A NOVEL DEEP CROSS-DOMAIN FRAMEWORK FOR FAULT DIAGNOSIS OF ROTARY MACHINERY IN PROGNOSTICS AND HEALTH MANAGEMENT

A Dissertation

presented to

the Faculty of Graduate School

at the University of Missouri-Columbia

In Partial Fulfilment

of the Requirements for the Degree

Doctor of Philosophy

By

BEHNOUSH REZAEIANJOUYBARI

Dr. Yi Shang, Dissertation Supervisor

May 2021

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

A NOVEL DEEP CROSS-DOMAIN FRAMEWORK FOR FAULT DIAGNOSIS OF ROTARY MACHINERY IN PROGNOSTICS AND HEALTH MANAGEMENT
presented by Behnoush Rezaeianjouybari,

a candidate for the degree of Doctor of Philosophy in Mechanical Engineering

and hereby certify that, in their opinion, it is worthy of acceptance.

ACKNOWLEDGEMENTS

I would like to appreciate my supervisor Dr. Yi Shang for his guidance and support in conducting this research at University of Missouri-Columbia.

I would like to express my sincerest gratitude to my committee members for their willingness to serve on my committee and for providing me with precious suggestions and insightful comments.

I would also like to thank my fellow officemates for their stimulating discussions, friendship, and all the fun we have had in the past years. Mr. David Ma, and Mr. Poorya Shobeiry, it would have been a lonely office without you. I wish you all the best and I look forward to working with you in the future.

Very special thanks are to my parents, Asieh and Yarali, and my sister, Mehrnoush, who love and encourage me by standing by me to overcome all difficulties in study and life. Your sacrificial love and gracious support are my drives to reach my goals in all my life. You have given me so much, and in return, so little. There is no way for me to express my utmost gratitude to you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AAE	Adversarial autoencoders
ACGAN	auxiliary classifier generative adversarial network
AdaBN	Adaptive batch normalization
AE	Autoencoder
AFSA	Artificial fish swarm algorithm
AHKL	Auto-balanced high-order Kullback-Leibler
AI	Artificial Intelligence
BLSTM	Bi-directional Long Short Term Memory
CAE	Contractive Autoencoder
CBLSTM	CNN- Bi-directional LSTM
CD	Contrastive divergence
CDBN	Convolutional Deep Belief Network
CLSTM	CNN-LSTM
CMD	Central Moment Discrepancy
CNN	Convolutional Neural Network
CORAL	Correlation alignment
CPS	Cyber-physical-systems
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
CVAE	Conditional variational autoencoder
CWRU	Case Western Reserve University
DA	Domain Adaptation
DAD	Deep Anomaly Detection
DAE	Denoising Autoencoder
DBM	Deep Boltzmann Machine
DBN	Deep Belief Network
DL	Deep Learning
DNN	Deep Neural Network
DQN	Deep Q-Network
EMA	Exponential moving average
FFT	Fast Fourier Transform

FTD-MSDA	Feature-level and task-specific distribution alignment multi-source domain adaptation
GAN	Generative Adversarial Network
GDA	Generalized discriminant analysis
GDBM	Gaussian Bernoulli DBM
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
GRU-ED	GRU Encoder-Decoder
HHT	Hilbert-Huang transform
HI	Health Indicator
HoMM	Higher-order Moment Matching
IaaS	Infrastructure as a Service
IIoT	Industrial Internet of Things
JSD	Jensen-Shannon Divergence
KAT	Konstruktions- und Antriebstechnik
KL	Kullback-Leibler
KNN	k-nearest neighbors
LSTM	Long Short Term Memory
LSTM-ED	LSTM Encoder-Decoder
MCMC	Markov chain Monte Carlo
MLP	Multi-Layer Perceptron
MMD	Maximum mean discrepancy
MSCNN	Multi-scale convolutional neural network
MSDA	Multi-source domain adaptation
NAS	Neural Architecture Search
OT	Optimal Transport
PaaS	Platform as a Service
PHM	Prognostics and Health Management
PSO	Particle Swarm Optimization
PSR	Phase Space Representation
RBF	Radial basis function
RBM	Restricted Boltzmann Machine
RKH	Reproducing kernel Hilbert
RKHS	Reproducing Kernel Hilbert Space

RL	Reinforcement Learning
RNN	Recurrent Neural Network
SaaS	Software as a Service
SAE	Stacked Autoencoder
SCDA	Smooth conditional distribution alignment
SDAE	Sparse Denoising Autoencoder
SDAE-NCL	Stacked denoising autoencoder network with negative correlation learning
SGD	Stochastic gradient descent
SML	Stochastic maximum likelihood
SML	Stochastic maximum likelihood
SNR	Signal to Noise Ratio
SPEV	Spectrum-principal-energy vector
SSAE	Sparse Stacked Autoencoder
SSDAE	Stacked sparse denoising autoencoder
STPN	Spatiotemporal pattern network
SVM	Support Vector Machine
SWD	Sliced Wasserstein distance
TDConvLSTM	Time-distributed Convolutional LSTM
TL	Transfer Learning
UDA	Unsupervised Domain Adaptation
WD	Wasserstein Distance

NOMENCLATURE

acc_i	ith Source-only accuracy on the target domain
C	Task-specific classifier
C_i^1	1st classifier in ith source classifier pair
C_i^2	2nd classifier in ith source classifier pair
D^i	ith source domain
D^t	Target domain
$\phi\,(.)$	Kernel map
$\mathcal{L}$	Loss function
m	Number of random projections in sliced Wasserstein distance
MMD^2	Maximum Mean Discrepancy distance
n	Batch size
N	Number of source domains
N_i	Number of ith source domain labeled samples
N_t	Number of target domain unlabeled samples
P_i	ith source domain distribution
P_t	Target domain distribution
$\mathcal{R}_\theta$	Radon transform
SW_p	p-Wasserstein distance
SWD_2	Sliced 2- Wasserstein distance
ω_i	ith source domain weight
W	Source domain weight vector
$\mathbf{X}_i$	ith source samples
$\mathbf{x}_i^j$	jth sample of ith source
$\mathbf{X}_t$	Target samples
$\mathbf{x}_t^j$	jth sample of the target
Y_i	ith source labels
y_i^j	jth sample label of ith source
y_t^j	jth sample label of the target

ABSTRACT

Improving the reliability of engineered systems is a crucial problem in many applications in various engineering fields, such as aerospace, nuclear energy, and water declination industries. This requires efficient and effective system health monitoring methods, including processing and analyzing massive machinery data to detect anomalies and performing diagnosis and prognosis. In recent years, deep learning has been a fast-growing field and has shown promising results for Prognostics and Health Management (PHM) in interpreting condition monitoring signals such as vibration, acoustic emission, and pressure due to its capacity to mine complex representations from raw data.

This doctoral research provides a systematic review of state-of-the-art deep learning-based PHM frameworks, an empirical analysis on bearing fault diagnosis benchmarks, and a novel multi-source domain adaptation framework. It emphasizes the most recent trends within the field and presents the benefits and potentials of state-of-the-art deep neural networks for system health management. Besides, the limitations and challenges of the existing technologies are discussed, which leads to opportunities for future research. The empirical study of the benchmarks highlights the evaluation results of the existing models on bearing fault diagnosis benchmark datasets in terms of various performance metrics such as accuracy and training time. The result of the study is very important for comparing or testing new models. A novel multi-source domain adaptation framework for fault diagnosis of rotary machinery is also proposed, which aligns the domains in both feature-level and task-level. The proposed framework transfers the knowledge from multiple labeled source domains into a single unlabeled target domain by reducing the feature distribution discrepancy between the target domain and each source domain. Besides, the model can be easily reduced to a single-source domain adaptation problem. Also, the model can be readily updated to unsupervised domain adaptation problems in other fields such as image classification and image segmentation.

Further, the proposed model is modified with a novel conditional weighting mechanism that aligns the class-conditional probability of the domains and reduces the effect of irrelevant source domain which is a critical issue in multi-source domain adaptation algorithms. The experimental verification results show the superiority of the proposed framework over state-of-the-art multi-source domain-adaptation models.

CHAPTER 1: INTRODUCTION

Recently, prognostics and health management (PHM) has emerged as a key technology to overcome the limitations of traditional reliability analysis. PHM focuses on utilizing sensory signals acquired from an engineered system to monitor the health condition, detect anomalies, diagnose the faults, and more importantly, predict the remaining useful life (RUL) of the system over its lifetime. This health information provides an advance warning of potential failures and a window of opportunity for implementing measures to avert catastrophic failures by reducing system downtime and maintenance costs.

In traditional maintenance models, the machinery is investigated and maintained via break-down-based or time-based strategies. These two strategies have two main disadvantages: (i) they can be extremely costly; and (ii) Their process can pose a safety risk to employees and other assets. Conversely. PHM is known to have strong economic benefits for owners, operators, and society. In PHM-based maintenance strategy, engineers predict when equipment failure might happen, and then perform maintenance to keep machines in operation. Modern systems are extensively complex and built with many interactive components and electronics, which highlights the importance of system reliability. Failure of any component will result in a catastrophic failure of the system. A viable PHM system framework gives early detection and isolation of the incipient fault of components/sub-systems. The outcome of an effective PHM model provides a tool to monitor the progression of the fault and to help in making assessment decisions and maintenance schedules

Availability of abundant data and exponentially increasing computational power provide significant opportunities for industry and academia to practice advanced data-driven frameworks to determine the patterns, classify faults, and assess system degradation trends. Numerous data-driven methods have been used in literature, including support vector machines (SVM) [1], random forest [2], principal component analysis [3], particle filtering [4], Hidden Markov Model (HMM) [5], and so on. However, all of these techniques require the experience of the expert and prior knowledge of signal processing to manually extract and select meaningful features. Also, they are not powerful enough to capture complex non-linear relationships that exist in real fault diagnosis and prognostics issues.

With the evolution of smart sensing, communication technologies, and complex engineered systems, the massive amount of data from various resources is rapidly generated and collected in real-time, which contains useful information about the degradation and health condition of the system. The performance of traditional algorithms is greatly impeded by the proliferation of multi-dimensional and heterogeneous data streams. Thus, more advanced analytic tools are necessary to adaptively and automatically mine the characteristics hidden in the real-time measured streams.

Deep learning, as a breakthrough in artificial intelligence, has been embraced by various areas such as medical image analysis, visual understanding, health care, computer vision, recommender systems, natural language processing, and automatic speech recognition. It can automatically process highly non-linear and complex feature abstraction from raw data via deep neural networks and eliminates the reliance on domain knowledge and manual feature engineering. Deep learning can automatically learn hierarchical representations of large-scale data, which makes it an effective tool for PHM applications, especially in the presence of high volume and multi-dimensional industrial data. Traditional data-driven frameworks require hand-crafted feature extraction and appropriate feature selection processes, which is highly dependent on the expertise of professionals and signal processing knowledge. Conventional frameworks cannot be updated in real-time and require a great deal of work dealing with large-scale data sets. Besides, the Deep learning algorithm makes it possible to integrate the PHM tasks such as feature extraction, feature selection, and classification/regression into an end-to-end architecture and jointly optimize all the tasks hierarchically.

This is a fast-growing area, and refined solutions and advanced models are being developed every few months. There is a need to present more current reviews to cover recent advances and suggested solutions in the PHM paradigm. In chapter 2, after a brief overview of traditional data-driven PHM and common deep learning architectures, we perform a systematic survey on the variety of deep neural networks that have been developed and explicitly deployed for fault diagnosis and RUL prediction of engineered. We address the role of benchmarking in Chapter 3, and provide a benchmarking study for bearing fault diagnosis with the well-known Case Western Reserve University (CWRU) public dataset. In chapter 4, we target the cross-domain fault diagnosis challenge and provide a novel deep framework for multi-source domain adaptation in PHM applications. In chapter 5, we improve the proposed framework by incorporating a novel conditional weighting mechanism for source domains.

The main contributions of this work can be summarized as follows:

In Chapter 2, We provided a comprehensive systematic survey of DL studies in PHM field and analyze the current status and the research trends. The second chapter of this work serves as a comprehensive reference for PHM community in terms of state-of-the-art models, datasets, existing challenges. and the research directions which are worth exploring in the future.

In Chapter 2, we categorized the available models into three classes of discriminative, generative, and hybrid, and with practical examples explain how these models especially the generative networks, can be effective to solve existing challenges.

In Chapter 3, following the review of the available studies, we carried out an empirical study of standard deep neural networks on the benchmark rolling-element bearing datasets. We utilized multiple performance metrics for a fair comparison. This chapter gives a complete view of explored solutions so far and helps the practitioners to select the DNN architecture(s) that better fit the resource constraints of practical deployments and applications.

In Chapter 4, we proposed a novel deep multi-source domain adaptation framework for fault diagnosis of rotary machinery, which aligns the domains in both feature-level and task-level. The proposed model can be easily reduced to a single-source domain adaptation problem. Also, the model can be readily updated to unsupervised domain adaptation problems in other fields such as image classification and image segmentation.

In Chapter 5, we modified the proposed MSDA framework by adding a novel conditional weighting mechanism to weight the relatedness of each source domain to the target, to reduce the influence of the negative transfer caused by an unrelated source domain. The weight vector is optimized as a composite function of network parameters during the training stage. At the same time, the proposed scheme aligns the conditional probability distribution of each source domain and the target domain.

CHAPTER 2: A SYSTEMATIC REVIEW OF DEEP LEARNING IN SYSTEM HEALTH MANAGEMENT

With the evolution of smart sensing, communication technologies, and complex engineered systems, the massive amount of data from various resources is rapidly generated and collected in real-time, which contains useful information about the degradation and health condition of the system. The performance of traditional algorithms is greatly impeded by the proliferation of multi-dimensional and heterogeneous data streams. Thus, more advanced analytic tools are necessary to adaptively and automatically mine the characteristics hidden in the real-time measured streams.

Deep learning, as a breakthrough in artificial intelligence, has been embraced by various areas such as medical image analysis, visual understanding, health care, computer vision, recommender systems, natural language processing, and automatic speech recognition. It can automatically process highly non-linear and complex feature abstraction from raw data via deep neural networks and eliminates the reliance on domain knowledge and manual feature engineering. Deep learning can automatically learn hierarchical representations of large-scale data, which makes it an effective tool for PHM applications, especially in the presence of high volume and multi-dimensional industrial data. Traditional data-driven frameworks require hand-crafted feature extraction and appropriate feature selection processes, which is highly dependent on the expertise of professionals and signal processing knowledge. Conventional frameworks cannot be updated in real-time and require a great deal of work dealing with large-scale data sets. Besides, Deep learning algorithm makes it possible to integrate the PHM tasks such as feature extraction, feature selection, and classification/regression into an end-to-end architecture and jointly optimize all the tasks in a hierarchical fashion.

To date, a few systematic surveys on deep learning and PHM have been published [6]–[9]. However, they are either component (or system) specific or not updated with more recent deep learning technologies. This is a fast-growing area, and refined solutions and advanced models are being developed every few months. There is a need to present more current reviews to cover recent advances and suggested solutions in the PHM paradigm. In this chapter, we review the variety of deep neural networks that have been developed and explicitly deployed for fault diagnosis and RUL prediction of engineered systems. A revisit of traditional data-driven PHM basics is given in section 2.3, followed by a brief introduction of standard deep neural networks in section 2.4.

An overview of common deep learning architectures is presented in Section 2.3, followed by a revisit of traditional data-driven PHM basics in Section 2.3. An overview of deep learning works for common PHM problems is given in Section 2.4. Section 2.5 through section 2.12 provides a comprehensive systematic survey of existing DL models in PHM. In section 2.13 we end the chapter by addressing current challenges and future directions. . This chapter provides a comprehensive reference for deep learning practitioners in the PHM society.

The main contributions of this chapter can be summarized as follows:

We categorized the available models into three classes of discriminative, generative, and hybrid, and with practical examples explain how these models especially the generative networks, can be effective to solve existing challenges.

We provide the community with a comprehensive reference of available resources in terms of datasets and existing algorithms.

We discuss the most significant available challenges such as imbalance classes, unlabeled data, insufficient data, domain shift, and explain how deep learning can be utilized to alleviate these problems.

2.1 METHODOLOGY OF SURVEY

The authors have conducted a thorough search in electronic databases Google Scholar, Scopus, Web of Science, IEEE explore, and Science direct using the keywords “fault detection” OR “fault diagnosis” OR “prognostics” OR “condition monitoring” OR “remaining useful life” AND either “deep learning” OR name of the particular deep network. The search retrieved 227 articles in the period between 2013 and September 2019. We have screened the articles carefully and eliminated the repeated studies. After applying the following exclusion criteria, a total of 137 studies were retained and analyzed:

EC1. Books, graduate theses, letters, and patents are not considered for review.

EC2. Conference entries and preprint papers are excluded unless those are highly-cited and not published in any journal.

EC3. Non-primary studies such as literature survey articles are not included.

EC4. Only unique studies are analyzed. For repeated studies with minor changes, the other copies of the study are excluded.

EC5. Studies that do not report the performance metric results are excluded.

EC6. Studies that do not contain validation or experimental verification are not considered for review.

Figure 2.13.1 shows the popularity of various deep learning architectures among PHM researchers and the distribution of the publications per year considering the variety of categories. There is a significant growth of related published papers in recent years.

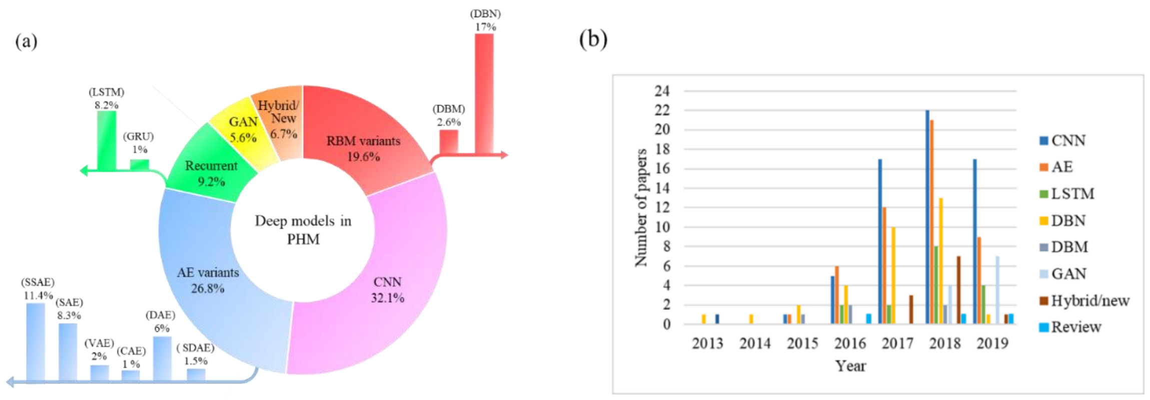


Figure 2.13.1 a) The density of various deep learning architectures in PHM from 2013 until September 2019, b) Breakdown of the papers in the year of publication.

2.2 BIBLIOMETRIC ANALYSIS

To get further insights into the structure of the paper, co-word analysis was undertaken based on related keywords in selected 127 research papers. 0Figure 2.13.2 is the projection of the co-occurrence relationship among the top 29 frequently occurring keywords, which is visualized in VOSviewer tool [6]. Keywords include well-known ***system types*** (battery, bearing, gearbox, bogie, aircraft engine, etc.), ***PHM tasks*** (fault detection, fault diagnosis, prognostics, RUL estimation, etc.), ***deep neural network categories*** (CNN, RNN, GAN, etc.), ***data types*** (vibration, current signal, and acoustic emission), and several ***learning problems*** in the realm of machine learning (transfer learning, domain adaptation, and unsupervised learning). Every keyword is represented by a colored circle. The size of the circle indicated the weight of the term occurrence in literature. Also, the weight of the links between the nodes represents the degree of co-occurrence of connected keywords.

VOSviewr uses a modularity-based clustering technique to group the most cooccurred keywords in the same cluster. We have merged the smaller clusters to eliminate unnecessary details. The final map is identified by three clusters (red, green, and blue), and all points with the same color are members of the same cluster. We have found the degree of centrality of each keyword to find the most representative keyword of each cluster: Fault detection (blue), Fault diagnosis (red), and Prognostics (green). Looking at these keywords and the terms within each cluster provide us a glimpse of the interconnection among various tasks and deep neural networks on the basis of available public datasets for practice. For example, at first glance, one can say fault diagnosis is much more studied than prognostics. Also, bearings seem to be the most studied components for PHM, as they are critical components of the engineered systems. The other reason lies in the availability of public datasets for rolling bearings. Moreover, the green cluster nodes indicate the nearness of the terms “prognostics”, “RUL estimation”, “RNN”, and “Battery”, showing the significance of prognostics for batteries. Furthermore, as expected and will be discussed, RNNs are the most commonly used networks for RUL prediction.

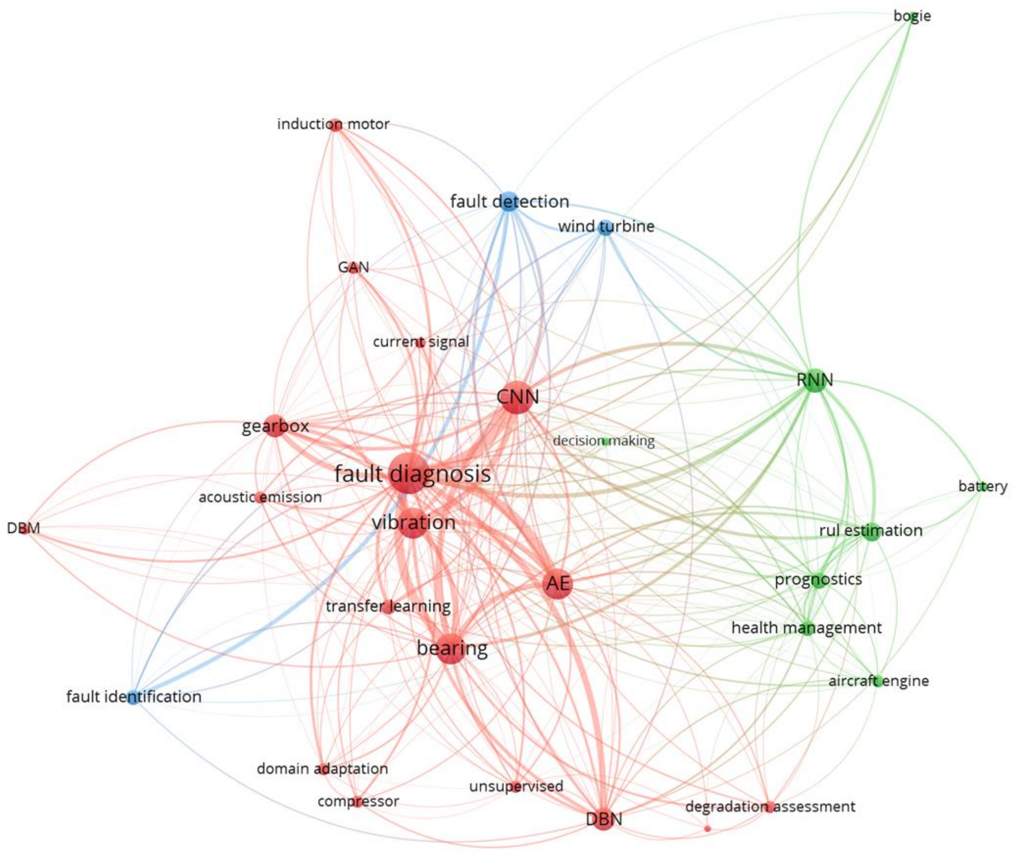


Figure 2.13.2 Network visualization for related keywords in the review content according to co-occurrence terms.

2.3 DATA-DRIVEN PROGNOSTICS AND HEALTH MANAGEMENT

PHM offers a wide range of tools for system health assessment and reliability improvement and involves many subareas in different aspects. This section presents a brief overview of the standard data-driven PHM framework including constituent parts, performance assessment metrics, and existing datasets so that the readers might use them for their model evaluation.

2.3.1 The modules of the traditional Prognostics and Health Management cycle

As demonstrated in Figure 2.13.3 , PHM is mainly considered as the combination of several tasks to reduce the total life-cycle cost of the equipment. The following paragraphs define the terminologies and the commonly used techniques:

Data Acquisition module comprises condition monitoring sensors (e.g., accelerometers, acoustic emission sensors, thermometers, etc.), data storage, and transmission devices, which provide initial monitoring information from machinery.

Feature extraction in PHM mostly refers to signal processing algorithms in time, frequency, and time-frequency domains to transform raw measurement data into the informative signature of the behavior of the system. Statistical time-series features such as RMS, kurtosis, crest factor, skewness and frequency domain signatures including spectral, envelope and cepstrum analysis are extensively used for stationary signals Time-frequency methods such as Short-time Fourier transform (STFT), Empirical mode decomposition (EMD), Wavelet packet transform (WPT), Hilbert-Huang transform (HHT), etc. on the other hand, achieve better results for non-stationary signal analysis [7]–[9].

Feature selection algorithm removes irrelevant and redundant features by selecting the optimal feature subset through filters, wrappers, or embedded methods [10].

Furthermore, **Dimensionality reduction** techniques such as principal component analysis (PCA), Linear Discriminant Analysis (LDA) and kernel principal component analysis and (KPCA) have been widely adopted to generate a new subset of lower-dimensional features while retaining intuitive information of the original features [11], [12].

In system health management discipline, anomaly refers to time instances that system behaves differently from normal, and the reason may or may not lies in an incipient fault or failure. Classical methods of **anomaly detection**, such as density-based techniques, support vector machines, Hidden Markov models, Bayesian networks, ensemble techniques, etc. have been broadly used in the system health assessment domain [13]–[15].

Diagnostics is the critical step after anomaly detection to identify the health status of the system by analyzing the severity levels of degradation. Classical supervised machine learning algorithms such as SVMs, random forest, k-nearest neighbors (KNN), artificial neural networks (ANN), etc. have been trained on labeled datasets to accurately classify fault types [1], [2], [16].

Prognostics refers to detecting incipient failures and associated RUL of the equipment to assess the reliability and support timely decision-making for maintenance operation. Numerous data-driven methods have been adopted to address prognosis in PHM cycle including ANNs, HMMs, particle filtering, Kalman filter variants, and regression methods [5], [17], [18].

Decision support represents the “health management” part of PHM which uses the outputs of Diagnostics and Prognostics for taking timely, appropriate maintenance and logistics decisions [19]. Mathematical programming, Markov decision process, and Reinforcement Learning (RL) techniques have been widely used to find the optimal maintenance action and the optimal time of applying it [20]–[22].

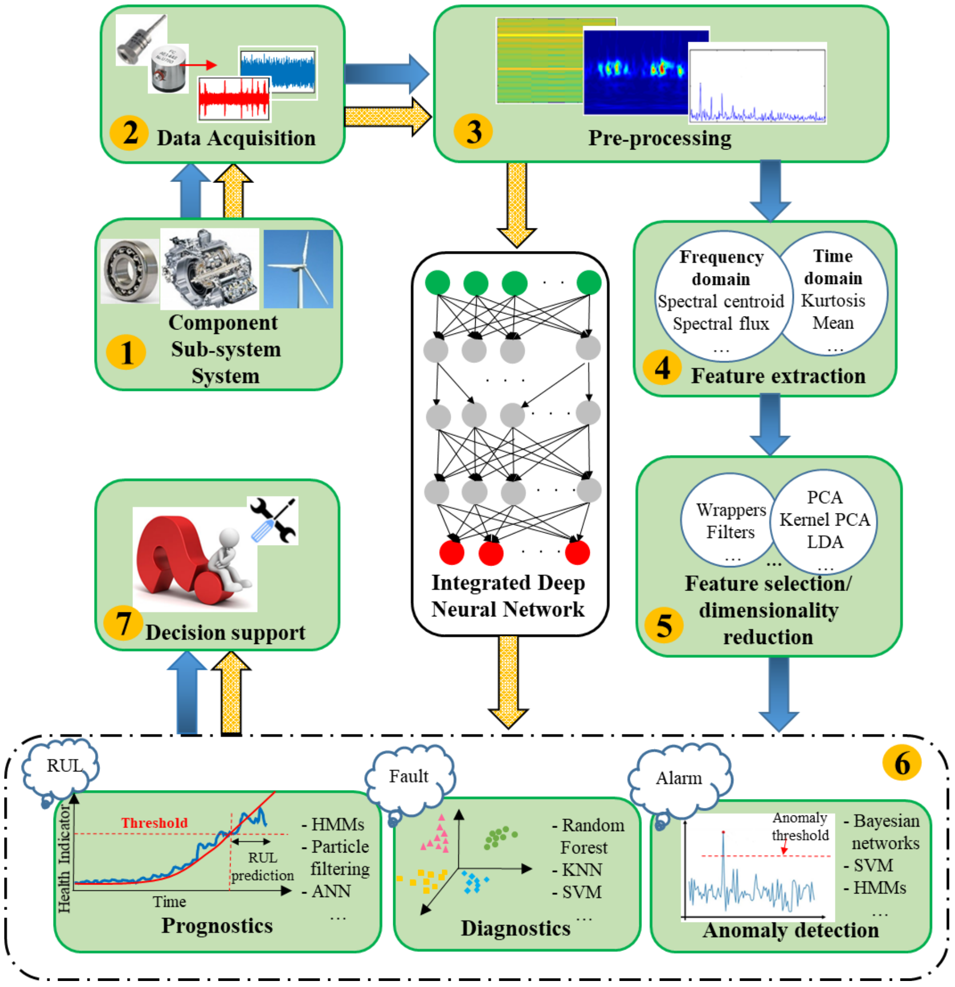


Figure 2.13.3 Modules of traditional data-driven PHM cycle vs. deep PHM model.

2.3.2 *Performance metrics*

A variety of indices are used to evaluate the prediction performance of the PHM model. Besides, depending on the complexity of the model, many researchers have proposed new measures for the evaluation of RUL prediction for prognostics. Table 2.13.1

summarizes the list of the most commonly used metrics. Various measures consider different aspects of the model depending on available information and problem requirements. Readers may refer to the last column references for more information.

Table 2.13.1 Performance metrics for PHM model evaluation.

Measure	Notation	Reference
Diagnostics		
Confusion matrix criteria:		
• Accuracy and Error rate	*ACC, ER*	Ali et al. [23]
• Precision	*PR*	Shao et al. [24]
• Sensitivity (Recall)	*SN*	Shao et al. [24]
• F1-score	*F1*	Shao et al. [24]
• Correlation coefficient	*CC*	Lou et al. [25]
Receiver operating characteristic (ROC) curve:		
• Area under the curve	*AUC*	Batista et al. [26]
Detection error trade-off curve	*DET*	Batista et al. [26]
Prognostics		
Offline evaluation; ground truth RUL data:		
• Mean absolute error	*MAE*	Zhu et al. [27]
• Root mean squared error	*RMSE*	Deutsch et al. [28]
• Mean absolute percentage error	*MAPE*	Deutsch et al. [28]
• Prediction horizon	*PH*	Saxena et al. [29]
• $\alpha - \mu$ accuracy	$\alpha - \mu$ *ACC*	Saxena et al. [29]
• Convergence	*Con*	Saxena et al. [29]
• Relative accuracy	*RA*	Saxena et al. [29]
• Confidence interval	*CI*	Chen et al. [30]
• Prognostic accuracy criterion	*PAC*	Nguyen et al. [31]
• Hybrid criterion	*HyC*	Nguyen et al. [31]
• Exponential Transformed accuracy	*ETA*	Nectoux et al. [32]
Offline evaluation; run-to-failure data:		
• Mean prediction error and standard deviation	*E, sd*	Zemouri et al. [33]
• Overall average bias	*OAB*	Zemouri et al. [33]
• Overall average variability	*OAV*	Zemouri et al. [33]
• Reproducibility	*Rep*	Zemouri et al. [33]
• Predictability	*Pred*	Javed et al. [34]
Online evaluation:		
• Online root mean squared error	*Online RMSE*	Hu et al. [35]
• Coverage rate	*CR*	Hu et al. [35]
• Average width	*AW*	Hu et al. [35]

2.3.3 Public datasets

Despite recent advances in data acquisition and sensor technology, acquiring enough high-quality data for data-driven approaches is still difficult and challenging. The long-term deterioration process and machinery break-down in-service makes it time-consuming and impracticable to collect high-resolution run-to-failure data. Moreover, the measurements collected during the out-of-service period of the machinery, usually do not reflect the real working situation behaviors.

To facilitate the PHM model development, Table 2.13.2 presents common published datasets for diagnostics and prognostics. Although the measurements are collected over accelerated degradation experiments under laboratory conditions, they provide a useful reference for PHM researchers.

2.4 DEEP NEURAL NETWORKS ARCHITECTURES

The idea behind deep neural networks is inspired by the hierarchical structure of the human brain, as they first learn simpler features, and then process them to represent more abstract features. The general structure of a deep neural network (DNN) known as feedforward mainly consists of an input layer, multiple hidden layers, and an output layer. In the multi-layer perceptron (MLP) network, as the simplest form of deep architecture, the output is computed straight forward along with the sequent layers of the model as long as input data is fed. In each neuron of the middle-hidden layers, the biased weighted sum of the previous layer outputs is put into a nonlinear function aka activation function to produce the output of that neuron. The hierarchical nature of representation learning in DL lets it find out desired but abstract underlying correlations and patterns among a large amount of data. In this section, we briefly discuss the fundamental concepts of deep learning and typical deep structures commonly applied to PHM in literature. Some mostly used terminologies have been defined in Table 2.13.3.

Table 2.13.2 Public datasets for system health management.

Dataset	Task	Comment	Link*
Bearing			
Lessmeier et al. [36]	Diagnostics	Motor currents and vibration signals over four different conditions.	1
CWRU [37]	Diagnostics	Delivers health-related vibration measurements of bearings at different locations under four various loading conditions of the motor.	2
PRONOSTIA (IEEE PHM'12) [32]	Prognostics: RUL prediction	Measures vibration and temperature and the rotating speed is stable.	3
IMS [38]	Prognostics: HI construction, RUL prediction	Compared with PRONOSTIA, the longer degradation process makes the data closer to the real industrial case. However, the RUL prediction gets more complicated.	3
Turbofan Engine			
CMAPSS (IEEE PHM'08) [39]	Prognostics: RUL prediction	Provides temperature, speed, pressure, and bleed measurements under six different operating conditions. Therefore, is the right candidate for multi-sensor fusion algorithms.	3
Gearbox			
PHM'09	Diagnostics	Unsupervised fault detection	4
Li-Ion Battery			
Idaho national lab [40]	Prognostics: State of health estimation	Battery aging experiment affords operational profiles data of four batteries at room temperature.	3
HIRF (IEEE PHM'15) [41]	Prognostics: state of health estimation	Battery current, voltage, and state of charge (SOC) available.	3
Randomized battery usage [42]	Prognostics: State of health estimation	Aging voltage and current data are collected over randomized discharge profiles.	3
Tool wear prediction			
PHM'10	Prognostics: Health assessment	Collected data include wear measurements, vibration, acoustic emission, and force readings.	5
Milling dataset	Prognostics: wear prediction, RUL prediction	Provides measurements from acoustic emission and vibration sensors.	3
Industrial plant			
PHM'15 [43]	Diagnostics	Contains time-series measurements for thirty plants with various components.	6
Bogie			
PHM'17	Diagnostics: Fault detection and isolation	Provides vibration data from different components of the vehicle.	7

* 1: Kat-Data Center, 2: Case Western Reserve University data center, 3: NASA Prognostics Center data repository, 4: https://www.phmsociety.org/references/datasets, 5: https://www.phmsociety.org/competition/phm/10, 6: https://www.phmsociety.org/events/conference/phm/15/data-challenge, 7: https://www.phmsociety.org/events/conference/phm/17/data-challenge

Table 2.13.3 Glossary.

