 year-operating period, the total costs of the preventive, breakdown, and preventive maintenances are estimated as follow

$$C_{preventive} = C_{monthly} N_1 + C_{biannually} N_2 + C_{energy}(E_{healthy} + \Delta E)R + C_{CO_2}(EI_{CO_2} + \Delta EI_{CO_2})R, \quad (9.26)$$

$$C_{breakdown} = C_{energy}(E_{healthy} + \Delta E)R + C_{CO_2}(EI_{healthy} + \Delta EI_{CO_2})R + T_{refurbishment} R, \quad (9.27)$$

$$C_{predictive} = C_{monthly} N_1 + C_{biannually} N_2 + C_{energy}(E_{healthy} + \Delta E)R + C_{CO_2}(EI_{healthy} + \Delta EI_{CO_2})R + T_{hardware}, \quad (9.28)$$

where $C_{monthly}$ is the monthly maintenance cost per intervention (\$/intervention), $C_{biannually}$ is the biannually maintenance cost per intervention (\$/intervention), N_1 is the number of intervention for the monthly tasks in Table 9.2, N_2 is the number of intervention for the biannually tasks in Table 9.2, $E_{healthy}$ is the amount of electricity consumed by a healthy motor (kWh), C_{CO2} is the cost of carbon dioxide emission (\$/kgCO_2), EI_{CO2} is the amount of CO_2 emission associated with $E_{healthy}$, R is the number of equipment, $T_{refurbishment}$ is the refurbishment cost including disassembly, shipping, and reinstallation, and $T_{hardware}$ is the hardware costs including data acquisition system and sensors. Here, in the equation, the cost associated with CO_2 emission is also included for the environmental consideration. The parameters are set to $C_{monthly}$=\$200, $C_{biannually}$=\$300, C_{CO2} =0.01

(\$/kgCO_2), $T_{refurbishment}$ =\$2,000, and $T_{hardware}$=\$6,231 (these are rough estimations based on the survey and [174]). N_1 and N_2 are set to 36 and 6 in preventive maintenance and 12 and 2 in predictive maintenance because there is less intervention in predictive maintenance (i.e., do only when necessary).

In order to estimate and compare the total cost of each maintenance strategies, two cases are examined: (1) different efficiency erosion models across the maintenance strategy and same efficiency drop over the operating period and (2). same efficiency erosion models across the maintenance strategies and different efficiency drops over the operating period.

Case 1: different efficiency erosion models across the maintenance strategies and same efficiency drop over the operating period

To calculate the costs associated with additional amounts of energy consumption and CO_2 emission, the cubic, exponential, and quadratic efficiency erosion models are adopted for preventive, breakdown, and predictive maintenance strategies, respectively. This implies the more amount of energy is consumed in the order of predictive, preventive, and breakdown maintenance over the 3 year-operating period. The efficiency loss, $\Delta\eta$, is assumed to be 40% for all strategies.

Table 9.3 Additional costs incurred by a fault in a motor for 3-year operating period.

Efficiency Loss (%)	Linear (\$)	Quadratic (\$)	Exponential (\$)	Cubic (\$)
0	0	0	0	0
5	232	154	234	231
10	483	319	493	478
15	756	498	781	742
20	1,054	691	1,102	1,027
25	1,381	901	1,465	1,334
30	1,743	1,132	1,878	1,667
35	2,145	1,385	2,353	2,030
40	2,598	1,668	2,908	2,428
45	3,111	1,985	3,566	2,866
50	3,703	2,346	4,361	3,355

Table 9.4 Additional CO_2 emission incurred by a fault in a motor for 3-year operating period.

Efficiency Loss (%)	Linear ($kgCO_2$)	Quadratic ($kgCO_2$)	Exponential ($kgCO_2$)	Cubic ($kgCO_2$)
0	0	0	0	0
5	3,498	2,323	3,532	3,479
10	7,281	4,816	7,429	7,196
15	11,389	7,502	11,757	11,180
20	15,873	10,409	16,597	15,463
25	20,797	13,573	22,055	20,087
30	26,238	17,038	28,270	25,100
35	32,297	20,859	35,426	30,561
40	39,106	25,108	43,776	36,546
45	46,839	29,882	53,677	43,149
50	55,736	35,311	65,650	50,497

Table 9.5 Comparison of total maintenance cost of three vacuum pumps when preventive, breakdown, and preventive maintenance strategy is adopted (3-year operating period); different efficiency erosion models across the maintenance strategy and same efficiency drop over the operating period.

Strategy	**Parameter**		**Cost**
Preventive	N_1	36	\$7,200
	N_2	6	\$1,800
	$E_{healthy}$	120,960 kWh	\$24,168
	ΔE_{cubic} (when $\Delta\eta$ = 40%)	30,488 kWh	\$7,284
	EI_{CO2}	121,250 $kgCO_2$	\$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 40%)	30,561 $kgCO_2$	\$1,096
	Total Cost		**\$45,186**
Breakdown	$E_{healthy}$	120,960 kWh	\$24,168
	$\Delta E_{exponential}$ (where $\Delta\eta$ = 40%)	35,341 kWh	\$8,724
	EI_{CO2}	121,250 $kgCO_2$	\$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 40%)	35,426 $kgCO_2$	\$1,313
	$T_{refurbishment}$		\$6,000
	Total Cost		**\$43,843**
Predictive	N_1	12	\$2,400
	N_2	2	\$600
	$E_{healthy}$	120,960 kWh	\$24,168
	$\Delta E_{quadratic}$ (when $\Delta\eta$ = 40%)	20,809 kWh	\$5,004
	EI_{CO2}	121,250 $kgCO_2$	\$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 40%)	20,859 $kgCO_2$	\$753
	$T_{hardware}$		\$6,231
	Total Cost		**\$42,794**

In Table 9.3 and Table 9.4, based on the four efficiency erosion models, the additional amounts of cost and CO_2 emission for different efficiency losses are presented. Here, the additional costs and the emission are incurred by an efficiency loss resulted from a fault in a motor. From the cubic, exponential, and quadratic efficiency erosion models, when $\Delta\eta$ is 35%, the amounts of additional cost and emission due to the efficiency loss are \$2,428/36,546kgCO_2, \$2,908/43,776kgCO_2, and \$1,668/25,108kgCO_2. Then, the total cost of each maintenance strategy can be calculated using Eqs. (9.26), (9.27), and (9.28), and they are presented in Table 9.5. As shown in the table, predictive maintenance is the most cost-effective option among the three strategies, followed by breakdown maintenance and preventive maintenance. Even if there is a substantial initial cost for the hardware purchase, a considerable amount of costs is saved from fewer maintenance works and less energy consumption in the predictive maintenance strategy. If excluding the emission associated costs (i.e., without EI_{CO2} and ΔEI_{CO2} in the cost models), the total costs of preventive, breakdown, and predictive maintenance are \$40,452, \$38,892, and \$38,403, which leads to the same conclusion that the preventive maintenance strategy is still the most cost-effective option.

Case 2: same efficiency erosion models across the maintenance strategies and different efficiency drops over the operating period

In this case, the cubic efficiency erosion model is selected for all maintenance strategies to calculate the costs associated with additional amounts of energy consumption and CO_2 emission. $\Delta\eta$ is assumed to be 20%, 40%, and 10% for preventive, breakdown, and predictive maintenance, respectively. This indicates the efficiency decreases more in the order of predictive, preventive, and breakdown maintenance over the 3 year-operating period.

From the cubic erosion model in Table 9.3 and Table 9.4, when $\Delta\eta$ is 20%, 40%, and 10% (i.e., the efficiency losses when preventive, breakdown, and preventive maintenance are employed), the amounts of additional cost and emission are \$1,027/15,463kgCO_2, \$2,428/36,546kgCO_2, and \$478/7,196kgCO_2. The total costs of each maintenance strategy are calculated using Eqs. (9.26), (9.27), and (9.28), and they are presented in Table 9.6. As shown in the table, the predictive maintenance is the most cost-effective option among the three strategies, which is the same result with the previous case. However, unlike the previous case, the preventive maintenance is more cost-effective than the breakdown maintenance. If excluding the emission associated costs (i.e., without EI_{CO2} and ΔEI_{CO2} in the cost model), the total cost of preventive, breakdown, and predictive maintenance costs are $36,249, $37,452, and $34,833, and the preventive maintenance is still the most cost-effective option.

