Towards Practical Data-driven Predictive Maintenance: A Robust and Generalizable Deep Learning Approach

Mohamed Ragab Mohamed Adam

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2022

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

25 Apr. 2022

.....

Date

.....

Mohamed Ragab Mohamed Adam

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accordance with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

25 Apr. 2022

Date

Prof. Kwoh Chee Keong

Authorship Attribution Statement

This thesis contains material from five papers published in the following peer-reviewed journals and accepted articles at conferences. In addition, one paper is under review in a peer-reviewed journal in which I am the first author

Chapter 3 is published as Mohamed Ragab, Zhenghua Chen, Min Wu, C. Kwoh, Ruqiang Yan, and Xiaoli Li. "Attention-based sequence to sequence model for machine remaining useful life prediction." Neurocomputing 466 (2021): 58-68.

The contributions of the co-authors are as follows:

- I designed the algorithm, implemented the code, conducted the experiments, and drafted the paper
- Dr. Zhenghua Chen and Dr. Min Wu participated in designing the experiments, analyzing the results, and polishing the manuscript.
- Prof. Ruqiang Yan helped in reviewing the manuscript
- Prof. Kwoh Chee Keong pointed out the research direction
- Prof. Li Xiaoli pointed out the research direction, participated in the technical discussion for developing the idea, and ensured the soundness and correctness of the proposed approach.

Chapter 4 is published in:

Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, and Xiaoli Li. "Adversarial transfer learning for machine remaining useful life prediction." In 2020 IEEE International Conference on Prognostics and Health Management (ICPHM), pp. 1-7. IEEE, 2020.

Mohamed Ragab, Zhenghua Chen, Min Wu, Chuan Sheng Foo, Chee Keong Kwoh, Ruqiang Yan, and Xiaoli Li. "Contrastive adversarial domain adaptation for machine remaining useful life prediction." IEEE Transactions on Industrial Informatics 17, no. 8 (2021): 5239-5249.

The contributions of the co-authors are as follows:

- I designed the algorithm, implemented the code, conducted the experiments, and drafted the paper
- Dr. Zhenghua Chen and Dr. Min Wu participated in designing the experiments, analyzing the results, and polishing the manuscript.

- Dr. Chuan Sheng Foo helped in reviewing the manuscript and refining the approach.
- Prof. Ruqiang Yan helped in reviewing the manuscript
- Prof. Kwoh Chee Keong and Prof. Xiaoli supervised the project and pointed out the research direction
- Prof. Li Xiaoli pointed out the research direction and participated in the technical discussion for developing the idea.

Chapter 5 is published in: Mohamed Ragab, Emadeldeen Eldele, Zhenghua Chen, Min Wu, C. Kwoh, and Xiaoli Li. "Self-supervised Autoregressive Domain Adaptation for Time Series Data." IEEE Transactions on Neural Networks and Learning Systems

The contributions of the co-authors are as follows:

- I designed the algorithm, implemented the code, conducted the experiments, and drafted the paper
- Emadeldeen Eldele worked alongside and helped in designing and executing the experiments.
- Dr. Zhenghua Chen and Dr. Min Wu participated in designing the experiments, analyzing the results, and polishing the manuscript.
- Prof. Kwoh Chee Keong and Prof. Li Xiaoli pointed out the research direction and participated in editing the manuscript

Chapter 6 is published as Mohamed Ragab, Zhenghua Chen, Min Wu, Haoliang Li, C. Kwoh, Ruqiang Yan, and Xiaoli Li. "Adversarial Multiple-Target Domain Adaptation for Fault Classification." IEEE Transactions on Instrumentation and Measurement 70 (2021): 1-11.

The contributions of the co-authors are as follows:

- I designed the algorithm, implemented the code, conducted the experiments, and drafted the paper
- Dr. Zhenghua Chen and Dr. Min Wu participated in designing the experiments, analyzing the results, and polishing the manuscript.
- Dr. Haoliang Li refined the approach and participated in editing the manuscript.
- Prof. Ruqiang Yan helped in reviewing the manuscript
- Prof. Kwoh Chee Keong pointed out the research direction and participated in editing the manuscript.
- Prof. Li Xiaoli participated in the technical discussion for developing the idea, ensured correctness and novelty of the idea, and helped in planning for the manuscript writing while revising and polishing the draft.