Term	Description
Cost function/Loss function	Loss function and cost function have been interchangeably used in the machine learning community. In the current review, loss function refers to the error term for a single training/validation/test set. However, the cost function is the average of loss function over the entire or a batch of training/validation/test set and may contain penalty terms.
Discriminative neural networks	Modeling the conditional probability of the output, given the observed data
Graphical models	Are probabilistic models in which the probabilistic distributions are expressed by graphs. each node in the graphs represents a random variable (or group of random variables), and the links express probabilistic relationships such as conditional probability between these variables [44].
Generative neural networks	Modeling the joint probability distribution of the input variables and the output variables. The term "generative" comes from the network's ability to generate random instances.
Machine learning	Is programming computers to optimize a performance criterion using example data or experience. The model is defined up to some parameters, and learning refers to optimize the parameters of the model. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data or both [45].
Representation learning	Automatically extracting meaningful representations or features required for machine learning tasks, as opposed to manual feature engineering techniques
Supervised learning	Machine learning algorithms that use labeled data to infer a mapping function from the input to the output
Unsupervised learning	Machine learning without labels or specific guidance

2.4.1 Restricted Boltzmann Machine

Restricted Boltzmann Machines (RBMs) are undirected bipartite graphical models consisting of n_x visible (input) units and n_h hidden units, and no intralayer connections are allowed, see Figure 2.13.4 RBMs often perform as powerful generative models trying to estimate the probability distribution of the input data. In other words, it learns a reconstructed version of the input data through stochastic processing units. From the supervised perspective, RBM often acts as a pre-processor for other models to carry out the classification task. However, there are several publications arguing the discriminative learning of RBM as a self-contained classifier [46]. The energy function E, and joint probability distribution P of a standard RBM with binary random visible variables $\mathbf{x} = \left[x_i\right]_{i=1}^{n_x}$ and binary random hidden variables $\mathbf{h} = \left[h_j\right]_{j=1}^{n_h}$ are defined as:

$$E(\mathbf{x},\mathbf{h}) = -\mathbf{x}^T\mathbf{W}\mathbf{h} - \mathbf{b}^T\mathbf{x} - \mathbf{c}^T\mathbf{h} \tag{2-1}$$

$$P(\mathbf{x},\mathbf{h}) = \frac{\exp\left\{-E(\mathbf{x},\mathbf{h})\right\}}{Z} \tag{2-2}$$

where $Z = \sum_x \sum_h \exp(-E(\mathbf{x},\mathbf{h}))$ is the partition function that ensures the normalization of P as a probability distribution. $\mathbf{W} = \left[w_{ij}\right]$, $\mathbf{b} = \left[b_i\right]$ and $\mathbf{c} = \left[c_j\right]$ are the weight matrix and bias vectors. n_x and n_h are number of units at input and hidden layers, respectively. RBMs are a particular case of Markov random fields and can be trained via Markov chain Monte Carlo (MCMC) methods. Among all, contrastive divergence (CD) is shown to reduce the training effort [47]. A practical guide to training RBMs is provided by Hinton [48]. In the following two sections, we briefly introduce two generative DNNs based on RBM, known as deep belief networks and deep Boltzmann machines.

2.4.2 Deep Belief Network

Deep belief networks (DBNs) are the first successful deep networks. They are formed by stacking multiple RBMs and model the joint distribution of observed data $\mathbf{x} = \mathbf{h}^0$ and l hidden layers $\mathbf{h}^k$ as:

$$P(\mathbf{x},\mathbf{h}^1,\mathbf{h}^2,\ldots,\mathbf{h}^l) = \left(\prod_{k=0}^{l-2} P(\mathbf{h}^k \left|\mathbf{h}^{k+1}\right.)\right)P(\mathbf{h}^{l-1},\mathbf{h}^l) \tag{2-3}$$

It can be seen in Figure 2.13.4 that the top two layers are non-directional and connections in the other layers are top-down directed. DBNs are trained through a two-steps process: the pre-training step and the fine-tuning step. A greedy layer-wise unsupervised algorithm in a down-top manner carries out the pre-training [49]. To this, first, the RBM which takes the data as input is trained via Contrastive Divergence and learns the representation of the input. Then, the previously trained features are treated as input variables of the second hidden layer and the layers are sequentially trained in the same way. Once the network has been initialized by pre-training, parameters can be fine-tuned with labeled data via a supervised up-down process [50].

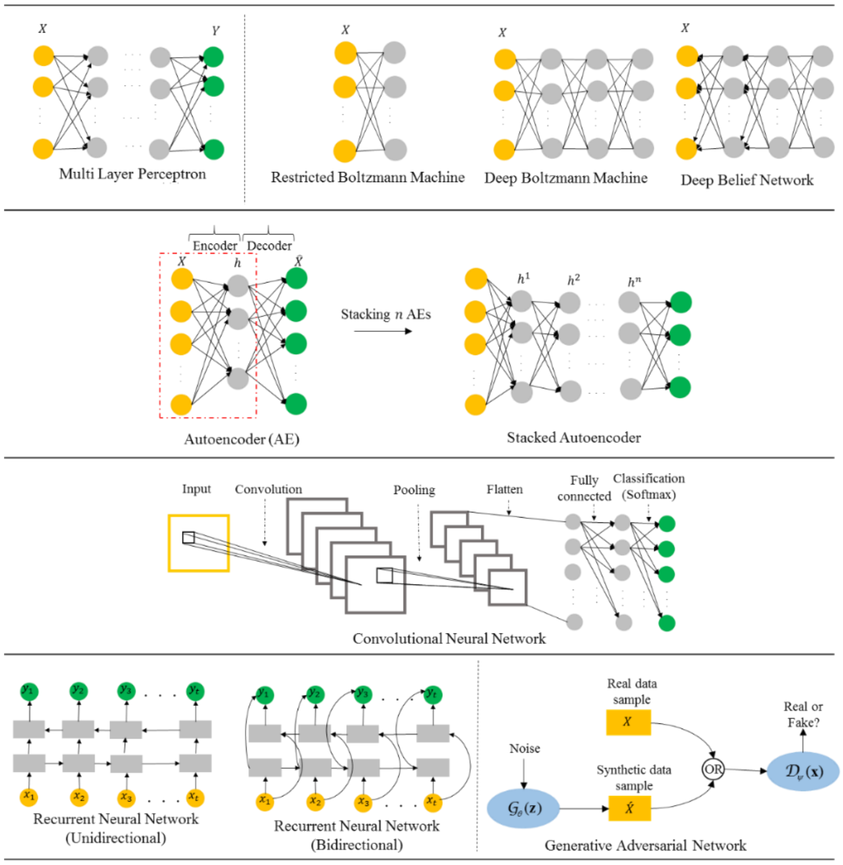


Figure 2.13.4 Typical deep architectures, rectangular hidden units represent recurrent cells and can be vanilla RNN, GRU or LSTM cells.

2.4.3 Deep Boltzmann Machine

Deep Boltzmann Machine (DBM) is another RMB-based deep generative model where layers are again arranged in a hierarchical manner [51]. Unlike DBN, in DBM all the connections are undirected, see Figure 2.13.4. DBM can be regarded as a deep RBM with multiple hidden layers, where units in odd-numbered layers are conditionally independent of even-numbered layers, and vice versa [52]. During the training process, a stochastic maximum likelihood (SML) based algorithm is used to jointly train all the layers by maximizing the lower bound on the likelihood [53]. Such a process seems leading to falling in poor local minima. Instead, Salakhutdinov and Hinton proposed a greedy layer-wise pre-training strategy, much similar to DBN, i.e., by treating the network as a stack of RBMs and pre-training them independently [51]. A final SML-based joint fine-tuning updates the parameter space.

DBMs are compelling deep models, but the high computational cost of the approximate inference makes large-scale learning of DBMs (sizeable datasets and/or larger DBMs) almost impractical. Some researchers focused on different methods to improve the efficiency of DBMs by accelerating the inference or improving the training process [54], [55].

2.4.4 Autoencoder

Autoencoders are unsupervised networks that are trained to reconstruct the input $\mathbf{x}$ on the output layer $\hat{\mathbf{x}}$ in a two-phase process: *encoding* learns a hidden representation of data $\mathbf{h}$ via a closed-form feature-extracting function $\mathbf{h} = f_\theta(\mathbf{x})$, and *decoding* maps $\mathbf{h}$ back into the input space to obtain a reconstruction of data $\hat{\mathbf{x}} = g_\theta(\mathbf{h})$. The subscript θ denotes network parameters $\theta = [\mathbf{W}, \mathbf{b}, \mathbf{W}', \mathbf{b}']$.

In the course of training a standard autoencoder, the parameters are found by minimizing the cost function over m training samples:

$$\mathcal{J}_{AE}(\theta) = \sum_{m} L(\mathbf{x}, g_\theta(f_\theta(\mathbf{x}))) \tag{2-4}$$

where L represents the loss function as a measure of the discrepancy between $\mathbf{x}$ and $\hat{\mathbf{x}}$ known as reconstruction error. Generally, single-layer autoencoders have limited representational power. Similar to RBMs, autoencoders can stack in a deep configuration called stacked autoencoder (SAE), which forwards the latent representation of the layer below as the input to the next layer, and training is done in a greedy layer-wise manner.

A significant drawback of the standard autoencoder is the tendency to learn identity functions without extracting meaningful information about the data, especially for the overcomplete case in which hidden layer has a dimension equal or greater than the input, i.e. $n_h \geq n_x$. Alternative variants are introduced to provide the solution by regularization, i.e., constraining the network to learn other properties besides the ability to reconstruct the data), or training autoencoders via generative modeling approaches. The resulting variants are discussed in sections 2.4.4.1-2.4.4.4.

2.4.4.1 Sparse Autoencoder

Sparse autoencoder exploits the inner structure of the data via including a sparsity constraint on the activation of the hidden units through the addition of Kullback-Leibler (KL) divergence term to the cost function [56]. Sparse representation improves the classification task performance by increasing the likelihood that different categories will be easily separable [57]. Overall, the corresponding cost function of a stacked sparse autoencoder (SSAE) is updated as:

$$\mathcal{J}_{SAE}(\theta) = \sum_{m} L(\mathbf{x}, g_\theta(f_\theta(\mathbf{x}))) + \beta \sum_{j=1}^{N} KL(\rho \| \hat{\rho}'_j) \tag{2-5}$$

where N is the hidden layer size, $\hat{\rho}_j$ is the average activation of the j-th hidden unit, and ρ is a specified sparsity parameter. The hyperparameter β determines the relative importance of the sparseness term in the cost function [58].

2.4.4.2 Denoising Autoencoder

The denoising autoencoder (DAE) is another regularized network to prevent the model from learning a trivial identity solution. Instead of adding the penalty to the cost function, DAE takes a noise-corrupted version of data $\tilde{\mathbf{x}}$ to reconstruct the input $\mathbf{x}$ and learn meaningful information by changing the reconstruction error term in the cost function, $L(\mathbf{x}, g_\theta(f_\theta(\tilde{\mathbf{x}}))$.

The input is first corrupted employing Binary or Gaussian noise, and the corrupted data $\tilde{\mathbf{x}}$ is then mapped to the hidden layer. Therefore, DAEs must undo the corruption process by capturing the input data distribution, rather than simply learning the identity. The learned representation is robust toward slight perturbations [59].

2.4.4.3 Contractive Autoencoder

Contractive autoencoder (CAE) proposed by Rifai et al. [60], adds the Frobenius norm of the Jacobian matrix of the latent space representation w.r.t the input, to the standard reconstruction loss $L(x, g_\theta(f_\theta(\mathbf{x})))$. The contractive cost function is updated as:

$$\mathcal{J}_{CAE}(\theta) = \sum_m L(\mathbf{x}, g_\theta(f_\theta(\mathbf{x}))) + \lambda \left\| J(\mathbf{x}) \right\|_F^2 \tag{2-6}$$

where $J(\mathbf{x}) = \dfrac{\partial f_\theta}{\partial x}(\mathbf{x})$ is the Jacobian matrix of the encoder at $\mathbf{x}$, and λ is the hyperparameter to control the regularization. Contractive autoencoders encourage the robustness of the representation by penalizing the sensitivity of the features rather than regularizing the reconstruction that offers better performance compared to other regularized models.

2.4.4.4 Variational Autoencoder

Variational autoencoders (VAEs) proposed by Kingma et al. [61] are directed generative models that use a variational inference framework to approximate the input data distribution $p(\mathbf{x})$ and can be trained with gradient-based methods [62]. VAEs are attractive deep models as they bridge the gap between neural networks and probability models, and make it possible to design generative models of large complex datasets.

Similar to any autoencoder, they have an encoder/decoder architecture, although the math behind the structure has little to do with other well-known autoencoders. Variational autoencoders are composed of two neural networks: inference network (encoder) with the weight parameters φ, and generative network (decoder) with the parameter space θ. For sampling, the network draws a sample of h from latent distribution, $h \sim p_\theta(\mathbf{h})$, which is same as the prior distribution in an inferential context and is usually defined as a centered isotropic Gaussian distribution, i.e. $\mathcal{N}(\mathbf{h}, 0, I)$. Then, the generative network is used to construct a parameterized distribution $p_\theta(\mathbf{x}|\mathbf{h})$, which can be Gaussian or Bernoulli, depending on the data type (binary or real-valued).

The posterior distribution of latent space over observed data, defined as $p_\theta(\mathbf{h}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{h}) p_\theta(\mathbf{h})$ is intractable and impossible to evaluate, therefore the recognition network (encoder) estimates the approximate distribution of $q_\varphi(\mathbf{h}|\mathbf{x})$ by minimizing KL divergence between the posterior distribution $p_\theta(\mathbf{h}|\mathbf{x})$ and approximate distribution [63]. The authors proposed a reparametrization Trick for generating samples via parametrizing $\mathbf{h}$ as a $h = g_\varphi(\boldsymbol{\varepsilon}, \mathbf{h})$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0,1)$ is an auxiliary variable. Reparameterization to $h = \mu_h(\mathbf{x}) + \sigma_h(\mathbf{x}) \odot \boldsymbol{\varepsilon}$ ($\odot$ is element-wise product) results in following training objective of the model to optimize the variational bound over the marginal log-likelihood of the observations [64]:

$$\mathcal{L}(\theta, \varphi, \mathbf{x}) = -KL(q_\varphi(\mathbf{h}|\mathbf{x}) \| p_\theta(\mathbf{h}) + \mathbb{E}_{q_\varphi(\mathbf{h}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{h})] \tag{2-7}$$

where the second term on the right-hand side is the reconstruction term. The reparametrization helps us to use backpropagation to train both the encoder and the decoder simultaneously.

2.4.5 Convolutional Neural Network

Convolutional neural networks (CNNs) are deep discriminative networks and have shown pleasing results in processing the data with grid-like topology. The key difference between CNNs and standard neural networks is that CNNs benefit parameter sharing scheme, which allows the network to look for specific features at different positions [62]. 2.4.4Figure 2.13.4 shows the schematic of a typical 2-D CNN characterized by three layers, i.e., convolutional, pooling and fully-connected layers. The convolutional layer carries out the convolution operation on the input data by sliding a filter (kernel) over the input to produce a feature map. The pooling layer aims to reduce the dimension of the feature map which reduces the number of the parameters and increases the shift-invariance property which leads to better robustness against noise [65]. The final fully connected layers map the data to a 1D feature vector, which can be either used by a classifier [66] or as a feature vector for further processing [66].

The training process works in the same way as standard neural networks using backpropagation. Based on various improvements of CNN schemes in structural reformulation and regularization perspective, deeper CNN architectures have been emerged and achieved successful results in different domains [67]–[69].

2.4.6 Recurrent neural network

Recurrent Neural Networks (RNNs) contain feedback loops to remember the information of former units and are the most suitable for sequential data such as natural language and time-series data. During the training process, the hidden unit h_t is sequentially updated based on the activation of the current input x_t at time t and previous hidden state h_{t-1} . Owing to the recursive structure, RNNs are capable of capturing long-term temporal dependencies from time series and sequential data but they suffer vanishing or exploding gradient problem, in the sense that during the propagation of the gradients back to the initial layers, the small gradients shrink and eventually vanish. On the other hand, if they are larger than one, they accumulate through numerous matrix multiplications and result in the model collapse [70]. Gated recurrent unit (GRU) and long short-term memory (LSTM) cells are popular variants of RNN that try to alleviate the aforementioned problem [62], see Figure 2.13.5. Bi-directional recurrent networks (BRNN), as shown in Figure 2.13.4 can increase the model capacity by sequencing the data in both forward and backward directions.

2.4.7 Generative adversarial network

Generative adversarial networks (GANs) proposed by Goodfellow et al. [71], are powerful generative models consisting of two neural networks: discriminator and generator. The generator $\mathcal{G}_\theta(\mathbf{z})$ as the generative part of the model learns the distribution of the inputs and creates the fake data, while the discriminator $\mathcal{D}_\psi(\mathbf{x})$ as the adversarial part, takes in both fake and real data and evaluates them for authenticity, Figure 2.13.4. The training process is similar to a min-max two-player game between discriminator and generator in game theory, that tries to reach a unique Nash-equilibrium of the players, i.e., the parameters set (θ,ψ) at which none of the players, neither the discriminator nor the generator changes their action, given the other player action. GANs produce appealing results, but they are commonly challenging to train and suffer from diverging behavior, mode collapse, and vanishing gradients issues [72], [73]. Considerable recent research has been devoted to finding solutions and a better understanding of the training dynamics [74]–[77].

The Original GAN model uses fully-connected networks for generator and discriminator. However, many recent studies developed more beneficial variants using AEs, CNNs and RNNs architectures. Considering the architecture of the networks or the choice of the loss function, different variants demonstrated a significant impact on the performance of the model. For more information about various GAN architectures, the reader may refer to [78].

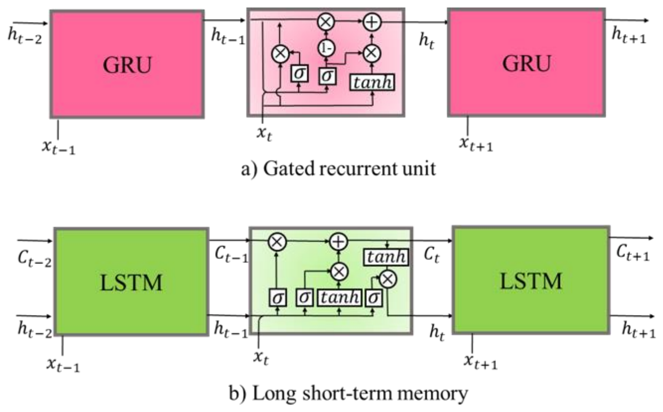


Figure 2.13.5 Two well-known recurrent cells: the sigmoid function is denoted as σ, and $\otimes$ is an element-wise operator. h_t and C_t indicate the hidden state and memory state at a time step t, respectively.

2.4.8 *Optimization in deep neural networks*

Artificial neural networks, as universal function approximators, are designed to learn any function. The multi-layered architecture of deep networks makes them able to handle complex non-linearly separable. However, the convergence or outstanding performance of deep learning is highly reliant on the critical factors such as activation function selection, weight initialization, hyperparameters (learning rate, number of layers and neurons at each layer and, etc.), optimization and regularization methods throughout training the model.

Activation functions are non-linear functions incorporated into artificial neural units (namely neurons, Figure 2.13.6), which receive the bias term and the weighted sum of the inputs from the previous layer and let deep neural networks to learn highly powerful representations over forward propagation and backpropagation algorithms. Table 2.13.4 **Table 2.13.4**summarizes popular activation functions in deep neural networks. Although choosing the proper function depends upon the type of the problem and depth of the network, it is recommended to start with ReLu (only for hidden layers), and then move over to alternatives if ReLu does not perform well.

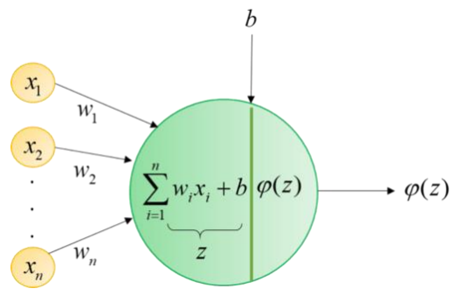


Figure 2.13.6 Schematic of a single neuron showing the inputs x_n , corresponding weight w_n , bias b , and activation function φ .

Table 2.13.4 Well-known activation functions in deep learning.

Function	Equation	Characteristics
Sigmoid	$\varphi(z) = 1/(1+e^{-z})$	Pros: normalized output between 0 and 1, smooth gradient, Cons: Computationally expensive, vanishing gradient, not zero-centered outputs, usually not used in deep models except in the output layer for binary classification
Hyperbolic Tangent (tanh)	$\varphi(z) = \tanh(z)$	Pros: zero-centered output, smooth gradient, noise-robust representation Cons: Computationally expensive, suffers from vanishing gradient but still better than sigmoid for hidden layers
Rectified Linear Unit (ReLu)	$\varphi(z) = \max(0, z)$	Pros: computationally efficient, no vanishing gradient, the most common function for hidden layers Cons: overfitting, bias shift because of non-zero mean activations, dying ReLu problem in which the inputs approach zero or negative values with zero gradients.
Leaky ReLu	$\varphi(z) = \max(\alpha z, z)$, α is a hyperparameter, and normally is 0.01.	Pros: solves dying ReLu issue Cons: not consistent results for negative values, similar to ReLu
Parametric ReLu	$\varphi(z) = \max(az, z)$, a is a learnable parameter	Pros: the results are better than Leaky ReLu by choosing proper a values Cons: it may reduce the overfitting problem with good a and appropriate regularization, similar to ReLu
Exponential Linear Unit (ELU)	$\varphi(z) = \begin{cases} z & if\ z > 0 \\ \alpha(e^z - 1) & if\ z \leq 0 \end{cases}$ α is hyperparameter	Pros: Solves bias shift problem, faster learning for deeper models Cons: saturates for large negative values and results in inactive neurons
Softmax	$\varphi(z)_i = e^{z_i} \Big/ \sum_{k=1}^{k=K} e^{z_k} \quad for\ i = 1, \ldots,$	Ranges the output between 0 and 1, different from other activation functions gives multiple outputs for the vector of inputs and usually used in the last layer for multi-class models

The optimization algorithm plays a vital role in training. The gradient descent method is the first-order optimization technique that is widely used for training. It converges slower than second-order alternatives such as Newton and Conjugate gradient methods. The traditional gradient descent technique runs through the whole training dataset to perform a single update of model parameters or weights. Hence, it can be slow and time-consuming to train a very large dataset. To remedy the issue, the usual practice is to perform the update for a sub-set of training samples [79]. The algorithm was traditionally called mini-batch gradient descent. However, now they are simply referred to as stochastic gradient descent (SGD). Despite the effective training process, SGD faces challenges such as the proper learning rate selection, dealing with the sparsity of data, and minimizing highly non-convex error functions while avoiding sub-optimal local minima. Various algorithms are proposed to tackle the challenges of SGD, see Table 2.13.5.

Table 2.13.5 Optimization algorithms in deep architectures*.

Parameter update	About
Vanilla SGD: $\theta_{t+1} = \theta_t - \eta g_t$	It reduces the variance of the parameters and leads to more stable convergence comparing to batch gradient descent, but the learning is slow.
Momentum: $\theta_{t+1} = \theta_t - v_{t+1}$, $v_{t+1} = \gamma v_t + \eta g_t$	Accelerates SGD learning process by accumulating the past gradients and moving in their directions at speed v .
Nesterov momentum: $\theta_{t+1} = \theta_t - v_{t+1}$, $v_{t+1} = \gamma v_t + \eta \tilde{g}_t$	Corrects standard momentum by evaluating the current gradient after applying the velocity at that time step.
AdaGrad: $\theta_{t+1} = \theta_t - \left(\eta / \sqrt{G_t + \delta}\right) \odot g_t$, $G_t = G_{t-1} + g_t \odot g_t$	AdaGrad is based on adapting the learning rates of all the parameters by dividing the rate by the square root of the sum of past and current squared gradients. The accumulation of gradients makes the learning rate to shrink to infinitesimally small values for deep networks.
RMSprop: $\theta_{t+1} = \theta_t - \left(\eta / \sqrt{\hat{G}_t + \delta}\right) \odot g_t$, $\hat{G}_t = \beta \hat{G}_{t-1} + (1-\beta) g_t \odot g_t$	Solves diminishing learning rate issue of AdaGrad by dividing the rate by exponentially weighted average of squared gradients.
Adam: $\theta_{t+1} = \theta_t - \eta \hat{s}_t / \left(\sqrt{\hat{r}_t} + \delta\right)$, $s_t = \beta_1 s_{t-1} + (1-\beta_1) g_t$ $r_t = \beta_1 r_{t-1} + (1-\beta_1) g_t \odot g_t$, $\hat{s}_t = s_t / 1-\beta_1^t$, $\hat{r}_t = r_t / 1-\beta_2^t$	It can be seen as the combination of RMSprop and momentum to enhance the falling learning rate problem in AdaGrad. Adam achieves huge improvement in terms of speed of training. However, it's shown to have convergence issue for some datasets. Several strategies have been proposed to benefit Adam optimizer while solving the convergence issue
AdaMax: $\theta_{t+1} = \theta_t - \eta \hat{s} / R_t$, $R_t = \max(\beta_2 R_{t-1}, \lvert g_t \rvert)$	AdaMax is a variation of Adam which uses the exponential moving average of gradients and past p-norm of gradients, and mostly used for the settings with sparse parameter updates.

*θ : parameters, η : learning rate, m : minibatch size, $f(x^{(i)};\theta)$: the predicted output where $x = \{x^{(i)}\}_{i=1}^{m}$ is the training minibatch, $y^{(i)}$ is the ground truth target, $\mathcal{L}$: loss function, $g_t = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta \mathcal{L}(f(x^{(i)};\theta_t), y^{(i)})$: gradient estimate at time-step t, $\odot$ denotes element-wise products, $\tilde{g}_t = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta \mathcal{L}(f(x^{(i)};\tilde{\theta}_t), y^{(i)})$: gradient estimate at interim point $\tilde{\theta}_t = \theta_t - \gamma v_t$, γ : momentum coefficient, δ : a small constant for numerical stability, β, β_1, β_2 : exponential decay rates, s : biased first moment estimator for Adam and AdaMax, r : biased second moment estimator for Adam, R_t : second moment estimator for AdaMax.

Another critical problem in the context of deep learning is to train the models that perform well on both training and test datasets. In this context, regularization is defined as "any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error [62]. A summary of the most common regularization algorithm is shown in Table 2.13.6.

Table 2.13.6 Regularization in training deep networks [62].

Technique	About
L2-Regularization	Aka weight decay, calculates the sum of the squared values of the weights, it shrinks the weight vector and is the most common norm penalty term added to the objective function.
L1-Regularization	The norm penalty includes the sum of the absolute values of the weights, encourages more sparse weights that L2 which is good for feature selection, as opposed to L2 does not provide clear algebraic solutions for quadratic approximations of the objective function.
Early stopping	The idea is to find model parameters for the best validation error by terminating the training as soon as the validation error starts to increase, and returning the model settings to the previous parameters with the lowest validation error.
Data augmentation	Make the training set larger by generating fake data, if possible.
Bagging	Or bootstrap aggregating, an ensemble learning technique in which several models are separately trained on bootstrapped samples, and then aggregated to make the averaged model. It reduces the variance and has a regularization effect by reducing the generalization error."
Dropout	A widely used regularization method in which the dependency among the units is reduced by randomly ignoring some units during the training.
Parameter sharing and tying	Constrain sets of parameters (in various models or components of one model) to be equal according to prior-knowledge on models' dependency. It is widely used for domain adaptation task. CNN networks also benefit parameter sharing to reduce the number of parameters.
Manifold regularization	Manifold regularization techniques, such as tangent prop, manifold tangent classifier, etc., are data-dependent methods and are based on the idea that data of the same classes mostly comes from the same low-dimensional manifolds

Table 2.13.7 gives a summary of the most well-known deep networks and their characteristics.

Table 2.13.7 Summary of deep baseline models and their characteristics.

Network	Representation learning	Merits	Demerits
RBM			
DBN	Unsupervised/ generative	• Achieves appealing results with raw vibration data, without considerable preprocessing effort	• Model performance is highly reliant on initialization of the parameters
DBM	Unsupervised/ generative	• Learns complex representations • Top-down feedback allows good uncertainty propagation	• Intensive computation of joint optimization
Autoencoder			
SAE	Unsupervised/ discriminative	• Easy implementation • Tractable optimization function • Dimensionality reduction technique*	• Not good in preserving the relationships among inputs • Risk of learning identity function without extracting meaningful information
SSAE	Unsupervised/ discriminative	• Better generalization by introducing sparse features • Easily separable classes due to sparse features	• Less robust than other regularized autoencoders
DAE	Unsupervised/ discriminative	• Learns a robust representation • Reconstructing the clean data from a corrupted input • Easy implementation	• Stochastic regularization • Partial robustness
CAE	Unsupervised/ discriminative	• More robust representation than DAE (encouraging robust features rather than robust reconstruction) • Analytical regularization • Deterministic gradient • More stable than DAE	• High computational cost • Not probing large perturbations
VAE	Unsupervised/ generative	• Learns the complex probability distributions of latent space other than fixed scalars, and can generate new instances • Data imputation ability to handle incomplete datasets	• Results are dependent on the expressiveness of the inference • Bad local optima issue
CNN	Supervised/ discriminative	• Preserves spatial information • Good for high-dimensional data	• Overfitting • Model accuracy is highly reliant on parameters initialization
RNN/BRNN		• Suitable for sequential and time-series data • Captures time dependencies of data • Stores temporal information	• Difficult training process • Vanishing/exploding gradient
GRU	Supervised/ discriminative	• Simpler than LSTM and computationally more efficient • Vanishing/exploding gradient remedy • Better generalization than LSTM with less data	• Slightly less control than LSTM over information flow
LSTM	Supervised/ discriminative	• Better than GRU in dealing with vanishing/exploding gradient issue • Remember longer sequences than GRU • Better generalization than GRU with more data	• Complicated structure • Slower training process compared to GRU
GAN	Unsupervised/ generative	• Not requiring Markov chains • Often generating the most realistic samples among other generative models • Active research area toward promising results	• Training instability • Difficult optimization • Mode collapse of the generator which leads to generating samples with the limited variety • Subjective evaluation

* All the variants have the property in common.

2.5 TAXONOMY OF DEEP LEARNING IN PHM

In recent years, research on the use of deep learning for representation learning, time series classification, and prediction in the field of PHM has gained growing attention. The DNN models can be mainly divided into three categories: generative models, discriminative models, and hybrid models, see

Generative models define a joint probability distribution over input and target variables and can be used to generate new instances from the underlying distribution of data. Among the models in this class are VAEs, DBMs, DBNs, and GANs. Discriminative models estimate the conditional probability distribution $P(\mathrm{y}|\mathrm{x})$, where y and x are the target variable (discrete class or scalar prediction) and observation variable, respectively. Discriminative models do not attempt to model the underlying distribution of the variables and only perform a mapping from the inputs to the desired targets [80]. CNNs, RNNs, and autoencoders (excluding VAEs) are common discriminative models in PHM.

In the current survey, the hybrid models are the deep architectures that combine generative models, discriminative models or both to enhance the performance of the model. In such architectures with a generative component, the generative part aid the discrimination either in optimization via providing a good initialization or by reducing the overall complexity of the model [81]. These architectures can take advantage of both discriminative and generative models. The application of the models mentioned above in PHM is thoroughly discussed in the following subsections. Other than the networks shown in Figure 2.13.7, several studies proposed new DNN architectures that have generative and/or discriminative natures. These models are discussed as "hybrid and emergent models" under section 2.12.

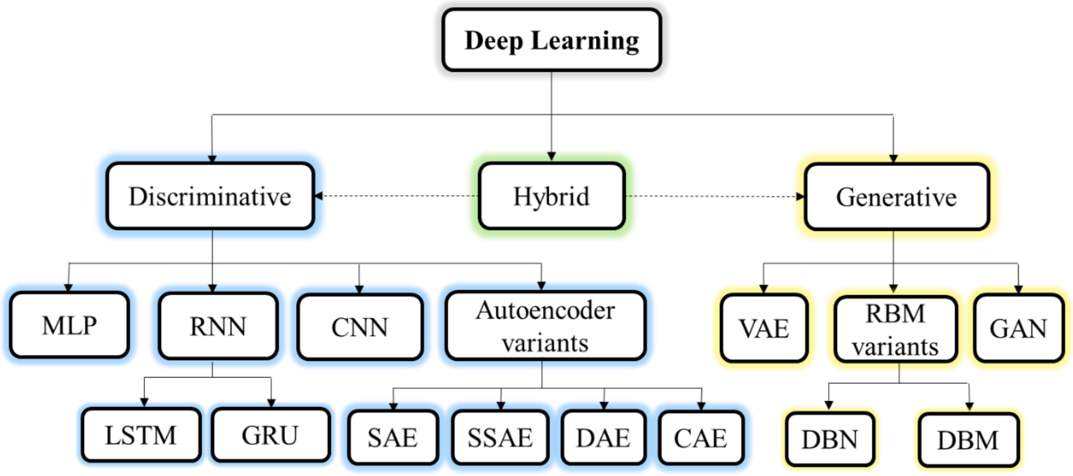


Figure 2.13.7 Taxonomy of deep learning architectures in PHM.

2.6 DEEP BELIEF NETWORKS

Deep belief networks are the first successfully trained deep networks and the first deep model applied in the PHM domain. Tamilselvan and Wang [82] developed a DBN-based multi-sensory fault diagnosis framework and leveraged the hierarchical architecture of DBN to handle heterogeneous sensory signals. Similarly, Tran et al. [83] used DBN classifier with Gaussian Bernoulli units for fault diagnosis of reciprocating compressor valves. They extracted time-domain and frequency-domain features of heterogeneous signals and applied generalized discriminant analysis (GDA) to reduce the dimensionality of the feature space.

Despite the substantial improvement of the mentioned studies over conventional models, hyperparameters of the model such as the number of layers, size of the layers, and learning rate have been randomly selected, which is shown to reduce the model efficiency significantly. To remedy the issue, Shao et al. [84] adopted a Particle swarm optimization (PSO) algorithm to decide the optimal hyperparameters for the fault diagnosis of the rolling element bearings. In their recent study, Tang et al. [85] proposed an adaptive learning rate with Nesterov momentum to accelerate network training and improve the performance.

Deep belief networks may act as intermediate feature extractors. Yuan et al. [86] trained two DBNs to learn intermediate representations of vibration and acoustic emission signals separately. They used the wavelet packet transform (WPT) features as the inputs of DBNs. Furthermore, Liang et al. [87] proposed a novel raw signal segmentation method named Grassmann manifold-angular central Gaussian distribution to capture fault impulse information. A DBN is adopted to reduce the feature space dimension and extract more discriminative features. In [88] a new vibration imaging method is used to capture fault information of the rotor system in different directions. Pretraining of a deep belief network with a vibration image is conducted in an unsupervised manner for high-level and scalable feature extraction. Deutsch and He [28] made use of DBN to predict the remaining useful life of rolling element bearings for prognostics applications.

There are other studies that addressed the application of DBN in system health assessment [89]–[93]. While most of them still need hand-crafted features and manual signal processing expertise, few studies used DBN as an end-to-end solution from raw input data and achieved comparative performance [94]–[96].

2.7 DEEP BOLTZMANN MACHINES

Although deep Boltzmann machines are powerful in capturing complex representations of data, particularly in cases of non-stationary signals and multi-sensory data with different modalities, their inference process is slow and costly, and the authors found limited studies that applied DBMs in PHM applications. Li et al. [97] adopted separate Gaussian Bernoulli DBMs (GDBMs) to extract high-level features of vibration signal in three modalities for gearbox fault diagnosis. A support vector classifier is used to fuse the representations towards the effective fault classification, and the model is verified with both spur and helical gearboxes. In [98] the authors applied DBMs to learn representations of acoustic emission and vibrations signals for fault diagnosis of the gearbox. A collaborative approach is proposed by Hu et al. [99] to deal with industrial fault diagnosis. A DBM turns the raw inputs to binary feature vectors, and the forest ensemble is used to concatenate the features. They utilized sliding windows to truncate the feature vectors, and a complete-random forest performs the classification. In their recent work, Wang et al. [100] leveraged DBM for the prognosis of the centrifugal compressor in smart manufacturing. Raw vibration signals are normalized through Gaussian neurons of DBM, and the model can learn the complex representation of the input sequence. The Particle Swarm Optimization algorithm searches the optimal hyperparameters, and a hybrid modified Liu-Storey conjugate gradient accelerated the pre-training step of the model.

2.8 DEEP AUTOENCODERS

After convolutional neural networks, deep Autoencoders are the most studied deep models in PHM applications. The earliest deep AE models, specifically stack multiple autoencoders to learn more complex representation of the data. For instance, Zhou et al. [101] utilized SAE for bearing fault classification problem. Shao et al. [102] adopted a deep AE with a modified maximum correntropy-based loss function for fault diagnosis of the gearbox and electrical locomotive roller bearings. The modified loss function is more robust to non-stationary noises and enhances the feature learning task. An artificial fish swarm algorithm (AFSA) is used to optimize the hyperparameters. In their other study [24], they proposed an ensemble deep AE model for intelligent fault diagnosis of rolling bearings. Firstly, raw vibration signals are fed to various SSAEs with different activation functions, and the Softmax classifier performs the fault diagnosis. A new combination strategy based on majority voting determines the threshold value for diagnostic accuracy of individual SSAEs, and the ensemble model is used for feature learning with the training samples. Furthermore, in [24] they proposed a novel deep autoencoder model with Gaussian wavelet activation functions and raw vibratory signals.

Regularized autoencoders enhance the generalization of the model and provide more robust representations. There have been numerous studies that leveraged regularized variants of autoencoders in PHM applications. For example, the rolling bearing fault diagnosis framework described in [103] benefits modified DAE with an improved norm penalty and new preprocessing method. Capturing the temporal dependency of the measurement data is a challenging task in vibration-based fault diagnosis. To address the issue, Jiang et al. [104] proposed a deep DAE-based model for wind turbine fault detection.