Table 9.6 Comparison of total maintenance cost of three vacuum pumps when preventive, breakdown, and preventive maintenance strategy is adopted (3-year operating period); same efficiency erosion models across the maintenance strategies and different efficiency drops over the operating period.

Strategy	Parameter		Cost
Preventive	N_1	36	$7,200
	N_2	6	$1,800
	$E_{healthy}$	120,960 kWh	$24,168
	ΔE_{cubic} (when $\Delta\eta$ = 20%)	15,426 kWh	$3,081
	EI_{CO2}	121,250 $kgCO_2$	$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 20%)	15,463 $kgCO_2$	$464
	Total Cost		**$40,351**
Breakdown	$E_{healthy}$	120,960 kWh	$24,168
	ΔE_{cubic} (when $\Delta\eta$ = 40%)	36,485 kWh	$7,284
	EI_{CO2}	121,250 $kgCO_2$	$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 40%)	36,546 $kgCO_2$	$1,096
	$T_{refurbishment}$		$6,000
	Total Cost		**$42,186**
Predictive	N_1	12	$2,400
	N_2	2	$600
	$E_{healthy}$	120,960 kWh	$24,168
	ΔE_{cubic} (when $\Delta\eta$ = 10%)	7,179 kWh	$1,434
	EI_{CO2}	121,250 $kgCO_2$	$3,638
	ΔEI_{CO2} (when $\Delta\eta$ = 10%)	7,196 $kgCO_2$	$216
	$T_{hardware}$		$6,231
	Total Cost		**$38,687**

10. SUMMARY AND CONCLUSION

Over the years, the smart manufacturing has been empowered by integrating cyber- and physical- systems with the advancement of computing infrastructures, AI, big data, cloud computing, IoT platform, etc. These new technologies help construct an integrative and collaborative system that responds in real time to meet changing conditions in the factory, supply network, and customer demand. The integrative and collaborative system is tremendously transforming the manufacturing plants, and equipment maintenance is one area where smart manufacturing can greatly improve in terms of cost, productivity, and product quality.

A maintenance practice in a factory has evolved from breakdown, preventive, to predictive maintenance over the years. The goal of maintenance is to ensure consistent production and operational excellence. In smart manufacturing, to achieve this goal, a health condition of the equipment is continuously monitored using sensor(s), and a necessary action is taken in a timely manner before catastrophic equipment failures (i.e., equipment health management). Beyond the health management, in equipment wellness, there is an active process of becoming aware of the health condition and of making choices toward a healthy and fulfilling life (i.e., continuous improvement). That means the root cause of failure is identified to prevent reoccurring problems, and the economic and environmental impacts associated with a defect/failure are also estimated to support decision making in physical asset management.

In this dissertation, the AI-driven predictive models for the condition monitoring of mechanical systems have been presented to study the wellness of mechanical systems. The proposed methods were applied to (1) multi-sensor signals collected during milling operations to quantify the level of tool wear in a machining process, (2) acceleration time-frequency images (spectrograms) to detect different faults from a power transmission mini-motor testbed based on a convolutional neural network (CNN), (3) vibration time series data to detect faults for various rotation speeds using a long short term memory (LSTM) augmented with an attention mechanism, and (4) statistical features extracted from vibration data to predict the time to failure of rotating machinery. To demonstrate the effectiveness of the proposed model, the methods were validated using the experimental data obtained from various sources.

Also, this dissertation has investigated the lifespan of products under different categories and the effectiveness of maintenance in their use stage. As a case study, vacuum pumps were examined. In this work, a power efficiency loss is used as a metric for decision making between repair and replacement. During the progress of a fault in a motor over operation hours, more power is lost in power transmission from an electric motor to driven equipment. Therefore, the additional energy consumption in a faulty motor compared to a healthy motor is presented as a function of efficiency loss, and the different efficiency erosion models are considered. Additionally, the economic loss and environmental impact (CO2 emission) associated with the additional energy consumption were quantified, and the total costs of different maintenance strategies were compared.

To sum up, this dissertation conducts the assessment of technical, environmental and economic performance of the AI-driven method for the wellness of mechanical systems (i.e., manufacturing equipment). The proposed method can help reduce the machine downtime and increase the RUL/TTF of a component, and these will save a cost by optimizing maintenance task and supply chain management while ensuring machine safety. Also, the method can play as a bridge to connect a large-scale machinery data and machine health condition, and ultimately, the proposed methods could be a promising solution for any types of condition monitoring problems. One potential long-term outcome of these works could be extending the life of manufacturing equipment by identifying root causes of machine failure at an early stage which is desirable from economic and environmental standpoints.

REFERENCES

[1] Progress towards the Sustainable Development. 2019.

[2] Simon T, Yang Y, Lee WJ, Zhao J, Li L, Zhao F. Reusable unit process life cycle inventory for manufacturing: stereolithography. Prod Eng 2019;13:675–84. https://doi.org/10.1007/s11740-019-00916-0.

[3] Cooper T. Beyond Recycling The longer life option. 1994.

[4] Gnanapragasam A, Cole C, Singh J, Cooper T. Consumer Perspectives on longevity and Reliability: A National Study of Purchasing Factors Across Eighteen Product Categories. Procedia CIRP, 2018, p. 910–5. https://doi.org/10.1016/j.procir.2017.11.151.

[5] Hiruta T, Uchida T, Yuda S, Umeda Y. A design method of data analytics process for condition based maintenance. CIRP Ann - Manuf Technol 2019;68:145–8. https://doi.org/10.1016/j.cirp.2019.04.049.

[6] Roy R, Stark R, Tracht K, Takata S, Mori M. Continuous maintenance and the future – Foundations and technological challenges. CIRP Ann - Manuf Technol 2016;65:667–88. https://doi.org/10.1016/j.cirp.2016.06.006.

[7] Environmental Protection Agency. Time Lag and Composition of Durable Goods. 2014.

[8] National Associtation of Home Builders. Study of Life Expectancy of Home Components. 2007.

[9] U.S. Government’s Office of Management and Budget. Useful Life and Disposal Value Table. 2003.

[10] Kara S, Manmek S, Herrmann C. Global manufacturing and the embodied energy of products. CIRP Ann - Manuf Technol 2010;59:29–32. https://doi.org/10.1016/j.cirp.2010.03.004.

[11] Ezema IC, Fagbenle OI, Olotuah AO. Estimating embodied energy in residential buildings in a nigerian context. Int J Appl Eng Res 2015;10:44140–9.

[12] Gazis E, Harrison GP. Life cycle energy and carbon analysis of domestic combined heat and power generators. IEEE TRONDHEIM POWERTECH, 2011, p. 1–6. https://doi.org/10.1109/PTC.2011.6019410.

[13] Life Cycle of a Computer n.d. https://sustainablecomputing.umich.edu/knowledge/life-cycle.php (accessed January 18, 2020).

[14] Yu J, Williams E, Ju M. Analysis of material and energy consumption of mobile phones in China. Energy Policy 2010;38:4135–41. https://doi.org/10.1016/j.enpol.2010.03.041.

[15] Sullivan JL, Hu J. Life cycle energy analysis for automobiles. 1995.

[16] Kleine R De. Life Cycle Optimization of Residential Air Conditioner Replacement. 2009.

[17] Tao F, Qi Q, Liu A, Kusiak A. Data-driven smart manufacturing. J Manuf Syst 2018;48:157–69. https://doi.org/10.1016/j.jmsy.2018.01.006.

[18] DeVor RE, Chang T, Sutherland JW. Statistical Quality Design and Control. 2nd ed. New York, NY, USA: Prentice Hall; 2006. https://doi.org/10.2307/1269537.