Chapter 7 is published as Mohamed Ragab, Zhenghua Chen, Wenyu Zhang, Emadeldeen Eldele, Min Wu, C. Kwoh, and Xiaoli Li. "Conditional Contrastive Domain Generalization for Fault Diagnosis." IEEE Transactions on Instrumentation and Measurement (Accepted)

The contributions of the co-authors are as follows:

- I designed the algorithm, implemented the code, conducted the experiments, and drafted the paper
- Dr. Wenyu Zhang helped in designing the framework and refining the approach
- Emadeldeen Eldele helped in executing the experiments.
- Dr. Zhenghua Chen and Dr. Min Wu participated in designing the experiments, analyzing the results, and polishing the manuscript.
- Prof. Kwoh Chee Keong pointed out the research direction and participated in editing the manuscript
- Prof. Li Xiaoli pointed out the research direction, validated the practicality of the proposed idea, and polished and revised the manuscript.

25 Apr. 2022

.....

Date

.....

Mohamed Ragab Mohamed Adam

Acknowledgements

I am sincerely indebted to Prof. Kwoh Chee Keong and Prof. Li Xiaoli for their kind supervision. I want to express my gratitude for their sustained guidance throughout the whole Ph.D. period. Their supervision has gone beyond technical support and included help and support during personal hardships.

I extend my heartfelt thanks to Dr. Zhenghua Chen and Dr. Wu Min for their mentoring and guidance throughout my Ph.D. study. Their valuable input and sustained support have definitely made me a better researcher.

I would also like to thank my co-authors Dr. Foo Chuan Sheng, Dr. Wenyu Zhang, and my colleague Eng. Emadeldeen Eldele. Their valuable feedback and their sustained support have contributed significantly toward the outcome of this work.

I would also like to share my sincere gratitude to my wife, Zeinab Moled, for being the cornerstone that kept me standing during my whole Ph.D. life. Last, I am incredibly indebted to my parents Mr. Ragab M. Adam and Mrs. Shadia Eissa for their unconditional love, and I will be eternally indebted to them.

"Do your little bit of good where you are; it's those little bits of good put together that overwhelm the world."

—Tutu, Desmond

To my dear family

Abstract

Predictive maintenance (PdM) is a prevailing maintenance strategy that aims to minimize downtime, reduce maintenance costs, and increase machine reliability. With the huge data availability in the era of Industry 4.0, data-driven predictive maintenance leverages Artificial Intelligence (AI) and historical data to forecast potential machine failure, enabling accurate in-advance maintenance schedules based on the machine condition. Recently, deep learning has shown widely acclaimed potential for data-driven predictive maintenance tasks. Nevertheless, deep learning only performs well under two main assumptions: (1) the availability of vast amounts of labeled data; (2) the training phase and deployment phase share the same data distribution or working condition. These constraints can significantly hinder the deep learning applicability to real-world environments. On the one hand, data labeling and annotation can be very laborious tasks, especially for sensor data that encompass complex underlying patterns. Furthermore, running to failure for vast and sophisticated machines can be costly and cause catastrophic consequences. This imposes an additional challenge on obtaining labeled data that identify the health status of the machine. On the other hand, under the premise of the real-world manufacturing environment, variation in working environments is inevitable. This can yield different sensor readings across the working environments even for the same industrial asset. As a result, the data generated from each working condition may have different statistics (i.e., distributions). Under those circumstances, a model that performs well under one working condition performs poorly when tested on data from different working conditions. Therefore, there is a need for more practical deep learning approaches that can handle the aforementioned limitations.

This thesis aims to develop robust and generalizable deep learning approaches that can work under real-world environments i.e., variability of working conditions and lack of labeled data. The focus of this thesis is on two main tasks of predictive maintenance: fault diagnosis and fault prognosis (i.e., Remaining Useful Life Estimation (RUL)) tasks.

The thesis' contributions are structured according to the predictive maintenance tasks. Part I includes two pieces of work to address the real-world challenges of the remaining useful life estimation task. First, an attention-based sequence-to-sequence model is proposed to exploit historical information better, improving robustness and generalization across different industrial assets. Second, a novel contrastive adversarial domain adaptation (CADA) approach is proposed to address the lack of labeled data coupled with the variability of working conditions. The proposed approach aims to transfer the knowledge learned from one labeled operating condition (source domain) to another operating condition (target domain) with no available labels while preserving the target-specific information during the adaptation process.