Firstly, the sliding window is applied to multi-sensory time-series data to capture the current and the past temporal information in a small time frame. Then, the robust multivariate reconstruction of the processed data is built via DAE.

Most of the studies above are based on the assumption of stationary operating conditions; however, real-case machinery work under varying conditions and the signals are non-stationary, which makes it challenging to extract fault features. Luo et al. [105] built SSAE for the early fault detection of CNC machine. The vibration signals are divided into fixed-length smaller samples using a sliding frame and labeled into impulse vs. non-impulse classes. The SSAE model is trained to determine the impulse responses of the data. The state-space model is adopted to estimate the dynamics of the machinery using impulse response data, and a dynamic property similarity-based health indicator is constructed for health monitoring tasks. In [106], the authors utilized SSAE for fault diagnosis of the gearbox with emerging new fault conditions. The proposed SSAE framework assigns new labels to the samples that deviate from Gaussian distribution and achieved higher accuracies compared to the standard SSAE model. Liu et al. [107] utilized SAE for multi-sensor fusion-based fault diagnosis of rotating machinery.

Wang et al. [108] adopted a batch normalization optimization method to reduce the internal covariate shift problem between hidden layers of SSAE for gearbox fault diagnosis and achieved the results superior to raw SSAE model. Sun et al. [109] used the compressed sensing idea with less measured data for SSAE-based fault diagnosis of rolling bearings. The selective stacked denoising autoencoder network with negative correlation learning (Selective-SDAE-NCL) is proposed by Yu [110] for gearbox fault diagnosis. In their model, the ensemble supervised fine-tuning of SDAE components via NCL is used to account for different aspects of the data. The PSO algorithm produces the optimal subset of SDAE components, Figure 2.13.8. Other PHM studies with stacked sparse denoising autoencoders (SSDAE) have been carried out by Jian et al. [111] for wind turbine fault diagnosis, Lu et al. [112] and Guo et al. [113] for fault diagnosis of rolling bearings, Shi et al. [114] for tool condition monitoring, Zhang et al. [115] for fault diagnosis of solid oxide fuel cell system.

Despite the achievements of DAE for automatic feature extraction in fault diagnosis applications, it is challenging to select the best corruption level. Some authors used contractive autoencoder (CAE) for more convenient and robust representation learning. For example, Shen et al. [116] proposed a CAE-based model for automatic feature learning in the gearbox and rolling bearing fault diagnosis problems. The input was frequency domain data and compared to other regularized autoencoders, they could obtain higher correlation coefficients under different signal to noise ratios (SNRs).

Contractive autoencoders penalize the sensitivity of the features and encourage the robustness of the representation rather than of the reconstruction, as in denoising autoencoders. Hence, they may provide better performance and generalization. However, CAE cannot probe large perturbations of the input. Shao et al. [117] suggested an enhanced feature learning method by combining the characteristics of DAE and CAE for fault diagnosis of electrical locomotive bearings. The raw vibration signals are fed into DAE to extract low-level fault features. Stack of multiple CAEs is then used for higher-level and robust feature extraction. Similarly, in [118], the authors leveraged a hybrid autoencoder representation learning based on DAE and CAE for fault diagnosis of rolling bearings from raw vibration signals.

Although the generative variant of autoencoders, known as variational autoencoder (VAE) has shown outstanding results in complex latent representation learning in varied applications, few studies used VAE in the field of PHM. Ping et al. [119] utilized VAE to extract deterioration features of complex rotary machinery. They proposed log-normally distributed latent variables instead of standard normal units to address heteroscedasticity issue of degradation data. In another study [120], the authors leveraged deep VAE for fault diagnosis of rolling bearings using raw vibration measurements. Zhan et al. [121] integrated VAE in a semi-supervised learning-based network with multiple association layers for fault diagnosis of the planetary gearbox. They applied wavelet packet transform to capture impulse components of the vibration signals, and trained the model with a combination of labeled and unlabeled samples. A conditional VAE (CVAE) network [122] is adopted for planetary gearbox fault diagnosis under noisy conditions. As opposed to standard VAE, CVAE models the features conditioned on some random variables and achieves a better reconstruction. Despite the successful examples above, there is still room for leveraging VAE in health monitoring applications, especially for dealing with heterogeneous and incomplete data in real industrial machinery [123].

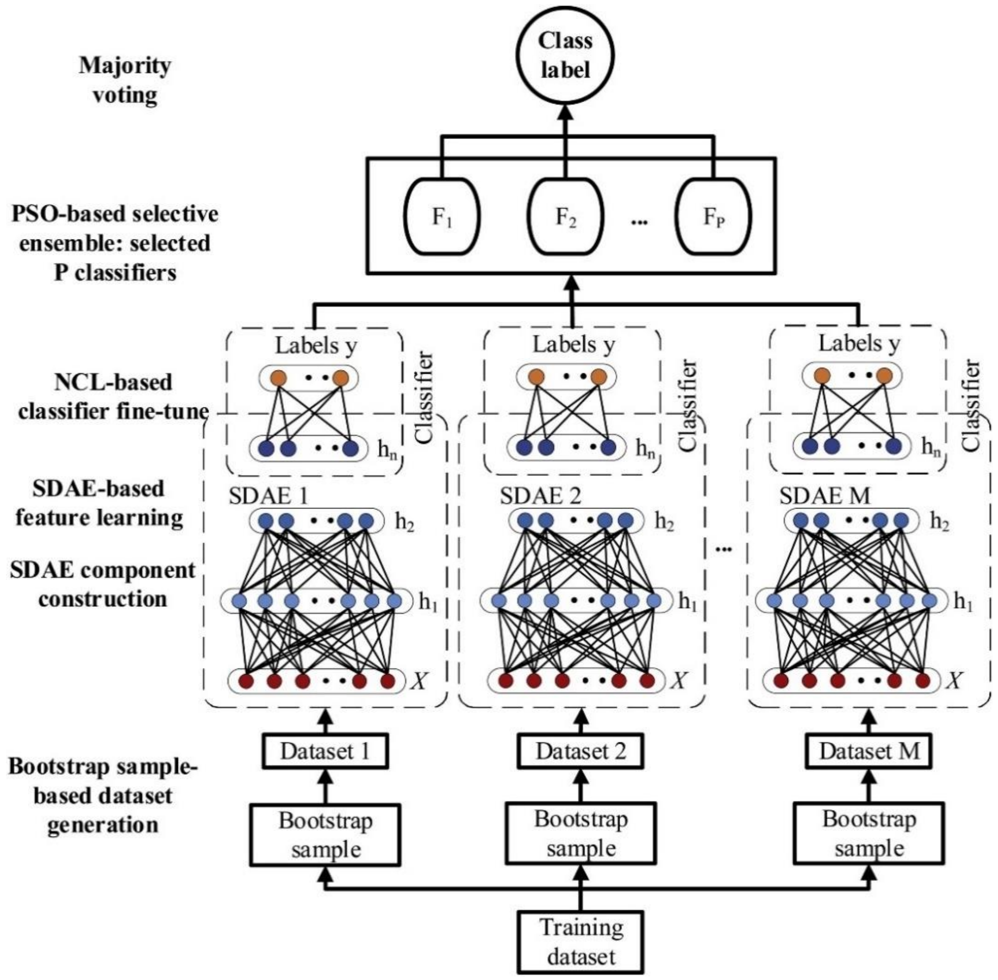

Class label
Majority voting
PSO-based selective ensemble: selected P classifiers
F1
F2
...
FP
NCL-based classifier fine-tune
Labels y
Classifier
hn
SDAE-based feature learning
SDAE component construction
SDAE 1
SDAE 2
SDAE M
h2
h1
X
Bootstrap sample-based dataset generation
Dataset 1
Dataset 2
Dataset M
Bootstrap sample
Training dataset

Figure 2.13.8 Selective-SDAE-NCL for gearbox fault diagnosis, [110]. First, the hierarchies of SDAEs are pre-trained with bootstrapped samples in an unsupervised fashion. Then, the ensemble supervised fine-tuning of SDAE components via NCL is carried out to account for different aspects of the data. Finally, The PSO algorithm produces the optimal subset of SDAE components.

2.9 CONVOLUTIONAL NEURAL NETWORKS

As shown in **Figure 2.13.1**, convolutional neural network (CNN) is the most applied deep model in the PHM field. Chen et al. [124] adopted 1-D CNN for gearbox fault identification. They fed time and frequency domain features of the vibration signal to the model and performed some parameter tunings to find the optimal architecture of CNN. Guo et al. [125] demonstrated a CNN-based health indicator (HI) construction method for rolling bearing prognosis. A novel outlier region removal technique is applied to reduce trend burr effect and enhance the prognostics performance. The new HI assessment metric named scale similarity comforts picking the proper failure threshold when HIs in the training set have different range scales. Similarly, the deep convolutional neural network was applied by Belmiloud et al. [126] for the RUL estimation of rolling bearings. Many studies have used 1-D CNN for fault classifications of rolling bearings [127]–[132]. Jing et al. [133] made use of 1-D CNN for multi-sensory fault diagnosis of the planetary gearbox. They utilized four types of signals including acoustic, vibration, current and instantaneous angular speed signals to integrate data-level, feature-level, and decision-level fusions into an optimized deep CNN. In [134], the authors used raw acoustic signals in time and frequency domains for gear fault diagnosis, and leveraged multi-channel CNN to fuse information from different microphones. Liu et al. [135] carried out the simultaneous diagnosis and prognosis of rolling bearing using a joint-loss CNN model. Zhang et al. [136] proposed a CNN-based fault diagnosis framework with residual blocks. The identity skip connections in the model allow the direct propagation of the information throughout the network and enhance high-level feature extraction.

Convolutional neural networks were originally designed for image analysis tasks. Hence, different researchers investigated approaches to preprocess and convert time-series data into 2-D inputs for the system health assessment, see Table 2.13.8 Several studies have used time-frequency analysis methods to transform vibration signals into image inputs. Han et al. [137] adopted multi-level wavelet packet matrices as the inputs to several parallel CNNs with shared parameters for gearbox fault diagnosis. Multi-level wavelet packet matrices incorporate non-stationary vibration information from multiple resolutions and cancel the need for level selection in WPT. Verstraete et al. [138] proposed a novel CNN architecture for rolling bearing fault diagnosis and compared the effectiveness of the model with three different time-frequency representations of raw signals as input images: spectrograms of short-time Fourier transform (STFT), scalograms of the continuous wavelet transform (CWT), and Hilbert-Huang transform (HHT) plots. The suggested network consists of two consecutive convolutional layers without any pooling layer between them, but there are pooling layers between stacks of two-layered convolutional blocks. The model achieves the same accuracies as standard CNNs for scalogram images with significantly less learnable parameters and computational cost but outperforms alternative CNN models for HHT images and spectrograms. In [139], Yoo and Baek demonstrated the Morlet-based CWT representation of vibration signals fed into CNN network to construct HI for remaining useful life estimation of rolling bearings. Although there is not a defined method to decide the best wavelet for various PHM scenarios, Morlet wavelets have shown effective results and high similarity to the impulse component of non-stationary signals of the faults in mechanical equipment. Zhu et al. [27] made use of binary interpolation to reduce the dimensionality of CWT image for bearing RUL estimation problem. Besides, they utilized a multi-scale convolutional neural network (MSCNN) that keeps global and local features synchronously by using the features of the last convolutional layer and the pooling layer before for prediction. A comparative study of the model with other CNN-based models verified the effectiveness of the proposed method.

Table 2.13.8 Summary of 2-D CNN based models for PHM applications. P: Prognostics, D: Diagnostics. *NRMSE*: Normalized *RMSE*.

Study	Task	Input	Architecture	Performance
Yoo et al. [139]	Bearing/P	Morlet-based CWT	CNN	*ETA*: 0.57
Zhu et al. [27]	Bearing/P	Morlet-based CWT	MSCNN	*ETA*: 0.3624, *MAE*: 1091.8, *NRMSE*: 0.3514
Zhu et al. [140]	Bearing/D	STFT	Inception	*ACC*: 97.15
Verstraete et al. [138]	Bearing/D	Morlet-based CWT HHT STFT	Doubled convolutional layers	*ACC*: 99.4 *ACC*: 97 *ACC*: 99.5
Wang et al. [141]	Gearbox/D	Morlet-based CWT	CNN	*ACC*: 99.58
Han et al. [137]	Gearbox/D	Multi-level WPT	Ensemble CNN	*ACC*: 96.48
Li et al. [142]	Bearing/P	STFT	Concatenated convolutional layers	-
Ding et al. [143]	Bearing/D	Energy-fluctuated image in WP phase space	MSCNN	*ACC*: 98.8
Ren et al. [144]	Bearing/P	Spectrum-Principal_Energy- Vector feature map	CNN	*RMSE*: 0.119
Hoang et al. [145]	Bearing/D	Gray-scale vibration image	CNN	*AC*C: 100 *ACC*: 97.74 under noisy and varying working conditions
Wen et al. [146]	Bearing/D	Gray-scale vibration image	LeNet-5	*ACC*: 99.79
Lu et al. [147]	Bearing/D	Matrix reconstruction-based feature map	CNN	*ACC*: 96.48
Hu et al. [148]	Bearing/D	Compressed sensing-based constructed image	Improved MSCNN	*ACC*: 99.4 with Gaussian measurement matrix

There are other studies that leveraged multi-scale layers in CNNs to capture more levels of abstraction in the data. In [143], Ding and He adopted the phase space reconstruction (PSR) technique to make the phase space image of the wavelet packets (WP) referred to as wavelet packet image (WPI). Combined with MSCNN, the proposed multi-scale feature learning method retains the energy fluctuations of the WP nodes, and hence, provides a robust fault diagnosis framework under fluctuating load conditions, Figure 2.13.9 Inspired by the Inception architecture of CNNs [68] and dynamic routing capsule net [149], a novel deep net with inception blocks is used by Zhu et al. [140] to address poor generalization of standard CNNs under varying working conditions.

Alternatively, a few studies adopted innovative methods to incorporate time and frequency information into the inputs. For example, Ren et al. [144] presented a new feature extraction approach named the spectrum-principal-energy vector (SPEV) for RUL estimation of rolling bearings. The vibration signal was subjected to FFT and then divided into 64 blocks. The maximum amplitude of each block was obtained to build spectrum-principal-energy-vector. The 64-dimensional vector at 64 time-steps was combined into a 64*64-dimensional feature map and fed into CNN followed by a deep feedforward network and final smoothing step to perform the regression task. Hoang and Kang [145] transformed raw vibration signals into gray-scale images. The normalized amplitude of each sample presents the intensity of the corresponding pixel in the vibration image. In [146], the authors utilized a closely similar method to convert the time-series measurements into a gray-scale vibration image. They used a CNN model based on LeNet-5 architecture, which is the earlier release of CNN for handwritten and machine-printed character recognition [150]. In [136], Zhang et al. proposed a residual learning-based CNN for bearing fault diagnosis task.

Larger-scale fault diagnosis in complex systems involves numerous disparate measurements from diverse sub-systems that make it challenging to capture the spatial dependency information between the components, especially under different operating conditions. Inspired by spatiotemporal pattern network (STPN), Han et al. [151] presented a spatiotemporal representation learning method to handle multivariate time series data for fault diagnosis of complex systems such as wind turbine with unseen fault conditions. The model. In the last step of preparing the inputs, Markov Machines are utilized to generate self-state and cross-state transition matrices that build the 2-D image of spatiotemporal features. Despite the achievements of the studies above, most of them lack enough information about the reasons for selecting certain architecture or pre-processing methods.

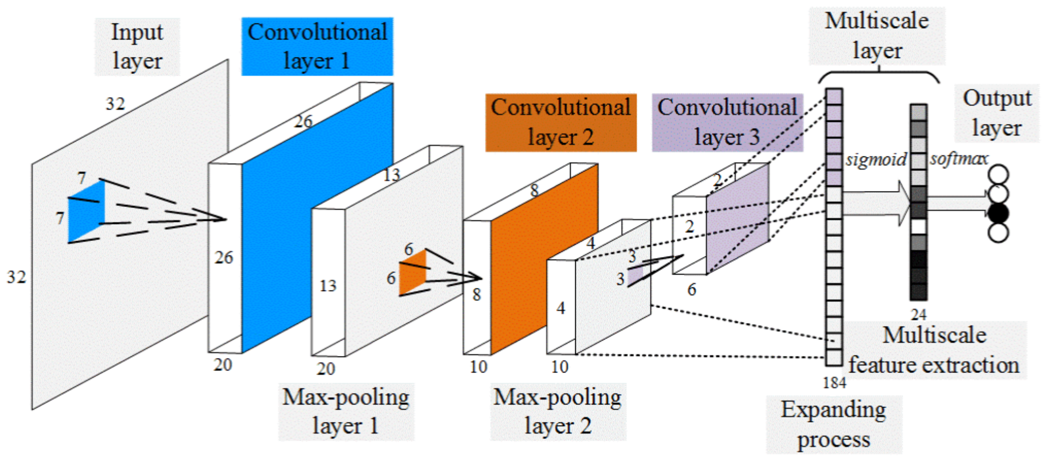


Figure 2.13.9 Deep CNN architecture with three convolutional layers, two max-pooling layers, and a multiscale layer. Multiscale layer combines the output of the last convolutional layer with the previous max-pooling layer [143].

2.10 RECURRENT NEURAL NETWORKS

Most of the system health management tasks deal with time-series measurements, and for a reliable diagnosis and prognosis framework, it's essential to capture the temporal information of the data. Owing to their internal memory and feedback loops, recurrent neural networks (RNN) can remember temporal dependencies and learn the dynamic behavior of the failure. However, vanilla RNNs (basic RNNs) suffer greatly from vanishing/exploding gradient issue and fail in learning long-term temporal dependencies. Gradient clipping technique is usually applied to limit the magnitude of the gradient by determining the threshold value. Also, different gating mechanisms are proposed to address the vanishing gradient. Long-Short-Term-Memory (LSTM) and Gated Recurrent Unit (GRU) are the two most well-known variants of RNN to remedy the issue above. Guo et al. [152] Made use of LSTM to build a health indicator for RUL prediction of rolling bearings. Firstly, they proposed a feature extraction method named related-similarity (RS) to range both frequency and time domain features from 0 to 1. After combining the RS features with time-frequency information, a linear combination of Correlation and Monotonicity metrics is adopted to select the sensitive features. Lastly, the sequence of the features is fed into RNN to construct the health indicator.

Adding more hidden layers to the RNN architecture results in deep RNN, which is much powerful in learning complex temporal dependencies of the sequential data, but introduces computational complexity to the model. Considering prognostics as a regression problem with sequence degradation index output, and the RNN's ability in handling complex sequence data, deep RNN-based health monitoring frameworks are proposed by the researchers and have shown effective results [153]–[157]. Zhang et al. [158] adopted bi-directional LSTM (BLSTM) with two hidden layers to track health index variations of the turbofan engine. Similarly, Huang et al. [159] proposed a BLSTM-based framework for RUL prediction of the engine under multiple operating conditions. Their model consists of two bi-directional LSTM networks, see Figure 2.13.10. The training set contains multi-sensory data, multi-operational data, and actual RUL values at N consecutive observation cycles. p and q denote the number of sensors and operating conditions (control settings, input settings, and etc.), respectively. Initially, multi-variate time-series are normalized and converted into desired sequenced data through a time-window processing method. Then, the normalized sensory sequences as main inputs of the model are fed into a deep BLSTM to extract degradation information with long-term dependencies. The working condition sequences are also normalized (named auxiliary inputs) and merged into the output features vector of the first BLSTM to arrange a new concatenated feature vector. The second BLSTM captures higher-level temporal information of the machinery deterioration, and multiple fully-connected layers, followed by the final regression layer complete the remaining-useful-life prediction task. The extensive comparative study with state-of-the-art deep models demonstrates the effectiveness of the proposed method in terms of machinery prognosis under complex operating variables. Standard Bi-directional LSTMs process the data sequence in both forward and backward directions, and at any point in time, the network utilizes the earlier processed observation and upcoming processed observation (by backward cells) simultaneously to perform intermediate prediction. However, the RUL estimation task requires a single prediction ahead of the whole given sequence. In [160], the authors targeted the mentioned requirement and presented a modified LSTM architecture named bidirectional handshaking LSTM, for RUL estimation from short sequences of the measurement data.

There is a limited number of studies in the literature that used GRU for PHM tasks. For example, Zhao et al. [161] adopted an enhanced bi-directional GRU model for three health monitoring case studies: tool wear prediction, gearbox fault diagnosis, and incipient fault diagnosis of rolling bearings. Firstly, the multi-sensory time-series are segmented into fixed-size windows followed by extracting local features in time, frequency, and time-frequency domains. Then, the local feature sequences are input to a bi-directional GRU to capture higher level and more discriminative information of the data. The authors concatenated the outputs of the GRU with the weighted average of the local feature sequence to avoid losing the mid-level information in the model. Kernel principal component analysis was used in [162] to fuse time, frequency, and time-frequency domains information of rolling bearing degradation. Then, the HI was smoothed through an exponentially weighted moving average technique, and fed to a hierarchical GRU-based recurrent network for future HI estimation and RUL prediction.

Although LSTM cells are the mostly used recurrent units in many applications, there is no evidence to show that one cell is superior to another. The GRUs are computationally less expensive and make the right choice for training smaller datasets. On the other hand, LSTMs may work better for bigger datasets to retain longer temporal information.

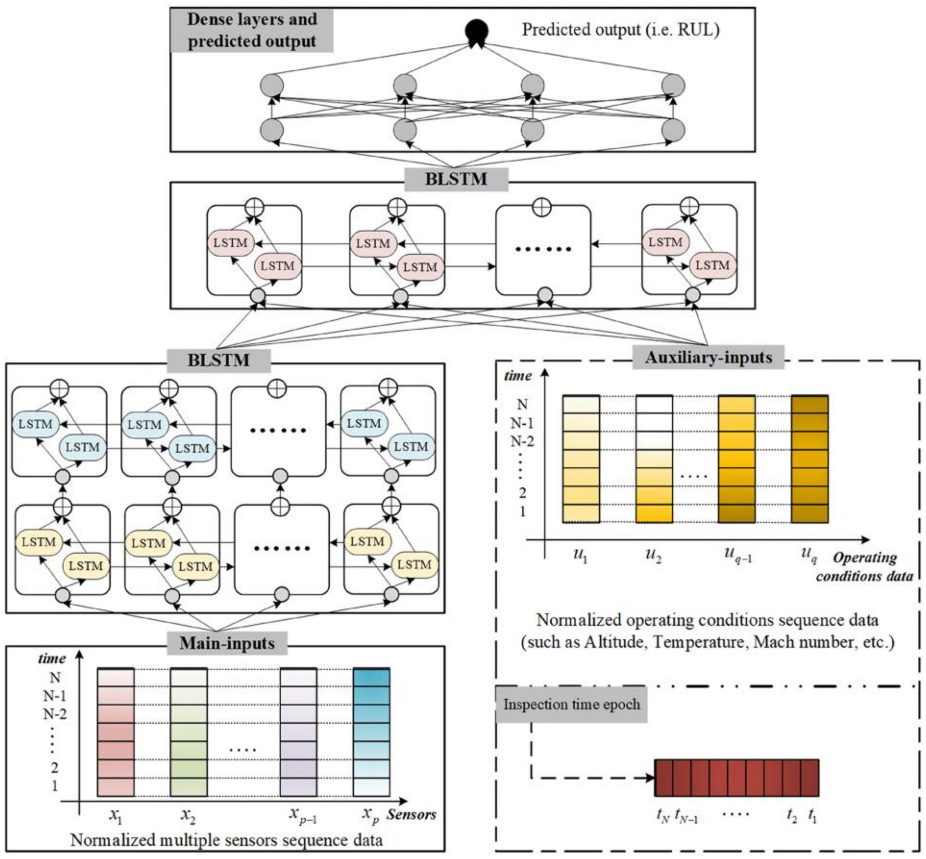

Dense layers and predicted output
Predicted output (i.e. RUL)
BLSTM
LSTM
BLSTM
Auxiliary-inputs
time
N
N-1
N-2
2
1
u_1
u_2
u_{q-1}
u_q
Operating conditions data
Normalized operating conditions sequence data (such as Altitude, Temperature, Mach number, etc.)
Inspection time epoch
t_N t_{N-1} .... t_2 t_1
Main-inputs
x_1
x_2
x_{p-1}
x_p
Sensors
Normalized multiple sensors sequence data

Figure 2.13.10 The BLSTM-based prognostics framework, adapted from [159]. Initially, normalized multi-variate time-series acquired from p sensors are fed into the first deep BLSTM to extract degradation information with long-term dependencies. Then, q operating conditions are sequenced and normalized within the inspection time frame t, through time-window processing method. Finally, the second BLSTM is fed with the concatenated vector of features from previous steps, accompanying actual RUL values, to capture more discriminative features.

2.11 GENERATIVE ADVERSARIAL NETWORKS

Generative adversarial networks (GANs) have attracted growing interest in various research areas, and have shown some advantages over other well-known deep generative models, e.g., VAE, in synthesizing excellent-quality samples. Besides, they are trained without any explicit density function, and no Markov chains are required neither in drawing the samples nor in training. Hence, there is no risk of chain breakage in high-dimensional space as in DBMs, and they have demonstrated appealing performance in dealing with the high-dimensional distribution of data. However, despite outstanding success in generating sharp synthetic images followed by an exceptional performance in the computer vision area, there are limited studies that investigated using GANs in other domains for time-series sensor data.

Recently, the PHM community has started to leverage GAN to enhance their model targeting two major concerns in industrial fault classification: The imbalanced distribution of health classes and insufficient labeled data; most of the fault diagnosis frameworks assume equal proportions of the data for all health conditions. However, the real machinery mostly works under normal conditions, and fault rarely happens. So there are abundant healthy class data, while faulty samples are limited. Also, it is unmanageable to stop the machinery during the operation and inspect the fault types. Therefore, the majority of the collected data are unlabeled. To face the first challenge, Li et al. [163] proposed an end-to-end 2-D CNN-based GAN model for bearing and gearbox fault diagnosis. In their model, the concatenated vector of the labels and the randomly generated noises are reshaped into 2-D feature maps and input to the generator. The generator consists of three consecutive deconvolution layers that map the inputs into the higher-resolution feature maps. It should be noted that in the CNN context, the term deconvolution refers to the transpose convolution (aka fractionally-strided convolution) and conducts up-sampling by padding the feature maps with the zeros. The generated data and the real data are fed into the discriminator that contains three convolutional layers without any pooling. The discriminator has the mission to find the true data and to identify the fault classes. The model was able to enrich the faulty classes data and handle the imbalanced data issue.

In [164], the authors established a semi-supervised anomaly detection algorithm for imbalanced industrial time-series based on an encoder-decoder-encoder structured generator with convolutional layers. The model is trained just with normal samples, and the test phase considers both normal and faulty conditions. The semi-supervised convolutional GAN combined with switchable normalization was used in [165] for vibration-based fault diagnosis of rolling bearings. Canceling the pooling layers and replacing the batch normalization with switchable normalization increased the training stability and introduced a high accuracy rate of 99.93% into the model for the bearing benchmark dataset.

The training process of the GANs suffers from instability issues, and they are prone to the mode collapse problem, which means the generator learns a limited subset of the modes and generates the same samples repeatedly. It has been shown that the design of the loss function significantly influences training stability and brings consequent issues into the model [78]. The original GAN structures use Jensen-Shannon Divergence (JSD) probability measurement metric which is proved to incur the vanishing gradient and mode collapse problems. Many studies in various fields designed alternative loss functions combined with enhanced architectures to overcome the mentioned challenges. Wang et al. [166] proposed a generalized imbalanced fault diagnosis framework based on Wasserstein generative adversarial network (WGAN). In WGAN, the Wasserstein loss provides the continuous gradient for generator training and solves the mode-collapse. Cabrera et al. [167] established an unsupervised GAN model selection mechanism to find the best WPT generator for reciprocating machinery fault diagnosis. In their model, the training process is guided by the dissimilarity of real and fake data clusters to enhance training stability. The obtained generator balances the 99% imbalanced dataset by producing more fault data. Zhou et al. [168] adopted a global optimization GAN framework to address the imbalanced class issue.

In a recent study [169], Shao et al. proposed an auxiliary classifier GAN (ACGAN) framework to augment the fault dataset, Figure 2.13.11. An auxiliary part is attached to the discriminator so that the enhanced discriminator can recognize the fake data and fault class labels simultaneously. The generator possesses 1D convolutional structure with batch normalization and generates the artificial data from random noise of latent variables with certain labels. The discriminator receives the generated data mixed with real samples to identify both source labels (1 or 0) and fault classes.

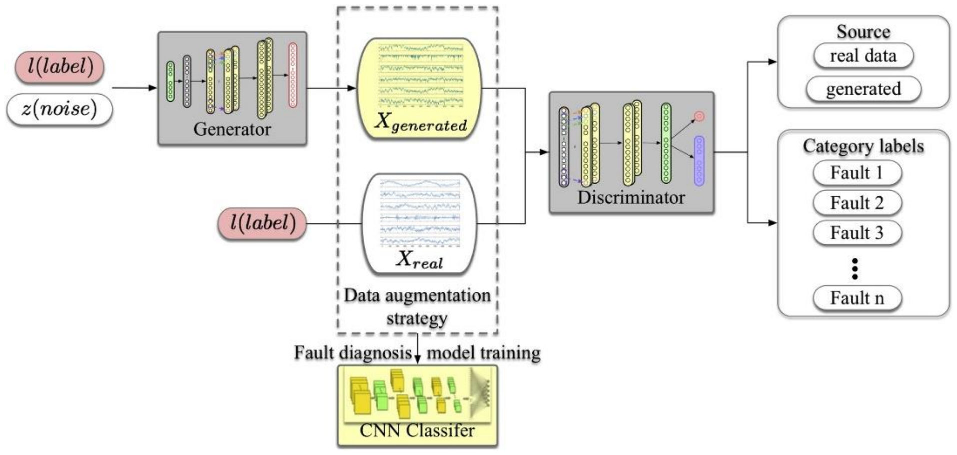


Figure 2.13.11 Auxiliary classifier GAN-based fault classification framework: First, the generator produces the samples from latent space. Then, the discriminator is trained with synthetic samples and real data. The modified loss function makes it possible for the discriminator to discriminate source labels (real or generated) and fault category labels simultaneously. Finally, the parameters of discriminator are frozen, and the generator is updated to produce more realistic sample

Many emerging generative models, such as Adversarial autoencoders (AAE) [170] and Wasserstein autoencoders (WAE) [171] are inspired by adversarial learning, and they have shown promising results in various domains [172]–[174]. However, GAN and adversarial training are somehow novel concepts and, despite offering great success in producing realistic images, the possible application of them in the context of time-series data is still very much open for future research opportunities in different directions.

2.12 HYBRID AND EMERGENT MODELS

Deep learning is a fast-growing field and there has been an enormous effort to develop new architectures that offer better performance. Many new models are the hybrid of standard architectures (i.e., CNN, RNN, AE, etc.) or rooted in the existing designs. A few studies, however, established novel ideas, but those are either mathematically complex or very application-specific. Likewise, the PHM community is actively developing more effective models. For instance, He et al. [175] established a bearing fault diagnosis framework hinged on Large Memory Storage and Retrieval Neural Networks (LAMSTAR). The LAMSTAR is a fast and deep dynamic neural network made of self-organizing-map (SOM) modules and has shown reliable results in various domains. They fed STFT of acoustic emission signals to the model and comparing to CNN-based diagnosis, achieved better performance.

Convolutional Deep Belief Network (CDBN) was originally proposed for visual recognition tasks and benefits the weight-sharing property of CNN to address the upscaling problem of DBN [176]. An improved CDBN with Gaussian visible units was used to learn representative fault features of rolling bearings [177]. The compressed sensing technique was adopted to enhance computation efficiency while preserving meaningful information. Standard CDBN suffers from error oscillation issue followed by weak generalization capability due to the limited number of Gibbs sampling steps in practical cases. An exponential moving average (EMA) weight smoothing method was employed to tackle the issue and enhance the learning algorithm. In their other study [178].

Dealing with multi-dimensional sensory data with internal dependencies is a critical challenge in most of the practical PHM frameworks. A few studies integrated convolutional and LSTM layers into a unified model to capture both spatial and temporal information of multi-dimensional time-series. Zhao et al. [179] adopted a CNN- Bi-directional LSTM (CBLSTM) network for tool wear prediction task. In [180], the authors established a CNN-LSTM (CLSTM) model with a class-imbalance-weighted loss function for imbalanced fault classification of cyber-physical-systems (CPS). Despite the satisfying results, the models above extract spatial and temporal information independently and pay less attention to feature changes between time steps. A Time-distributed Convolutional LSTM (TDConvLSTM) was proposed by Qiao et al. [181] to learn spatiotemporal information of multi-channel time-series measurements. They segmented the normalized raw data into subsequences and fed them into the model with ConvLSTM cells instead of vanilla LSTM units. The first Conv-LSTM layer simultaneously learns local spatial and temporal information inside a subsequence. Stacking a holistic Conv-LSTM unit on top of the previous layer extracts the spatiotemporal information between the subsequences.

It has been shown that the encoder-decoder structured RNNs introduce multiple improvements to the complex sequence-to-sequence tasks analogous to the RUL estimation from time-series measurements for prognostics application. Malhotra et al. [182] proposed the LSTM Encoder-Decoder (LSTM-ED) framework for unsupervised health indicator construction of the system. Similarly, the GRU Encoder-Decoder (GRU-ED) network on RUL estimation of the turbofan engine dataset demonstrated significantly robust results encountering different noise levels [183]. Moreover, DAE network with GRU hidden units has indicated the fault diagnosis accuracy superior to the standard GRU network [184].

Table 2.13.9 Survey of hybrid and emergent models where measures are rounded to two decimal places.

Publication	Task/Dataset*	Model	Performance
Shao et al. [177]	Bearing fault diagnosis/*NA*	CDBN	*ACC*: 97.37
Shao et al. [185]	Bearing fault diagnosis/ *NA*	CDBN	*ACC*:97.44
Wu et al. [180]	Fault diagnosis/PHM'15	CLSTM	*PR*: 98.42, *SN*: 98.46, *F1*: 0.98
Zhao et al. [179]	Tool wear prediction/PHM'10	CBLSTM	*RMSE*: 10.8, *MAE*: 8.1
Qiao et al. [181]	Gearbox fault diagnosis/ *NA* Tool wear prediction/PHM'10	TDConvLSTM	*ACC*: 97.56 *RMSE:* 10.22, *MAE*: 7.50
Yoon et al. [186]	RUL estimation/CMAPSS	LSTM-VAE	*MAE*: 28.1 ± 3.4
Malhotra et al. [182]	HI construction/CMAPSS	LSTM-ED	*MAE*: 18, *MAPE*: 39
Gugulothu et al. [183]	HI construction/CMAPSS	GRU-ED	*MAE*: 17, *MAPE*: 39
Liu et al. [184]	Fault diagnosis/CWRU	DAE-GRU	No noise- *ACC*: 99.75 1dB SNR-*ACC*:96.98
Chen and Li [187]	Bearing fault diagnosis/ *NA*	SSAE-DBN	*ACC*: 91.76
Lu et al. [188]	Early fault detection/IMS	AE-LSTM	Fault alarm at 527 signal snapshot
Ellefsen et al. [189]	RUL estimation/CMAPSS	RBM-LSTM	Lowest *RMSE*: 12.56, Highest *RMSE*: 22.66 Lowest *ETA*: 231, Highest *ETA*: 2840
Li et al. [190]	Cross-domain fault diagnosis/ CWRU	CNN-Generative	Lowest *ACC*: 69.4 Highest *ACC*: 84.5
Zhang et al. [191]	Cross-domain fault diagnosis/ CWRU	Adversarial CNN	Lowest *SN*: 73.75 Highest *SN*: 98.88
Han et al. [174]	Fault diagnosis/ PHM'09	Adversarial CNN	Seen condition-*ACC*:99.4, *PR*:99.3, F1:0.99 Unseen condition: *ACC*:92.5, *PR*: 92.3, *F1*:0.92
He and He [175]	Bearing fault diagnosis/ *NA*	LAMSTAR	Lowest *ACC*: 96 Highest *ACC*: 100
Yu et al. [192]	RUL estimation/CMAPSS, Milling dataset	BLSTM-ED	*ETA*: 273, RMSE: *14.74* RMSE: *7.14*

* Refer to Table 2.13.2for public datasets information.

0Table 2.13.9 provides a summary of the hybrid and emerging models. Adversarial twists of standard CNN, RNN, and AE have recently brought attention to deep learning research, and there are few related studies in the PHM field.

2.13 TRANSFER LEARNING

Compared to conventional data-driven approaches, deep learning techniques remove the burden of manual feature engineering and achieve state-of-the-art results. Despite the marvelous performance, the majority of the studies are based upon the assumption that the training and test data are drawn from the same distributions. However, in the real industry, the data are collected under different operating and environmental conditions and during different time intervals, often resulting in feature space difference or distribution shift across training and testing datasets. Moreover, labeling the industrial data is costly and error-prone, and requires huge human labor and expertise. Therefore, there is no sufficient annotated data to train reliable models. Transfer Learning (TL) focuses on improving the model by transferring the knowledge or utilizing the transferable features from one or more training datasets to execute the relevant new task in the testing dataset.