[19] Thoben K-D, Wiesner S, Wuest T. “Industrie 4.0” and Smart Manufacturing – A Review of Research Issues and Application Examples. Internantional J Autom Technol 2017;11:4–19. https://doi.org/10.20965/ijat.2017.p0004.

[20] Lu SCY. Machine learning approaches to knowledge synthesis and integration tasks for advanced engineering automation. Comput Ind 1990;15:105–20. https://doi.org/10.1016/0166-3615(90)90088-7.

[21] Wuest T, Weimer D, Irgens C, Thoben K-D. Machine learning in manufacturing: advantages, challenges, and applications. Prod Manuf Res 2016;4:23–45. https://doi.org/10.1080/21693277.2016.1192517.

[22] Moyne J, Iskandar J. Big Data Analytics for Smart Manufacturing: Case Studies in Semiconductor Manufacturing. Processes 2017;5:1–20. https://doi.org/10.3390/pr5030039.

[23] Lee J, Bagheri B, Kao HA. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. Manuf Lett 2015;3:18–23. https://doi.org/10.1016/j.mfglet.2014.12.001.

[24] Kaiser KA, Gebraeel NZ. Predictive maintenance management using sensor-based degradation models. IEEE Trans Syst Man, Cybern Part ASystems Humans 2009;39:840–9. https://doi.org/10.1109/TSMCA.2009.2016429.

[25] Coleman C, Damofaran S, Deuel E. Predictive maintenance and the smart factory. 2017.

[26] Kang HS, Lee JY, Choi S, Kim H, Park JH, Son JY, et al. Smart Manufacturing : Past Research , Present Findings , and Future Directions. Int J Precis Eng Manuf - Green Technol 2016;3:111–28. https://doi.org/10.1007/s40684-016-0015-5.

[27] Shin I, Lee J, Lee JY, Jung K, Kwon D, Youn BD, et al. A Framework for Prognostics and Health Management Applications toward Smart Manufacturing Systems. Int J Precis Eng Manuf - Green Technol 2018;5:535–54. https://doi.org/10.1007/s40684-018-0055-0.

[28] Sondalini M. Plant and Equipment Wellness: A Process for Exceptional Equipment Reliability and Maximum Life Cycle Profits. Engineers Media; 2009.

[29] EERE. O&M Best Practices Guide,: Chapter 5 Types of Maintenance Programs 2010:1–9.

[30] Agapiou JS, Stephenson DA. Analytical and Experimental Studies of Drill Temperatures. J Eng Ind 1994;116:54–60. https://doi.org/10.1115/1.2901809.

[31] Kothuru A, Nooka SP, Liu R. Application of Audible Sound Signals for Tool Wear Monitoring Using Machine Learning Techniques in End Milling. Int J Adv Manuf Technol 2018;95:3797–808. https://doi.org/https://doi.org/10.1007/s00170-017-1460-1.

[32] Cao H, Zhang X, Chen X. The concept and progress of intelligent spindles: A review. Int J Mach Tools Manuf 2017;112:21–52. https://doi.org/10.1016/j.ijmachtools.2016.10.005.

[33] Abele E, Altintas Y, Brecher C. Machine tool spindle units. CIRP Ann - Manuf Technol 2010;59:781–802. https://doi.org/10.1016/j.cirp.2010.05.002.

[34] Byrne G, O'Donnell GE. An Integrated Force Sensor Solution for Process Monitoring of Drilling Operations. CIRP Ann 2007;56:89–92. https://doi.org/10.1016/j.cirp.2007.05.023.

[35] Hoshi T. Damage monitoring of ball bearing. CIRP Ann Technol 2006;55:427–30. https://doi.org/10.1016/S0007-8506(07)60451-X.

[36] Qiu M, Chen L, Li Y, Yan J. Bearing Tribology: Principles and Applications. Springer; 2016.

[37] Peng Y, Dong M, Zuo MJ. Current status of machine prognostics in condition-based maintenance: A review. Int J Adv Manuf Technol 2010;50:297–313. https://doi.org/10.1007/s00170-009-2482-0.

[38] Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX. Deep learning and its applications to machine health monitoring. Mech Syst Signal Process 2019;115:213–37. https://doi.org/10.1016/j.ymssp.2018.05.050.

[39] Jia F, Lei Y, Guo L, Lin J, Xing S. A neural network constructed by deep learning technique and its application to intelligent fault diagnosis of machines. Neurocomputing 2018;272:619–28. https://doi.org/10.1016/j.neucom.2017.07.032.

[40] Pimenov D. Y, Bustillo A, Mikolajczyk T. Artificial intelligence for automatic prediction of required surface roughness by monitoring wear on face mill teeth. J Intell Manuf 2018;29:1045–61. https://doi.org/10.1007/s10845-017-1381-8.

[41] Lee WJ, Mendis GP, Triebe M, Sutherland J. Monitoring of a machining process using kernel principal component analysis and kernel density estimation. J Intell Manuf 2020;31:1175–89.

[42] Lee WJ, Wu H, Yun H, Kim H, Jun MBG, Sutherland JW. Predictive maintenance of machine tool systems using artificial intelligence techniques applied to machine condition data. Procedia CIRP 2019;80:506–11. https://doi.org/10.1016/j.procir.2018.12.019.

[43] Ren L, Sun Y, Cui J, Zhang L. Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. J Manuf Syst 2018;48:71–7. https://doi.org/10.1016/j.jmsy.2018.04.008.

[44] Huo Z, Zhang Y, Francq P, Shu L, Huang J. Incipient Fault Diagnosis of Roller Bearing Using Optimized Wavelet Transform Based Multi-Speed Vibration Signatures. IEEE Access 2017;5:19442–56. https://doi.org/10.1109/ACCESS.2017.2661967.

[45] Verstraete D, Ferrada A, Droguett EL, Meruane V, Modarres M. Deep Learning Enabled Fault Diagnosis Using Time-Frequency Image Analysis of Rolling Element Bearings. Shock Vib 2017;2017:1–17.

[46] Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX. Deep learning and its applications to machine health monitoring. Mech Syst Signal Process 2019;115:213–37. https://doi.org/10.1016/j.ymssp.2018.05.050.

[47] Zhu X, Bain M. B-CNN: Branch Convolutional Neural Network for Hierarchical Classification. CoRR 2017;abs/1709.0.

[48] LeCun Y, Bottou L, Bengio Y, Haffiner P. Gradient-Based Learning Applied to Document Recognition. Proc. IEEE, vol. 86, 1998, p. 2278–324.

[49] Li LJ, Su H, Lim Y, Fei-Fei L. Objects as attributes for scene classification. 2012. https://doi.org/10.1007/978-3-642-35749-7_5.

[50] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going Deeper with Convolutions. 2015 IEEE Conf Comput Vis Pattern Recognit 2015;9:1–9. https://doi.org/10.1002/jctb.4820.

[51] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. 2016 IEEE Conf Comput Vis Pattern Recognit 2016:770–8.

[52] Wu H, Yu Z, Wang Y. Real-time FDM machine condition monitoring and diagnosis based on acoustic emission and hidden semi-Markov model. Int J Adv Manuf Technol 2017;90:2027–36. https://doi.org/10.1007/s00170-016-9548-6.

[53] Pezzani CM, Fontana JM, Donolo PD, De Angelo CH, Bossio GR, Silva LI. SVM-Based System for Broken Rotor Bar Detection in Induction Motors. 2018 IEEE ANDESCON, IEEE; 2018, p. 1–6. https://doi.org/10.1109/ANDESCON.2018.8564627.

[54] Bhat NN, Dutta S, Vashisth T, Pal S, Pal SK, Sen R. Tool condition monitoring by SVM classification of machined surface images in turning. Int J Adv Manuf Technol 2016;83:1487–502. https://doi.org/10.1007/s00170-015-7441-3.

[55] Kane P V., Andhare AB. Application of psychoacoustics for gear fault diagnosis using artificial neural network. J Low Freq Noise, Vib Act Control 2016;35:207–20. https://doi.org/10.1177/0263092316660915.