Part II focuses on the fault diagnosis task and includes three pieces of work that have been developed to address the limitations of deep learning in real-world environments. The thesis first proposes a novel **S**e**L**f-supervised **A**uto**R**egressive **D**omain **A**daptation (SLARDA) to address the lack of labeled data while explicitly considering the dynamics of time series data during both feature learning and domain alignment. Subsequently, towards a more scalable domain adaptation approach, a novel adversarial multiple domain adaptation (AMDA) method for single-source multiple targets (1SmT) is proposed to handle multiple unlabeled working conditions concurrently. Last, unlike previous domain adaptation approaches that assume access to unlabeled data from the target working condition, a novel contrastive domain generalization (CDG) is designed to generalize to unseen working conditions with no prior data available. The main goal of this thesis is to pave the way of deep learning under real-world environments, towards practical data-driven predictive maintenance.

Despite the significant thesis's contributions, there remain some obstacles that hinder the applicability of data-driven predictive maintenance. These obstacles have been regarded in the future works of the presented thesis, which can be briefed as follows: considering the class-label shifts across different working environments; (2) quantifying the uncertainty of the proposed data-driven approaches; (3) realizing explainable deep learning approach that can provide reasons behind the decisions and give descriptions to the end-users.

Contents

List of Figures

List of Tables

Symbols and Acronyms

Symbols

$\mathcal{R}^n$	the n-dimensional Euclidean space
$\odot$	the element-wise product
$\mathcal{X}$	the input space
$\mathcal{Y}$	the label space
$\mathcal{H}$	the feature space
$\mathcal{D}_S$	the source Domain
$\mathcal{D}_T$	the target Domain
$P(X)$	the marginal distribution of the data
$P_S(X)$	the marginal distribution of the source domain
$P_T(X)$	the marginal distribution of the target domain
P_{XY}	the joint distribution defined on $\mathcal{X} \times \mathcal{Y}$
$P(Y\|X)$	the conditional distribution
$\mathbb{E}$	the Expected value
$\bar{x}$	the vector with the average of all components of x as each element
$\|\cdot\|$	the 2-norm of a vector or matrix in Euclidean space
W	the weight matrix
$\otimes$	the Kronecker product
$\langle \cdot, \cdot \rangle$	the inner product of two vectors
$\circ$	the composition of functions
$x_{i,k}$	the i-th component of a vector x at time
k	
$\mathbb{1}$	an indicator function that return 1 when argument is true

Acronyms

PM	Preventive Maintenance
PdM	Predictive Maintenance
CBM	Condition-based Monitring
FD	Fault Diagnosis
FP	Fault Prognosis
RUL	Remaining Useful Life
AI	Artificial Intelligence
SVM	Support Vector Machine
PCA	Principle Component Analysis
ANN	Artificial Neural Network
KNN	K-nearest Neighbor
ELM	Extreme Learning Machine
LDA	Linear Discriminant Analysis
RF	Random Forest
PF	Particle Filter
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-term Memory
RBM	Restricted Boltzman machines
UDA	Unsupervised Domain Adaptation
DG	Domain Generalization
RMSE	Root Mean Square Error
i.i.d.	independent and identically distributed

Chapter 1

Introduction

1.1 Background

Industrial machines are ubiquitous in our daily life. It can be found in trains, planes, ships, factories, and a wide range of manufacturing industries. The failure of these machines can have catastrophic consequences in terms of revenue, productivity, and safety. For instance, in automotive manufacturing, one minute of unplanned downtime can cost up to $20k, while a whole downtime event can cost around $2 Million [4]. Therefore, proper maintenance is a crucial step towards reducing downtime, lowering economic loss, and increasing machines' usability. Three maintenance strategies have been commonly adopted to increase reliability and reduce operational costs namely reactive maintenance (RM), preventive maintenance (PM), and predictive maintenance (PdM). In the reactive maintenance approach, the machine is used until it reaches its maximum limits, while repairs are performed only after failure. This approach can be affordable with simple and cheap systems and machines, yet it can be costly and unsafe for complex industrial assets. For instance, the failure of complex wind turbines not only requires major efforts to be replaced but also imposes life-threatening consequences, leading to losses in both economic and human resources [5]. The preventive maintenance strategy tackles this issue by scheduling regular maintenance for industrial assents to prevent machines from failure. However, regular maintenance imposes costs