Deep TL techniques have gained growing attention recently in the computer vision field and achieved excellent results for object classification, object recognition, and semantic segmentation applications. Although some studies investigated the TL-related deep learning approaches for PHM to address different issues such as insufficient training data, class imbalance, cross-domain fault diagnosis, and covariate-shift, it is still in its infancy stage.

As opposed to traditional machine learning algorithms, the performance of deep models highly depends on the availability of massive training data to learn the latent pattern of data. However, in many domains specially PHM, it is extremely difficult to collect large-scale labeled datasets. Also, it requires extensive computational power to train the model on large-scale datasets. Recently, researchers leveraged the knowledge learned by various pre-trained models on large benchmark datasets and performed transferring the knowledge to other applications to tackle the issues mentioned above. A plethora of modified deep

CNN architectures have been trained on large-scale image datasets such as ImageNet [193]. Inception Net, GoogleNet, LeNet, AlexNet, ResNet, and VGG are some examples [194].

The idea behind TL is to fine-tune pre-trained models on new tasks in the target domain. Hence, the network on the new model can be initialized by transferred parameters instead of training from scratch. In literature, a recipe has been proposed to use pre-trained CNN architectures based on the similarity of source (pre-trained) domain and target domain, and size of the dataset, Figure 2.13.1 [195]. At present, researchers in the PHM community have begun using pre-trained models for fault diagnosis and tasks and achieved promising results. Wen et al. [196] fine-tuned all layers of an AlexNet pre-trained model on bearing fault diagnosis task. In their model, the final fully connected layer is replaced with a classifier layer with four neurons (number of bearing fault conditions). They provided a comprehensive comparison of their proposed model with eight different time-frequency image inputs and various training/testing data set ratios. In another study [197], the authors utilized a VGG-16 pre-trained network for bearing fault diagnosis. They froze bottom blocks of the network and fine-tuned the last three layers of VGG-16 with a supervised classifier layer. As most of the pre-trained networks require RGB images with three channels as inputs, it is important to pre-process the data accordingly. Wen et al. [196] proposed a signal-to-image method to convert time-domain signals into an RGB image. They have transferred the first 49 layers of a pre-trained ResNet-50 and fine-tuned the model after adding a fully-connected layer and a softmax classifier. Several studies have taken somehow similar strategies and achieved interesting results, Table 2.13.1. Despite promising results, more research is required to capture temporal features for time-series classification/prediction. TimeNet and ConvTimeNet pre-trained models are two interesting examples [198], [199].

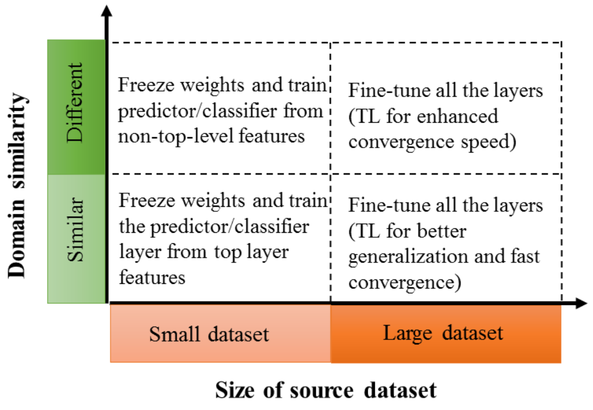

Domain similarity
Different
Similar
Freeze weights and train predictor/classifier from non-top-level features
Fine-tune all the layers (TL for enhanced convergence speed)
Freeze weights and train the predictor/classifier layer from top layer features
Fine-tune all the layers (TL for better generalization and fast convergence)
Small dataset
Large dataset
Size of source dataset

Figure 2.13.1 The general criterion to be considered in using pre-trained models for transfer learning [195]**.**

Table 2.13.1 Summary of deep transfer learning studies for PHM applications with pre-trained CNN architectures*.

Study	Dataset	Input	Architecture	Performance
Shao et al. [197]	CWRU bearing/D	Three-channel augmented WT time-frequency image	VGG-16	1. *ACC*=100 (training and testing data from same working loads) 2. *ACC*=98.80 (training and test data from different working loads)
Xu et al. [200]	CWRU bearing/D	Gray-scale CWT image	LeNet-5	*ACC*=99.08
Ma et a. [201]	CWRU bearing/D	Frequency slice wavelet transform image	AlexNet	1. *ACC*=99.89 (without SNR) 2. *ACC*=82.70~98.18 for SNR from -4~4 dB
Wen et al. [202]	1. CWRU bearing/D 2. KAT bearing/D 3. Lu et al. bearing/D	Time-domain RGB image	ResNet-50	1. *ACC*=99.99 2. *ACC*=98.95 3. *ACC*=99.2
Wang et al. [203]	1. Unknown bearing/D 2. CWRU bearing/D	Trained and compared with eight different time-frequency images	AlexNet	1. Highest *ACC*=100, Lowest *ACC*=92.57 (for HHT image, with 5% of images trained) 2. Highest *ACC*=100, Lowest *ACC*=76.10 (for Fast Kurtogram image, with 5% of images trained)
Wen et al. [196]	CWRU bearing/D	RGB time-domain image	VGG-19	*ACC*=99.17
Mao et al. [204]	PRONOSTIA bearing/Incipient fault detection	Three-channel vibration image	VGG-16	*NA*

* Refer to Table 2.4.1 for public datasets information.

2.14 HARDWARE, SOFTWARE AND COMPUTING RESOURCES

Although deep learning has shown good results on PHM problems, its applicability is impaired by high computational demand. Appropriate hardware and software are required to support effective training in complex settings [205]. In this section, we discuss three main enablers of deep learning, i.e. parallel computing, advanced libraries, and cloud/edge computing.

2.14.1 Parallel computing

Compared to traditional machine learning algorithms, deep architectures involve much larger parameter space, which should be updated at each training epoch, requiring a huge amount of matrix operations and an abundance of processing power. Parallel computing facilitates executing massive operations simultaneously. Central Processing Units (CPU), even the latest and powerful chips, have a limited number of processing units (cores) and low parallelism capability. Hence, they are not efficient for implementing deep models, and it may take weeks for them to come up with the results.

Graphics Processing Units (GPU) are originally dedicated to processing graphics and high-quality 3D games. Compared with CPU, GPU has thousands of highly specialized cores that are adept at processing matrices. Thanks to Compute Unified Device Architecture (CUDA) platform and NVIDIA CUDA Deep Neural Network (cuDNN) library, researchers and data scientists have recently identified that GPUs can be turned into a powerful general-purpose computing engine to accelerate the training process through parallelism [205]. They offer higher memory bandwidth and dramatically speed up the training.

Moreover, Tensor Processing Units (TPU) are application-specific integrated circuits recently developed by Google and specifically act as machine learning accelerators. TPUs have demonstrated higher processing speeds compared to GPUs, but they are less flexible and limited to models in TensorFlow library. [206] provides a comprehensive evaluation of TPU and compares it with GPU and CPU in terms of performance and speed.

2.14.2 Platforms

The success of deep learning is highly reliant on the development of state-of-the-art tools including frameworks and libraries. Various frameworks play critical roles in generating new models with high scalability by offering different features in terms of pre-trained models, multi-GPU processing, and training/test speed. Table 2.14.1 presents the most well-known frameworks, showing the supported programming language and the strong point of each. The tools are ranked based on the popularity and ratings of the users in GitHub website [207], which is a collaborative code-hosting platform for developers. A comprehensive discussion and comparison of deep learning tools can be found in [208], [209].

2.14.3Could computing

Cloud computing is a general term for on-demand computing resources including power, storage, and services over the internet (known as cloud) with high scalability and reliability and, can be seen as the evolution of cluster and grid computing. In the PHM context, cloud resources can be basically used by the researchers to develop, train and deploy their deep models in real-time at any scale. Public cloud vendors are rapidly improving their capabilities by offering advanced analytics services in the pay-as-you-go basis that are practical for both newcomers and experienced data analysts, see Table 2.14.2.

In the bigger picture, cloud computing as a key enabler of “Big Data Analytics” and “Industrial Internet of Thing (IIoT)” technologies alongside with other technologies such as “advanced sensors,” “wireless communications,” “advanced manufacturing“, and “robotics” in cyber-physical-systems (CPS) concept plays a critical role in moving the manufacturing systems to an intelligent level which is known as “smart Manufacturing” toward “Industry 4.0 (fourth industrial revolution)” goal [210].

Integrating PHM in the smart manufacturing paradigm goes beyond the monitoring and data analysis for an individual component. It is challenging and requires a vast amount of computational resources to determine and manage the interactions between components, sub-systems, and systems. Cloud computing is transferring traditional manufacturing and condition monitoring frameworks into service-oriented models [211], [212]. Edge computing and fog computing are recent extensions of cloud computing that tackle high-latency, security, and bandwidth issues by processing the data in the layers of IIoT, which are closer to data resources [213].

Table 2.14.1 Mainstream deep learning tools, rankings are based on the stars and forks in GitHub [207]**. API: application programming interface.**

Ranking	Library	API	GitHub URL	Comment
1	TensorFlow	Python, C++, Java, Go, JavaScript	https://github.com/tensorflow/tensorflow	The most popular library with complete functionality and several interface support. In CPU computing, it has shown better scalability compared to other libraries [214].
2	Keras	Python, R	https://github.com/keras-team/keras	High-level API integrating with TensorFlow, CNTK, and Theano.
3	Caffe/Caffe 2	Python, Matlab, C++	https://github.com/BVLC/caffe https://github.com/facebookarchive/caffe2	Extended for use in Hadoop with Spark.
4	MXNet	Python, C++, Matlab, R, Julia, Scala, Perl	https://github.com/apache/incubator-mxnet	Difficult to learn, but highly scalable and memory-efficient.
5	Theano	Python	https://github.com/Theano/Theano	No longer supported after release 1.0.0 (November 2017). Supports tensor and sparse operations, python 2 and python 3, GPU computation, and SIMD parallelism on CPU.
6	CNTK	Python, C++, and C#	https://github.com/Microsoft/CNTK	The user can easily combine different models such as CNNs and RNNs and supports transfer between different platforms (Caffe2, MXNet, and PyTorch) [215].
7	Deeplearning4j (DL4J)	Java, Scala, Clojure or Koltin	https://github.com/eclipse/deeplearning4j	Accelerates training through built-in integration with Apache Hadoop and Spark. Using Keras API bridges the gap between JVM languages and Python, and it can import models from TensorFlow, Theano, CNTK, and Caffe [216].
8	PyTorch	Python	https://github.com/pytorch/pytorch	Pre-trained models are available. However, no visualization tool available.
9	Chainer	Python	https://github.com/chainer/chainer	-
10	Torch7	Lua	https://github.com/torch/torch7	Focused on GPU computation acceleration. Despite that, not beating CNTK, Caffe and MXNet for GPU accelerated implementation [217].

Table 2.14.2 Existing cloud environments.

Provider	ML and DL Services	URL
Microsoft Azure	ML Studio	https://azure.microsoft.com
Google Cloud Platform (GCP)	Cloud AutoML, Google ML engine	https://cloud.google.com/
Amazon Web Services (AWS)	Amazon ML, Deep Learning AMI	https://aws.amazon.com/
IBM Cloud	Watson ML Studio	https://www.ibm.com/cloud
Oracle Cloud	Oracle ML	https://www.oracle.com/

2.15 SUMMARY AND CONCLUDING REMARKS

In this chapter, a detailed systematic review has been carried out on various aspects of employing deep neural networks in the context of fault detection, diagnostics, and prognostics. From the above discussion, it is clear that DL algorithms have brought new perspectives to data-driven methods in terms of model performance, learning complex representations, big data analysis, and handling the raw data with minimum preprocessing efforts. However, despite promising results, DL for PHM application has a long way to go to replace well-established data-driven techniques in the industries. In addition to the barriers that are addressed in the literature and thoroughly reviewed through the survey, the authors have identified several significant challenges in exploiting the potentials of DL toward employing a reliable, scalable and applicable PHM model to realize industry 4.0 goals. To outline the future research directions, we conclude with the key related challenges and associated opportunities to the researchers.

1. Data scarcity: As a matter of fact, DL algorithms are known to be data-hungry, and their superior performance depends upon the availability of abundant data, which is rarely feasible in most of the situations. In recent years, several approaches have been suggested to tackle the limitations imposed by the small size of datasets in terms of model generalization and optimization. Data augmentation techniques have shown great success in enhancing the size of training datasets by generating synthetic data. Basic augmentation techniques such as window cropping, wrapping, and flipping have been widely utilized to generate new data sequences from the original time-series [218]. Also, advanced techniques such as generative algorithms can be used to generate new data that look close to real data. However, new generative models are required to create valid time-series data in time and frequency domain considering the temporal dependency of the data. Transfer learning is an active research direction that helps to deal with a small amount of data by transferring knowledge from one domain to another domain. It will mitigate the need for training the model from scratch by fine-tuning a pre-trained model on a new domain. Moreover, one-shot learning approaches as an effective solution, support learning from one annotated training sample either by defining a new loss function or creating an external memory which can encode and retrieve new data.

2. Industrial data characteristics: The success of deep models is highly reliant on the quality and variety of the collected data. The evolution of smart sensors and IIoT technologies has somehow eased the industrial data scarcity issue. Nevertheless, more data means more noise and uncertainty associated with the operating environment, various data sources, and data transmission that need to be addressed. Moreover, when it comes to big industrial data, the challenges regarding incomplete data, unlabeled data, imbalanced classes, and unseen classes get more critical and severe. As mentioned earlier, a few works put forth the effort to mitigate the issues using augmentation techniques through generative algorithms, specifically GANs, and have achieved interesting results. However, most of the methods consider a moderately imbalanced scenario and ignore the challenge of significantly under-represented class which is what exists in real industrial applications. Also, real-world data come from various sensors and are mostly non-structured, multi-modal and heterogeneous which make the model much more complex. Further research is required in the future to leverage heterogeneous information in deep models while not reducing the training efficiency.

3. Data analysis: Data preprocessing and visualization play critical roles in the performance of machine learning and deep learning algorithms. The quality of the model is highly sensitive to the quality of the training data. Preprocessing aspects range from simple normalization, standardization, and data segmentation to more complex processing tasks such as labeling, dealing with incomplete data, handling outliers, and missing values. Chen et al. proposed a deep transfer learning-based framework to tackle missing value issue by transferring a well-trained structurally complete fault diagnosis model to the missing data model [219]. In the absence of insufficient labeled data, synthetic data generation with generative deep networks such as VAEs and GANs provide a fast and cheap solution to produce new labeled data. Moreover, for deep learning-based fault diagnosis, the spatial distribution of the features reflects the quality of the disparity of the fault features and directly affects the classification accuracy. Hence, effective visualization techniques are necessary to analyze the quality of the features. Zemouri et al. proposed a 2D visualization model based on deep convolutional VAE for the classification task [220].

4. Model selection: Choosing an optimal network architecture is an important issue. Most of the reviewed papers have not justified using certain architecture to solve specific problems. Up to now, the majority of employed networks have been designed manually by human experts which is error-prone and time-consuming. Although DL promises considerable benefits in terms of automated solutions, still it remarkably depends on choosing a wide range of hyperparameters. There is a paucity of literature that used evolutionary algorithms to optimize the hyperparameter setting. However, the authors have not found any literature in PHM regarding neural architecture search (NAS) which is a recent trend in machine learning and has shown significant success in automating network design for image classification and semantic segmentation tasks. It is important to investigate the possibilities of developing more automated models in terms of hyperparameter and architecture selection for large-scale real industrial data.

5. *Black-box tool:* Despite the accurate prediction promising results, many of the companies still are unwilling to adopt DL. The reason lies behind the black-box nature of DL algorithms that imposes a lack of "transparency" and "interpretability" to the model and especially the decision-making part of the PHM cycle. There is no inherent understanding of the underlying process and the reason for making certain decisions. In other words, companies cannot trust something that they don't understand and don't have control over it. In recent years, several efforts have been made to tackle the aforementioned issue. Explainable deep learning is a new paradigm to open the black-box and increase the transparency of the model. These techniques are roughly categorized into two types: The former approaches utilize a relatively simple model to interpret complex deep learning models. The latter methods, build intrinsic interpretable deep architecture by incorporating attention mechanisms in intermediate layers [221].

6. Cross-domain prediction: The majority of the reviewed papers train the model with public data that are collected under laboratory conditions. Even proposed domain adaptation techniques, mainly focused on transferring the knowledge from one working condition to another working condition in laboratory collected data. Transfer learning to tackle distribution mismatch among various domains including real industrial equipment and artificial laboratory faults is still ongoing research area within deep learning paradigm and further focus on domain adaptation with multiple source domains is required to reach superior domain generalization ability which creates feasible models in practice.

7. Real-time realization: While advanced hardware, DL architectures and computing paradigms (Cloud, edge, and fog) have revolutionized large-scale learning in recent years, emerging computing challenges have risen for real-time training and deployment of DL algorithms. There are few relevant works in the realm of PHM which have exploited cloud computing capabilities for faster offline training while speeding up the inference (i.e. deployment) is of more concern of an applicable PHM model. Real industrial data come in continuous streams and their distribution characteristics are in dynamic change over time which limits accurate and real-time inference of the data. Thus, the model needs to cope with the concept drift of continuously evolving new data within the incremental learning settings. However, typical DL algorithms greatly suffer from forgetting, which refers to the complete loss of previously learned knowledge in favor of learning the new information during the sequential training process. New algorithms and hardware architectures are needed to facilitate continuous learning of non-stationary sequence data while retaining the general-domain knowledge of the pre-trained model.

8. The role of benchmarking: At the moment, many DL architectures, algorithms, platforms and frameworks are being used to solve specific PHM problems which previously deemed unsolvable. The variety of available algorithms, models, software, and hardware systems raises the need for benchmarking infrastructure that enables a fair comparison of workloads with respect to the time and cost of both training and inference. Currently, some authors have carried out a comparative analysis of different techniques. However, they have compared their deep models with classical machine learning algorithms and have focused solely on generic performance metrics such as accuracy and classification (See Table 2.13.1). There is a need to build novel metrics that incorporate runtime performance, model accuracy, and robustness across various architectures and DL frameworks.

Following the existing challenges mentioned above, the authors address "the benchmarking" and "cross-domain prediction" in subsequent sections.

CHAPTER 3: EMPIRICAL STUDY OF DEEP LEARNING ALGORITHMS ON BEARING FAULT DIAGNOSIS BENCHMARKS

At the moment, many DL architectures, algorithms, platforms, and frameworks are being used to solve specific PHM problems which previously deemed unsolvable. The variety of available algorithms, models, software, and hardware systems raises the need for benchmarking infrastructure that enables a fair comparison of the existing networks with respect to the performance and workloads. This chapter evaluates various DL-based fault diagnosis algorithms for two benchmark bearing datasets and provides the benchmark accuracy to make future studies more comparable. Based on the empirical study, we highlight some evaluation results which are very important for comparing or testing new models.

3.1 DATASETS

In the field of intelligent fault diagnosis, most of the researchers evaluate their model on their dataset, and the publicly available datasets, have not been investigated in depth. For comprehensive performance comparisons, it is important to utilize different kinds of representative datasets. We collected two publicly available datasets for bearing fault diagnosis tasks and give a detailed discussion about their adaptability. The description of all these datasets is listed as follows.

3.1.1 Dataset1: Case Western Reserve University (CWRU) bearing dataset

Two accelerometers capture the vibration response of the motor/bearing system for both drive end (DE) bearing and fan end (FE) bearing, see **Figure 3.1.1** . Rolling element bearing has four major parts: inner race, outer race, rolling element, and cage. The fault might happen at any of these parts. CWRU data set contains bearing data for ball fault, inner race fault, and outer race fault. Single point faults were introduced to these parts using electro-discharge machining with fault diameters of 7 mils, 14 mils, 21 mils (1 mil=0.001 inch). The bearing data with no faults (normal) is also available. So, we have **ten fault classes** in total. Data is collected at 12k sampling rate for both bearings and 48k sampling rate for DE bearing faults. Also, data is recorded for motor loads of 0, 1, 2, and 3 horsepower (hp) (See 2.4.2Table 3.1.1). 3.1.2**Figure 3.1.2** shows raw vibration signals for all then classes for one working loads: 1 h. Also, Table 3.1.2 lists dataset sizes for different classes and load conditions for drive-end bearings.

Table 3.1.1 Description of CWRU rolling-element bearing dataset classes*.

	Fault Location										**Motor load (hp)**
	Ball			Inner Race			Outer Race			Normal	
Fault diameter (mils)	7	14	21	7	14	21	7	14	21	0	0,1,2,3
Fault label	1	2	3	4	5	6	7	8	9	10	

*** Dataset size varies from 250k~480k for each fault class.**

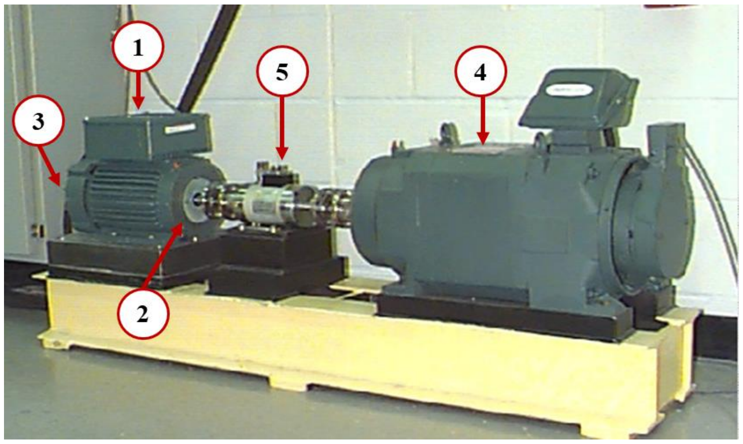


Figure 3.1.1 CWRU bearing test set-up [37]: 1- Electric motor, 2- Drive-end bearing, 3- Fan-end bearing, 4- Dynamometer, 5- Torque transducer and encoder.

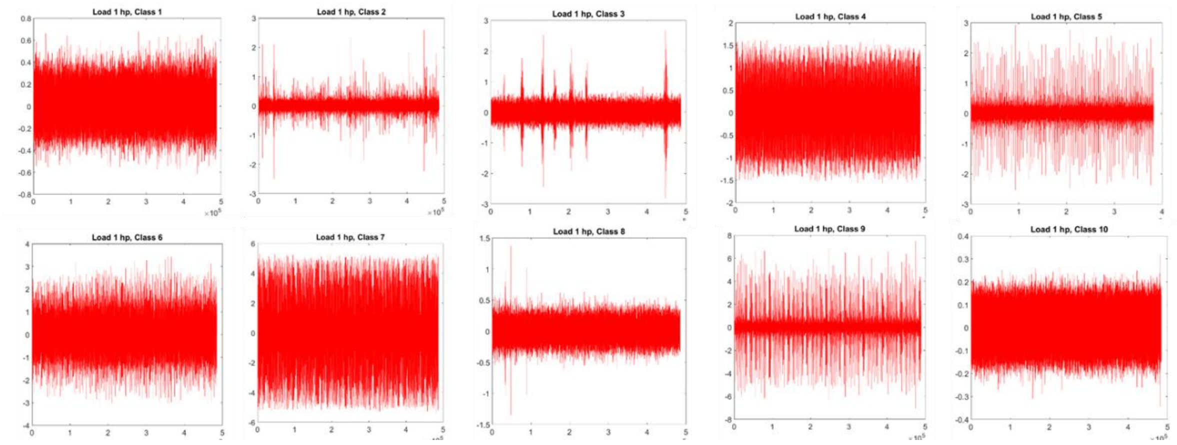


Figure 3.1.2 CWRU dataset: raw vibration signal for different classes under motor loads 1 hp.

Table 3.1.2 CWRU: Datasets sizes for drive-end bearing.

Working Load (hp)	**Data size**									
	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10
0 (12K)	122571	121846	121991	121265	121846	122136	121991	121846	122426	243938
0 (48K)	244739	249146	243938	243938	63788	244339	243538	245140	246342	
1 (12K)	121410	122136	121701	121991	121846	121556	122426	122136	121991	483903
1 (48K)	487384	486224	486804	486224	381890	485063	486804	484483	489125	
2 (12K)	121556	121991	122136	122136	121846	121846	121410	121846	122281	485063
2 (48K)	486804	487384	487384	485643	487964	491446	486804	486804	487964	
3 (12K)	121556	122136	122136	122917	121701	121991	122571	121991	121991	485643
3 (48K)	488545	486804	486804	485643	485063	489125	487364	488545	489125	

3.1.2 Dataset 2: Paderborn University KAT bearing dataset

KAT bearing dataset is another well-known public dataset for bearing fault diagnosis [36]. The dataset is provided by KAT datacenter at Paderborn University [36]. It includes measured motor currents and vibration signals. Experiments were performed on 26 damaged bearing states and 6 undamaged states. Figure 3.1.3 shows the experimental test rig. Among the damaged bearings, 14 were naturally damaged by accelerated life tests, and 12 were artificially damaged by EMD or drilling. Also, the data is recorded for four different operating conditions in terms of radial force, load torque, and rotating speed (See

2.4.2Table 3.1.3 For artificial damages, recordings for two fault severity levels in both outer race and inner race are available. Hence we have **five health condition classes**: 1- healthy, 2- IR1, 3- IR2, 4- OR1, and 5- OR2 (see

Table 3.1.4). In case of real damages, recordings for three pitting severity levels in the inner race, one plastic deform severity level in the inner race, two pitting severity levels in the outer race, and one plastic deform severity level in the outer race are available. Hence we have **eight health condition classes**: 1- Healthy, 2- IR1, 3- IR2, 4- IR3, 5- IR4, 6- OR1, 7- OR2, and 8- OR3 (see Table 3.1.5). Available datasets are not balanced between classes. Hence, we don't use all given datasets for all the categories.

Table 3.1.3 Operating conditions for KAT dataset

Working Condition	Rotating speed (rpm)	Load torque (Nm)	Radial force (N)
1	1500	0.7	1000
2	900	0.7	1000
3	1500	0.1	1000
4	1500	0.7	400

Table 3.1.4 KAT dataset: Bearings with artificial damages, H: healthy, OR: outer ring, IR: inner ring, for each bearing 20 time-series of 256k points are available.

Bearing	Fault location	Fault level	Fault label
K001	-	-	H
K002	-	-	H
K003	-	-	H
K004	-	-	H
K005	-	-	H
K006	-	-	H
KA01	Outer ring	1	OR
KA03	Outer ring	2	OR
KA05	Outer ring	1	OR
KA06	Outer ring	2	OR
KA07	Outer ring	1	OR
KA08	Outer ring	2	OR
KA09	Outer ring	2	OR
KI01	Inner ring	1	IR
KI03	Inner ring	1	IR
KI05	Inner ring	1	IR
KI07	Inner ring	2	IR
KI08	Inner ring	2	IR

Table 3.1.5 KAT dataset: Bearings with real damages, OR: outer ring, IR: inner ring, for each bearing 20 time-series of 256k points are available.

Bearing	Fault type	Fault location	Fault level	Fault label
KA04	Pitting	Outer ring	1	OR
KA15	Plastic deform	Outer ring	1	OR
KA16	Pitting	Outer ring	2	OR
KA22	Pitting	Outer ring	1	OR
KA30	Plastic deform	Outer ring	1	OR
KB23	Pitting	Inner ring	2	IR
KB24	Pitting	Inner ring	3	IR
KB27	Plastic deform	Inner ring	1	IR
KI04	Pitting	Inner ring	1	IR
KI14	Pitting	Inner ring	1	IR
KI16	Pitting	Inner ring	3	IR
KI17	Pitting	Inner ring	1	IR
KI18	Pitting	Inner ring	2	IR
KI21	Pitting	Inner ring	1	IR

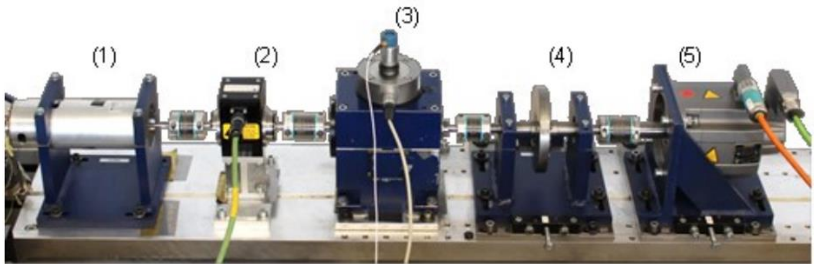


Figure 3.1.3 Modular KAT test rig: (1) motor, (2) torque measurement shaft, (3) bearing set module, (4) flywheel, (5) load motor [36].

3.2 DATA PREPARATION

The type of input data and the way of preparing the data have a great impact on the performance of DL models. In the current work, we evaluate the models with raw input data. We first use a sampling window of size 2048 with an overlap of 124 points to segment the data series. Then, [-1,1] normalization method is adopted to normalize the segmented samples.

3.3 DEEP NEURAL NETWORKS

In this empirical study, we investigate four deep neural network models for bearing fault diagnosis: MLP, Autoenocer, CNN, and RNN. Models are discussed below. **Figure 3.3.1** represents the detailed architecture of the applied models:

- *MLP:* In the multi-layer perceptron (MLP) network, as the simplest form of deep architecture, the output is computed straight forward along with the sequent layers of the model as long as input data is fed. In each neuron of the middle-hidden layers, the biased weighted sum of the previous layer outputs is put into an activation function to produce the output of that neuron. In the current study, we adopted an MLP network with five hidden layers accompanying 50% dropout rate and batch normalization for each layer.

- *Autoencoder:* Autoencoders are unsupervised networks that are trained to reconstruct the input $\mathbf{x}$ on the output layer $\hat{\mathbf{x}}$ in a two-phase process: *encoding* learns a hidden representation of data $\mathbf{h}$ via a feature-extracting function, and *decoding* maps $\mathbf{h}$ back into the input space to obtain a reconstruction of data. Autoencoders can stack in a deep configuration called stacked autoencoder (SAE), which exploits the inner structure of the data via including a sparsity constraint on the activation of the hidden units through the addition of Kullback-Leibler (KL) divergence term to the cost function. The denoising autoencoder (DAE) is another regularized network to prevent the model from learning a trivial identity solution. Instead of adding the penalty to the cost function, DAE takes a noise-corrupted version of data $\tilde{\mathbf{x}}$ to reconstruct the input $\mathbf{x}$ and learn meaningful information by changing the reconstruction error term in the cost function. The input is first corrupted by employing Binary or Gaussian noise and then fed to the hidden layer. *Herein*, we adopted autoencoder networks with nine hidden layers accompanying batch normalization for each layer. We use same network structures for DAE, SAE, and AE, and their differences lie in the loss function and inputs. DAE takes the corrupted input (with Gaussian noise), and learns to reconstruct the clean input. We utilized the loss functions introduced in 2.4.4.

- *CNN:* Convolutional neural networks (CNNs) are deep discriminative networks and have shown good results in processing data with grid-like topology. The key difference between CNNs and standard neural networks is that CNNs benefit from parameter sharing, which allows the network to look for specific features at different positions [62]. The convolutional layer carries out the convolution operation on the input data by sliding a filter (kernel) over the input to produce a feature map. The pooling layer aims to reduce the dimension of the feature map, which reduces the number of the parameters and increases the shift-invariance property. In the current study, we use three 1D convolutional layers, accompanying batch normalization layers, followed by one max-pooling layer, one dropout layer (50%), four fully-connected layers, and softmax classification layer.

- *RNN:* Recurrent Neural Networks (RNNs) contain feedback loops to remember the information of former units and are the most suitable for sequential data such as natural language and time-series data. During the training process, the hidden unit h_t is sequentially updated based on the activation of the current input x_t at time t and previous hidden state h_{t-1}. RNNs are capable of capturing long-term temporal dependencies from time series and sequential data. Gated recurrent unit (GRU) and long short-term memory (LSTM) cells are popular variants of RNN that try to alleviate the aforementioned problem [62], see Figure 2.13.4. Bi-directional recurrent networks (BRNN) increase the model capacity by sequencing the data in both forward and backward directions. Herein, we adopted one layer bi-directional LSTM, followed by two fully-connected layers and softmax classification layer. For GRU network implementation, we have used the same network architecture.

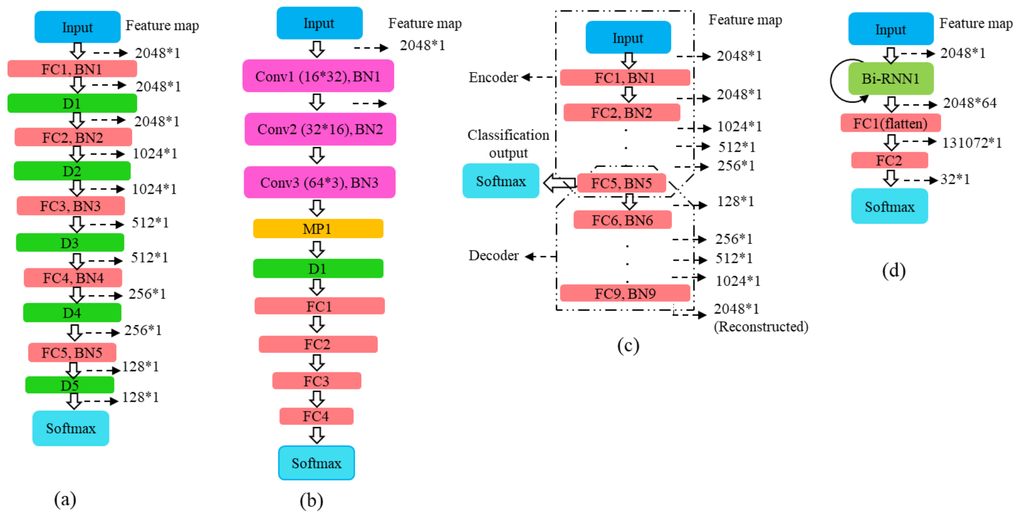


Figure 3.3.1 Various DL architectures for empirical study: (a) MLP, (b) CNN, (c) AE, (d) RNN.

3.4 METRIC

It is a rather challenging task to evaluate the performance of intelligent diagnosis algorithms with suitable evaluation metrics. In this work, we use the overall **accuracy** to evaluate the performance of algorithms. The overall accuracy is defined as the number of correctly classified samples divided by the total number of samples. The average accuracy is defined as the average classification accuracy of each category. Since the performance of DL-based intelligent diagnosis algorithms fluctuates during the training process, to obtain reliable results and show the best overall accuracy that the model can achieve, we repeat each experiment five times. We also report the **training time** of the models to evaluate the workload of certain networks in terms of computation time.

3.5 HYPERPARAMETERS

The success of deep neural networks requires choosing optimal hyperparameters, network architectures, and regularization techniques. Currently, there is no easy way of finding the optimal settings- specifically, learning rate, momentum, weight decay, and batch size. The conventional methods perform a grid search or random search to find the optimal configuration of the parameters by exploring the hyperparameter space [222]. Both these approaches are computationally expensive and time-consuming. Smith [223] proposed the cyclical learning rate (CLR) and cyclical momentum (CM) which are based on the balance between underfitting versus overfitting. Underfitting is when the machine learning model is unable to reduce the error for either the test or training set. Overfitting happens when the machine learning model fits the training data to the extent that it negatively impacts the model generalization on the test data. Small learning rates can lead to overfitting. On the other hand, large learning rates help the regularization of the training, but it may lead to training divergence.

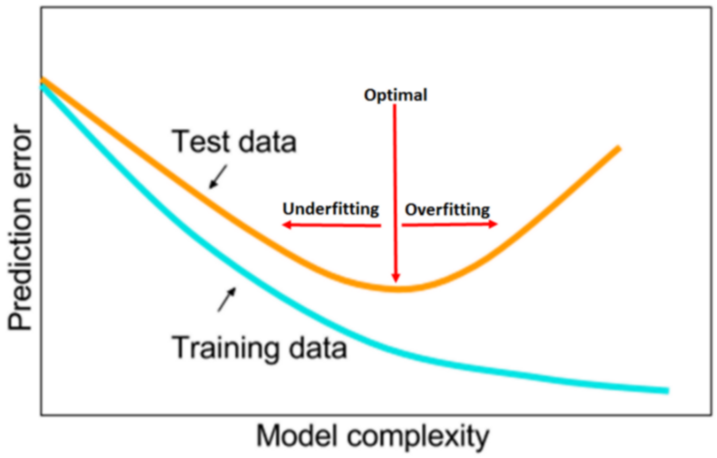


Figure 3.5.1 The tradeoff between underfitting and overfitting. Model complexity refers to the capacity or powerfulness of the machine learning model [224].