[56] Jing L, Zhao M, Li P, Xu X. A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. Measurement 2017;111:1–10. https://doi.org/10.1016/j.measurement.2017.07.017.

[57] Chen Z, Li C, Sanchez R. Gearbox Fault Identification and Classification with Convolutional Neural Networks. Shock Vib 2015;2015:1–10.

[58] Ince T, Kiranyaz S, Member S, Eren L. Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks. IEEE Trans Ind Electron 2016;63:7067–75. https://doi.org/10.1109/TIE.2016.2582729.

[59] Jing L, Zhao M, Li P, Xu X. A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. Meas J Int Meas Confed 2017;111:1–10. https://doi.org/10.1016/j.measurement.2017.07.017.

[60] Eren L, Turker I, Kiranyaz S. A Generic Intelligent Bearing Fault Diagnosis System Using Compact Adaptive 1D CNN Classifier. J Signal Process Syst 2019;91:179–89.

[61] Janssens O, Slavkovikj V, Vervisch B, Stockman K, Loccufier M, Verstockt S, et al. Convolutional Neural Network Based Fault Detection for Rotating Machinery. J Sound Vib 2016;377:331–45. https://doi.org/10.1016/j.jsv.2016.05.027.

[62] Cacciola S, Agud IM, Bottasso CL. Detection of rotor imbalance, including root cause, severity and location. J. Phys. Conf. Ser., vol. 753, 2016. https://doi.org/10.1088/1742-6596/753/7/072003.

[63] Jia F, Lei Y, Lin J, Zhou X, Lu N. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. Mech Syst Signal Process 2016;72–73:303–15. https://doi.org/10.1016/j.ymssp.2015.10.025.

[64] Khan S, Yairi T. A review on the application of deep learning in system health management. Mech Syst Signal Process 2018;107:241–65. https://doi.org/10.1016/j.ymssp.2017.11.024.

[65] Kusiak A. Convolutional and generative adversarial neural networks in manufacturing. Int J Prod Res 2019;0:1–11. https://doi.org/10.1080/00207543.2019.1662133.

[66] Park J, Hamadache M, Ha JM, Kim Y, Na K, Youn BD. A positive energy residual (PER) based planetary gear fault detection method under variable speed conditions. Mech Syst Signal Process 2019;117:347–60.

[67] Wang S, Wan J, Zhang D, Li D, Zhang C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. Comput Networks 2016;101:158–68. https://doi.org/10.1016/j.comnet.2015.12.017.

[68] Wang J, Sutherland JW. A Data-Driven Model for Energy Consumption in the Sintering Process 2018;138:1–12. https://doi.org/10.1115/1.4033661.

[69] Elangovan M, Sakthivel NR, Saravanamurugan S, Nair BB, Sugumaran V. Machine Learning Approach to the Prediction of Surface Roughness using Statistical Features of Vibration Signal Acquired in Turning. Procedia - Procedia Comput Sci 2015;50:282–8. https://doi.org/10.1016/j.procs.2015.04.047.

[70] Shaban Y, Yacout S, Balazinski M. Tool Wear Monitoring and Alarm System Based on Pattern Recognition With Logical Analysis of Data. J Manuf Sci Eng Trans ASME 2015;137. https://doi.org/10.1115/1.4029955.

[71] Segreto T, Simeone A, Teti R. Multiple sensor monitoring in nickel alloy turning for tool wear assessment via sensor fusion. Procedia CIRP, vol. 12, Elsevier B.V.; 2013, p. 85–90. https://doi.org/10.1016/j.procir.2013.09.016.

[72] Mesina OS, Langari R. A Neuro-Fuzzy System for Tool Condition Monitoring in Metal Cutting. J Manuf Sci Eng 2002;123:312. https://doi.org/10.1115/1.1363599.

[73] Wu D, Jennings C, Terpenny J, Kumara S, Gao RX. Cloud-Based Parallel Machine Learning for Tool Wear Prediction. J Manuf Sci Eng 2018;140:041005. https://doi.org/10.1115/1.4038002.

[74] Hui Y V, Leung LC, Linn R. Optimal machining conditions with costs of quality and tool maintenance for turning 2010;7543. https://doi.org/10.1080/00207540010011063.

[75] Abellan-Nebot JV, Subirón RF. A review of machining monitoring systems based on artificial intelligence process models. Int J Adv Manuf Technol 2010;47:237–57. https://doi.org/10.1007/s00170-009-2191-8.

[76] Vapnik VN. An overview of statistical learning theory. IEEE Trans Neural Networks 1999;10:988–99. https://doi.org/10.1109/72.788640.

[77] Agogino A, Goebel K. Milling Data Set. 2007.

[78] Goebel K. Management of uncertainty in sensor validation , sensor fusion , and diagnosis of mechanical systems using soft computing techniques. University of California at Berkeley, 1996.

[79] Yu J. Machine Tool Condition Monitoring Based on an Adaptive Gaussian Mixture Model. J Manuf Sci Eng 2012;134:1–13. https://doi.org/10.1115/1.4006093.

[80] Joliffe IT, Morgan BJT. Principal component analysis and exploratory factor analysis. Stat Methods Med Res 1992;1:69–95.

[81] Kulkarni SR, Harman G. Statistical Learning Theory : A Tutorial. Wiley Interdiscip Rev Comput Stat 2011;3:543–56.

[82] Amami R, Ayed D Ben, Ellouze N. Practical Selection of SVM Supervised Parameters with Different Feature Representations for Vowel Recognition. Wiley Interdiscip Rev Comput Stat 2013;7:418–24. https://doi.org/10.4156/jdcta.vol7.issue9.50.

[83] Kramer O. Machine learning for evolution strategies. vol. 20. Springer; 2016.

[84] Kramer O, Ciaurri DE, Koziel S. Derivative-free optimization. Comput. Optim. methods algorithms, Springer; 2011, p. 61–83.

[85] Back T. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford university press; 1996.

[86] Taguchi G, Elsayed EA, Hsiang TC. Quality engineering in production systems. McGraw-Hill College; 1989.

[87] Wysk RA, Chang TC. An Introduction to Automated Process Planning Systems. Prentice Hall 1985;5:162–85.

[88] Goel T, Vaidyanathan R, Haftka RT, Shyy W, Queipo N V, Tucker K. Response surface approximation of Pareto optimal front in multi-objective optimization. Comput Methods Appl Mech Eng 2007;196:879–93.

[89] Dutta S, Pal SK, Sen R. Tool Condition Monitoring in Turning by Applying Machine Vision. J Manuf Sci Eng 2015;138:051008. https://doi.org/10.1115/1.4031770.

[90] Kurada S, Bradley C. A review of machine vision sensors for tool condition monitoring. Comput Ind 1997;34:55–72. https://doi.org/10.1016/s0166-3615(96)00075-9.

[91] Cho S, Binsaeid S, Asfour S. Design of multisensor fusion-based tool condition monitoring system in end milling. Int J Adv Manuf Technol 2010;46:681–94. https://doi.org/10.1007/s00170-009-2110-z.

[92] Mesina OS, Langari R. A Neuro-Fuzzy System for Tool. ASME J Manuf Sci Eng 2000;123:312–8. https://doi.org/10.1115/1.1363599.

[93] Dimla Snr. DE. The correlation of vibration signal features to cutting tool wear in a metal turning operation. Int J Adv Manuf Technol 2002;19:705–13. https://doi.org/10.1007/s001700200080.

[94] Li X, Yuan Z. Tool Wear Monitoring with Wavelet Packet Transform Fuzzy Clustering Method. Wear 1989;219:145–54.

[95] Li X. Detection of tool flute breakage in end milling using feed-motor current signatures. IEEE/ASME Trans. Mechatronics, vol. 6, 2001, p. 491–8. https://doi.org/10.1109/3516.974863.

[96] Chen S, Jen YW. Data fusion neural network for tool condition monitoring in CNC milling machining. Int J Mach Tools Manuf 2000;40:381–400.