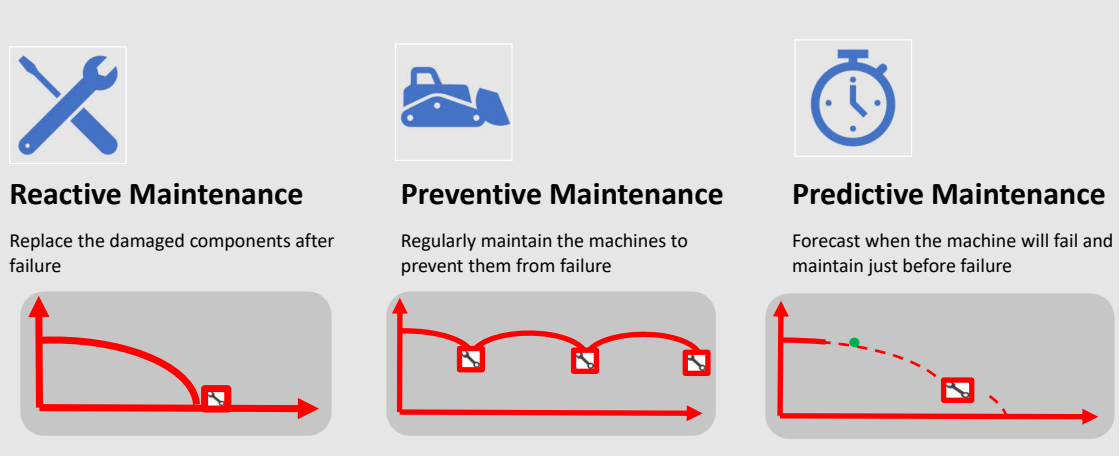


FIGURE 1.1: Maintenance Strategies

overhead by increasing the machines' downtime. Furthermore, with more conservative maintenance schedules, components that are still usable can be replaced, causing additional maintenance costs [6].

Recently, predictive maintenance (PdM) forecasts future failure events, enabling prior maintenance planning, preventing catastrophic failures, and reducing machines' downtime. PdM goes beyond fixed maintenance schedules by leveraging historical sensor data and expert knowledge to forecast time and the location of failure events. Figure 1.1 illustrates different maintenance strategies

Realization of predictive maintenance under realistic environments can be challenging as it requires accurate prediction of the incipient failure time. In particular, over-early prediction of the failure time causes the removal of usable components, imposing unnecessary costs on the maintenance process. In contrast, late predictions of failure events can cause catastrophic accidents and losses due to unplanned asset failures. Predictive maintenance mainly involves two essential tasks, namely, the fault diagnosis and the fault prognosis tasks. Fault diagnosis is the process of detecting, isolating, and identifying the current degradation of a specific component or machine based on physical changes while fault prognosis is the process of predicting the potential degradation or failure of certain equipment or machine. Prognostics aim to complement the diagnostics capabilities by providing prior failure warnings enough to perform the appropriate action. Reliable prediction of *the remaining useful life* (RUL) of equipment is a milestone for prognostics tasks.

Extensive research has been conducted on fault diagnosis and prognosis tasks during the past years. Approaches for fault diagnosis and prognosis can be classified

into two broad categories, namely, model-driven approaches [7], and data-driven approaches [8]. Specifically, model-driven approaches rely on domain expertise to model the physics of the monitored equipment/machine. Hence, it requires a strong theoretical understanding to model the equipment behavior and its detailed degradation process. These approaches operate by observing the behavior of the current process and comparing it with a reference model provided by the system operation under nominal conditions. A residual vector represents the difference between observed measurements and normal behavior. This vector is employed to detect and isolate faults. In particular, the generated residuals are examined to measure the probability of failure given the residual vector, then a decision-making criterion is followed to determine the final decision about failure. The main advantage of these techniques is that they can detect unexpected faulty situations [9]. However, as the complexity of the machine continues to evolve, it is becoming increasingly challenging to get accurate physical models, even with domain expertise.

With the huge data availability in Industry 4.0 era, data-driven approaches have shown great potential in modeling the dynamics of complex systems. For example, it can help intelligently monitor machine health status, identify root causes of failures, and make maintenance decisions. To do so, engineering domain expertise has been employed to extract different prominent features from the data such as frequency domain features, entropy-based features, and empirical modes features. Then, different machine learning algorithms such as support vector machines [10], artificial neural networks [11], and k-nearest neighbors [12] are used to rationally map the different features to the corresponding fault modes. However, these approaches suffer from extensive efforts for feature engineering that requires advanced domain knowledge. Moreover, the manually extracted features can be sensitive to the variability of different working parameters, which are common in the real industrial environment. Recently deep learning with automatic feature extraction ability has been applied for fault diagnosis and prognosis. Deep learning approaches have significantly outperformed traditional machine learning approaches. Various deep learning methods have been used, e.g., convolutional neural network (CNN) [13], autoencoder, [14] and recurrent neural network (RNN) [15].