Smith’s cyclical method examines the training’s test/validation loss for clues of underfitting and overfitting to strive for the optimal set of hyperparameters. Specifically, it is based on changing the learning rate within a range of values rather than using a step-wise decreasing value. That is, one specifies the minimum and maximum learning rate boundaries, and the learning rate cyclically varies between this range, see

Figure 3.5.2. LR range test is proposed to estimate the minimum and maximum bounds. According to the LR range test, one may run the model for several epochs while letting the learning rate increase linearly between low and high LR values. Next, plot the accuracy vs. learning rate, and note the learning rate value when the accuracy starts to increase and when the accuracy starts to fall. These two values make good choices for minimum and maximum bounds.

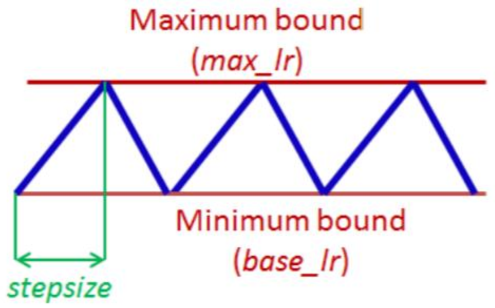


Figure 3.5.2 Triangular learning rate policy. The blue lines represent learning rate values changing between bounds. The input parameter step size is the number of iterations in half a cycle [223].

The learning rate and momentum are closely related. Momentum is designed to accelerate network training but its effect on updating the weights is of the same magnitude as the learning rate. Hence, the optimal training process requires an optimal combination of learning rate and momentum. According to smith's experiments, the optimal training process includes a combination of an increasing cyclical learning rate and decreasing cyclical momentum. An initial small learning rate permits convergence to begin, and a decreasing cyclical momentum allows the learning rate to become larger in the early to middle parts of training [224].

In the current study, we adopt two hyperparameter selection strategies: *1- constant hyperparameters, 2- cyclical learning rate*. In the first setting, we set the learning rate, momentum, weight decay, and the batch size to constant values of 10^{-3}, 0.9, 10^{-4} and 128 respectively. In the latter, we adopt an increasing cyclical learning rate along with constant momentum of 0.9, constant weight decay and of 10^{-4} and the constant batch size of 128. We perform an LR range test to find the maximum learning rate, and then, set the minimum bound as a tenth of the maximum. We execute short runs of momentum values to find the best momentum and start the decreasing cyclical momentum at this maximum value. For both settings, we run the models for 100 epochs.

3.6 EXPERIMENTS

We define four sets of experiments based on the input types and hyperparameter settings. As mentioned earlier, we report accuracy and training time for each experiment to compare the effect of hyperparameter tunning on model performance and training cost:

Experiment 1: Raw normalized inputs with fixed hyperparameters.

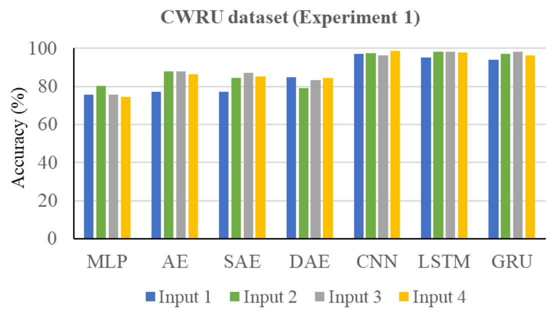
Experiment 2: Raw normalized inputs with cyclical learning rate.

For cyclical learning rate, we use triangular learning rate adjustment, in which the cycle amplitude is decreased by half after each period (batch iteration), see Figure 3.7.3. All the experiments are done using Pytorch 1.5 and were running on NVIDIA GTX 2060 GPU. The reported experimental results are averaged by 5 trials to reduce the effect of randomness, and the results are given in the form of $\mu \pm sd$, where μ and sd stand for mean value and standard deviation of 5 trials.

3.7 RESULTS AND DISCUSSION

The results are given in Figure 3.7.1-3.7.4, and Table 3.7.1-3.7.8. From the results, we can observe that MLP gives the lowest accuracy for both datasets. In almost all the experiments, CNN, and LSTM achieve higher accuracy which is comparable to each other. However, LSTM and GRU are pretty slow, and take so much time to learn. For instance, in experiment 1 of CWRU input 4, the training time of LSTM is nearly 281 times higher than CNN. Among three autoencoder variants, DAE seems to give higher accuracy compared to other two variants for both datasets. The training time of DAE is slightly higher than AE and SAE. The observations of RNN variants are nearly close to each other. However, GRU networks take a little bit of lower training time compared to LSTMs.

For all the inputs, we carried out a separate LR range test to find the minimum and maximum learning rate boundaries. We can observe that nearly in most of the cases, the cyclical learning rate gives a better accuracy compared to the fixed learning rate. Besides, as it is shown in Figure 3.7.3, in most cases CLR provides quicker convergence compared to the fixed learning rate. One reason might be because increasing the learning rate is an effective way of escaping saddle points. By cycling the learning rate, we're guaranteeing that such an increase will take place if we end up in a saddle point [224].



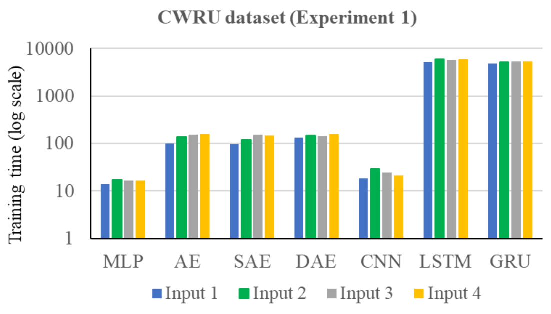


Figure 3.7.1 Experiment 1: test accuracy for CWRU (left), training time for CWRU dataset (right).

Table 3.7.1 Experiment 1: comparison of test accuracy for CWRU dataset.

			AE				RNN	
CWRU Input	# Classes	MLP	AE	SAE	DAE	CNN	LSTM	GRU
1	10	75.6±0.8	77.1±0.5	77.3±0.3	84.7±0.1	**97.1**±0.4	95.1±0.2	94.1±0.1
2	10	80.1±0.9	88.0±0.2	84.3±0.2	79.0±0.5	97.5±0.8	**98.2**±0.5	97.2±0.5
3	10	75.6±1.0	87.9±0.1	87.3±0.1	83.3±0.6	96.1±0.5	**98.3**±0.1	**98.3**±0.2
4	10	74.6±1.1	86.5±0.4	85.2±0.1	84.3±0.2	**98.6**±0.8	97.9±0.1	96.3±0.1

Table 3.7.2 Experiment 1: comparison of training time for CWRU dataset.

			AE				RNN	
CWRU Input	# Classes	MLP *(sec)*	AE *(Sec)*	SAE *(Sec)*	DAE *(sec)*	CNN *(sec)*	LSTM *(sec)*	GRU *(sec)*
1	10	**14.0**	101.1	97.5	130.8	18.2	5240.0	4904.1
2	10	**17.0**	135.6	120.4	148.7	29.1	5941.2	5256.5
3	10	**16.7**	152.1	152.9	145.0	24.5	5831.3	5380.4
4	10	**16.9**	155.9	149.3	157.3	21.0	5905.0	5401.0

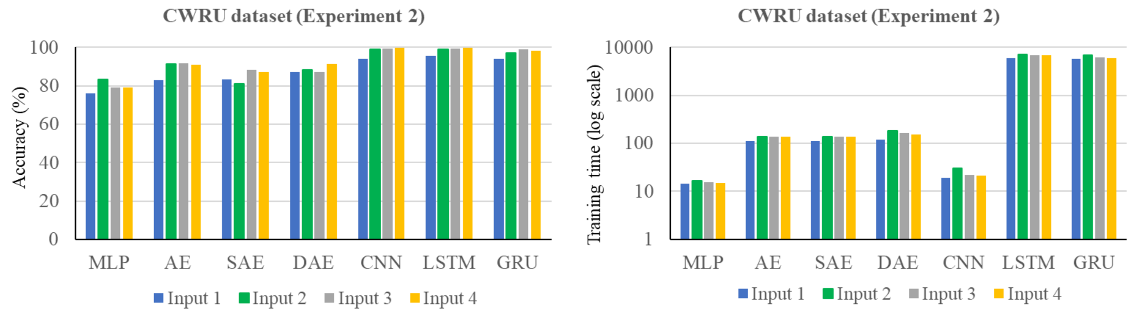


Figure 3.7.2 Experiment 2: test accuracy for CWRU (left), training time for CWRU dataset (right).

Table 3.7.3 Experiment 2: comparison of test accuracy for CWRU dataset.

			AE				RNN	
CWRU Input	# Classes	MLP	AE	SAE	DAE	CNN	LSTM	GRU
1	10	75.9±0.4	82.9±0.4	83.2±0.5	87.3±0.2	94.1±0.3	**95.6**±0.4	94.0±0.2
2	10	83.1±0.7	91.3±0.2	81.0±0.3	88.3±0.1	**99.1**±0.2	97.3±0.2	97.2±0.3
3	10	79.1±0.2	91.8±0.3	88.1±0.2	87.1±0.3	99.2±0.2	**99.3**±0.3	98.9±0.1
4	10	79.1±0.4	91.1±0.1	87.1±0.2	91.2±0.4	**99.8**±0.1	98.9±0.2	98.3±0.3

Table 3.7.4 Experiment 2: comparison of training time for CWRU dataset.

			AE				RNN	
CWRU Input	# Classes	MLP *(sec)*	AE *(Sec)*	SAE *(Sec)*	DAE *(sec)*	CNN *(sec)*	LSTM *(sec)*	GRU *(sec)*
1	10	**14.2**	111.0	111.6	120.0	19.1	6054.1	5840.2
2	10	**16.6**	135.8	120.5	180.5	30.3	7021.0	6845.6
3	10	**15.5**	137.5	130.1	162.3	22.0	6940.5	6154.0
4	10	**15.1**	136.9	125.3	151.1	21.2	6854.0	6021.2

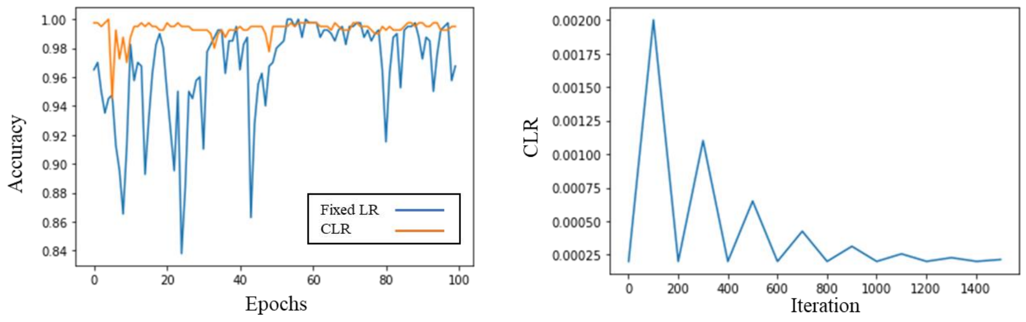


Figure 3.7.3 CWRU, Input 3: test accuracy for fixed learning rate vs. cyclical learning rate (left), triangular learning rate adjustment, cycle amplitude is decreased by half after each period (right).

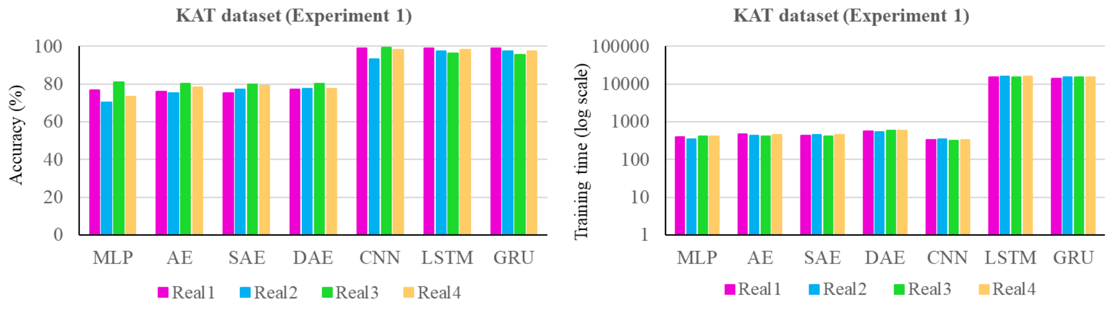


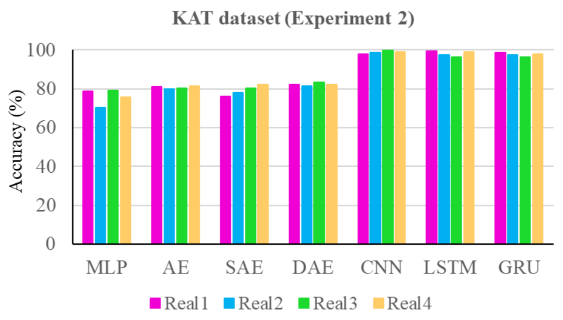
Figure 3.7.4 Experiment 1: test accuracy for KAT (left), training time for KAT dataset (right).

Table 3.7.5 Experiment 1: comparison of test accuracy for KAT dataset.

			AE				RNN	
KAT Input	# Classes	MLP	AE	SAE	DAE	CNN	LSTM	GRU
Real-1	8	76.7±0.1	76.2±0.2	75.3±0.4	77.1±0.1	99.0±0.1	98.9±0.3	**99.1**±0.2
Real-2	8	70.1±0.1	75.1±0.2	77.3±0.3	77.4±0.0	93.1±0.2	**97.4**±0.4	**97.4**±0.1
Real-3	8	81.1±0.2	80.3±0.3	79.8±0.3	80.1±0.1	**99.5**±0.2	96.2±0.3	95.5±0.3
Real-4	8	73.5±0.3	78.3±0.1	79.1±0.2	77.5±0.0	**98.2**±0.1	98.1±0.1	97.5±0.4

Table 3.7.6 Experiment 1: comparison of training time for KAT dataset.

			AE				RNN	
KAT Input	# Classes	MLP *(sec)*	AE *(Sec)*	SAE *(Sec)*	DAE *(sec)*	CNN *(sec)*	LSTM *(sec)*	GRU *(sec)*
Real-1	8	396.0	477.0	425.1	550.1	**328.8**	15,103.5	14,148.2
Real-2	8	**339.1**	435.4	450.2	538.5	348.9	16,051.1	14,984.9
Real-3	8	414.3	419.0	419.2	580.2	**320.5**	15,503.2	15,054.2
Real-4	8	412.8	450.1	450.4	585.9	**324.9**	16,131.0	15,121.5



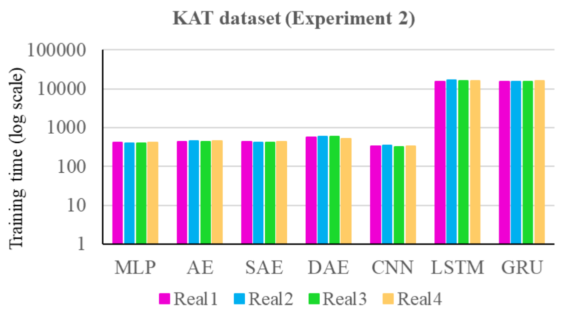


Figure 3.7.5 Experiment 2: test accuracy for KAT (left), training time for KAT dataset (right).

Table 3.7.7 Experiment 2: comparison of test accuracy for KAT dataset.

			AE				**RNN**	
KAT Input	# Classes	MLP	AE	SAE	DAE	CNN	LSTM	GRU
Real-1	8	78.6±0.2	81.1±0.2	76.2±0.1	82.3±0.1	98.0±0.3	**99.5**±0.1	98.5±0.3
Real-2	8	70.3±0.5	79.8±0.2	78.0±0.3	81.5±0.1	**98.6**±0.2	97.6±0.2	97.3±0.3
Real-3	8	79.0±0.2	80.1±0.1	80.3±0.2	83.3±0.0	**99.6**±0.2	96.1±0.2	96.3±0.2
Real-4	8	75.6±0.1	81.5±0.3	82.3±0.3	82.1±0.2	98.8±0.1	**98.9**±0.1	97.9±0.2

Table 3.7.8 Experiment 2: comparison of training time for KAT dataset.

			AE				**RNN**	
KAT Input	# Classes	MLP *(sec)*	AE *(Sec)*	SAE *(Sec)*	DAE *(sec)*	CNN *(sec)*	LSTM *(sec)*	GRU *(sec)*
Real-1	8	405.1	430.0	430.2	550.3	**326.9**	15,472.9	15,012.0
Real-2	8	401.8	447.5	420.2	580.7	**339.9**	16,413.0	15,455.9
Real-3	8	395.2	430.9	417.1	600.2	**321.1**	15,914.7	15,233.2
Real-4	8	415.3	451.2	434.3	510.1	**327.5**	16,257.6	15,656.9

3.8 SUMMARY

An empirical analysis of two bearing benchmark datasets was performed to evaluate the performance of various standard deep neural networks for the bearing fault diagnosis task. The models are trained with both fixed and cycling learning rates to evaluate the effect of hyperparameter tuning on the accuracy and training time. The results show that cyclical learning rate provides quicker convergence compared to the fixed learning rate, resulting in comparable accuracy within less training epochs. According to the results, CNNs provide the best accuracy in the least training time. The RNNs achieve a performance comparable to the CNNs. However, they are slow and time-consuming to be trained. Among three autoencoder variants, DAE seems to give higher accuracy compared to the other two variants for both datasets. The training time of DAE is slightly higher than AE and SAE. The observations of RNN variants are nearly close to each other. However, GRU networks take a little bit of lower training time compared to LSTMs.

CHAPTER 4: A NOVEL CROSS-DOMAIN FRAMEWORK FOR FAULT DIAGNOSIS

Compared to conventional data-driven approaches, deep learning techniques remove the burden of manual feature engineering and achieve state-of-the-art results. Despite the marvelous performance, the success of existing deep learning algorithms is highly reliant on the availability of large-scale labeled datasets. However, data annotation is time-consuming, error-prone, and prohibitive for most real-world industrial datasets. Also, the majority of the studies are based upon the assumption that the training and test data are drawn from the same distributions. However, in the real industry, the data are collected under different operating and environmental conditions and during different time intervals, often resulting in feature space difference or distribution shift across training and testing datasets (Domain shift issue).

Recently, researchers have tried to address the aforementioned problem by training a domain-invariant fault diagnosis model. Combining deep learning (DL) algorithms with Domain adaptation (DA) as a transfer learning (TL) method achieved promising results in different fields [225], [226]. A few studies have been conducted in intelligent fault diagnosis of rotary machinery based on domain-invariant feature learning. Some unsupervised domain adaptation (UDA) methods incorporated discrepancy measure loss into the neural network to diminish the shift between the two domains. For example, Lu et al. [227] utilized Maximum Mean Discrepancy (MMD) loss to align conditional and marginal distributions in multiple feature extraction layers. Lu et al. [228] adopted a weight regularization term and MMD metric to learn the shared-subspace while preserving the discriminative information of the original data for semi-supervised fault diagnosis of rotating machinery components only with the normal class of target domain during the training. Wang et al. [229] incorporated the Correlation Alignment (CORAL) distance loss of both marginal and conditional distributions into denoising autoencoder objective function to learn domain-invariant and discriminative representations.

Other models benefit adversarial learning schemes to learn transferable features by training a couple of the networks in opposite direction: Domain discriminator minimized the loss to improve domain discriminative ability of the network. On the other hand, the feature extractor maximizes the loss to fool the discriminator and learn transferable mappings indistinguishable by the discriminator [174], [191], [230]–[232]. A few studies utilized some mentioned approaches simultaneously to enhance model performance [11]–[15].

Despite the outstanding performance, the previous studies suffer from two major drawbacks as follows: Firstly, these studies are based upon the assumption that there is only one source domain, while in real industrial applications labeled data comes from different sources with various underlying distributions. Secondly, features are transferred across varying working conditions of bearings in the same machinery in the previous studies, which may affect the generalization ability of the model over an identical bearing of different machines.

To address the issues above, we proposed a novel multi-source domain adaptation framework for fault diagnosis of rotary machinery, which aligns the domains in both feature-level and task-level. The proposed model can be easily reduced to a single-source domain adaptation problem. Also, the model can be readily updated to unsupervised domain adaptation problems in other fields such as image classification and image segmentation. The main contributions of this chapter are summarized below:

- A novel multi-source domain adaptation approach is proposed to diminish the domain shift between unlabeled target domain and various labeled source domains. The proposed model uses task-specific decision boundaries based on sliced Wasserstein discrepancy metric and optimal transport theory (OT).

- The global feature extractor generates the low-level fault features and local feature extractors minimize MMD loss to align source samples and target samples.

- The proposed framework utilizes raw data in an end-to-end fashion. Hence, it does not require manual feature engineering and signal processing effort.

- We extensively evaluate our method on benchmark bearing datasets and show its superiority by comparing it with existing approaches.

4.1 RELATED WORKS

The domain divergence can be caused by the distribution shift or feature space difference. The first setting is referred to as homogenous DA, while the latter case denotes the heterogeneous DA. Also, considering labeled, partially labeled, or no-labeled target dataset available in the training stage, the settings can be categorized into supervised, semi-supervised, or unsupervised, respectively, Figure 3.8.1 summarizes the major DA settings and approaches in the machinery health monitoring field.

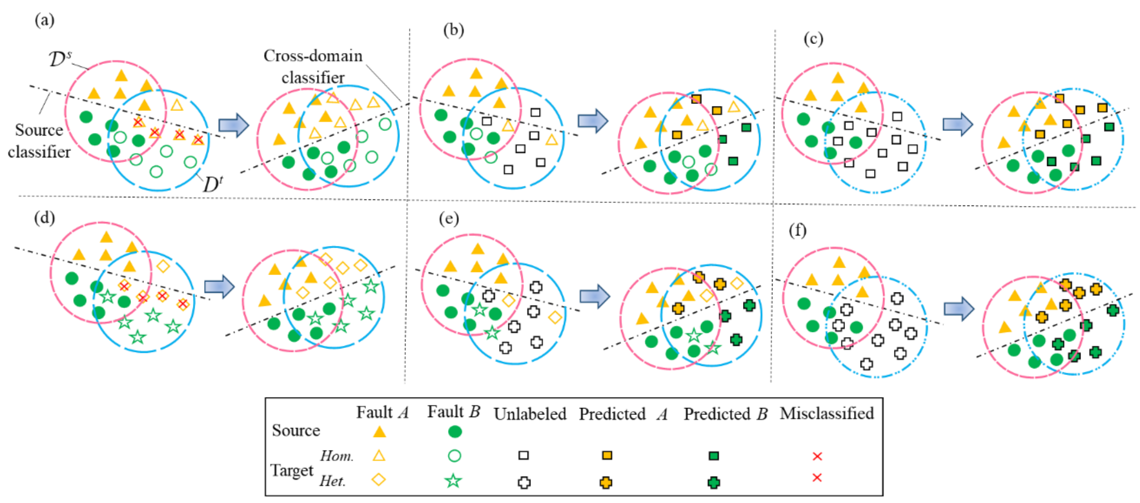


Figure 3.8.1 Different Domain adaptation scenarios: (a) homogenously supervised, (b) homogenous semi-supervised, (c) homogenous unsupervised, (d) heterogeneous supervised, (e) heterogeneous semi-supervised, and (f) heterogeneous unsupervised, *Hom:* homogenous, *Het:* heterogeneous.

4.1.1 Unsupervised Single-Source Domain Adaptation (UDA)

Single-source unsupervised domain adaptation focuses on reducing high generalization error by establishing knowledge transfer from one labeled source domain (training dataset) to an unlabeled target domain (Testing dataset). UDA methods can be classified into four categories of discrepancy-based, adversarial, reconstruction-based, and hybrid methods [238]:

- *Discrepancy-based methods* align the features by explicitly defining and reducing a distribution discrepancy measure on the corresponding latent layers. Some authors adopted discrepancy-based methods to enhance model performance through fine-tuning with labeled or unlabeled target data. A few studies carried-out the fine-tuning by adjusting the architecture of the network through adaptive batch normalization (AdaBN) [239] and reweighting the weak learner [240] techniques. However, most of the discrepancy-based methods utilize pre-defined distance metrics such as maximum mean discrepancy (MMD) [190], [228], [241], [242] KL divergence [243], and correlation alignment (CORAL) [244] to learn a domain-invariant representation by reducing the shift between two domains.

- *Adversarial learning* inspired by GAN models achieved great success. The core idea is to ensure that the classifier is fooled with synthetic labeled target data or cannot differentiate between the source and target domains through a generative or non-generative adversarial process. In generative adversarial processes, generators render synthetic target data a, which are paired with synthetic source data to share labels or appear as if they were sampled from the target domain while maintaining paired labels, or something else. Then, synthetic data with labels are used to train the target model as if no DA were required. In non-generative models, a domain discriminator that classifies whether a data point is drawn from the source or target domain is used to encourage domain confusion through an adversarial objective to minimize the distance between the empirical source and target mapping distributions [238]. In the PHM paradigm, Domain adversarial neural network (DANN) [174], Wasserstein distance-based adversarial networks [231], [245] have been utilized for bearing fault diagnosis.

- *Reconstruction-based methods* use encoder-decoder or GAN architectures to create a shared representation between the domains while preserving the discriminative information of each domain. Li et al. [246] well-trained and tested the source data on the hierarchy of SAEs in an unsupervised fashion for intelligent fault diagnosis. The nonnegative-constraint term was employed to the reconstruction error of layer-wise unsupervised training and the Softmax classifier cost function to enhance the sparsity of the model. Transferring the parameters to a similar model followed by the fine-tuning process tackles the scarce annotated data issue. In another study, Cycle-GAN network is adapted for bearing fault diagnosis [247]. The network learns one mapping from source to target and a reverse mapping from target to source. The cycle consistency loss measures the reconstruction error after two generating steps.

- *Hybrid approaches* use some of the mentioned approaches simultaneously to enhance model performance. In [233], the authors combined the reconstruction-based SSDAE network with MMD statistic for bearing fault diagnosis. Wang et al. [229] incorporated the CORAL distance loss of both marginal and conditional distributions into deep DAE objective function to learn domain-invariant and discriminative features from low-level to higher-level hierarchical latent layers.

Table 3.8.1 provides a summary of single-source deep domain adaptation approaches for PHM applications.

Table 3.8.1 Summary of single-source deep domain adaptation approaches for PHM applications, based on [238] **categorization.**

Approach	Unsupervised	Semi-supervised	Supervised
Discrepancy-based			
MMD	Li et al. [242], Li et al. [248], Zhang et al. [241], Yang et al. [249], Xiao et al. [250], Han et al. [251]	Lu et al. [228], Li et al. [190]	-
AdaBN	Zhang et al. [132]	-	Chen et al. [252]
Re-weighting	-	-	Xiao et al. [253]
AHKL divergence	Qian et al. [243]	-	-
Adversarial-based			
Discriminative	Da Costa et al. [230], Zhang et al. [191], Cheng et al. [231], Li et al. [254]	Han et al. [174], Li et. al. [255]	-
Generative	-	-	-
Reconstruction-based			
Encoder-decoder	-	-	Li et al. [246]
GAN architectures	Xie et al. [247]	-	-
Hybrid	Wang et al. [229], Sun et al. [233], Wen et al. [235], Sun et al. [234]	-	-

4.1.2 Unsupervised Multi-Source Domain Adaptation (MSDA)

The UDA methods mentioned in the previous section, mainly focus on target vs. single source. However, in real industrial applications, labeled data comes from different sources with various underlying distributions. As a result, the distribution shift among various sources should also be considered. In such cases, the naive application of the single-source UDA algorithms may lead to suboptimal solutions. Multi-source DA (MSDA) algorithms retrieve knowledge from various source domains and perform the predictions for the target domain. There are several theoretical analyses for existing MSDA algorithms [256]. For instance, Blitzer et al. [257] defined an error bound for empirical risk minimization based on a weighted combination of the source domains. In another study, Mansour et al. [258] proposed both linear and weighted mixtures of source datasets for MSDA task. Early MSDA approaches utilized shallow models either to combine prelearned classifiers or to learn a shared latent space. However, the recent development of deep learning and deeper neural network architectures opens doors to more practical MSDA scenarios. Several existing deep MSDA either attempt to learn a domain-invariant latent space by applying discrepancy-based loss functions or adversarial learning [259], [260]. There are fewer algorithms that focus on aligning the classifiers by learning task-specific decision boundaries [261]. In the current work, we proposed a novel model to simultaneously align latent representations and shape task-specific decision boundaries. Our proposed model aligns the features of each source and the target by minimizing the MMD measure. At the same time, Sliced Wasserstein discrepancy is used to learn task-specific decision boundaries for the classifiers. To the best of our knowledge, our work is one of the earliest attempts of unsupervised multi-source domain adaptation in PHM applications.

4.2 PROPOSED FEATURE-LEVEL AND TASK-SPECIFIC FEATURE DISTRIBUTION ALIGNMENT MULTI-SOURCE DOMAIN ADAPTATION (FT-MSDA)

4.3 NOTATIONS AND PROBLEM SETTING

Definition 1 (Domain) A domain $\mathcal{D}$ is composed of a m-dimensional feature space $\mathcal{X}$ and the marginal probability distribution P(**x**), i.e. $\mathcal{D}$={ $\mathcal{X}$, P(**x**)}, where **x** ∈ $\mathcal{X}$.

Definition 2 (Task) Given specific domain $\mathcal{D}$, task $\mathcal{T}$ is composed of a label space $\mathcal{Y}$ and the classifier function $f(\mathbf{x})$ where can be viewed as the conditional probability distribution P(y|**x**), i.e. $\mathcal{T} = \{ \mathcal{Y}, \mathrm{P}(y|\mathbf{x})\}$ where y ∈ $\mathcal{Y}$.

Problem Suppose we have N labeled source domains $\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^N$, and one fully unlabeled target domain $\mathcal{D}^t$. For the ith source domain $\mathcal{D}^i$, the observed samples and corresponding labels drawn from the source distribution $P_i(\mathbf{x}_i, y_i)$ are $\mathbf{X}_i = \{\mathbf{x}_i^j\}_{j=1}^{N_i} \in \mathcal{X}_i$ and $Y_i = \{y_i^j\}_{j=1}^{N_i}$, where N_i in the number of source samples, and $\mathcal{X}_i$ denotes the source feature space. The unlabeled target samples are drawn from the target distribution $P_t(\mathbf{x}_t, y_t)$, and are $\mathbf{X}_t = \{\mathbf{x}_t^j\}_{i=1}^{N_t} \in \mathcal{X}_t$, where N_t in the number of target samples, and $\mathcal{X}_t$ denotes the target feature space, and $P_i(\mathbf{x}_i) \neq P_t(\mathbf{x}_t)$, $P_i(y_i|\mathbf{x}_i) \neq P_t(y_t|\mathbf{x}_t)$.

In the current setup, we assume that the data from various domains are observed in the same feature space, *i.e.* $\mathbf{x}_i^j \in \mathbb{R}^d$, $\mathbf{x}_t^j \in \mathbb{R}^d$ (homogeneity condition). Also, we assume the closed-set scenario in which the source domains and the target domain share the same labels, *i.e.* $y_t^j \in \mathcal{Y}$ and $y_i^j \in \mathcal{Y}$, where $\mathcal{Y} = \{1, 2, \dots, c\}$ is the label space, and c is the number of possible health condition classes. We aim to build a domain adaptation model that learns the function f^t which is trained based on $\{\mathbf{X}_i, Y_i\}_{i=1}^{N}$ and $\{\mathbf{X}_t\}$, and can correctly predict the sample from the target domain.

4.4 PRELIMINARIES

This section discusses the details of mainstream approaches in unsupervised domain adaptation that are related to our approach.

4.4.1 Feature Distribution Matching

The moment matching techniques have been widely studied by the researchers to learn domain-invariant representations by reducing domain discrepancy through various distribution discrepancy metrics. Maximum Mean Discrepancy (MMD), as the most widely used metric, measures the First-order (Mean) statistics between the features of different domains in a projected space [241], [242], [248]–[251]. Reproducing Kernel Hilbert Space (RKHS) is a commonly used projected space which corresponds to a kernel. Correlation alignment (CORAL) metric is also proposed to align the second-order statistics of the source and target distributions [262]. Central Moment Discrepancy (CMD) [263] and Higher-order Moment Matching (HoMM) [264] align higher-order statistics of the distributions to guarantee fine-grained alignment [264]. In the current study, we focus on matching the feature distribution for multiple domains.

4.4.2 Maximum Mean Discrepancy

In unsupervised domain adaptation, the target sample labels are unavailable. Hence, one common strategy for UDA is to learn domain invariant representation via minimizing the domain distribution discrepancy. Many of the criteria to estimate the distance between the distributions such as KL divergence are parametric, and an intermediate density estimate is usually required. To avoid such a non-trivial task, a non-parametric distance estimate between distributions is desirable. Maximum Mean Discrepancy (MMD) is an effective non-parametric metric for comparing the distributions based on the first-order statistics between the features of different domains [265]. Given two distributions s and t, by mapping the data to a reproducing kernel Hilbert space (RKHS), the MMD between s and t is defined as:

$$MMD^2(s,t) = \sup_{\|\phi\|_{\mathcal{H}\leq 1}} \left\| E_{\boldsymbol{x}_s \sim s}[\phi(\boldsymbol{x}_s)] - E_{\boldsymbol{x}_t \sim t}[\phi(\boldsymbol{x}_t)] \right\|_{\mathcal{H}}^2 \qquad (4\text{-}1)$$

Where $E_{x_s \sim s}$ is the expectation for the distribution s, and $\|\phi\|_{\mathcal{H} \leq 1}$ is a set of mapping functions into RKHS $\mathcal{H}$. For two set of samples $D^s = \{\mathbf{x}_i^s\}_{i=1}^N$ and $D^t = \{\mathbf{x}_i^t\}_{i=1}^M$ drawn *i.i.d.* from the distributions s and t, respectively, the empirical MMD can be estimated as [265]:

$$MMD^2(D^s, D^t) = \left\| \frac{1}{N} \sum_{i=1}^{N} \phi(\boldsymbol{x}_s^i) - \frac{1}{M} \sum_{j=1}^{M} \phi(\boldsymbol{x}_t^j) \right\|_{\mathcal{H}}^2 \tag{4-2}$$

Where the feature map $\phi\,(.)$ is associated with the kernel map $k(\boldsymbol{x}_s, \boldsymbol{x}_t)$, i.e. $\langle \phi\,(\boldsymbol{x}_s), \phi\,(\boldsymbol{x}_t) \rangle = k(\boldsymbol{x}_s, \boldsymbol{x}_t)$. The optimal kernel choice is critical to ensure the test correctly distinguishes unlike distributions with high probability [266].

4.4.3 Sliced Wasserstein Distance

The Wasserstein distance – arising from the optimal transport- has been recently used in designing loss functions for deep DA applications. As opposed to popular probability distances used in the adversarial process such as total variation distance, KL, and Jensen-Shannon (JS) divergence, Wasserstein distance provides a continuous mapping and usable gradient everywhere [245]. In [231], the domain critic loss uses Wasserstein distance between the source and target distributions to optimize the CNN-based shared latent representation on a pre-trained CNN (on the source data) feature extractor.

Wasserstein distance is defined in form of a linear programming optimization and solving the problem is computationally expensive for high-dimensional data. Sliced Wasserstein distance (SWD) is proposed to make the optimization more computationally efficient [267]. The idea behind the Sliced Wasserstein distance is to first obtain a family of one-dimensional representations for a higher-dimensional probability distribution through projections/slicing, and then calculate the distance between two input distributions as a functional on the Wasserstein distance of their one-dimensional representations

Following [267], the sliced p-Wasserstein distance between two distributions and Q_s and Q_t is defined as:

$$SW_p(Q_s, Q_t) = \left(\int_{\mathbb{S}^{d-1}} W_p^p(\mathcal{R}_\theta Q_s, \mathcal{R}_\theta Q_t) d\theta \right)^{\frac{1}{p}} \qquad (4\text{-}3)$$

Where $\mathcal{R}_\theta$ denotes the linear one-dimensional projection (i.e. Radon transform) which maps Q_s and Q_t over all possible directions θ on the unit sphere $\mathbb{S}^{d-1} \in \mathbb{R}^d$. In this sense, the distance is obtained by solving several one-dimensional optimal transport problems, which have closed-form solutions [268], and has been shown to satisfy the properties of non-negativity, the identity of indiscernible, symmetry, and subadditivity. Hence, it is a reliable discrepancy metric between probability distributions [267].