[97] Wang G, Zhang Y, Liu C, Xie Q, Xu Y. A new tool wear monitoring method based on multi-scale PCA. J Intell Manuf 2019;30:113–22. https://doi.org/10.1007/s10845-016-1235-9.

[98] Ketelaere B, Hubert M, Schmitt E. Overview of PCA-based statistical process-monitoring methods for time-dependent, high-dimensional data. J Qual Technol 2015;47:318–35. https://doi.org/10.1080/00224065.2015.11918137.

[99] Abdi H, Williams LJ. Principal component analysis. Wiley Interdiscip Rev Comput Stat 2010;2:433–59. https://doi.org/10.1002/wics.101.

[100] Zhang X, Gao H, Xu M. On-line Tool Condition Monitoring Based on PCA and Integrated Neural Networks for Cold Blast Machining Operation. Int Conf Appl Mech Mech Mater Eng 2016. https://doi.org/10.12783/dtmse/ammme2016/6891.

[101] Lever J, Krzywinski M, Altman N. Points of Significance: Principal component analysis. Nat Methods 2017;14:641–2. https://doi.org/10.1038/nmeth.4346.

[102] Schölkopf B, Smola A, Müller K-R. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. Neural Comput 1998;10:1299–319. https://doi.org/10.1162/089976698300017467.

[103] Choi SW, Lee C, Lee J, Hyun J, Lee I. Fault detection and identification of nonlinear processes based on kernel PCA. Chemom Intell Lab Syst 2005;75:55–67. https://doi.org/10.1016/j.chemolab.2004.05.001.

[104] Lee JM, Yoo CK, Choi SW, Vanrolleghem PA, Lee IB. Nonlinear process monitoring using wavelet kernel principal component analysis. Chem Eng Sci 2004;59:223–34. https://doi.org/10.1109/ICSAI.2012.6223652.

[105] Vanhatalo E, Kulahci M, Bergquist B. On the Structure of Dynamic Principal Component Analysis Used in Statistical Process Monitoring. Chemom Intell Lab Syst 2017;167:1–11. https://doi.org/10.1016/j.chemolab.2017.05.016.

[106] Jiang Q, Yan X. Weighted kernel principal component analysis based on probability density estimation and moving window and its application in nonlinear chemical process monitoring. Chemom Intell Lab Syst 2013;127:121–31. https://doi.org/10.1016/j.chemolab.2013.06.013.

[107] Wang Q. Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models. CoRR, 2014.

[108] Amami R, Ben Ayed D, Ellouze N. Practical Selection of SVM Supervised Parameters with Different Feature Representations for Vowel Recognition. Int J Digit Content Technol Its Appl 2013;7:418–24. https://doi.org/10.4156/jdcta.vol7.issue9.50.

[109] Scholkopf B, Smola a J, Muller KR. Kernel Principal Component Analysis. Int Conf Artif Neural Networks 1997;1327:583–8. https://doi.org/10.1162/089976698300017467.

[110] Samuel RT, Cao Y. Fault detection in a multivariate process based on kernel PCA and kernel density estimation. Syst Sci Control Eng 2014;4:165–74. https://doi.org/10.1109/IConAC.2014.6935477.

[111] Chen Q, Wynne RJ, Goulding P, Sandoz D. The application of principal component analysis and kernel density estimation to enhance process monitoring. Control Eng Pr 2000;8:531–43.

[112] Silverman BW. Density Estimation for Statistics and Data Analysis 2002:1–22.

[113] Qehaja N, Kyçyku A. Tool Life Modeling Based on Cutting Parameters and Work Material Hardness in Turning Process. Sci. Proc. XIV Int. Congr., vol. IV, 2017, p. 278–81.

[114] Twardowski P, Legutko S, Krolczyk GM, Hloch S. Investigation of wear and tool life of coated carbide and cubic boron nitride cutting tools in high speed milling. Adv Mech Eng 2015;7:1–9. https://doi.org/10.1177/1687814015590216.

[115] Tao F, Qi Q, Liu A, Kusiak A. Data-driven smart manufacturing. J Manuf Syst 2018;48:157–69. https://doi.org/10.1016/j.jmsy.2018.01.006.

[116] Lee WJ, Wu H, Yun H, Kim H, Jun MBG, Sutherland JW. Predictive Maintenance of Machine Tool Systems Using Artificial Intelligence Techniques Applied to Machine Condition Data. Procedia CIRP 2019;80:506–11. https://doi.org/10.1016/j.procir.2018.12.019.

[117] Ren L, Cui J, Sun Y, Cheng X. Multi-bearing remaining useful life collaborative prediction: A deep learning approach. J Manuf Syst 2017;43:248–56. https://doi.org/10.1016/j.jmsy.2017.02.013.

[118] Yu J. Health condition monitoring of machines based on hidden markov model and contribution analysis. IEEE Trans Instrum Meas 2012;61:2200–11. https://doi.org/10.1109/TIM.2012.2184015.

[119] Mahmood F, Toots M, Öfverstedt L-G, Skoglund U. Algorithm and Architecture Optimization for 2D Discrete Fourier Transforms with Simultaneous Edge Artifact Removal. Int. J. Reconfigurable Comput., vol. 3, 2018, p. 1–17.

[120] Hoang D, Kang H. ScienceDirect Rolling element bearing fault diagnosis using convolutional neural network and vibration image. Cogn Syst Res 2019;53:42–50. https://doi.org/10.1016/j.cogsys.2018.03.002.

[121] Sharma N, Jain V, Mishra A. Analysis of Convolutional Neural Networks for Document Image Classification. Procedia Comput. Sci., vol. 132, Elsevier B.V.; 2018, p. 377–84. https://doi.org/10.1016/j.procs.2018.05.198.

[122] Scherer D, Andreas M, Behnke S. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. Int. Conf. Artif. neural networks, 2010, p. 92–101.

[123] Bouvrie J. Notes on Convolutional Neural Networks 2006.

[124] Krizhevsky A, Ilya Sutskever, Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks Alex. Proc IEEE 2012;86:2278–323. https://doi.org/10.1109/5.726791.

[125] Girdhar P, Scheffer C. Practical Machinery Vibration Analysis and Predictive Maintenance. 2004. https://doi.org/http://dx.doi.org/10.1016/B978-075066275-8/50001-1.

[126] Rajendra BR, Bhaskar S. Condition Monitoring of Gear Box by Using Motor. Int J Sci Res Publ 2013;3:1–6.

[127] U.S. DOE. Energy Tips : MOTOR SYSTEMS. 2012.

[128] Jung J, Member S, Lee S Bin, Member S, Lim C, Cho C, et al. Induction Motors With Sleeve Bearings. IEEE Trans Energy Convers 2016;31:1377–86. https://doi.org/10.1109/TEC.2016.2583473.

[129] Ganchev P, Malehorn D, Bigbee WL, Gopalakrishnan V. Transfer learning of classification rules for biomarker discovery and verification from molecular profiling studies. J Biomed Inform 2011;44:S17–23. https://doi.org/10.1016/j.jbi.2011.04.009.

[130] Kim DB. An approach for composing predictive models from disparate knowledge sources in smart manufacturing environments. J Intell Manuf 2019;30:1999–2012. https://doi.org/10.1007/s10845-017-1366-7.

[131] CESMII n.d. https://www.cesmii.org/cesmii-announces-first-rfp-project-selections (accessed July 28, 2019).

[132] Seevers JP, Johst J, Weiß T, Meschede H, Hesselbach J. Automatic time series segmentation as the basis for unsupervised, non-intrusive load monitoring of machine tools. Procedia CIRP, vol. 81, Elsevier B.V.; 2019, p. 695–700. https://doi.org/10.1016/j.procir.2019.03.178.

[133] LSM - Purdue University n.d. https://engineering.purdue.edu/LSM (accessed September 15, 2020).

[134] Zheng S, Ristovski K, Farahat A, Gupta C. Long Short-Term Memory Network for Remaining Useful Life estimation. 2017 IEEE Int Conf Progn Heal Manag ICPHM 2017 2017:88–95. https://doi.org/10.1109/ICPHM.2017.7998311.