1.2 Research Questions

Deep learning has shown strong potential and achieved state-of-the-art performance on predictive maintenance tasks. Nevertheless, only 15% of the manufacturing industry is adopting the predictive maintenance strategy [16]. One reason for this deficiency is the incapacity of existing deep learning approaches to handle real-world and dynamic predictive maintenance environments. This thesis aims to mitigate this issue and realize a more reliable deep learning approaches that can handle real-world environments, towards expanding the applicability of predictive maintenance in various manufacturing industries. To do so, this thesis aims to address the following questions:

Q1: Deep learning methods usually focus on the latest information while ignoring historical information of long sequence data to perform the predictive maintenance tasks. However, historical information can be critical for modeling temporal dynamics. Under this context, how can the historical information be better exploited to improve the robustness and reliability of the deep learning model?

Q2: Deep learning only performs well under two main conditions: (1) the training data and testing data are collected under the same working condition; (2) the availability of a large amount of labeled data. However, in reality, the variability of machines or working conditions between training and testing is inevitable. Besides, data labeling can be laborious and time-consuming. Thus, how can deep learning perform well under varying working conditions with only unlabeled data available for the target working condition?

Q3: Real-world manufacturing environment usually involves various machines and working conditions. Even with a more robust deep learning that can generalize well to unlabeled data from different working conditions, a new model needs to be trained for each new working condition. This shows limited scalability in handling multiple working conditions since different models should be trained for different target working conditions, which is not a viable solution in practice. Therefore, how can a more scalable deep learning approach be realized to generalize to multiple working conditions concurrently?

Q4: All the above situations assume access to either labeled or unlabeled data from the target working condition. However, in reality, i the model can be tested on an unknown working condition with no prior data available. Thus, how can the deep learning model leverage historical data from other working conditions to generalize well to new unseen working condition?

1.3 Research Objectives

The main purpose of this thesis is to address the above questions and develop deep learning algorithms that can handle the dynamics of a real-world manufacturing environment. Particularly, the variability of working conditions and the lack of sufficient labeled data hinder the applicability of deep learning to real-world scenarios. Therefore, this thesis aims to develop robust and generalizable deep learning models that can handle both variability of working environments and the lack of labeled data. The thesis objectives are summarized as follows:

O1: To develop a deep learning model that can exploit historical information to have more accurate and robust predictions. Capturing the long-term dependency of the sequence data is a key to accurately predicting the incipient failure time. However, existing deep learning approaches tend to only focus on the latest information when modeling the input sensory data. Therefore, there is a need to develop a deep learning methodology to better exploit the hidden information in the sequence and capture the long-term dependencies to improve the predictive capability of the deep learning model.

O2: To equip deep learning model with domain adaptation capability to generalize well on the different working environments with no labeled data available. Deep learning approaches work under the assumption that the training phase and the deployment phase are conducted under the same working environment, which does not hold in many practical scenarios. We consider two working environments A and B each with certain parameters from different distributions. The model trained on working condition A usually fails to generalize well to working condition B. In reality, this situation is very likely to happen due to the varying environments. Besides, as data annotation is laborious and time-consuming, only unlabeled data can

be attainable for some working conditions. Thus, there is an urgent need for adaptive deep learning models that can handle the scarcity of labeled data problems. Therefore, there is an urgent need for a model that can handle a new working environment with no labels available.

O3: To devise more scalable deep learning models that can generalize to multiple working conditions simultaneously. Prior domain adaptation models are designed to handle only a single working condition at a time. Thus, a new model needs to be retrained for each new working condition. This can have limited scalability and applicability in dynamic environments with multiple working conditions. Hence, there is a need for a more scalable solution to multiple working conditions simultaneously.

O4: To propose a ubiquitous deep learning model that perfectly leverages historical data to generalize to new unseen environments with no prior data available. In the realistic industrial environment, the deep learning model can be tested on new unseen working conditions with no prior data available for the training. In this situation, even a scalable domain adaptive model can be unfeasible as it mainly assumes access to unlabeled data from the target working condition. Therefore, there is a need for a ubiquitous model that performs well in new unseen working conditions on the fly without any prior data available for the training.