Using samples $\mathcal{F} \sim Q_s$ and $\mathcal{G} \sim Q_t$ ($|\mathcal{F}| = |\mathcal{G}| = N$), let σ_s and σ_t be the permutations that sort N projected samples such that [265]:

$$\mathcal{R}_\theta \mathcal{F}_{\sigma_s(i)} \leq \mathcal{R}_\theta \mathcal{F}_{\sigma_s(i+1)}, \quad \forall i \in \{1 \leq i < N-1\} \qquad (4\text{-}4)$$

$$\mathcal{R}_\theta \mathcal{G}_{\sigma_t(i)} \leq \mathcal{R}_\theta \mathcal{G}_{\sigma_t(i+1)}, \quad \forall i \in \{1 \leq i < N-1\} \qquad (4\text{-}5)$$

Then, the sliced 2-Wasserstein distance can be approximated by calculating the average distance between the sorted samples:

$$SWD_2(\mathcal{F}, \mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{i=1}^{|\mathcal{F}|} \left\| \mathcal{R}_\theta \mathcal{F}_{\sigma_s(i)}, \mathcal{R}_\theta \mathcal{G}_{\sigma_t(i)} \right\|_2^2 \qquad (4\text{-}6)$$

4.4.4 Co-training and Maximum Classifier Discrepancy

In co-training, the learners will be trained alternately to maximize the mutual agreement on two distinct views of unlabeled data, and it is closely related to unsupervised domain adaptation (UDA)applications [269]. Based on the co-training idea, Saito et al. proposed the Maximum Classifier Discrepancy adversarial training method to align the distributions of the target domain by considering task-specific decision boundaries [270]. They train a feature extractor G and two classifiers C_1 and C_2 with different characteristics that produce corresponding logits $p_1(y|\boldsymbol{x})$, $p_2(y|\boldsymbol{x})$, for input $\boldsymbol{x}$. Two classifiers try to find the target samples that cannot be supported by the source, and they are trained to classify the source samples, simultaneously. Hence, the feature extractor generates discriminative features for target samples by taking into account the relationship between the decision boundary and target samples.

see Figure 4.4.1 [271]. The authors proposed using L1-distance as the discrepancy loss between two classifiers which is not helpful when the supports of two probability distributions do not overlap.

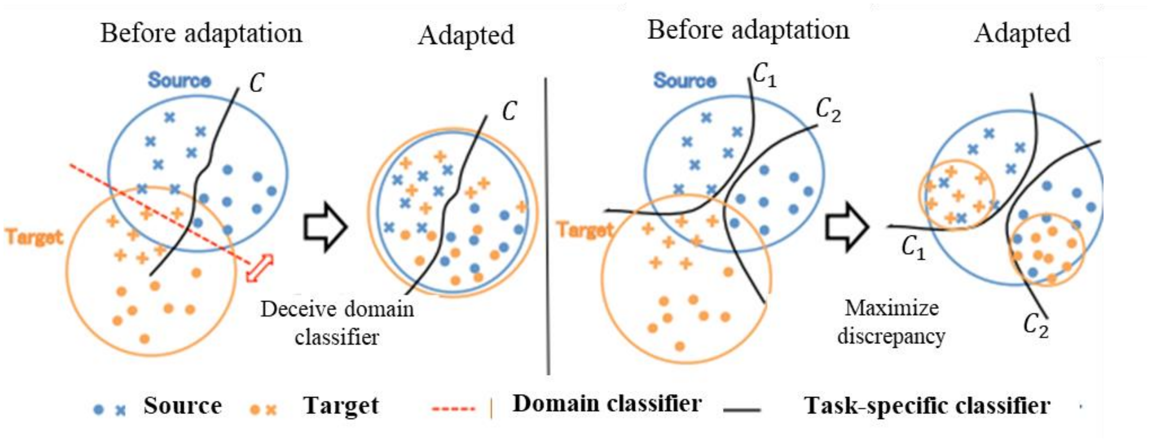


Figure 4.4.1 Comparison of standard domain classifier method (left) and maximum classifier discrepancy method (right) [271]**.**

4.5 ARCHITECTURE

The architecture of the proposed feature-level and task-specific distribution alignment multi-source domain adaptation (FTD-MSDA) framework is illustrated in 0Figure 4.5.1. It is composed of a global feature extractor G, (N) local feature extractors $F = \{F_1, ..., ,F_N\}$, N task-specific classifier sets $C = \{(C_1^1, C_1^2), ..., (C_i^1, C_i^2)\}, (i = 1, ... N)$, and the aggregation component which delivers the final prediction in the inference phase. N denotes the number of source domains.

Given a batch of samples $\{\mathbf{x}^j, \mathbf{y}^j\}, j = \mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^N$ from source domains and $\{\mathbf{x}^t\}$ from the target domain, the shared feature extractor maps the samples into a common feature space and globally aligns the features. Then, the local feature extractors map each pair of source and target domain data into a domain-specific feature space. The MMD loss is utilized to learn domain-invariant representations of the samples. Each task-specific predictor set $C_i = (C_i^1, C_i^2)$, is a pair of Softmax classifiers, that receives domain-invariant features $F_i(G(x))$ for ith source domain. The cross-entropy loss is used for each classifier. Following the co-training method, we define a discrepancy loss between the outputs of C_i^1 and C_i^2. The reliability of the network is highly reliant on the discrepancy loss. As mentioned earlier, in comparison to many well-known metrics, the Wasserstein metric takes into account the properties of the underlying geometry and can compare the distributions that don't share support [245]. Motivated by that, we propose to use a sliced Wasserstein discrepancy loss between the outputs of C_i^1 and C_i^2. In comparison to the original Wasserstein distance, SWD is more computationally efficient. In the inference stage, we adapt an ensemble scheme to aggregate the classifier outputs. The final prediction is a weighted average of paired classifier predictions.

Figure 4.5.2 demonstrates an intuitive example of representation scattering for the proposed adversarial approach with $N = 2$. During the training process, the global feature extractor and local feature extractors try to align the marginal probability distributions of each source-target pair by minimizing the MMD loss. On the other hand, during adversarial task-level adaptation, C_i^1 and C_i^2 for each source act as discriminators to consider the relationship between class boundaries and target samples. The final goal is to find those target features that are far from the support of that certain source. [270]. To this end, the adversarial training process utilizes the disagreement between C_i^1 and C_i^2 and avoids generating target features in this disagreement region. Hence, the final target distribution is pushed close to source distributions (See Figure 4.5.2 Right).

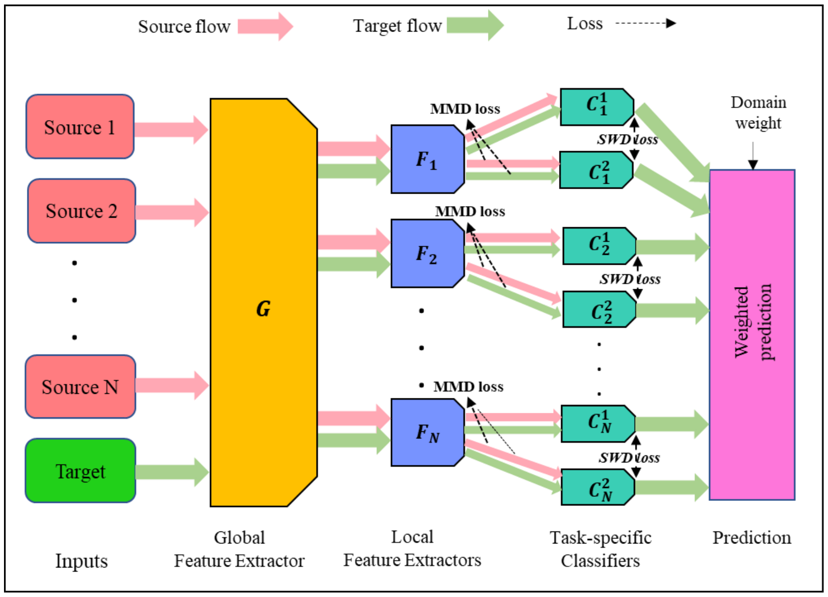


Figure 4.5.1 The proposed Feature-level and Task-specific distribution alignment multi-source domain adaptation

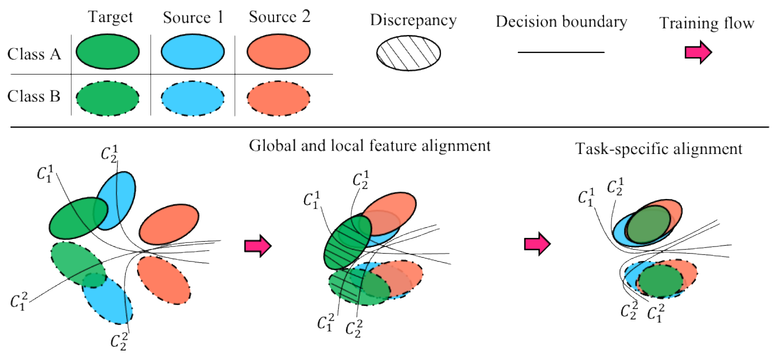


Figure 4.5.2 Analysis of adversarial training for the proposed method with two sources and binary classification.

4.6 TRAINING OBJECTIVE

The optimization procedure of the network consists of three steps:

Step 1: Given $X_1, \ldots, X_N$ as collections of *i.i.d* labeled samples from $\mathcal{D}^s = \{\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^N\}$, and X_t as the collection of unlabeled samples from the target domain $\mathcal{D}^t$, the MMD loss between $\mathcal{D}^s$ and $\mathcal{D}^t$ is defined as:

$$\begin{aligned}\mathcal{L}_{MMD^2}(D^s, D^T) &= \frac{1}{N}\sum_{i=1}^{N} \|E(X_i) - E(X_t)\|_2) \\ &= \frac{1}{N}\sum_{i=1}^{N} \left\|F_i\big(G(X_i)\big) - F_i\big(G(X_t)\big)\right\|_2)\end{aligned} \tag{4-7}$$

The multi-class cross-entropy loss can be derived as:

$$\mathcal{L}_{cls} = \sum_{i=1}^{N} \frac{1}{N_i} \sum_{j=1}^{N_i} \mathcal{L}_C^{\mathcal{D}^i}[C_i\left(F_i\big(G(\mathbf{x}_i^j)\big), y_i^j\right] \tag{4-8}$$

Where

$$\mathcal{L}_C^{\mathcal{D}^i}(\mathrm{X}_i, \mathrm{Y}_i) = -E_{(\boldsymbol{x_i}, y_i)\sim(\mathrm{X}_i, \mathrm{Y}_i)} \sum_{k=1}^{c} 1(y_i = k). \log\ p(y|\boldsymbol{x}_i) \tag{4-9}$$

In this step, we train G, F and C to classify the multi-source samples. The objective is:

$$\min_{G,F,C} \mathcal{L}_{cls} + \lambda \min_{F} \mathcal{L}_{MMD^2}(D^s, D^T) \tag{4-10}$$

λ is a balancing factor.

Step 2: The SWD discrepancy loss is defined as:

$$\mathcal{L}_{SWD}(X^t) = \sum_{i=1}^{N} \frac{1}{mn} \sum_{j=1}^{m} \sum_{k=1}^{n} \left\| \mathcal{R}_{\theta_j}(P_{C_i^1})_{\sigma_{P_{C_i^1}}(k)} - \mathcal{R}_{\theta_j}(P_{C_i^2})_{\sigma_{P_{C_i^2}}(k)} \right\|_2^2 \tag{4-11}$$

Where $P_{C_i^1}$ and $P_{C_i^2}$ are the outputs of C_i^1 and C_i^2. m and n denote the number of random projections and the batch size, respectively.

In step 2, we freeze the parameters of G and F, and maximize the SWD discrepancy between the outputs of two classifiers on target samples. The objective function is:

$$\min_{C} \mathcal{L}_{cls} - \mathcal{L}_{SWD} \tag{4-12}$$

Step 3: In the last step, we freeze the parameters of the classifiers and update G and F on target samples to minimize the SWD discrepancy between the classifiers:

$$\min_{G,F} \mathcal{L}_{SWD}(X^t) \tag{4-13}$$

The last step makes the target feature space closer to the source. The steps above are periodically done until the network convergence is achieved. The whole process is demonstrated in Algorithm 1.

Algorithm 1 Feature-level and task-specific distribution alignment multi-source domain adaptation (FT-MSDA)

```
Input: Labeled source domains data $\{\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^N\}$, unlabeled target data $\mathcal{D}^t$, number of random projections $m$, batch size $n$, randomly initialized $G$, $F = \{F_i\}_{i=1}^N$, and $C = \{(C_i^1, C_i^2)\}_{i=1}^N$
    while $G$, $F$, and $C$ have not converged do
    Step 1: Train $G$, $F$, and $C$ to classify the multi-source samples with equation (4-9)
    Step 2: Freeze $G$ and $F$ parameters, train $C$ on unlabeled target samples to maximize SWD discrepancy (4-11)
    Step 3: Train $G$ and $F$ on unlabeled target samples to minimize SWD discrepancy (4-12)
    end while
```

4.7 INFERENCE PHASE

In the testing stage, we utilize the ensemble scheme to aggregate the classifier outputs. We define a weight vector $W = (w_1, w_1, \dots, w_N)$, $\sum_{i=1}^N w_i = 1$, the final prediction is a weighted average of paired classifier predictions. As proposed in [271], we use source-only accuracies between ith domain and target domain to derive the weight vector, *i.e.* $w_i = acc_i / \sum_{j=1}^N acc_j$.

4.8 EXPERIMENTS

We evaluate the performance of the proposed MSDA framework for bearing fault diagnosis tasks and compare our proposed model with several state-of-the-art existing domain adaptation frameworks. We perform the evaluations on the well-known CWRU and KAT public datasets which have been introduced in chapter 3.

4.8.1 Datasets

- **CWRU** is introduced in chapter 3. It contains vibration measurements for two bearings, and the data is recorded under four various motor loads: 0, 1 hp, 2 hp, 3 hp. Changing the operating conditions will change the data distribution, and for that reason, the model which is trained on one motor load would perform poorly on the other motor load data. We use the drive-end bearing data. Datasets from motor load 0 hp, 1hp, 2hp, and 3hp have been defined as dataset A, B, C, and D respectively. All four datasets include ten health condition classes of the drive-end bearing installed on the motor which are acquired at 48 kHz sampling frequency.

- **Paderborn University KAT bearing dataset** is another well-known public dataset for bearing fault diagnosis [36]. The dataset is provided by KAT datacenter in Paderborn University [36]. It includes measured motor currents and vibration signals. Experiments were performed on 26 damaged bearing states and 6 undamaged states. Among the damaged bearings, 14 were naturally damaged by accelerated life tests, and 12 were artificially damaged by EMD or drilling.

For both artificial and real damages (see Chapter 3), the faults at two locations (outer race and inner race) with 1~3 damage severity levels are available. For the current experiment, based on the location of the damage, three classes are defined for each dataset: Healthy, Inner race, and Outer Race which are collected under operating conditions 1, 2, and 3. The operating conditions are called KAT1, KAT2, and KAT3.

4.8.2 Components architectures

As mentioned earlier, we evaluate the proposed model with raw vibration signals. 1-dimensional CNNs have been used as feature extractor layers, see Figure 4.8.1. However, according to the task of interest and type of inputs, one can leverage various architectures for individual components, i.e. 2-D CNN, RNNs, etc. For each task-specific classifier, we used two fully connected layers in the combination of batch-normalization layers and the softmax prediction layer. Table 4.8.1 shows the architecture of network components.

Table 4.8.1 Proposed architecture network layers: BN, MP, and D stand for batch normalization, Max Pooling, and Dropout layers respectively. The number preceding Conv1D shows the number of filters, and the number after Conv1D illustrates the filter size. C refers to the number of classes

Component	Network architecture
Input	Data layer (2048)
Global feature extractor	32*Conv1D1*(64*1), *BN1*(32), *MP1*(4*2),*D1* (0.5), 32*Conv1D2*(3*1), *BN2*(32), *MP2*(4*2),*D2* (0.5)
Local feature extractors	64*Conv1D3*(3*1), *BN3*(64), *MP3*(4*2),*D3* (0.5), 64*Conv1D4*(3*1), *BN4*(64), *MP4*(4*2),*D4* (0.5)
Task-specific classifiers	*FC1*(320), *BN4*(320), *FC2*(200), *FC3*(c)

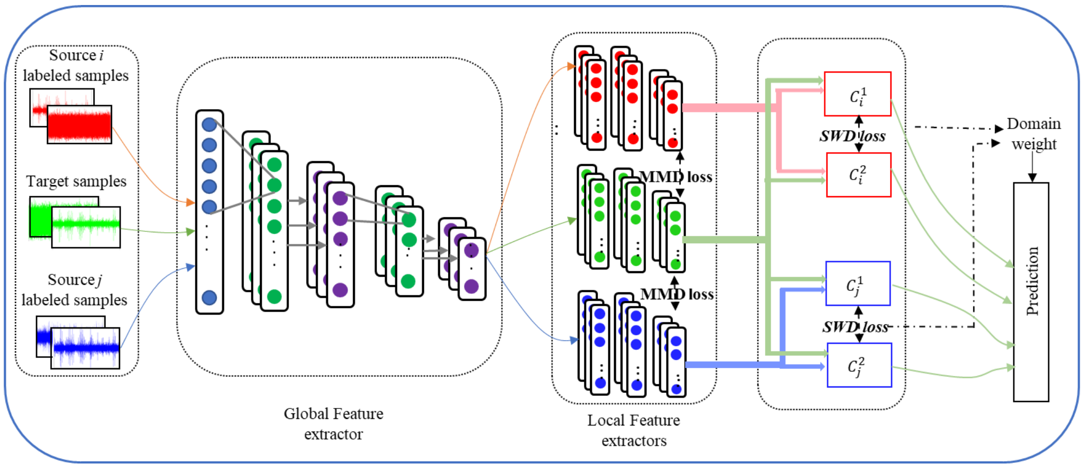


Figure 4.8.1 Proposed framework components for bearing fault diagnosis task.

4.8.3 Experimental set-up

To demonstrate the effectiveness of the proposed approach, we define three sets of experiments based on CWRU and KAT datasets. All the experiments are done using Pytorch 1.5 and were running on NVIDIA GTX 2060 GPU. The reported experimental results are averaged by 5 trials to reduce the effect of randomness, and the results are given in the form of $\mu \pm sd$, where μ and sd stand for mean value and standard deviation of 5 trials. For the sake of comparison. Some results are taken from the published paper. Hence, they are not in the form of $\mu \pm sd$. All the experiments are trained on mini-batches of 128, and the learning rate was 1e-4.

Experiment 1: CWRU datasets from motor loads 0 hp, 1hp, 2hp, and 3hp have been defined as dataset A, B, C, and D respectively. All four datasets include ten health condition classes of the drive-end bearing installed on the motor which are acquired at 48 kHz sampling frequency. Raw signals from 10 health conditions are segmented into samples of 2048 points. A simple sample augmentation method with an overlap length of 128~200 points is used to avoid losing information and to balance datasets across domains, see Figure 4.8.3. Hence, each dataset contains 3600 i.e. 360 samples for each class. We take 2500 and 1100 samples of each domain as training and testing subsets respectively. We then take turns to set one domain as the target domain and the rest as the source domains.

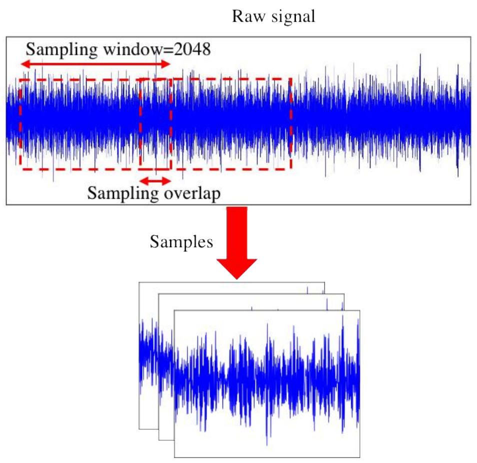


Figure 4.8.3 Sample preparation, the signals are segmented into samples of 2048 with an overlap of 128~200 points.

Experiment 2: For both artificial and real damages of the KAT dataset, the faults at two locations (outer race and inner race) with 1~3 damage severity levels are available. For the current experiment, based on the location of the damage, three classes are defined for each dataset: Healthy, Inner race, and Outer Race which are collected under operating conditions 1, 2, and 3, see Table 4.8.2. The operating conditions are called KAT1, KAT2, and KAT3. The data preparation step is similar to the previous experiment for CWRU datasets. We draw 7800 samples (2600 samples in each class) for each dataset. We take 4950 and 2850 samples of each domain as training and testing subsets respectively.

Experiment 3: CWRU dataset consists of nine faulty classes at the inner race, outer race, and ball with three different fault severity levels. KAT dataset does not have the measurement for fault at the ball element. Hence, we have combined three various fault severities at the inner race into one big class of inner race to match the number of classes with the KAT dataset for domain adaptation. Similarly, three various fault severities at the outer race are combined into one big class of outer race. With this new categorization, both KAT and CWRU datasets share the same label space of three classes: Healthy, Inner race, and Outer race. This experiment evaluates the performance of the proposed approach for multi-source DA across various machinery. We draw 7800 samples (2600 samples in each class) for each dataset. We draw 5850 samples (1950 samples in each class) for each domain. We take 4500 and 1350 samples of each domain as training and testing subsets respectively.

4.8.4 Comparative studies and Results

In this section, we present the adaptation result on two benchmark fault diagnosis datasets to evaluate the proposed FTD-MSDA: CWRU and KAT. We compare the results with state-of-the-art single-source and multi-source unsupervised domain adaptation algorithms. To the best of our knowledge, this study is an early attempt at multi-source domain adaptation in the PHM field. Hence, for the sake of comparison, the authors utilized the existing state-of-the-art multi-source domain adaptation algorithms in other fields such as semantic segmentation and image classification. We used two standards for reporting the results: 1- Single-source best: which utilizes existing single-source domain adaptation methods on our experiment datasets and reports the best source transfer accuracy on the target set. 2- Multisource: Reports the performance of multi-source domain adaptation algorithms. Moreover, we compare the results with another version of our proposed model in which we replace SWD metric with standard L1-distance, to evaluate the effect of discrepancy distance on the maximum classifier discrepancy technique.

For experiment #1, the proposed model achieves the average accuracy of 95.03%, outperforming other baselines, see Figure 4.8.4, Figure 4.8.5, and Table 4.8.2. One interesting observation is the transfer performance for dataset A. For "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$" transfer scenario, the highest performance is achieved by the single-source UDA algorithm of CLAN [272] which carries out an adaptive class-level domain adaptation to mitigate the effect of global alignment in terms of intra-class compactness between the domains. The poor performance of multi-source domain adaptation algorithms for dataset A is probably due to the presence of “negative transfer”, which causes the samples from different domains but of the same class to be mapped farther away in the feature space [273].

In Table 4.8.3, we compared the class-wise performance of “without domain adaptation” and “Proposed multi-source domain adaptation” methods for the transfer scenario "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$" . The individual class accuracies are obviousely improved for all classes except for C5. Although the overall performance of the model improved, the influence of negative transfer for this class is obvious and explains why the highest accuracy (for this specific transfer scenario) is achieved by a class-wise domain adaptation algorithm (i.e. CLAN). In the presence of negative transfer, the training of source leads to worse performance of the target. More the domains are different, more negative transfer is likely to occur. In experiment#1 scenario, there is a huge domain shift between working condition A and working condition D, as the motor rotation speeds are highly different. The SWD metric handles larger domain discrepancies by obtaining more stable gradients. However, this observation emphasizes the importance of finding the related source domains for multi-source algorithms to avoid negative transfer phenomena by mitigating the effect of the unrelated source.

For experiment #2 and experiment #3, we compared our proposed FTD-MSDA method with several well-known SDA and MSDA algorithms, see Table 4.8.4 and Table 4.8.5.

For "$\mathbf{KAT1}, \mathbf{KAT2} \rightarrow \boldsymbol{KAT}\mathbf{3}$" transfer task, the highest performance of 98.9 is achieved with the proposed FTD-MSDA model (Figure 4.8.6). However, the highest accuracy of 99.0 for "$\mathbf{KAT1}, \mathbf{KAT2} \rightarrow \boldsymbol{KAT}\mathbf{3}$" is achieved by MDDA which utilizes a source distilling mechanism to select the source training samples that are closer to the target [273]. In experiment #3, the average lower accuracy compared with the previous two experiments confirms the larger domain shift across different machinery (see Figure 4.8.7). According to the results, the proposed method and M3SDA [259] deliver a comparable performance that is higher than other state-of-the-art algorithms. For all three experiments, using L1-distance in the proposed model gives a lower accuracy compared to SWD, which was expected.

In addition to classification accuracy, we have compared the proposed algorithm with existing MSDA algorithms in terms of training time and inference time (See Table 4.8.6). According to the results, the computational cost of the model is comparable to state-of-the-art algorithms.

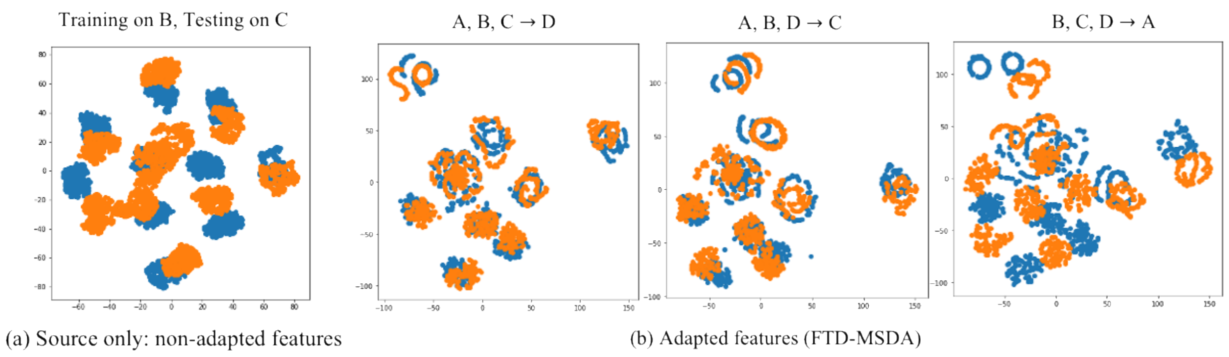


Figure 4.8.4 (a) t-SNE feature visualization of the non-adapted model for experiment #1 (b) Feature-level and task-specific distribution alignment Multi-Source Domain Adaptation (FTD-MSDA): t-SNE feature visualization for experiment 1, target features are shown in orange. The models are trained using multiple sources. But for the sake of visualization, only the features of one source domain with the highest accuracy among all the sources have been shown in blue.

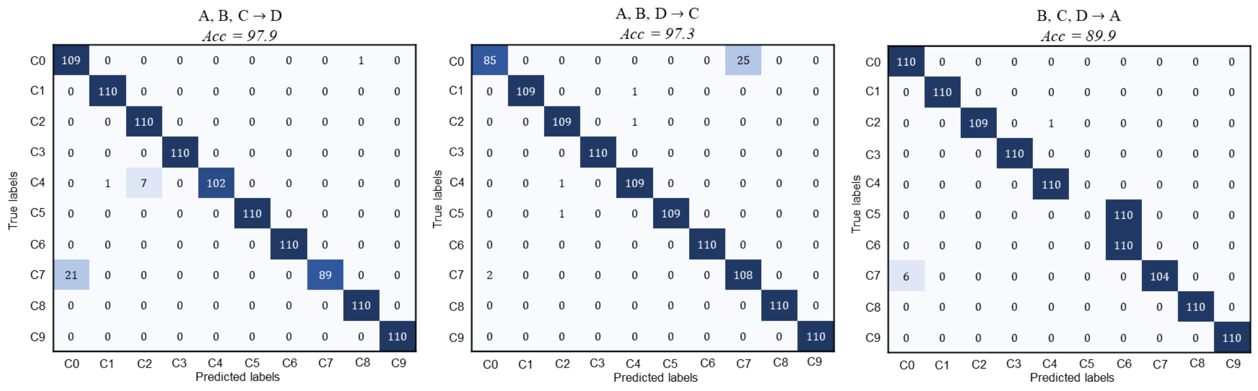


Figure 4.8.5 Proposed Multi-source domain adaptation: test confusion matrices for experiment #1.

Table 4.8.2 Classification accuracies (%) for experiment #1.

Standards	Model	Number of classes	A, B, C → D	A, B, D → C	B, C, D → A	Average
Single source best	Source only	10	80.8	83.2	81.2	81.7
	Li et al. [242]	10	84.8	*NA*	84.4	84.6
	Cheng et al. [231]	10	92.9	91.5	93.3	92.6
	Sun et al. [233]	15	82.9	90.3	88.7	87.3
	TCA [274]	10	79.8±0.2	65.5±0.5	68.9±0.4	71.4
	DDC [275]	10	88.2±0.4	91.1±0.2	90.6±0.2	90.0
	WDGRL [276]	10	87.3±0.2	86.9±0.1	85.0±0.1	86.4
	DANN [277]	10	86.8±0.3	87.4±0.1	85.9±0.1	86.7
	CLAN [272]	10	94.9±0.1	95.2±0.0	**93.7**±0.1	94.6
Multi-source	DCTN [278]	10	93.5±0.1	93.9±0.0	81.3±0.2	89.6
	M3SDA [259]	10	96.1±0.0	95.4±0.0	86.3±0.1	92.6
	MDDA [273]	10	97.5±0.2	96.9±0.2	88.2±0.1	94.5
	FTD-MSDA(Ours with L1)	10	96.2±0.1	96.9±0.1	87.6±0.2	93.6
	FTD-MSDA(Ours)	10	**97.9**±0.4	**97.3**±0.1	89.9±0.4	**95.03**

Table 4.8.3 Class-wise accuracies (%) for experiment #1 (B, C, D → A transfer scenario).

MSDA Scenario	Class	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
B, C, D → A	Without DA	80.3	87.1	83.9	86.5	85.5	61.7	86.2	78.9	83.1	82.6
	FTD-MSDA(Ours)	100.0	100.0	99.1	100.0	100.0	0.0	100.0	94.5	100.0	100.0
	Improvement %	**19.7**	**12.9**	**15.2**	**13.5**	**14.5**	-61.7	**13.2**	**15.6**	**16.9**	**17.4**

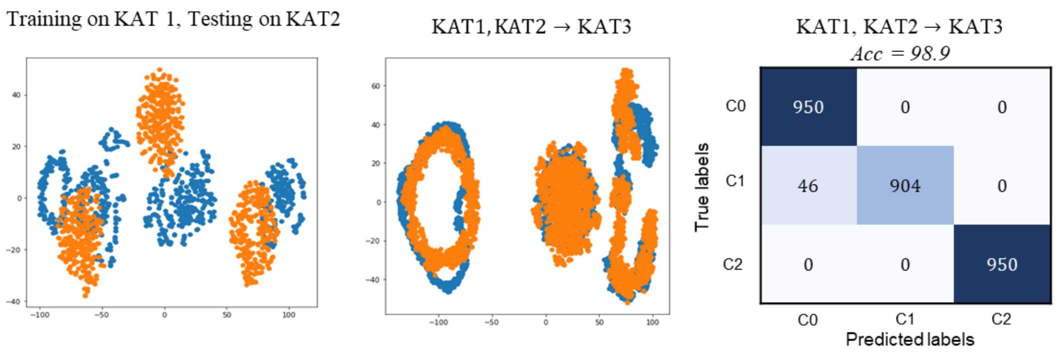


Figure 4.8.6 (a) t-SNE feature visualization of the non-adapted model for experiment #2 (left) (b) Feature-level and task-specific distribution alignment Multi-Source Domain Adaptation (FTD-MSDA): t-SNE feature visualization for experiment 2, target features are shown in orange. The models are trained using multiple sources. But for the sake of visualization, only the features of one source domain with the highest accuracy among all the sources have been shown in blue (middle). (c) test confusion matrix, experiment 2 (right).

Table 4.8.4 Classification accuracies (%) for experiment #2.

Standards	Model	Number of classes	KAT1, KAT2 → KAT3	KAT2, KAT3 → KAT1	Average
Single source best	Source only	3	89.9±0.1	90.8±0.3	90.3
	DDC [275]	3	94.0±0.3	96.2±0.0	95.1
	WDGRL [276]	3	96.5±0.1	96.0±0.2	96.3
	DANN [277]	3	97.0±0.1	98.1±0.2	97.6
	CLAN [272]	3	98.3±0.0	98.0±0.1	98.1
Multi-source	DCTN [278]	3	97.9±0.1	96.9±0.0	97.4
	M3SDA [259]	3	98.1±0.0	95.4±0.0	96.8
	MDDA [273]	3	98.7±0.2	**99.0**±0.1	**98.8**
	FTD-MSDA(Ours with L1)	3	98.0±0.1	98.1±0.2	98.0
	FTD-MSDA(Ours)	3	**98.9**±0.0	98.3±0.1	98.6

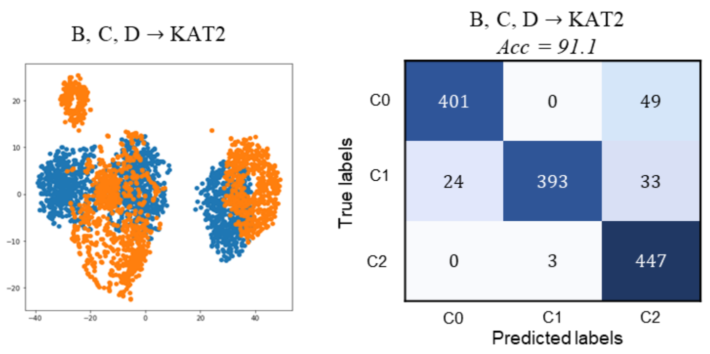


Figure 4.8.7 (a) t-SNE feature visualization of feature-level and task-specific distribution alignment Multi-Source Domain Adaptation (FTD-MSDA for experiment #3, target features are shown in orange. The models are trained using multiple sources. But for the sake of visualization, only the features of one source domain with the highest accuracy among all the sources have been shown in blue (left). (b) test confusion matrix, experiment 3 (right).

Table 4.8.5 Classification accuracies (%) for experiment #3.

Standards	Model	Number of classes	B, C, D → KAT2	KAT1, KAT2, KAT3 → B	Average
Single source best	Source only	3	79.0±0.3	78.0±0.2	78.5
	DDC [275]	3	82.1±0.2	84.6±0.1	83.3
	WDGRL [276]	3	84.1±0.1	83.9±0.3	84.0
	DANN [277]	3	86.1±0.2	86.6±0.1	86.3
	CLAN [272]	3	86.5±0.0	88.0±0.2	87.0
Multi-source	DCTN [278]	3	88.7±0.2	87.9±0.1	88.3
	M3SDA [259]	3	91.0±0.1	**89.4**±0.0	**90.2**
	MDDA [273]	3	88.9±0.3	**89.4**±0.2	89.1
	FTD-MSDA(Ours with L1)	3	87.3±0.1	86.9±0.0	87.1
	FTD-MSDA(Ours)	3	**91.1**±0.1	89.3±0.2	**90.2**

Table 4.8.6 Computational cost comparison (second) for experiment #1 .

	DCTN [278]	M3SDA [259]	MDDA [273]	FTD-MSDA (Ours with L1)	FTD-MSDA (Ours)
Training time (per iteration)	2.19	1.12	2.34	1.10	**0.91**
Inference time	3.1	**2.8**	3.8	2.9	**2.8**

4.9 SUMMARY

In this chapter, we proposed a new FTD-MSDA deep learning framework to align multiple source domains with the target domain. The global feature extractor aligns the lower-level features across the domains. Further, we incorporated the maximum mean discrepancy loss to map each pair of source and target domain data into a domain-specific feature space. The task-specific classifier pairs act as a model discriminator to detect the samples that are far from the support of each source. We proposed a Sliced Wasserstein Distance discrepancy loss between the probabilistic outputs of each classifier set. The model is adversarially trained in an end-to-end fashion. The results are reported into single-source best and multi-source standards. The former uses existing single-source domain adaptation methods on our experiment datasets and reports the best source transfer accuracy on the target set. The latter compares the results with existing state-of-the-art MSDA algorithms. Our experimental results show that the proposed model outperforms the state-of-the-art domain adaptation algorithms. Moreover, for "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$" transfer task, the highest performance is achieved by the single-source domain adaptation. The poor performance of all multi-source domain adaptation algorithms for dataset A is due to negative transfer caused by the presence of an unrelated source domain in the multi-source domain adaptation algorithm. It emphasized the effect of source domain weighting in multi-source domain algorithms, which will be addressed in Chapter 5.

CHAPTER 5: PROPOSED FTD-MSDA WITH A NOVEL WEIGHTING STRATEGY

In this chapter, we add a novel weighting strategy to our proposed FTD-MSDA framework to identify the most relevant sources and eliminate the influence of unrelated sources. The weight parameter is defined based on the conditional distribution discrepancy over the target domain and each source domain. The proposed weighting scheme considers the source domain relevance and intra-class alignment at the same time. The final predictor uses the weighted aggregation of source classifiers to perform a more accurate prediction.

5.1 NOTATIONS AND PROBLEM SETTING

Definition 1 (Domain) A domain $\mathcal{D}$ is composed of a m-dimensional feature space $\mathcal{X}$ and the marginal probability distribution P($\mathbf{x}$), i.e. $\mathcal{D}$={ $\mathcal{X}$, P($\mathbf{x}$)}, where $\mathbf{x} \in \mathcal{X}$.