[135] Zhang J, Wang P, Yan R, Gao RX. Long short-term memory for machine remaining life prediction. J Manuf Syst 2018;48:78–86. https://doi.org/10.1016/j.jmsy.2018.05.011.

[136] Peng ZK, Chu FL. Application of the wavelet transform in machine condition monitoring and fault diagnostics : a review with bibliography 2004;18:199–221. https://doi.org/10.1016/S0888-3270(03)00075-X.

[137] Zhang Z, Wang Y, Wang K. Fault diagnosis and prognosis using wavelet packet decomposition, Fourier transform and artificial neural network. J Intell Manuf 2013;24:1213–27. https://doi.org/10.1007/s10845-012-0657-2.

[138] Colah n.d. https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed August 12, 2019).

[139] NN SVG n.d. http://alexlenail.me/NN-SVG/LeNet.html (accessed August 12, 2019).

[140] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. Int. Conf. Learn. Represent., 2015, p. 1–15.

[141] Hu Q, Si XS, Zhang QH, Qin AS. A rotating machinery fault diagnosis method based on multi-scale dimensionless indicators and random forests. Mech Syst Signal Process 2020;139:106609.

[142] Moshrefzadeh A. Condition monitoring and intelligent diagnosis of rolling element bearings under constant/variable load and speed conditions. Mech Syst Signal Process 2021;149:107153. https://doi.org/10.1016/j.ymssp.2020.107153.

[143] Qin A, Zhang Q, Hu Q, Sun G, He J, Lin S. Remaining Useful Life Prediction for Rotating Machinery Based on Optimal Degradation Indicator. Shock Vib 2017.

[144] Bektas O, Jones JA, Sankararaman S, Roychoudhury I, Goebel K. A neural network filtering approach for similarity-based remaining useful life estimation. Int J Adv Manuf Technol 2019;101:87–103.

[145] Huang Z, Xu Z, Ke X, Wang W, Sun Y. Remaining useful life prediction for an adaptive skew-Wiener process model. Mech Syst Signal Process 2017;87:294–306. https://doi.org/10.1016/j.ymssp.2016.10.027.

[146] Kim S, Kim NH, Choi JH. Prediction of remaining useful life by data augmentation technique based on dynamic time warping. Mech Syst Signal Process 2020;136:106486. https://doi.org/10.1016/j.ymssp.2019.106486.

[147] Lei Y, Yang B, Jiang X, Jia F, Li N, Nandi AK. Applications of machine learning to machine fault diagnosis: A review and roadmap. Mech Syst Signal Process 2020;138:106587. https://doi.org/10.1016/j.ymssp.2019.106587.

[148] Lee WJ, Mendis GP, Sutherland JW. Development of an intelligent tool condition monitoring system to identify manufacturing tradeoffs and optimal machining conditions. Procedia Manuf 2019;33:256–63.

[149] Ren L, Cui J, Sun Y, Cheng X. Multi-bearing remaining useful life collaborative prediction: A deep learning approach. J Manuf Syst 2017;43:248–56.

[150] Ben Ali J, Chebel-Morello B, Saidi L, Malinowski S, Fnaiech F. Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network. Mech Syst Signal Process 2015;56:150–72.

[151] Guo L, Li N, Jia F, Lei Y, Lin J. A recurrent neural network based health indicator for remaining useful life prediction of bearings. Neurocomputing 2017;240:98–109.

[152] Li X, Duan F, Mba D, Bennett I. Multidimensional prognostics for rotating machinery: A review. Adv Mech Eng 2017;9:1–20.

[153] Yang F, Habibullah MS, Shen Y. Remaining useful life prediction of induction motors using nonlinear degradation of health index. Mech Syst Signal Process 2021;148:107183. https://doi.org/10.1016/j.ymssp.2020.107183.

[154] Li X, Yang Y, Bennett I, Mba D. Condition monitoring of rotating machines under time-varying conditions based on adaptive canonical variate analysis. Mech Syst Signal Process 2019;131:348–63. https://doi.org/10.1016/j.ymssp.2019.05.048.

[155] Hong S, Zhou Z. Application of Gaussian Process Regression for bearing degradation assessment. 2012 6th Int. Conf. New Trends Inf. Sci. Serv. Sci. Data Min., IEEE; 2012, p. 644–8.

[156] Choosing the Right Type of Accelerometer 2017. https://www.mouser.com/pdfdocs/choosing-the-right-accelerometer-white-paper.pdf (accessed September 15, 2020).

[157] Silverman BW. Density Estimation for Statistics and Data Analysis. New York: Chapman and Hall/CRC; 1986.

[158] Bishop CM. Pattern Recognition and Machine Learning. New York :Springer; 2006.

[159] Chati YS, Balakrishnan H. A Gaussian Process Regression approach to model aircraft engine fuel flow rate. 2017 ACM/IEEE 8th Int. Conf. Cyber-Physical Syst., IEEE; 2017, p. 131–40.

[160] Lee WJ, Wu H, Huang A, Sutherland JW. Learning via acceleration spectrograms of a DC motor system with application to condition monitoring. Int J Adv Manuf Technol 2020;106:803–16.

[161] Holt CC. Forecasting seasonals and trends by exponentially weighted moving averages. Int J Forecast 2004;20:5–10. https://doi.org/10.1016/j.ijforecast.2003.09.015.

[162] Saidur R. A review on electrical motors energy use and energy savings. Renew Sustain Energy Rev 2010;14:877–98. https://doi.org/10.1016/j.rser.2009.10.018.

[163] Mohsen MS, Akash BA. Energy analysis of the steel making industry. Int J Energy Res 1998;22:1049–54.

[164] Li Y, Liu M, Lau J, Zhang B. A novel method to determine the motor efficiency under variable speed operations and partial load conditions. Appl Energy 2015;144:234–40. https://doi.org/10.1016/j.apenergy.2015.01.064.

[165] Jones T, Lalemand T. Motor Efficiency, Selection, and Management. A Guid B Ind Effic Programs Consort Energy Effic Bost 2013:8–16.

[166] Mahlia TMI, Yanti PAA. Cost efficiency analysis and emission reduction by implementation of energy efficiency standards for electric motors. J Clean Prod 2010;18:365–74. https://doi.org/10.1016/j.jclepro.2009.11.005.

[167] Lu B, Habetler TG, Harley RG. A Nonintrusive and In-Service Motor-Efficiency Estimation Method Using Air-Gap Torque With Considerations of Condition Monitoring 2008;44:1666–74.

[168] Singh G, Anil TC, Naikan VNA. Efficiency monitoring as a strategy for cost effective maintenance of induction motors for minimizing carbon emission and energy consumption. Reliab Eng Syst Saf 2018;000:1–9. https://doi.org/10.1016/j.ress.2018.02.015.

[169] Lee WJ, Mendis GP, Sutherland JW. Development of an Intelligent Tool Condition Monitoring System to Identify Manufacturing Tradeoffs and Optimal Machining Conditions. Procedia Manuf., vol. 33, Elsevier B.V.; 2019, p. 256–63.

[170] QDP Drystar ® Vacuum Pumps Instruction Manual n.d. https://www.idealvac.com/files/manualsII/Edwards_DRYSTAR_QDP-40_QDP-80_Operating_Instructions.pdf (accessed March 30, 2021).

[171] US Department of Energy. Determining Electric Motor Load Ranges. vol. 1. 2014.

[172] Indiana Profile n.d. https://www.eia.gov/state/print.php?sid=IN (accessed March 11, 2021).

[173] Frequently Asked Questions (FAQs) - U.S. Energy Information Administration (EIA) n.d. https://www.eia.gov/tools/faqs/faq.php?id=74&t=11 (accessed March 11, 2021).

[174] Stiglitz JE, Stern N, Duan M, Edenhofer O, Giraud G, Heal GM, et al. Report of the high-level commission on carbon prices 2017.