1.4 Thesis Contributions

This thesis presents robust and generalizable deep learning models that can address real-world manufacturing problems. The major contributions of this thesis are as follows:

- **A novel attention-based sequence-to-sequence model with an auxiliary task is developed to better handle longer sequences and empower the predictive capabilities of the model.** In prognostics tasks, longer sequences can be more informative to better improve the prediction performance. Therefore, focusing on the most relevant information of the long sequences can be critical to the model's performance. In this thesis,

an attention-based sequence model is proposed to better handle longer sequences while focusing on the most relevant parts of the sequence. First, the proposed approach jointly optimizes a novel auxiliary reconstruction task with the RUL prediction task to empower the prediction capability. Specifically, the model is optimized to predict the future sequences given the current sequence; simultaneously, it is optimized to accurately predict the RUL values. Second, attention-based decoding is developed to focus on the most relevant information of the sequence during the training process. Last, a new dual latent feature representation is developed to integrate the encoder and decoder features, enriching the feature representation.

- **A novel domain adaptive deep learning method is proposed to generalize to different working conditions with no labeled data available for fault prognosis task.** As annotating sensory data is very laborious and time-consuming, scarcity of labeled data can be a serious problem. However, labeled data can still be attainable from simulated situations or historical data. Thus, this thesis aims to leverage readily labeled data from one working condition (i.e., source domain) to improve the performance on different unlabeled working conditions (i.e., target domain). To address this issue, a novel contrastive adversarial domain adaptation approach is proposed. First, an adversarial training technique is developed to find domain invariant feature representation for the source and target domains, improving the performance on the target domain without access to labeled data. However, forcing the target domain features to be invariant to source domain features without any constraints can remove the target-specific information and hinder the model performance on the target domain. To tackle this, a contrastive adaptation module is proposed to preserve the target-specific information during the adversarial adaptation step. Specifically, the proposed approach aims to maximize the mutual information between the input space and the feature space of the target data, preserving the structure of target features during the adaptation process.

- **A novel self-supervised aggressive domain adaptive model is proposed to improve the generalization on different unlabeled working conditions for fault diagnosis tasks.** The temporal information of time-series sensory signals can be critical when identifying different fault types. Thus, this approach aims to consider the temporal information to

improve features transferability, as well as, adaptation performance on unlabeled working conditions. First, self-supervised pretraining is proposed to improve the feature transferability of the source domain. Subsequently, an autoregressive domain discriminator is proposed to consider the temporal dimension between the source and target features during the adversarial alignment step. Last, to further refine the adaptation process, a class-conditional alignment technique is proposed to align the fine-grained distributions within the source and target domains.

- **A novel scalable multi-target domain adaptation approach is proposed for the fault classification task.** While adapting to single unlabeled working conditions has been addressed, there is still a need for a scalable solution that can handle multiple working conditions concurrently. This thesis proposes a novel scalable single-source multiple-target domain adaptation approach that can concurrently generalize to multiple working environments. The proposed method leverages a shared network among the multiple target domains while training an adversarial discriminator network to align the features of the multiple target domains with the source domain.

- **A novel contrastive domain generalization approach is proposed to handle unseen working environments with no prior data.** While all the above-proposed approaches address practical scenarios, they all assume that either labeled or unlabeled data from the target working condition is readily available during the training process. However, this assumption may not hold in practice as the model can encounter new working conditions with no prior data available. This thesis proposes a novel domain generalization approach that leverages historical data from other working conditions to realize a ubiquitous deep learning model that generalizes to an unseen working condition on the fly without requiring data availability or additional training. Particularly, this approach leverages data from multiple historical domains to learn environment-independent class representation, enabling better generalization on the new working environment.