Definition 2 (Task) Given specific domain $\mathcal{D}$, task $\mathcal{T}$ is composed of a label space $\mathcal{Y}$ and the classifier function $f(\mathbf{x})$ where can be viewed as the conditional probability distribution P($y|\mathbf{x}$), i.e. $\mathcal{T} = \{ \mathcal{Y}, \mathrm{P}(y|\mathbf{x})\}$ where y $\in \mathcal{Y}$.

Problem Suppose we have N labeled source domains $\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^N$, and one fully unlabeled target domain $\mathcal{D}^t$. For the ith source domain $\mathcal{D}^i$, the observed samples and corresponding labels drawn from the source distribution $P_i(\mathbf{x}_i, y_i)$ are $\mathbf{X}_i = \{\mathbf{x}_i^j\}_{j=1}^{N_i} \in \mathcal{X}_i$ and $Y_i = \{y_i^j\}_{j=1}^{N_i}$, where N_i in the number of source samples, and $\mathcal{X}_i$ denotes the source feature space. The unlabeled target samples are drawn from the target distribution $P_t(\mathbf{x}_t, y_t)$, and are $\mathbf{X}_t = \{\mathbf{x}_t^j\}_{i=1}^{N_t} \in \mathcal{X}_t$, where N_t in the number of target samples, and $\mathcal{X}_t$ denotes the target feature space, and $P_i(\mathbf{x}_i) \neq P_t(\mathbf{x}_t)$, $P_i(y_i|\mathbf{x}_i) \neq P_t(y_t|\mathbf{x}_t)$.

In the current setup, we assume that the data from various domains are observed in the same feature space, *i.e.* $\mathbf{x}_i^j \in \mathbb{R}^d$, $\mathbf{x}_t^j \in \mathbb{R}^d$ (homogeneity condition). Also, we assume the closed-set scenario in which the source domains and the target domain share the same labels, *i.e.* $y_t^j \in \mathcal{Y}$ and $y_i^j \in \mathcal{Y}$, where $\mathcal{Y} = \{1, 2, \ldots, c\}$ is the label space, and c is the number of possible health condition classes. We aim to build a domain adaptation model that learns the function f^t which is trained based on $\{\mathbf{X}_i, Y_i\}_{i=1}^N$ and $\{\mathbf{X}_t\}$, and can correctly predict the sample from the target domain.

5.2 TRAINING OBJECTIVE

In the current work, we propose to refine the proposed FTD-MSDA by adding a source weighting strategy based on the conditional probability distribution over each source domain and target domain. Since estimating the posterior conditional probability distribution, i.e. $\mathrm{P}(y|\mathbf{x})$ is nontrivial to compute, we explore class-conditional probability, i.e. $\mathrm{P}(\mathbf{x}|y)$ in terms of the Bayesian formula [279].

The MMD between source k and target class-conditional probabilities is defined as:

$$M_{C_k} = \frac{1}{C}\sum_{c=1}^{C}\left\|\frac{1}{N_k^c}\sum_{\mathbf{x}_i^c \in \mathcal{D}^{k(c)}} F_k(G(\mathbf{x}_i^c)) - \frac{1}{N_t^c}\sum_{\mathbf{x}_t^c \in \mathcal{D}^{t(c)}}^{N_t^c} F_k(G(\mathbf{x}_t^c)\right\|_2 \tag{5-1}$$

Where $\mathcal{D}^{k(c)}$ is the set of samples belonging to class c with labels y_i^c. Correspondingly, $\mathcal{D}^{t(c)}$ represents the set of samples in class c with pseudo labels $\tilde{y}_t^c$. Also, N_k^c and N_t^c denote the number of samples in class c for source k and target domain, respectively. Target pseudo labels $\tilde{y}_t$ are estimated from the average prediction of source classifiers, and will be updated at each iteration.

we define the source vector as $\boldsymbol{\omega} = \{\omega_i\}_{i=1}^{N}$, where N denotes the number of source domains:

$$\boldsymbol{\omega}_k = \frac{1}{N-1}\sum_{\substack{i=1\\ i\neq k}}^{N}\left(\frac{2}{1+\exp\left(-2M_{C_i}\right)} - 1\right) \tag{5-2}$$

Which scales M_{C_i} to $[0, 1)$. Following what is suggested in [280], we define the weight of each source as a function of other sources' divergence (not that of M_{C_k}) from the target domain. This way, minimizing $\boldsymbol{\omega}_k$ will not minimize M_{C_k} but leads to minimization of divergence for other more related sources which results in conditional distribution alignment as well.

The modified cross-entropy loss is then defined as:

$$\mathcal{L}'_{cls} = \sum_{i=1}^{N}\frac{\omega_i}{N_i}\sum_{j=1}^{N_i}\mathcal{L}_C^{\mathcal{D}^i}[C_i\,(F_i(G(\mathbf{x}_i^j)), y_i^j] \tag{5-3}$$

Where

$$\mathcal{L}_C^{\mathcal{D}^i}(\mathrm{X}_i, \mathrm{Y}_i) = -E_{(\boldsymbol{x}_i, y_i)\sim(\mathrm{X}_i,\mathrm{Y}_i)}\sum_{k=1}^{c}1(y_i = k).\log\, p(y|\boldsymbol{x}_i) \tag{5-4}$$

With the changes made above, the modified FTD-MSDA network i.e. weighted FTD-MSDA is trained over the following three steps:

Step 1: Given $X_1, \dots, X_N$ as collections of *i.i.d* labeled samples from $\mathcal{D}^s = \{\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^N\}$, and X_t as the collection of unlabeled samples from the target domain $\mathcal{D}^t$, the MMD loss between $\mathcal{D}^s$ and $\mathcal{D}^t$ is defined as:

$$\begin{aligned}\mathcal{L}_{MMD^2}(D^s, D^T) &= \frac{1}{N}\sum_{i=1}^{N} \|E(X_i) - E(X_t)\|_2) \\ &= \frac{1}{N}\sum_{i=1}^{N} \left\|F_i\big(G(X_i)\big) - F_i\big(G(X_t)\big)\right\|_2)\end{aligned} \tag{5-5}$$

In this step, we train G, F and C to classify the multi-source samples. The objective is:

$$\min_{G,F,C} \mathcal{L}'_{cls} + \lambda \min_{F} \mathcal{L}_{MMD^2}(D^s, D^T) \tag{5-6}$$

λ is a balancing factor.

Step 2: The SWD discrepancy loss is defined as in chapter 4:

$$\mathcal{L}_{SWD}(X^t) = \sum_{i=1}^{N} \frac{1}{mn} \sum_{j=1}^{m} \sum_{k=1}^{n} \left\| \mathcal{R}_{\theta_j}(P_{C_i^1})_{\sigma_{P_{C_i^1}}(k)} - \mathcal{R}_{\theta_j}(P_{C_i^2})_{\sigma_{P_{C_i^2}}(k)} \right\|_2^2 \tag{5-7}$$

Where $P_{C_i^1}$ and $P_{C_i^2}$ are the outputs of C_i^1 and C_i^2. m and n denote the number of random projections and the batch size, respectively.

In step 2, we freeze the parameters of G and F, and maximize the SWD discrepancy between the outputs of two classifiers on target samples. The objective function is:

$$\min_{C} \mathcal{L}'_{cls} - \mathcal{L}_{SWD} \tag{5-8}$$

Step 3: In the last step, we freeze the parameters of the classifiers and update G and F on target samples to minimize the SWD discrepancy between the classifiers (similar to the originally proposed network):

$$\min_{G,F} \mathcal{L}_{SWD}(X^t) \tag{5-9}$$

Algorithm 2 demonstrates the training process.

```
Algorithm 2 Weighted Feature-level and task-specific distribution alignment multi-source domain adaptation (Weighted FT-MSDA)

Input: Labeled source domains data $\{\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^N\}$, unlabeled target data $\mathcal{D}^t$, number of random projections $m$, batch size $n$, randomly initialized $G$, $F = \{F_i\}_{i=1}^N$, and $C = \{(C_i^1, C_i^2)\}_{i=1}^N$
    while $G$, $F$, and $C$ have not converged do
    Step 1: Train $G$, $F$, and $C$ to classify the multi-source samples with equation (5-6)
    Step 2: Freeze $G$ and $F$ parameters, train $C$ on unlabeled target samples to maximize SWD discrepancy (5-8)
    Step 3: Train $G$ and $F$ on unlabeled target samples to minimize SWD discrepancy (5-9)
    end while
```

The proposed weighting strategy is closely related to the scheme presented in [279]. However, their proposed weight for semi-supervised MSDA ranges between $[0.5, 1)$, as opposed to our proposed weight which ranges between $[0, 1)$ and discards the effect of totally irrelevant source domain to avoid negative transfer issue. Also, our proposed weight does not need any labeled target samples which suit better to more real-case MSDA scenarios. Moreover, it provides larger derivatives which may lead to faster convergence of the network. The averaged prediction of the classifiers is used for the inference stage.

5.3 COMPARATIVE STUDIES AND RESULTS

We evaluate the performance of the proposed MSDA framework for bearing fault diagnosis tasks and compare our proposed model with several state-of-the-art existing domain adaptation frameworks. We perform the evaluations on the well-known CWRU which has been introduced in Chapter 3. The data preparation step is similar to **Experiment**

1 in Chapter 4. Raw signals from 10 health conditions are segmented into samples of 2048 points. A simple sample augmentation method with an overlap length of 128~200 points is used to avoid losing information and to balance datasets across domains. Hence, each dataset contains 3600 i.e. 360 samples for each class. We take 2500 and 1100 samples of each domain as training and testing subsets respectively.

As seen in Table 5.3.1, the proposed weighted model achieves an average accuracy of 97.3%, not only outperforming other baselines, it is 2.3% better than the originally proposed framework. For "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$" transfer scenario, our modified model gives 93.9% accuracy which is slightly higher than the single-source CLAN baseline [272], see Figure 5.3.1.

In Table 5.3.2, we compared the class-wise performance of "without domain adaptation" and "Proposed weighted multi-source domain adaptation" methods for the transfer scenario "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$". Similar to the original model, the individual class accuracies are improved for all classes, but for C5, a slightly negative transfer can be seen, i.e. -6.2%. However, compared to the original model the class-wise accuracy for C5 is increased by 55.5%, which shows the effect of class-conditional distribution alignment of the proposed weighting scheme.

Figure 5.3.2 indicates the weight of sources concerning training epochs for A, B, C → D adaptation scenario. As it can be seen, in the beginning, the weights are dropping drastically. After a while, the effect of classification loss surpasses which makes the conditional MMD larger, and makes the class-conditional distribution alignment harder.

After the classification gets stable, the samples become orderly, and the weights decrease. As expected, as domain A is the most irrelevant source, it converges to the lowest weight.

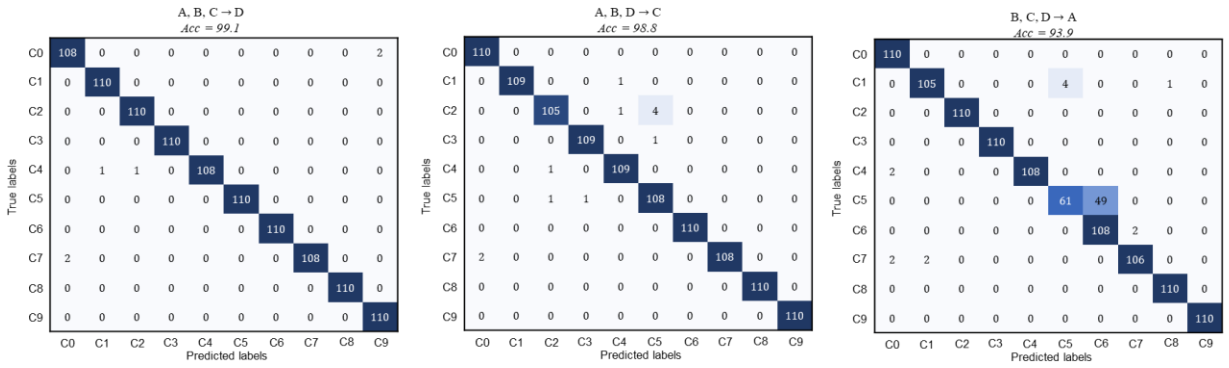


Figure 5.3.1 Proposed weighted FTD-MSDA: test confusion matrices for CWRU dataset.

Table 5.3.1 Classification accuracies (%) for CWRU dataset.

Standards	Model	Number of classes	A, B, C → D	A, B, D → C	B, C, D → A	Average
Single source best	Source only	10	80.8	83.2	81.2	81.7
	Li et al. [242]	10	84.8	*NA*	84.4	84.6
	Cheng et al. [231]	10	92.9	91.5	93.3	92.6
	Sun et al. [233]	15	82.9	90.3	88.7	87.3
	TCA [274]	10	79.8±0.2	65.5±0.5	68.9±0.4	71.4
	DDC [275]	10	88.2±0.4	91.1±0.2	90.6±0.2	90.0
	WDGRL [276]	10	87.3±0.2	86.9±0.1	85.0±0.1	86.4
	DANN [277]	10	86.8±0.3	87.4±0.1	85.9±0.1	86.7
	CLAN [272]	10	94.9±0.1	95.2±0.0	93.7±0.1	94.6
Multi-source	DCTN [278]	10	93.5±0.1	93.9±0.0	81.3±0.2	89.6
	M3SDA [259]	10	96.1±0.0	95.4±0.0	86.3±0.1	92.6
	MDDA [273]	10	97.5±0.2	96.9±0.2	88.2±0.1	94.5
	FTD-MSDA(Ours with L1)	10	96.2±0.1	96.9±0.1	87.6±0.2	93.6
	FTD-MSDA(Ours)	10	97.9±0.4	97.3±0.1	89.9±0.4	95.03
	Weighted FTD-MSDA(Ours)	10	**99.1**±0.0	**98.8**±0.0	**93.9**±0.1	**97.3**

Table 5.3.2 Class-wise accuracies (%) for CWRU dataset (B, C, D → A transfer scenario).

Scenario	Class	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
B, C, D → A	Without DA	80.3	87.1	83.9	86.5	85.5	61.7	86.2	78.9	83.1	82.6
	FTD-MSDA	100.0	100.0	99.1	100.0	100.0	0.0	100.0	94.5	100.0	100.0
	Weighted FTD-MSDA	100.0	95.4	100.0	100.0	98.2	55.5	98.2	96.4	100.0	100.0
	Improvement %	**19.7**	**8.3**	**16.1**	**13.5**	**12.7**	-6.2	**12.0**	**17.5**	**16.9**	**17.4**

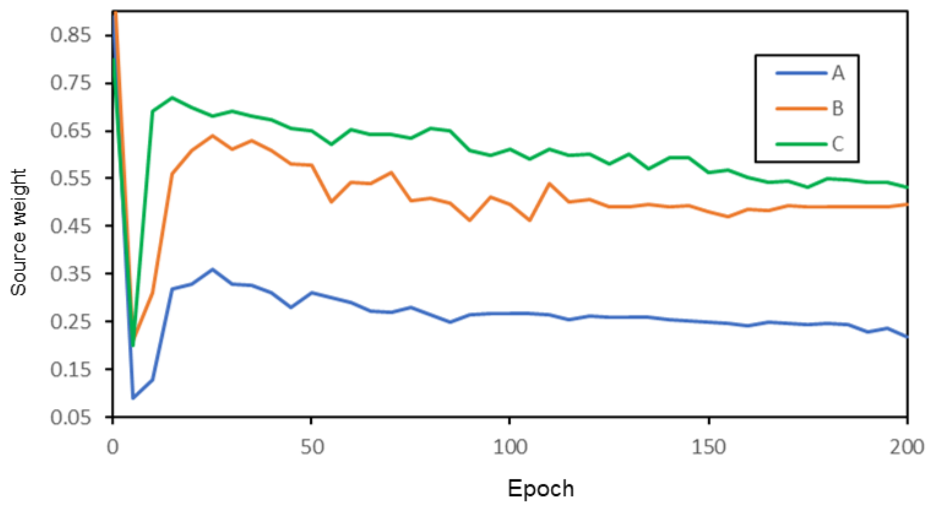


Figure 5.3.2 Source domain weight versus training epoch.

5.4 SUMMARY

In this chapter, we added a novel class-conditional weighting mechanism to our proposed FTD-MSDA framework. The weight is defined as a function of the maximum mean discrepancy between the conditional probability distribution of the source domain and target domain. The proposed weighting scheme also adjusts intra-class feature distribution that reduces the effect of class-wise negative transfer.

The model is verified on CWRU dataset. Results are reported into single-source best and multi-source standards. The former uses existing single-source domain adaptation methods on our experiment datasets and reports the best source transfer accuracy on the target set. The latter compares the results with existing state-of-the-art MSDA algorithms. Our experimental results show that the proposed weighting scheme enhances the performance of the model, and outperforms the originally proposed model and the state-of-the-art domain adaptation algorithms. Moreover, for "$\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \rightarrow \boldsymbol{A}$" transfer task, the effect of the negative transfer is considerably reduced.

CHAPTER 6: CONCLUSIONS AND RECOMMENDATION FOR FUTURE WORK

In this research, a detailed systematic review has been carried out on various aspects of employing deep neural networks in the context of fault detection, diagnostics, and prognostics. To the best of our knowledge, this is the first systematic review of deep learning within the PHM paradigm that categorized the models into three classes of discriminative, generative, and hybrid to reflect the role of generative models in solving many existing challenges. All previous surveys just briefly mentioned GANs, transfer learning, and domain adaptation as new models and techniques in the context of deep learning for future opportunities. Besides, deep learning is a fast-growing area and new algorithms and solutions are rapidly proposed and applied by researchers across the world. Moreover, this study discusses the available datasets, hardware, platforms and libraries, and cloud solutions. Therefore, it can be used as a comprehensive reference for deep learning practitioners in PHM.

Even though DL algorithms have brought new perspectives to data-driven methods in terms of model performance, learning complex representations, big data analysis, and handling the raw data with minimum preprocessing efforts. However, it was seen that DL in the PHM field is still in the infancy stage, and it brings many opportunities and future research direction in terms of data scarcity, explainability, interpretability, model selection, benchmarking, and so on.

We carried out an extensive empirical study on existing deep learning models for bearing fault diagnosis tasks. We explored various hyperparameter tunning techniques and the impact of the choices on deep learning model performance. The comparisons were made based on classification accuracy, and training time. It was seen that CNNs provide the best accuracy in the least training time. The RNNs achieve a performance comparable to the CNNs. However, they are slow and time-consuming to be trained. To provide a better grasp and understanding in terms of model selection, it is recommended to extend the empirical studies on traditional machine learning fault classification methods such as Logistic regression, SVM, Decision trees, and ensemble methods. Also, it is worth exploring to see how signal processing and feature engineering techniques may improve the performance of the models.

In literature, the majority of related studies are based upon the assumption that the training and test data are drawn from the same distributions. However, in the real industry, the data are collected under different operating and environmental conditions and during different time intervals, often resulting in feature space difference or distribution shift across training (source) and testing datasets (target) aka Domain shift issue. In this doctoral research, we proposed a novel deep multi-source domain adaptation framework for fault diagnosis of rotary machinery, which aligns the domains in both feature-level and task-level. The proposed model achieves ~ 4.5% average accuracy higher than baseline multisource domain adaptation models. Also, it enhances the average classification accuracy by ~ 10 % compared to baseline single source domain adaptation models.

Moreover, we added a novel source domain weighting scheme to the proposed model which is based on class-conditional distribution alignment of each source domain and target domain. Higher is the divergence between a certain source and target, that source receives the lower weight. The proposed scheme reduces the negative transfer phenomenon by matching the distribution of inter-class features. The proposed weighting scheme improves the average accuracy by ~2.2 (as compared to the originally proposed model). Hence, in total the proposed weighted FTD-MSDA model enhances the average classification accuracy by ~6.7% (over baseline multi-source domain adaptation networks).

It is recommended to extend the proposed model for the open-set domain adaptation task in which the target domain has private label sets besides the identical (shared) label sets. It would be also interesting to perform an ablation study to investigate the effectiveness of various parameters and loss terms on the convergence and performance of the proposed MSDA network.

The results of this doctoral research have been published in the following references:

- Rezaeianjouybari, Behnoush, and Yi Shang. "Deep learning for prognostics and health management: State of the art, challenges, and opportunities." Measurement 163 (2020): 107929.

- Rezaeianjouybari, Behnoush, and Yi Shang. "A Novel Deep Multi-Source Domain Adaptation Framework for Bearing Fault Diagnosis Based on Feature-level and Task-specific Distribution Alignment." Measurement (2021): 109359.

- Rezaeianjouybari, Behnoush, and Yi Shang.” An Empirical Study of Machine learning and Deep Learning Algorithms on Bearing Fault diagnosis Benchmarks.” Under review for IMECE2021.

REFERENCES

[1] V. Sugumaran, G. R. Sabareesh, and K. I. Ramachandran, "Fault diagnostics of roller bearing using kernel based neighborhood score multi-class support vector machine," *Expert Syst. Appl.*, vol. 34, no. 4, pp. 3090–3098, 2008.

[2] D. Cabrera *et al.*, "Fault diagnosis of spur gearbox based on random forest and wavelet packet decomposition," *Front. Mech. Eng.*, vol. 10, no. 3, pp. 277–286, 2015.

[3] T. Wang, H. Xu, J. Han, E. Elbouchikhi, and M. E. H. Benbouzid, "Cascaded H-bridge multilevel inverter system fault diagnosis using a PCA and multiclass relevance vector machine approach," *IEEE Trans. Power Electron.*, vol. 30, no. 12, pp. 7006–7018, 2015.

[4] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Péra, and N. Zerhouni, "Particle filter-based prognostics: Review, discussion and perspectives," *Mech. Syst. Signal Process.*, vol. 72, pp. 2–31, 2016.

[5] A. Soualhi, G. Clerc, H. Razik, and F. Guillet, "Hidden Markov models for the prediction of impending faults," *IEEE Trans. Ind. Electron.*, vol. 63, no. 5, pp. 3271–3281, 2016.

[6] N. J. Van Eck and L. Waltman, "VOSviewer manual," *Leiden: Univeristeit Leiden*, vol. 1, no. 1, pp. 1–53, 2013.

[7] R. Yan, R. X. Gao, and X. Chen, "Wavelets for fault diagnosis of rotary machines : A review with applications," *Signal Processing*, vol. 96, pp. 1–15, 2014.

[8] Z. Feng, M. Liang, and F. Chu, "Recent advances in time–frequency analysis

methods for machinery fault diagnosis: A review with application examples," *Mech. Syst. Signal Process.*, vol. 38, no. 1, pp. 165–205, 2013.

[9] Y. Wang, J. Xiang, R. Markert, and M. Liang, "Spectral kurtosis for fault detection, diagnosis and prognostics of rotating machines: A review with applications," *Mech. Syst. Signal Process.*, vol. 66, pp. 679–698, 2016.

[10] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2015, pp. 1200–1205.

[11] R. Zimroz and A. Bartkowiak, "Two simple multivariate procedures for monitoring planetary gearboxes in non-stationary operating conditions," *Mech. Syst. Signal Process.*, vol. 38, no. 1, pp. 237–247, 2013.

[12] G. Cheng, X. Chen, H. Li, P. Li, and H. Liu, "Study on planetary gear fault diagnosis based on entropy feature fusion of ensemble empirical mode decomposition," *Measurement*, vol. 91, pp. 140–154, 2016.

[13] W. D. Fisher, T. K. Camp, and V. V Krzhizhanovskaya, "Anomaly detection in earth dam and levee passive seismic data using support vector machines and automatic feature selection," *J. Comput. Sci.*, vol. 20, pp. 143–153, 2017.

[14] R. Moghaddass and S. Sheng, "An anomaly detection framework for dynamic systems using a Bayesian hierarchical framework," *Appl. Energy*, vol. 240, pp. 561–582, 2019.

[15] S. Lee, J.-W. Park, D.-S. Kim, I. Jeon, and D.-C. Baek, "Anomaly detection of tripod shafts using modified Mahalanobis distance," *J. Mech. Sci. Technol.*, vol.

32, no. 6, pp. 2473–2478, 2018.

[16] M. S. Safizadeh and S. K. Latifi, “Using multi-sensor data fusion for vibration fault diagnosis of rolling element bearings by accelerometer and load cell,” *Inf. Fusion*, vol. 18, pp. 1–8, 2014.

[17] R. K. Singleton, E. G. Strangas, and S. Aviyente, “Extended Kalman filtering for remaining-useful-life estimation of bearings,” *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1781–1790, 2014.

[18] F. Di Maio, K. L. Tsui, and E. Zio, “Combining relevance vector machines and exponential regression for bearing residual life estimation,” *Mech. Syst. Signal Process.*, vol. 31, pp. 405–427, 2012.

[19] G. Niu, *Data-Driven Technology for Engineering Systems Health Management*. Springer, 2017.

[20] N. Aissani, B. Beldjilali, and D. Trentesaux, “Dynamic scheduling of maintenance tasks in the petroleum industry: A reinforcement approach,” *Eng. Appl. Artif. Intell.*, vol. 22, no. 7, pp. 1089–1103, 2009.

[21] S. Wu, N. Gebraeel, M. A. Lawley, and Y. Yih, “A neural network integrated decision support system for condition-based optimal predictive maintenance policy,” *IEEE Trans. Syst. Man, Cybern. A Syst. Humans*, vol. 37, no. 2, pp. 226–236, 2007.

[22] G. K. Chan and S. Asgarpoor, “Optimum maintenance policy with Markov processes,” *Electr. power Syst. Res.*, vol. 76, no. 6–7, pp. 452–456, 2006.

[23] J. Ben Ali, N. Fnaiech, L. Saidi, B. Chebel-Morello, and F. Fnaiech, “Application of empirical mode decomposition and artificial neural network for automatic

bearing fault diagnosis based on vibration signals," *Appl. Acoust.*, vol. 89, pp. 16–27, 2015.

[24] H. Shao, J. Hongkai, L. Ying, and L. Xingqiu, "A novel method for intelligent fault diagnosis of rolling bearings using ensemble deep auto-encoders," *Mech. Syst. Signal Process.*, vol. 102, pp. 278–297, 2018.

[25] X. Lou and K. A. Loparo, "Bearing fault diagnosis based on wavelet transform and fuzzy inference," *Mech. Syst. Signal Process.*, vol. 18, no. 5, pp. 1077–1095, 2004.

[26] L. Batista, B. Badri, R. Sabourin, and M. Thomas, "A classifier fusion system for bearing fault diagnosis," *Expert Syst. Appl.*, vol. 40, no. 17, pp. 6788–6797, 2013.

[27] J. Zhu, N. Chen, and W. Peng, "Estimation of Bearing Remaining Useful Life Based on Multiscale Convolutional Neural Network," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 190–193, 2018.

[28] J. Deutsch and D. He, "Using Deep Learning-Based Approach to Predict Remaining Useful Life of Rotating Components," *IEEE Trans. NEURAL NETWORKS Learn. Syst.*, vol. 48, no. 1, pp. 11–20, 2018.

[29] A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, "Metrics for offline evaluation of prognostic performance," *Int. J. Progn. Heal. Manag.*, vol. 1, no. 1, pp. 4–23, 2010.

[30] N. Chen and K. L. Tsui, "Condition monitoring and remaining useful life prediction using degradation signals: Revisited," *IiE Trans.*, vol. 45, no. 9, pp. 939–952, 2013.

[31] K. T. P. Nguyen, M. Fouladirad, and A. Grall, "New methodology for improving the inspection policies for degradation model selection according to prognostic

measures," *IEEE Trans. Reliab.*, vol. 67, no. 3, pp. 1269–1280, 2018.

[32] P. Nectoux *et al.*, "PRONOSTIA : An experimental platform for bearings accelerated degradation tests .," in *IEEEInternational Conference on Prognostics and Health Management*, 2012, pp. 1–8.

[33] R. Zemouri and R. Gouriveau, "Towards accurate and reproducible predictions for prognostic: an approach combining a RRBF network and an autoregressive model," *IFAC Proc. Vol.*, vol. 43, no. 3, pp. 140–145, 2010.

[34] K. Javed, R. Gouriveau, and N. Zerhouni, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2626–2639, 2015.

[35] Y. Hu, P. Baraldi, F. Di Maio, and E. Zio, "Online performance assessment method for a model-based prognostic approach," *IEEE Trans. Reliab.*, vol. 65, no. 2, pp. 718–735, 2015.

[36] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: a benchmark data set for data-driven classification," *Third Eur. Conf. Progn. Heal. Manag. Soc. 2016*, no. Cm, pp. 152–156, 2016.

[37] "Case Western Reserve University Bearing vibration Data," 2015. [Online]. Available: http://csegroups.case.edu/bearingdatacenter/pages/12k-drive-end-bearing-fault-data.

[38] J. Lee, H. Qiu, G. Yu, and J. Lin, "Rexnord technical services: Bearing data set," *Moffett Field, CA IMS, Univ. Cincinnati. NASA Ames Progn. Data Repos. NASA Ames*, 2007.

[39] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," *2008 Int. Conf. Progn. Heal. Manag. PHM 2008*, 2008.

[40] B. Saha and K. Goebel, "Uncertainty management for diagnostics and prognostics of batteries using Bayesian techniques," in *2008 IEEE Aerospace Conference*, 2008, pp. 1–8.

[41] E. F. Hogge *et al.*, "Verification of a remaining flying time prediction system for small electric aircraft," in *Annual Conference of the Prognostics and Health Management, PHM 2015*, 2015.

[42] B. Bole, C. S. Kulkarni, and M. Daigle, "Adaptation of an electrochemistry-based li-ion battery model to account for deterioration observed under randomized use," in *Annual Conference of the Prognostics and Health Management, PHM 2014*, 2014.

[43] W. Xiao, "A probabilistic machine learning approach to detect industrial plant faults," *arXiv Prepr. arXiv1603.05770*, 2016.

[44] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[45] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[46] H. Larochelle, M. Mandel, and Y. Bengio, "Learning Algorithms for the Classification Restricted Boltzmann Machine," vol. 13, pp. 643–669, 2012.

[47] A. Miguel and G. E. Hinton, "On Contrastive Divergence Learning," vol. 0.

[48] G. Hinton and G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," 2010.

[49] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy Layer-Wise

Training of Deep Networks,” *Adv. Neural Inf. Process. Syst.*, no. 1, 2007.

[50] J. Xu, H. Li, S. Zhou, J. Xu, H. Li, and S. Zhou, “An Overview of Deep Generative Models An Overview of Deep Generative Models,” no. December, pp. 37–41, 2014.

[51] R. Salakhutdinov and G. Hinton, “Deep Boltzmann Machines,” no. 3, pp. 448–455, 2009.

[52] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Neurocomputing Deep learning for visual understanding : A review,” vol. 187, pp. 27–48, 2016.

[53] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning : A Review and New Perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[54] R. Salakhutdinov and H. Larochelle, “Efficient learning of deep Boltzmann machines,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 693–700.

[55] I. Goodfellow, M. Mirza, A. Courville, and Y. Bengio, “Multi-prediction deep Boltzmann machines,” in *Advances in Neural Information Processing Systems*, 2013, pp. 548–556.

[56] A. Ng, “CS294A Lecture notes Sparse autoencoder,” pp. 1–19.

[57] M. Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, “Efficient learning of sparse representations with an energy-based model,” in *Advances in neural information processing systems*, 2007, pp. 1137–1144.

[58] M. Shu and A. Fyshe, “Sparse autoencoders for word decoding from magnetoencephalography,” in *Proceedings of the third NIPS Workshop on*

Machine Learning and Interpretation in NeuroImaging (MLINI), 2013.

[59] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[60] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, “Contractive auto-encoders: Explicit invariance during feature extraction,” in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 2011, pp. 833–840.

[61] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv Prepr. arXiv1312.6114*, 2013.

[62] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[63] D. I. J. Im, S. Ahn, R. Memisevic, and Y. Bengio, “Denoising criterion for variational auto-encoding framework,” in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[64] B. R. Kiran, D. M. Thomas, and R. Parakkal, “An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos,” *J. Imaging*, vol. 4, no. 2, pp. 1–15, 2018.

[65] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “Neurocomputing A survey of deep neural network architectures and their applications ☆,” *Neurocomputing*, vol. 234, no. October 2016, pp. 11–26, 2017.

[66] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv Prepr. arXiv1408.5882*, 2014.

[67] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the

inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[68] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[69] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv Prepr. arXiv1409.1556*, 2014.

[70] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 6, no. 02, pp. 107–116, 1998.

[71] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[72] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv Prepr. arXiv1705.07215*, 2017.

[73] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," *arXiv Prepr. arXiv1701.00160*, 2016.

[74] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually Converge?," *arXiv Prepr. arXiv1801.04406*, 2018.

[75] J. Li, A. Madry, J. Peebles, and L. Schmidt, "Towards understanding the dynamics of generative adversarial networks," *arXiv Prepr. arXiv1706.09884*, 2017.

[76] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223.

[77] B. R. Jouibary, K. G. Osgouie, and A. Meghdari, "Employing Neural Networks for Manipulability Optimization of the Dual-Arm Cam-Lock Robot," in *ASME 2010*

International Mechanical Engineering Congress and Exposition, 2010, pp. 587–595.

[78] Z. Wang, Q. She, and T. E. Ward, "Generative Adversarial Networks: A Survey and Taxonomy," *arXiv Prepr. arXiv1906.01529*, no. 2, pp. 1–16, 2019.

[79] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv Prepr. arXiv1609.04747*, 2016.

[80] T. Jebara, *Machine learning: discriminative and generative*, vol. 755. Springer Science & Business Media, 2012.

[81] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Trans. Signal Inf. Process.*, vol. 3, 2014.

[82] P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification," *Reliab. Eng. Syst. Saf.*, vol. 115, pp. 124–135, 2013.

[83] V. T. Tran, F. Althobiani, and A. Ball, "An approach to fault diagnosis of reciprocating compressor valves using Teager – Kaiser energy operator and deep belief networks," *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4113–4122, 2014.

[84] H. Shao, H. Jiang, X. Zhang, and M. Niu, "Rolling bearing fault diagnosis using an optimization deep belief network," *Meas. Sci. Technol.*, vol. 26, no. 11, 2015.

[85] S. Tang, C. Shen, D. Wang, S. Li, W. Huang, and Z. Zhu, "Adaptive deep feature learning network with Nesterov momentum and its application to rotating machinery fault diagnosis," *Neurocomputing*, vol. 305, pp. 1–14, 2018.

[86] N. Yuan, W. Yang, B. Kang, S. Xu, and C. Li, "Signal fusion-based deep fast random forest method for machine health assessment," *J. Manuf. Syst.*, vol. 48, no. February, pp. 1–8, 2018.

[87] J. Liang, Y. Zhang, J. Zhong, and H. Yang, “A novel multi-segment feature fusion based fault classification approach for rotating machinery,” *Mech. Syst. Signal Process.*, vol. 122, pp. 19–41, 2019.

[88] H. Oh, J. H. Jung, B. C. Jeon, and B. D. Youn, “Scalable and Unsupervised Feature Engineering Using Vibration-Imaging and Deep Learning for Rotor System Diagnosis,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 4, pp. 3539–3549, 2018.

[89] Y. Qin, X. Wang, and J. Zou, “The optimized deep belief networks with improved logistic Sigmoid units and their application in fault diagnosis for planetary gearboxes of wind turbines,” *IEEE Trans. Ind. Electron.*, vol. PP, no. CD, pp. 1–1, 2018.

[90] X. Zhao and M. Jia, “A novel deep fuzzy clustering neural network model and its application in rolling bearing fault recognition,” *Meas. Sci. Technol.*, vol. 29, no. 12, p. 125005, 2018.

[91] Q. Tang, Y. Chai, J. Qu, and H. Ren, “Fisher discriminative sparse representation based on DBN for fault diagnosis of complex system,” *Appl. Sci.*, vol. 8, no. 5, 2018.

[92] Z. Gao, C. Ma, D. Song, and Y. Liu, “Deep quantum inspired neural network with application to aircraft fuel system fault diagnosis,” *Neurocomputing*, vol. 238, pp. 13–23, 2017.

[93] J. He, S. Yang, and C. Gan, “Unsupervised fault diagnosis of a gear transmission chain using a deep belief network,” *Sensors*, vol. 17, no. 7, pp. 1–21, 2017.

[94] Z. Liu, Z. Jia, C. M. Vong, S. Bu, J. Han, and X. Tang, “Capturing High-

Discriminative Fault Features for Electronics-Rich Analog System via Deep Learning," *IEEE Trans. Ind. Informatics*, vol. 13, no. 3, pp. 1213–1226, 2017.

[95] J. Xie, G. Du, C. Shen, N. Chen, L. Chen, and Z. Zhu, "An End-to-End Model Based on Improved Adaptive Deep Belief Network and Its Application to Bearing Fault Diagnosis," *IEEE Access*, vol. 6, pp. 63584–63596, 2018.