[175] Lee WJ, Xia K, Denton NL, Ribeiro B, Sutherland JW. Development of a speed invariant deep learning model with application to condition monitoring of rotating machinery. J Intell Manuf 2020;32:393–406. https://doi.org/10.1007/s10845-020-01578-x.

APPENDIX

The detailed calculation to obtain Eq. (9.22) in Section 9.2 is presented here. First, using Eq. (9.18), ΔE_{cubic} can be calculated as follows

$$\Delta E_{cubic} = \int_0^T \Delta P_{cubic}(t)\, dt = \int_0^T \left(\frac{1}{\frac{d}{P_{\text{output}}}(t^3 - \frac{3}{2}Tt^2 + \frac{3}{4}T^2 t) + \frac{1}{P_{input\_t_1}}} - P_{input\_t_1}\right) dt. \tag{10.1}$$

Let's α and β are $\frac{d}{P_{output}}$ and $\frac{T^3\alpha}{8} + \frac{1}{P_{input\_t_1}}$, then, above equation can be rewritten as follows

$$\int_0^T \left(\frac{1}{\alpha(t-\frac{T}{2})^3 + \beta}\right) dt - \int_0^T P_{input\_t_1} dt = \int_0^T \left(\frac{1}{\alpha(t-\frac{T}{2})^3 + \beta}\right) dt - P_{input\_t_1} T. \tag{10.2}$$

To solve the integral part, substitute $x = (t - \frac{T}{2})$ and $dt = dx$ as follows

$$\int_0^T \left(\frac{1}{\alpha(t-\frac{T}{2})^3 + \beta}\right) dt = \int_{-T/2}^{T/2} \frac{1}{\alpha x^3 + \beta}\, dx. \tag{10.3}$$

If factoring the denominator and doing partial fraction decomposition,

$$\int \frac{1}{(\alpha x + \alpha^{2/3}\beta^{1/3})(x^2 - \frac{\beta^{1/3}x}{\alpha^{1/3}} + \frac{\beta^{2/3}}{\alpha^{2/3}})}\, dx, \tag{10.4}$$

$$\int \frac{\alpha^{2/3}}{3\beta^{2/3}(\alpha x + \alpha^{2/3}\beta^{1/3})} - \frac{\alpha^{7/3}x - 2\alpha^2\beta^{1/3}}{3\alpha^2\beta^{2/3}(\alpha^{2/3}x^2 - \alpha^{1/3}\beta^{1/3}x + b^{2/3})} dx. \tag{10.5}$$

From the above equation, let's solve the following part first:

$$\frac{\alpha^{2/3}}{3\beta^{2/3}} \int \frac{1}{(\alpha x + \alpha^{2/3}\beta^{1/3})} dx. \tag{10.6}$$

If substituting $u = \alpha x + \alpha^{2/3}\beta^{1/3}$ and $dx = \frac{du}{\alpha}$, and undoing the substitution $u = \alpha x + \alpha^{2/3}\beta^{1/3}$,

$$\frac{\alpha^{2/3}}{3\beta^{2/3}} \int \frac{1}{(\alpha x + \alpha^{2/3}\beta^{1/3})} dx = \frac{\alpha^{2/3}}{3\beta^{2/3}} \int \frac{1}{\alpha u} du = \frac{\alpha^{2/3}}{3\beta^{2/3}} \frac{\ln(u)}{\alpha} = \frac{\alpha^{2/3}}{3\beta^{2/3}} \frac{\ln(\alpha x + \alpha^{2/3}\beta^{1/3})}{\alpha}. \tag{10.7}$$

For next, let's solve the second part in Eq. (10.5), which is

$$\frac{1}{3\alpha^2\beta^{2/3}}\int\frac{\alpha^{7/3}x-2\alpha^2\beta^{1/3}}{(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}dx. \tag{10.8}$$

If rewriting the denominator as $\alpha^{7/3}x-2\alpha^2\beta^{1/3}=\frac{\alpha^{5/3}(2\alpha^{2/3}x-\alpha^{1/3}\beta^{1/3})}{2}-\frac{3\alpha^2\beta^{1/3}}{2}$ and splitting the above equation into two parts (let's consider the coefficient, $\frac{1}{3\alpha^2\beta^{2/3}}$, later),

$$\int\frac{\alpha^{7/3}x-2\alpha^2\beta^{1/3}}{(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}dx=\int\frac{\alpha^{5/3}(2\alpha^{2/3}x-\alpha^{1/3}\beta^{1/3})}{2(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}-\frac{3\alpha^2\beta^{1/3}}{2(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}dx. \tag{10.9}$$

Let's solve the first part of the above equation. If putting $u=\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3}$ and $dx=\frac{du}{2\alpha^{2/3}x-\alpha^{1/3}\beta^{1/3}}$ in the first part of the above equation, then

$$\frac{\alpha^2}{2}\int\frac{(2\alpha^{1/3}x-\beta^{1/3})}{(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}dx=\frac{\alpha^2}{2}\int\frac{1}{\alpha^{1/3}u}du=\frac{\alpha^2}{2}\frac{\ln(u)}{\alpha^{1/3}}. \tag{10.10}$$

Undoing the substitution $u=\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3}$, then

$$\frac{\alpha^2}{2}\frac{\ln(u)}{\alpha^{1/3}}=\frac{\alpha^{5/3}}{2}\ln(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3}). \tag{10.11}$$

Next, let's solve the second part of Eq. (10.9), which is

$$-\frac{3\alpha^2\beta^{1/3}}{2}\int\frac{1}{(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3})}dx. \tag{10.12}$$

Rewriting the denominator, $\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+b^{2/3}$, as $(\alpha^{1/3}x-\frac{\beta^{1/3}}{2})^2+\frac{3}{4}b^{2/3}$, and substituting $u=\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}}$ and $dx=\frac{du}{2\alpha^{1/3}/\sqrt{3}\beta^{1/3}}$ as follows

$$\begin{aligned}&-\frac{3\alpha^2\beta^{1/3}}{2}\int\frac{1}{(\alpha^{1/3}x-\frac{\beta^{1/3}}{2})^2+\frac{3}{4}b^{2/3}}dx=-\frac{3\alpha^2\beta^{1/3}}{2}\int\frac{\sqrt{3}\beta^{1/3}}{2\alpha^{1/3}(\frac{3\beta^{2/3}u^2}{4}+\frac{3\beta^{2/3}}{4})}du\\&=-\frac{3\alpha^2\beta^{1/3}}{2}\int\frac{2}{\sqrt{3}\alpha^{1/3}\beta^{1/3}}\frac{1}{u^2+1}du=\frac{3\alpha^2\beta^{1/3}}{2}\frac{2}{\sqrt{3}\alpha^{1/3}\beta^{1/3}}\arctan(u).\end{aligned} \tag{10.13}$$

Undo the substitution, $u=\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}}$, then

$$-\frac{3\alpha^{5/3}}{\sqrt{3}}\arctan(\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}}). \tag{10.14}$$

Using Eqs. (10.11) and (10.14), Eq. (10.9) can be further written as

$$\frac{\alpha^{5/3}}{2}\ln(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3})-\sqrt{3}\alpha^{5/3}\arctan(\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}}). \tag{10.15}$$

And, multiplying the coefficient, $\frac{1}{3\alpha^2\beta^{2/3}}$, to the above equation, then (10.8) can be written as

$$\frac{\ln(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3})}{6\alpha^{1/3}\beta^{2/3}}-\frac{\arctan(\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}})}{\sqrt{3}\alpha^{1/3}\beta^{2/3}}. \tag{10.16}$$

Using Eqs. (10.7) and (10.16), Eq. (10.5) is solved as follows

$$\frac{\ln(\alpha x+\alpha^{2/3}\beta^{1/3})}{3\alpha^{1/3}\beta^{2/3}}-\frac{\ln(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3})}{6\alpha^{1/3}\beta^{2/3}}+\frac{\arctan(\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}})}{\sqrt{3}\alpha^{1/3}\beta^{2/3}}+C \tag{10.17}$$

where C is the constant. Then, using (10.17), Eq. (10.3) can be solved as

$$\int_{-T/2}^{T/2}\frac{1}{\alpha x^3+\beta}\,dx=\left[\frac{\ln(\alpha x+\alpha^{2/3}\beta^{1/3})}{3\alpha^{1/3}\beta^{2/3}}-\frac{\ln(\alpha^{2/3}x^2-\alpha^{1/3}\beta^{1/3}x+\beta^{2/3})}{6\alpha^{1/3}\beta^{2/3}}+\frac{\arctan(\frac{2\alpha^{1/3}x-\beta^{1/3}}{\sqrt{3}\beta^{1/3}})}{\sqrt{3}\alpha^{1/3}\beta^{2/3}}\right]_{-T/2}^{T/2}. \tag{10.18}$$

Finally, after solving the above equation, ΔE_{cubic} is obtained as follows

$$\Delta E_{cubic} = \frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^2}T^2 + 2\sqrt[3]{\alpha}\sqrt[3]{\beta}T + 4\sqrt[3]{\beta^2}\right|}{4}\right)}{6\sqrt[3]{a}\beta^{\frac{2}{3}}} -$$

$$\frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^2}T^2 - 2\sqrt[3]{\alpha}\sqrt[3]{\beta}T + 4\sqrt[3]{\beta^2}\right|}{4}\right)}{6\sqrt[3]{\alpha}\beta^{\frac{2}{3}}} + \frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^{-1}}\right|\left|\sqrt[3]{\alpha}T + 2\sqrt[3]{\beta}\right|}{2}\right)}{3\sqrt[3]{\alpha}\beta^{\frac{2}{3}}} - \frac{\ln\left(\frac{\left|\sqrt[3]{\alpha^{-1}}\right|\left|\sqrt[3]{\alpha}T - 2\sqrt[3]{\beta}\right|}{2}\right)}{3\sqrt[3]{\alpha}\beta^{\frac{2}{3}}}$$

$$+\frac{\arctan\left(\frac{\sqrt{3}\sqrt[3]{\alpha}\sqrt[3]{\beta^2}T + \sqrt{3}\beta}{3\beta}\right)}{\sqrt{3}\sqrt[3]{\alpha}\beta^{\frac{2}{3}}} + \frac{\arctan\left(\frac{\sqrt{3}\sqrt[3]{\alpha}\sqrt[3]{b^2}T - \sqrt{3}\beta}{3\beta}\right)}{\sqrt{3}\sqrt[3]{\alpha}\beta^{\frac{2}{3}}} - P_{input\_t_1}T.$$

(10.19)

VITA

Wo Jae Lee
Research Assistant
Environmental and Ecological Engineering
Purdue University

Wo Jae Lee is a Ph.D. student in Environmental and Ecological Engineering (EEE) at Purdue University. He earned his Bachelor's degree in Civil, Environmental and Architectural Engineering from Korea University, and double master degrees in Mechanical Engineering and Environmental and Ecological Engineering (EEE) from Purdue University. He joined Purdue University under the supervision of Professor John Sutherland in 2016. During his Ph.D., he was involved in the research project funded by the "WHIN" (Wabash Heartland Innovation Network), whose vision is to make the Wabash Heartland (10-county region in the state of Indiana) the global epicenter for next-generation manufacturing empowered by smart "Internet of Things" platforms. His research was funded by Indiana Manufacturing Competitiveness Center (IN-MaC) between 2016 and 2017 and by WHIN between 2018 and 2021. Wo Jae's doctoral research and most of his activities within the research projects have focused on enhancing the reliability of manufacturing equipment through predictive equipment health maintenance. During his Ph.D., he has produced 6 journal papers (5 published and 1 under review) and 6 conference papers (5 published and 1 under review). He has co-authored papers with researchers from several different fields (e.g., Industrial Engineering, Mechanical Engineering, Electrical and Computer Engineering, Materials Engineering, and Computer Science). In addition to his research, Mr. Lee has provided service in support of both the profession and the university. He participated in a national competition, Reusable Abstraction of Manufacturing Process (RAMP) organized by National Institute of Standards and Technology at ASME MSEC 2018, and his group was awarded first place. He also was awarded three travel awards (two from NSF and one from Prognostics and Health Management Society). He served as the Chair of the Outreach Committee for the EEE Graduate Student Organization as well as a member of EEE's Space and Safety Committee.

PUBLICATIONS

1. Simon, T. R., Lee, W. J., Spurgeon, B. E., Boor, B. E., and Zhao, F., 2018, "An Experimental Study on the Energy Consumption and Emission Profile of Fused Deposition Modeling Process," *46th SME North American Manufacturing Research Conference*
2. Lee, W. J., Mendis, G. P., and Sutherland, J. W., 2019, "Development of an Intelligent Tool Condition Monitoring System to Identify Manufacturing Tradeoffs and Optimal Machining Conditions," *16th Global Conference on Sustainable Manufacturing*
3. Simon, T. R., Yang, Y., Lee, W. J., Zhao, J., Li, L., and Zhao, F., 2019, "Reusable unit process life cycle inventory for manufacturing: stereolithography," *Production Engineering Research and Development*
4. Lee, W. J., and Sutherland, J. W., 2019, "Deep Learning Enable Diagnostics and Prognostics of Machine Health Condition," *2019 Annual Conference of the Prognostics and Health Management Society*
5. Lee, W. J., Wu, H., Yun, H., Kim, H., Jun, M. B.G., and Sutherland, J. W., 2019, "Predictive Maintenance of Machine Tool Systems Using Artificial Intelligence Techniques Applied to Machine Condition Data," *26th CIRP Life Cycle Engineering (LCE) Conference*
6. Lee, W. J., Mendis, G., Triebe., M., and Sutherland, J. W., 2020, "Monitoring of a Machining Process using Multi Level Kernel Principal Component Analysis and Kernel Density Estimation," *Journal of Intelligent Manufacturing*
7. Joung, B. G., Lee, W. J., Huang, A., and Sutherland, J. W., 2020, "Development and Application of a Method for Real Time Motor Fault Detection," *CIRP Through-Life Engineering Services Conference*
8. Lee, W. J., Wu, H., Huang, A., and Sutherland, J. W., 2020, "Learning via Acceleration Spectrograms of DC Motor System with Application to Condition Monitoring," *The International Journal of Advanced Manufacturing Technology*
9. Lee, W. J., Xia, K., Denton, N., Ribeiro, B., Sutherland, J. W., 2020 "A Development of Speed Invariant Deep Learning Model with Application to Condition Monitoring of Rotating Machinery," *Journal of Intelligent Manufacturing*
10. Yildirim, A., Grant J., Eshaghi, S., Lee, W. J., Maruthamuthu, M., Verma, M., Sutherland, J. W., Cakmak, M., 2020, "Roll-to-Roll (R2R) Production of Large-Area High-Performance Piezoelectric Films Based on Vertically Aligned Nanocolumn Forests," *Advanced Materials Technologies*
11. Lee, W. J., Helwani, K., Tenneti, S., Krishnaswamy, A., 2021, "Robust Audio Anomaly Detection," *International Conference on Learning Representations Workshop (accepted)*
12. Lee, W. J., Raghunathan, N, Hosler, R., Sutherland, J. W., 2021, "Time to Failure Prediction of Rotating Machinery using Dynamic Feature Extraction and Gaussian Process Regression," *Journal of Intelligent Manufacturing* (*under review*)
13. Lee, W. J., Helwani, K., Tenneti, S., Krishnaswamy, A., 2021, "Robust Multiresolutional Recurrent Architecture with Application to Multivariate Time Series Forecasting and Anomaly Detection," *International Conference on Machine Learning* (*under review*)
14. Mustafa, Lee, W. J., Raghunathan, N, Mousoulis, C., Sutherland, J. W., Bagchi, S., 2021 "Anomaly Detection and Transferring Learning across sensors in IoT system," arXiv preprint arXiv:2102.05814 (*pre-print*)