[96] G. Zhao, X. Liu, B. Zhang, Y. Liu, G. Niu, and C. Hu, "A novel approach for analog circuit fault diagnosis based on Deep Belief Network," *Measurement*, vol. 121, no. August 2017, pp. 170–178, 2018.

[97] C. Li, R. Sanchez, G. Zurita, M. Cerrada, D. Cabrera, and R. E. Vásquez, "Multimodal deep support vector classification with homologous features and its application to gearbox fault diagnosis," *Neurocomputing*, vol. 168, pp. 119–127, 2015.

[98] C. Li, R. V. Sanchez, G. Zurita, M. Cerrada, D. Cabrera, and R. E. Vásquez, "Gearbox fault diagnosis based on deep random forest fusion of acoustic and vibratory signals," *Mech. Syst. Signal Process.*, vol. 76–77, pp. 283–293, 2016.

[99] G. Hu, H. Li, Y. Xia, and L. Luo, "A deep Boltzmann machine and multi-grained scanning forest ensemble collaborative method and its application to industrial fault diagnosis," *Comput. Ind.*, vol. 100, pp. 287–296, 2018.

[100] J. Wang, K. Wang, Y. Wang, Z. Huang, and R. Xue, "Deep Boltzmann machine based condition prediction for smart manufacturing," *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 3, pp. 851–861, 2019.

[101] F. Zhou, Y. Gao, and C. Wen, "A Novel Multimode Fault Classification Method Based on Deep Learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes*

Artif. Intell. Lect. Notes Bioinformatics), pp. 442–452, 2017.

[102] H. Shao, J. Hongkai, Z. Huiwei, and W. Fuan, “A novel deep autoencoder feature learning method for rotating machinery fault diagnosis,” *Mech. Syst. Signal Process.*, vol. 95, pp. 187–204, 2017.

[103] Z. Meng, X. Zhan, J. Li, and Z. Pan, “An enhancement denoising autoencoder for rolling bearing fault diagnosis,” *Measurement*, vol. 130, pp. 448–454, 2018.

[104] G. Jiang, P. Xie, H. He, and J. Yan, “Wind Turbine Fault Detection Using a Denoising Autoencoder with Temporal Information,” *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 1, pp. 89–100, 2018.

[105] B. Luo, H. Wang, H. Liu, B. Li, and F. Peng, “Early Fault Detection of Machine Tools Based on Deep Learning and Dynamic Identification,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 509–518, 2019.

[106] S. Zhang, M. Wang, W. Li, J. Luo, and Z. Lin, “Deep learning with emerging new labels for fault diagnosis,” *IEEE Access*, vol. 7, pp. 1–1, 2018.

[107] J. Liu, Y. Hu, Y. Wang, B. Wu, J. Fan, and Z. Hu, “An integrated multi-sensor fusion-based deep feature learning approach for rotating machinery diagnosis,” *Meas. Sci. Technol.*, vol. 29, no. 5, 2018.

[108] J. Wang, S. Li, Z. An, X. Jiang, W. Qian, and S. Ji, “Batch-normalized deep neural networks for achieving fast intelligent fault diagnosis of machines,” *Neurocomputing*, vol. 329, pp. 53–65, 2019.

[109] J. Sun, C. Yan, and J. Wen, “Intelligent Bearing Fault Diagnosis Method Combining Compressed Data Acquisition and Deep Learning,” *IEEE Trans. Instrum. Meas.*, vol. 67, no. 1, pp. 185–195, 2018.

[110] J. Yu, “A selective deep stacked denoising autoencoders ensemble with negative correlation learning for gearbox fault diagnosis,” *Comput. Ind.*, vol. 108, pp. 62–72, 2019.

[111] G. Jiang, H. He, P. Xie, and Y. Tang, “Stacked Multilevel-Denoising Autoencoders : A New Representation Learning Approach for Wind Turbine Gearbox Fault Diagnosis,” *Ieee Trans. Instrum. Meas.*, vol. 66, no. 9, pp. 2391–2402, 2017.

[112] C. Lu, Z. Wang, W. Qin, and J. Ma, “Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identi fi cation,” *Signal Processing*, vol. 130, pp. 377–388, 2017.

[113] X. Guo, C. Shen, and L. Chen, “Deep Fault Recognizer : An Integrated Model to Denoise and Extract Features for Fault Diagnosis in Rotating Machinery,” *Appl. Sci.*, 2017.

[114] C. Shi, G. Panoutsos, B. Luo, H. Liu, B. Li, and X. Lin, “Using multiple feature spaces-based deep learning for tool condition monitoring in ultra-precision manufacturing,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 5, pp. 3794–3803, 2019.

[115] Z. Zhang, S. Li, Y. Xiao, and Y. Yang, “Intelligent simultaneous fault diagnosis for solid oxide fuel cell system based on deep learning,” *Appl. Energy*, vol. 233–234, no. October 2018, pp. 930–942, 2019.

[116] C. Shen, Y. Qi, J. Wang, G. Cai, and Z. Zhu, “An automatic and robust features learning method for rotating machinery fault diagnosis based on contractive autoencoder,” *Eng. Appl. Artif. Intell.*, vol. 76, no. 8, pp. 170–184, 2018.

[117] H. Shao, H. Jiang, F. Wang, and H. Zhao, “An enhancement deep feature fusion

method for rotating machinery fault diagnosis," *Knowledge-Based Syst.*, vol. 119, pp. 200–220, 2017.

[118] W. Jiang, J. Zhou, H. Liu, and Y. Shan, "A multi-step progressive fault diagnosis method for rolling element bearing based on energy entropy theory and hybrid ensemble auto-encoder," *ISA Trans.*, vol. 87, pp. 235–250, 2018.

[119] G. Ping, J. Chen, T. Pan, and J. Pan, "Degradation feature extraction using multi-source monitoring data via logarithmic normal distribution based variational auto-encoder," *Comput. Ind.*, vol. 109, pp. 72–82, 2019.

[120] G. San Martin, E. López Droguett, V. Meruane, and M. das Chagas Moura, "Deep variational auto-encoders: A promising tool for dimensionality reduction and ball bearing elements fault diagnosis," *Struct. Heal. Monit.*, 2018.

[121] K. Zhang, B. Tang, Y. Qin, and L. Deng, "Fault diagnosis of planetary gearbox using a novel semi-supervised method of multiple association layers networks," *Mech. Syst. Signal Process.*, vol. 131, pp. 243–260, 2019.

[122] Y. ren Wang, Q. Jin, G. dong Sun, and C. fei Sun, "Planetary gearbox fault feature learning using conditional variational neural networks under noise environment," *Knowledge-Based Syst.*, vol. 163, pp. 438–449, 2019.

[123] A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera, "Handling incomplete heterogeneous data using VAEs," *arXiv Prepr. arXiv1807.03653*, 2018.

[124] Z. Chen, R. V. Sánchez, and C. Li, "Gearbox fault identification and classification with convolutional neural networks," *Shock Vib.*, vol. 2015, no. 1, pp. 1–18, 2015.

[125] L. Guo, Y. Lei, N. Li, T. Yan, and N. Li, "Machinery health indicator construction based on convolutional neural networks considering trend burr," *Neurocomputing,*

vol. 292, pp. 142–150, 2018.

[126] D. Belmiloud, T. Benkedjouh, M. Lachi, A. Laggoun, and J. P. Dron, "Deep convolutional neural networks for Bearings failure prediction and temperature correlation," *J. Vibroengineering*, vol. 20, no. 8, pp. 2878–2891, 2018.

[127] L. Eren, T. Ince, and S. Kiranyaz, "A Generic Intelligent Bearing Fault Diagnosis System Using Compact Adaptive 1D CNN Classifier," *J. Signal Process. Syst.*, pp. 1–11, 2018.

[128] J. Pan, Y. Zi, J. Chen, Z. Zhou, and B. Wang, "LiftingNet: A Novel Deep Learning Network with Layerwise Feature Learning from Noisy Mechanical Data for Fault Classification," *IEEE Trans. Ind. Electron.*, vol. 65, no. 6, pp. 4973–4982, 2018.

[129] H. Jiang, F. Wang, H. Shao, and H. Zhang, "Rolling bearing fault identification using multilayer deep learning convolutional neural network," *J. Vibroengineering*, pp. 138–149, 2017.

[130] Y. Chen, G. Peng, C. Xie, W. Zhang, C. Li, and S. Liu, "ACDIN: Bridging the gap between artificial and real bearing damages for bearing fault diagnosis," *Neurocomputing*, vol. 294, pp. 61–71, 2018.

[131] L. Eren, "Bearing fault detection by one-dimensional convolutional neural networks," *Math. Probl. Eng.*, vol. 2017, 2017.

[132] W. Zhang, G. Peng, C. Li, Y. Chen, and Z. Zhang, "A New Deep Learning Model for Fault Diagnosis with Good Anti-Noise and Domain Adaptation," *Sensors*, 2017.

[133] L. Jing, T. Wang, M. Zhao, and P. Wang, "An Adaptive Multi-Sensor Data Fusion Method Based on Deep Convolutional Neural Networks for," *Sensors*, 2017.

[134] Y. Yao *et al.*, “End-To-End Convolutional Neural Network Model for Gear Fault Diagnosis Based on Sound Signals,” *Appl. Sci.*, vol. 8, no. 9, p. 1584, 2018.

[135] M. Xia, T. Li, L. Xu, L. Liu, and C. W. De Silva, “Fault Diagnosis for Rotating Machinery Using Multiple Sensors and Convolutional Neural Networks,” *IEEE/ASME Trans. Mechatronics*, vol. 4435, no. c, pp. 1–9, 2017.

[136] W. Zhang, X. Li, and Q. Ding, “Deep residual learning-based fault diagnosis method for rotating machinery,” *ISA Trans.*, 2018.

[137] Y. Han, B. Tang, and L. Deng, “Multi-level wavelet packet fusion in dynamic ensemble convolutional neural network for fault diagnosis,” *Measurement*, vol. 127, no. February, pp. 246–255, 2018.

[138] D. Verstraete, A. Ferrada, E. L. Droguett, V. Meruane, and M. Modarres, “Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings,” *Shock Vib.*, vol. 2017, 2017.

[139] Y. Yoo and J.-G. Baek, “A Novel Image Feature for the Remaining Useful Lifetime Prediction of Bearings Based on Continuous Wavelet Transform and Convolutional Neural Network,” *Appl. Sci.*, vol. 8, no. 7, p. 1102, 2018.

[140] Z. Zhu, G. Peng, Y. Chen, and H. Gao, “A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis,” *Neurocomputing*, vol. 323, pp. 62–75, 2019.

[141] P. Wang, Ananya, R. Yan, and R. X. Gao, “Virtualization and deep recognition for system fault classification,” *J. Manuf. Syst.*, vol. 44, pp. 310–316, 2017.

[142] X. Li, W. Zhang, and Q. Ding, “Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction,” *Reliab. Eng. Syst. Saf.*,

vol. 182, no. July 2018, pp. 208–218, 2019.

[143] X. Ding and Q. He, “Energy-Fluctuated Multiscale Feature Learning With Deep ConvNet for Intelligent,” *IEEE Trans. Instrum. Meas.*, vol. 66, no. 8, pp. 1926–1935, 2017.

[144] L. Ren, Y. Sun, H. Wang, and L. Zhang, “Prediction of bearing remaining useful life with deep convolution neural network,” *IEEE Access*, vol. 6, pp. 13041–13049, 2018.

[145] D. T. Hoang and H. J. Kang, “Rolling element bearing fault diagnosis using convolutional neural network and vibration image,” *Cogn. Syst. Res.*, vol. 53, pp. 42–50, 2019.

[146] L. Wen, X. Li, L. Gao, and Y. Zhang, “A New Convolutional Neural Network-Based Data-Driven Fault Diagnosis Method,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5990–5998, 2018.

[147] C. Lu, Z. Wang, and B. Zhou, “Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification,” *Adv. Eng. Informatics*, vol. 32, pp. 139–151, 2017.

[148] Z.-X. Hu, Y. Wang, M.-F. Ge, and J. Liu, “Data-driven Fault Diagnosis Method based on Compressed Sensing and Improved Multi-scale Network,” *IEEE Trans. Ind. Electron.*, vol. PP, no. 1, pp. 1–1, 2019.

[149] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in *Advances in neural information processing systems*, 2017, pp. 3856–3866.

[150] Y. LeCun, “LeNet-5, convolutional neural networks,” *URL http//yann. lecun. com/exdb/lenet*, vol. 20, p. 5, 2015.

[151] T. Han, C. Liu, L. Wu, S. Sarkar, and D. Jiang, “An adaptive spatiotemporal feature learning approach for fault diagnosis in complex systems,” *Mech. Syst. Signal Process.*, vol. 117, pp. 170–187, 2019.

[152] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, “A recurrent neural network based health indicator for remaining useful life prediction of bearings,” *Neurocomputing*, vol. 240, pp. 98–109, 2017.

[153] W. Peng, Z.-S. Ye, and N. Chen, “Bayesian Deep Learning based Health Prognostics Towards Prognostics Uncertainty,” *IEEE Trans. Ind. Electron.*, vol. PP, no. c, pp. 1–1, 2019.

[154] R. Ma, T. Yang, E. Breaz, Z. Li, P. Briois, and F. Gao, “Data-driven proton exchange membrane fuel cell degradation predication through deep learning method,” *Appl. Energy*, vol. 231, no. March, pp. 102–115, 2018.

[155] Y. Zhang, R. Xiong, H. He, and M. G. Pecht, “Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5695–5705, 2018.

[156] Y. Wu, M. Yuan, S. Dong, L. Lin, and Y. Liu, “Remaining useful life estimation of engineered systems using vanilla LSTM neural networks,” *Neurocomputing*, vol. 275, pp. 167–179, 2018.

[157] J. Wu, K. Hu, Y. Cheng, H. Zhu, X. Shao, and Y. Wang, “Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network,” *ISA Trans.*, 2019.

[158] J. Zhang, P. Wang, R. Yan, and R. X. Gao, “Long short-term memory for machine remaining life prediction,” *J. Manuf. Syst.*, vol. 48, pp. 78–86, 2018.

[159] C.-G. Huang, H.-Z. Huang, and Y.-F. Li, “A Bi-Directional LSTM prognostics method under multiple operational conditions,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 11, pp. 1–1, 2019.

[160] A. Elsheikh, S. Yacout, and M. S. Ouali, “Bidirectional handshaking LSTM for remaining useful life prediction,” *Neurocomputing*, vol. 323, pp. 148–156, 2019.

[161] R. Y. Rui Zhao, Dongzhe Wang, “Machine Health Monitoring Using Local Feature-Based Gated Recurrent Unit Networks,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1539–1548, 2018.

[162] X. Li, H. Jiang, X. Xiong, and H. Shao, “Rolling bearing health prognosis using a modified health index based hierarchical gated recurrent unit network,” *Mech. Mach. Theory*, vol. 133, pp. 229–249, 2019.

[163] Q. Li, L. Chen, S. Changqing, and Y. Bingru, “Enhanced generative adversarial networks for fault diagnosis of rotating machinery with imbalanced data,” *Meas. Sci. Technol.*, vol. 30, no. 11, 2019.

[164] W. Jiang, C. Cheng, B. Zhou, G. Ma, and Y. Yuan, “A Novel GAN-based Fault Diagnosis Approach for Imbalanced Industrial Time Series,” *arXiv Prepr. arXiv1904.00575*, pp. 1–6, 2019.

[165] D. Zhao, F. Liu, and H. Meng, “Bearing Fault Diagnosis Based on the Switchable Normalization SSGAN with 1-D Representation of Vibration Signals as Input,” *Sensors*, vol. 19, no. 9, 2019.

[166] J. Wang, S. Li, B. Han, Z. An, H. Bao, and S. Ji, “Generalization of Deep Neural Networks for Imbalanced Fault Classification of Machinery Using Generative Adversarial Networks,” *IEEE Access*, vol. PP, pp. 1–1, 2019.

[167] D. Cabrera *et al.*, “Generative Adversarial Networks Selection Approach for Extremely Imbalanced Fault Diagnosis of Reciprocating Machinery,” *IEEE Access*, vol. 7, pp. 70643–70653, 2019.

[168] F. Zhou, S. Yang, H. Fujita, D. Chen, and C. Wen, “Deep learning fault diagnosis method based on global optimization GAN for unbalanced data,” *Knowledge-Based Syst.*, vol. 187, p. 104837, 2020.

[169] S. Shao, P. Wang, and R. Yan, “Generative adversarial networks for data augmentation in machine fault diagnosis,” *Comput. Ind.*, vol. 106, pp. 85–93, 2019.

[170] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, “Adversarial autoencoders,” *arXiv Prepr. arXiv1511.05644*, 2015.

[171] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, “Wasserstein auto-encoders,” *arXiv Prepr. arXiv1711.01558*, 2017.

[172] H. Liu, J. Zhou, Y. Xu, Y. Zheng, X. Peng, and W. Jiang, “Unsupervised fault diagnosis of rolling bearings using a deep neural network based on generative adversarial networks,” *Neurocomputing*, vol. 315, pp. 412–424, 2018.

[173] C. Cheng, B. Zhou, G. Ma, D. Wu, and Y. Yuan, “Wasserstein Distance based Deep Adversarial Transfer Learning for Intelligent Fault Diagnosis,” *arXiv Prepr. arXiv1903.06753*, 2019.

[174] T. Han, C. Liu, W. Yang, and D. Jiang, “A novel adversarial learning framework in deep convolutional neural network for intelligent diagnosis of mechanical faults,” *Knowledge-Based Syst.*, vol. 165, pp. 474–487, 2019.

[175] M. He and D. He, “Deep learning based approach for bearing fault diagnosis,”

IEEE Trans. Ind. Appl., vol. 53, no. 3, pp. 3057–3065, 2017.

[176] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 609–616.

[177] H. Shao, H. Jiang, H. Zhang, W. Duan, T. Liang, and S. Wu, “Rolling bearing fault feature learning using improved convolutional deep belief network with compressed sensing,” *Mech. Syst. Signal Process.*, vol. 100, pp. 743–765, 2018.

[178] D. Park, S. Kim, Y. An, and J. Y. Jung, “Lired: A light-weight real-time fault detection system for edge computing using LSTM recurrent neural networks,” *Sensors (Switzerland)*, vol. 18, no. 7, 2018.

[179] R. Zhao, R. Yan, J. Wang, and K. Mao, “Learning to monitor machine health with convolutional Bi-directional LSTM networks,” *Sensors*, vol. 17, no. 2, pp. 1–18, 2017.

[180] Z. Wu, Y. Guo, and W. Lin, “A Weighted Deep Representation Learning Model for Imbalanced Fault Diagnosis in Cyber-Physical Systems,” *Sensors*, 2018.

[181] H. Qiao, T. Wang, P. Wang, S. Qiao, and L. Zhang, “A time-distributed spatiotemporal feature learning method for machine health monitoring with multi-sensor time series,” *Sensors*, vol. 18, no. 9, 2018.

[182] P. Malhotra *et al.*, “Multi-Sensor Prognostics using an Unsupervised Health Index based on LSTM Encoder-Decoder,” *arXiv Prepr. arXiv1608.06154*, 2016.

[183] N. Gugulothu, V. TV, P. Malhotra, and G. S. Lovekesh Vig, Puneet Agarwal, “Predicting Remaining Useful Life using Time Series Embeddings based on

Recurrent Neural Networks," *Work. Proc. New Secur. Paradig.*, 2017.

[184] H. Liu, J. Zhou, Y. Zheng, W. Jiang, and Y. Zhang, "Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders," *ISA Trans.*, vol. 77, pp. 167–178, 2018.

[185] H. Shao, H. Jiang, H. Zhang, and T. Liang, "Electric locomotive bearing fault diagnosis using novel convolutional deep belief network," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2727–2736, 2018.

[186] A. S. Yoon *et al.*, "Semi-supervised Learning with Deep Generative Models for Asset Failure Prediction," *arXiv Prepr. arXiv1709.00845*, 2017.

[187] Z. Chen and W. Li, "Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 7, pp. 1693–1702, 2017.

[188] W. Lu, Y. Li, Y. Cheng, D. Meng, B. Liang, and P. Zhou, "Early Fault Detection Approach With Deep Architectures," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 7, pp. 1679–1689, 2018.

[189] A. Listou Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliab. Eng. Syst. Saf.*, vol. 183, no. June 2018, pp. 240–251, 2019.

[190] X. Li, W. Zhang, and Q. Ding, "Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 7, pp. 5525–5534, 2019.

[191] B. Zhang, W. Li, J. Hao, X.-L. Li, and M. Zhang, "Adversarial adaptive 1-D

convolutional neural networks for bearing fault diagnosis under varying working condition," *arXiv Prepr. arXiv1805.00778*, pp. 1–19, 2018.

[192] W. Yu, I. Y. Kim, and C. Mechefske, "Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme," *Mech. Syst. Signal Process.*, vol. 129, pp. 764–780, 2019.

[193] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

[194] "Model Zoo: Pre-trained networks." .

[195] M. Z. Alom *et al.*, "The history began from alexnet: A comprehensive survey on deep learning approaches," *arXiv Prepr. arXiv1803.01164*, 2018.

[196] L. Wen, X. Li, X. Li, and L. Gao, "A New Transfer Learning Based on VGG-19 Network for Fault Diagnosis," in *2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2019, pp. 205–209.

[197] S. Shao, S. McAleer, R. Yan, and P. Baldi, "Highly Accurate Machine Fault Diagnosis Using Deep Transfer Learning," *IEEE Trans. Ind. Informatics*, vol. 15, no. 4, pp. 2446–2455, 2019.

[198] P. Malhotra, V. TV, L. Vig, P. Agarwal, and G. Shroff, "TimeNet: Pre-trained deep recurrent neural network for time series classification. arXiv 2017," *arXiv Prepr. arXiv1706.08838*.

[199] K. Kashiparekh, J. Narwariya, P. Malhotra, L. Vig, and G. Shroff, "ConvTimeNet: A pre-trained deep convolutional neural network for time series classification," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.

[200] G. Xu, M. Liu, Z. Jiang, D. Söffker, and W. Shen, "Bearing fault diagnosis method based on deep convolutional neural network and random forest ensemble learning," *Sensors*, vol. 19, no. 5, p. 1088, 2019.

[201] P. Ma, H. Zhang, W. Fan, C. Wang, G. Wen, and X. Zhang, "A novel bearing fault diagnosis method based on 2D image representation and transfer learning-convolutional neural network," *Meas. Sci. Technol.*, vol. 30, no. 5, p. 55402, 2019.

[202] L. Wen, X. Li, and L. Gao, "A transfer convolutional neural network for fault diagnosis based on ResNet-50," *Neural Comput. Appl.*, pp. 1–14, 2019.

[203] J. Wang, Z. Mo, H. Zhang, and Q. Miao, "A deep learning method for bearing fault diagnosis based on time-frequency image," *IEEE Access*, vol. 7, pp. 42373–42383, 2019.

[204] W. Mao, L. Ding, S. Tian, and X. Liang, "Online detection for bearing incipient fault based on deep transfer learning," *Measurement*, vol. 152, p. 107278, 2020.

[205] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[206] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017, pp. 1–12.

[207] "The world's leading software development platform , GitHub." [Online]. Available: https://github.com/.

[208] Z. Wang, K. Liu, J. Li, Y. Zhu, and Y. Zhang, "Various Frameworks and Libraries of Machine Learning and Deep Learning: A Survey," *Arch. Comput. Methods*

Eng., pp. 1–24, 2019.

[209] J. Zacharias, M. Barz, and D. Sonntag, “A Survey on Deep Learning Toolkits and Libraries for Intelligent User Interfaces,” *arXiv Prepr. arXiv1803.04818*, 2018.

[210] P. Zheng *et al.*, “Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives,” *Front. Mech. Eng.*, vol. 13, no. 2, pp. 137–150, 2018.

[211] Q. Qi and F. Tao, “A Smart Manufacturing Service System Based on Edge Computing, Fog Computing, and Cloud Computing,” *IEEE Access*, vol. 7, pp. 86769–86777, 2019.

[212] J. Watkins, C. Teubert, and J. Ossenfort, “Prognostics As-A-Service: A Scalable Cloud Architecture for Prognostics,” in *11th Annual Conference Prognostics and Health Management Society*, 2019.

[213] L. Li, K. Ota, and M. Dong, “Deep learning for smart industry: Efficient manufacture inspection system with fog computing,” *IEEE Trans. Ind. Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.

[214] S. Shi, Q. Wang, P. Xu, and X. Chu, “Benchmarking state-of-the-art deep learning software tools,” in *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*, 2016, pp. 99–104.

[215] “The Microsoft Cognitive Toolkit.” [Online]. Available: https://www.microsoft.com/en-us/cognitive-toolkit/.

[216] “Deeplearning4j: Open-source distributed deep learning for the JVM.” [Online]. Available: https://deeplearning4j.org/.

[217] S. Shi, Q. Wang, P. Xu, and X. Chu, “Benchmarking state-of-the-art deep learning

software tools," *Proc. - 2016 7th Int. Conf. Cloud Comput. Big Data, CCBD 2016*, pp. 99–104, 2017.

[218] X. Li, W. Zhang, Q. Ding, and J.-Q. Sun, "Intelligent rotating machinery fault diagnosis based on deep learning using data augmentation," *J. Intell. Manuf.*, vol. 31, no. 2, pp. 433–452, 2020.

[219] D. Chen, S. Yang, and F. Zhou, "Transfer learning based fault diagnosis with missing data due to multi-rate sampling," *Sensors*, vol. 19, no. 8, p. 1826, 2019.

[220] R. Zemouri, M. Lévesque, N. Amyot, C. Hudon, O. Kokoko, and A. Tahan, "Deep Convolutional Variational Autoencoder as a 2D-Visualization tool for Partial Discharge Source Classification in Hydrogenerators," *IEEE Access*, 2019.

[221] A. Brown, A. Tuor, B. Hutchinson, and N. Nichols, "Recurrent neural network attention mechanisms for interpretable system log anomaly detection," in *Proceedings of the First Workshop on Machine Learning for Computing Systems*, 2018, pp. 1–8.

[222] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 281–305, 2012.

[223] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 464–472.

[224] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1--learning rate, batch size, momentum, and weight decay," *arXiv Prepr. arXiv1803.09820*, 2018.

[225] G. Csurka, "A comprehensive survey on domain adaptation for visual

applications," *Adv. Comput. Vis. Pattern Recognit.*, no. 9783319583464, pp. 1–35, 2017.

[226] B. Rezaeianjouybari and Y. Shang, "Deep learning for prognostics and health management: State of the art, challenges, and opportunities," *Measurement*, p. 107929, 2020.

[227] N. Lu, H. Xiao, Y. Sun, M. Han, and Y. Wang, "A New Method for Intelligent Fault Diagnosis of Machines based on Unsupervised Domain Adaptation," *Neurocomputing*, 2020.

[228] W. Lu *et al.*, "Deep model based domain adaptation for fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2296–2305, 2017.

[229] X. Wang, H. He, and L. Li, "A Hierarchical Deep Domain Adaptation Approach for Fault Diagnosis of Power Plant Thermal System," *IEEE Trans. Ind. Informatics*, vol. PP, no. XX, pp. 1–1, 2019.

[230] P. R. de O. da Costa, A. Akcay, Y. Zhang, and U. Kaymak, "Remaining Useful Lifetime Prediction via Deep Domain Adaptation," *arXiv Prepr. arXiv1907.07480*, pp. 1–30, 2019.

[231] C. Cheng, B. Zhou, G. Ma, D. Wu, and Y. Yuan, "Wasserstein Distance based Deep Adversarial Transfer Learning for Intelligent Fault Diagnosis," *arXiv Prepr. arXiv1903.06753*, pp. 1–11, 2019.

[232] C. Cheng, B. Zhou, G. Ma, D. Wu, and Y. Yuan, "Wasserstein distance based deep adversarial transfer learning for intelligent fault diagnosis with unlabeled or insufficient labeled data," *Neurocomputing*, vol. 409, pp. 35–45, 2020.

[233] M. Sun, H. Wang, P. Liu, S. Huang, and P. Fan, "A sparse stacked denoising

autoencoder with optimized transfer learning applied to the fault diagnosis of rolling bearings,” *Measurement*, vol. 146, pp. 305–314, 2019.

[234] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, “Deep Transfer Learning Based on Sparse Autoencoder for Remaining Useful Life Prediction of Tool in Manufacturing,” *IEEE Trans. Ind. Informatics*, vol. 15, no. 4, pp. 2416–2425, 2019.

[235] L. Wen, L. Gao, and X. Li, “A New Deep Transfer Learning Based on Sparse Auto-Encoder for Fault Diagnosis,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 49, no. 1, pp. 136–144, 2017.

[236] Q. Li, B. Tang, L. Deng, Y. Wu, and Y. Wang, “Deep balanced domain adaptation neural networks for fault diagnosis of planetary gearboxes with limited labeled data,” *Measurement*, vol. 156, p. 107570, 2020.

[237] X. Li, X. D. Jia, W. Zhang, H. Ma, Z. Luo, and X. Li, “Intelligent cross-machine fault diagnosis approach with deep auto-encoder and domain adaptation,” *Neurocomputing*, vol. 383, pp. 235–247, 2020.

[238] M. Wang and W. Deng, “Deep visual domain adaptation : A survey,” *Neurocomputing*, vol. 312, pp. 135–153, 2018.

[239] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, “Revisiting batch normalization for practical domain adaptation,” *arXiv Prepr. arXiv1603.04779*, 2016.

[240] A. Rozantsev, M. Salzmann, and P. Fua, “Beyond sharing weights for deep domain adaptation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 4, pp. 801–814, 2018.

[241] B. Zhang, W. Li, X. L. Li, and S. K. Ng, “Intelligent Fault Diagnosis under

Varying Working Conditions Based on Domain Adaptive Convolutional Neural Networks,” *IEEE Access*, vol. 6, pp. 66367–66384, 2018.

[242] X. Li, W. Zhang, and Q. Ding, “A robust intelligent fault diagnosis method for rolling element bearings base d on deep distance metric learning,” *Neurocomputing*, vol. 30, pp. 77–95, 2018.

[243] W. Qian, S. Li, and X. Jiang, “Deep transfer network for rotating machine fault analysis,” *Pattern Recognit.*, vol. 96, 2019.

[244] J. An, P. Ai, and D. Liu, “Deep Domain Adaptation Model for Bearing Fault Diagnosis with Domain Alignment and Discriminative Feature Learning,” *Shock Vib.*, vol. 2020, 2020.

[245] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” *arXiv Prepr. arXiv1701.07875*, 2017.

[246] X. Li, H. Jiang, K. Zhao, and R. Wang, “A Deep Transfer Nonnegativity-Constraint Sparse Autoencoder for Rolling Bearing Fault Diagnosis With Few Labeled Data,” *IEEE Access*, vol. 7, pp. 91216–91224, 2019.

[247] Y. Xie and T. Zhang, “A Transfer Learning Strategy for Rotation Machinery Fault Diagnosis based on Cycle-Consistent Generative Adversarial Networks,” *Proc. 2018 Chinese Autom. Congr. CAC 2018*, pp. 1309–1313, 2019.

[248] X. Li, W. Zhang, Q. Ding, and J. Q. Sun, “Multi-Layer domain adaptation method for rolling bearing fault diagnosis,” *Signal Processing*, vol. 157, pp. 180–197, 2019.

[249] B. Yang, Y. Lei, F. Jia, and S. Xing, “An intelligent fault diagnosis approach based on transfer learning from laboratory bearings to locomotive bearings,” *Mech. Syst.*

Signal Process., vol. 122, pp. 692–706, 2019.

[250] D. Xiao, Y. Huang, L. Zhao, C. Qin, H. Shi, and C. Liu, “Domain Adaptive Motor fault diagnosis using Deep Transfer Learning,” *IEEE Access*, vol. 7, pp. 1–1, 2019.

[251] T. Han, C. Liu, W. Yang, and D. Jiang, “Deep transfer network with joint distribution adaptation: a new intelligent fault diagnosis framework for industry application,” *ISA Trans.*, 2019.

[252] Z. Chen, K. Gryllias, and W. Li, “Intelligent Fault Diagnosis for Rotary Machinery Using Transferable Convolutional Neural Network,” *IEEE Trans. Ind. Informatics*, vol. 3203, no. c, pp. 1–1, 2019.

[253] D. Xiao, Y. Huang, C. Qin, Z. Liu, Y. Li, and C. Liu, “Transfer learning with convolutional neural networks for small sample size problem in machinery fault diagnosis,” *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, p. 0954406219840381, 2019.

[254] X. Li, W. Zhang, N.-X. Xu, and Q. Ding, “Deep learning-based machinery fault diagnostics with domain adaptation across sensors at different places,” *IEEE Trans. Ind. Electron.*, 2019.

[255] X. Li, W. Zhang, Q. Ding, and X. Li, “Diagnosing rotating machines with weakly supervised data using deep transfer learning,” *IEEE Trans. Ind. Informatics*, 2019.

[256] S. Sun, H. Shi, and Y. Wu, “A survey of multi-source domain adaptation,” *Inf. Fusion*, vol. 24, pp. 84–92, 2015.

[257] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, “Learning bounds for domain adaptation,” in *Advances in neural information processing systems*, 2008, pp. 129–136.

[258] Y. Mansour, M. Mohri, and A. Rostamizadeh, “Domain adaptation with multiple sources,” in *Advances in neural information processing systems*, 2009, pp. 1041–1048.

[259] X. Peng, K. Saenko, and B. Wang, “Moment Matching for Multi-Source Domain Adaptation,” no. 2.

[260] J. Liu, J. Li, and K. Lu, “Coupled local–global adaptation for multi-source transfer learning,” *Neurocomputing*, vol. 275, pp. 247–254, 2018.

[261] H. Wang, W. Yang, Z. Lin, and Y. Yu, “TMDA: Task-Specific Multi-source Domain Adaptation via Clustering Embedded Adversarial Training,” in *2019 IEEE International Conference on Data Mining (ICDM)*, 2019, pp. 1372–1377.

[262] B. Sun, J. Feng, and K. Saenko, “Return of frustratingly easy domain adaptation,” *arXiv Prepr. arXiv1511.05547*, 2015.

[263] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, “Central moment discrepancy (cmd) for domain-invariant representation learning,” *arXiv Prepr. arXiv1702.08811*, 2017.

[264] C. Chen *et al.*, “HoMM: Higher-order Moment Matching for Unsupervised Domain Adaptation,” *order*, vol. 1, no. 10, p. 20, 2020.

[265] I. Deshpande, Z. Zhang, and A. G. Schwing, “Generative modeling using the sliced wasserstein distance,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3483–3491.

[266] A. Gretton *et al.*, “Optimal kernel choice for large-scale two-sample tests,” in *Advances in neural information processing systems*, 2012, pp. 1205–1213.

[267] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister, “Sliced and radon wasserstein

barycenters of measures," *J. Math. Imaging Vis.*, vol. 51, no. 1, pp. 22–45, 2015.

[268] K. Nadjahi, A. Durmus, U. Simsekli, and R. Badeau, "Asymptotic guarantees for learning generative models with the sliced-wasserstein distance," in *Advances in Neural Information Processing Systems*, 2019, pp. 250–260.

[269] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv Prepr. arXiv1304.5634*, 2013.

[270] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum Classifier Discrepancy for Unsupervised Domain Adaptation," 2018.

[271] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3723–3732.

[272] Y. Luo, L. Zheng, T. Guan, and J. Yu, "Taking A Closer Look at Domain Shift : Category-level Adversaries for Semantics Consistent Domain Adaptation," pp. 2507–2516.

[273] S. Zhao *et al.*, "Multi-source Distilling Domain Adaptation," *arXiv Prepr. arXiv1911.11554*, 2019.

[274] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[275] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv Prepr. arXiv1412.3474*, 2014.

[276] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation

learning for domain adaptation," *arXiv Prepr. arXiv1707.01217*, 2017.

[277] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2030–2096, 2016.

[278] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3964–3973.

[279] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2200–2207.

[280] Y. Yao, X. Li, Y. Zhang, and Y. Ye, "Multi-source Heterogeneous Domain Adaptation with Conditional Weighting Adversarial Network," *arXiv Prepr. arXiv2008.02714*, 2020.

VITA

Behnoush Rezaeianjouybari was born in Iran. She completed her Bachelors of Science in Mechanical Engineering at the University of Mazandaran, graduating in June 2008. In January 2012 Behnoush Received a Master of Science in Mechanical Engineering from the Sharif University of Technology for her thesis work, "Optimization of kinematic redundancy and workspace analysis of a dual-arm cam-lock robot.". Finally, She received a Doctoral Degree also in Mechanical Engineering from the University of Missouri for her work, " A novel deep cross-domain framework for fault diagnosis of rotary machinery in prognostics and health management". During her Ph.D. at MU, she was responsible for numerous major and minor research focusing on various machine learning-based approaches for sensor-based fault classification, anomaly detection, and time-series forecasting that can be used for predictive maintenance and failure resilience decision making. Behnoush started her career outside of academia in April 2021 working for FIGS, Los Angeles Metropolitan Area as a Data Scientist.