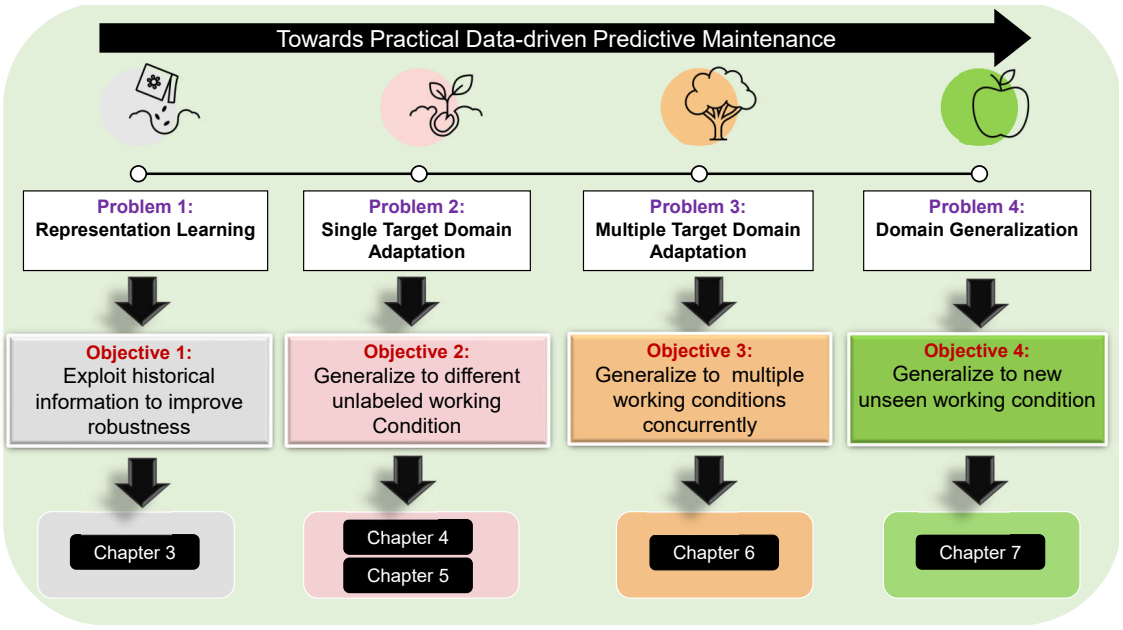


FIGURE 1.2: The overall aims and objectives of the thesis are summarized. Chapter 3 presents the representation learning technique to improve the model's robustness. Chapters 4 and 5 realize domain adaptation approaches to handle the lack of labeled data and the distribution shift problem. Chapter 6 presents a scalable domain adaptation to handle the dynamic change in working conditions. Last, Chapter 7 proposes a domain generalization approach that can handle to new-unseen working conditions.

1.5 Thesis Organization

This thesis presents the above contributions to address the aforementioned challenges of real-world data-driven predictive maintenance. Fig. 1.2 summarizes the research problems and objectives of each chapter in the thesis. The thesis is organized as follows:

- Chapter 2 provides a comprehensive overview of various machine learning and deep learning for the predictive maintenance tasks i.e., fault diagnosis, and prognosis tasks. Besides, the limitations and advantages of various approaches are also reviewed.

- Chapter 3 introduces an Attention-Based Sequence to Sequence Model for Machine Remaining Useful Life Prediction that advances the existing state-of-the-art methods. This model leverages attention-based decoding to focus on the most relevant features of the input sequence. In addition, a novel auxiliary task of predicting future sequences is presented to empower the predictive

capability of the proposed model. Last, a dual-latent feature representation approach is presented to enrich the feature representation of the proposed model [17].

- Chapter 4 introduces an adversarial domain adaptation approach for remaining useful life estimation. This model utilizes adversarial learning techniques to bridge the domain gap between different working conditions. Furthermore, a contrastive domain adaptation module is developed to preserve the target-specific structure and further improve stability and performance [18].

- Chapter 5 introduces a Self-supervised Autoregressive Domain Adaptation for Fault Classification [19]. This chapter first demonstrates the effectiveness of self-supervised learning in improving the transferability of source pretraining. In addition, an Autoregressive domain discriminator is presented to consider the temporal dimensions in the features during the domain alignment step. Last, a class-conditional alignment technique is proposed to align the fine-grained class distributions among different domains.

- Chapter 6 introduces an Adversarial Multiple-Target Domain Adaptation for Fault Classification [20], a new model that addresses the limited scalability of existing domain adaptation methods via generalizing to multiple target domains concurrently. The key idea is to find feature representations of multiple target domains that can be invariant from the source domain.

- Chapter 7 introduces a Conditional Contrastive Domain Generalization for Fault Diagnosis, a more realistic solution that can generalize to unseen domains (i.e., working conditions) with no prior data available [21]. This approach aims to leverage historical data from different working conditions and learn an environment-independent class representation that can be robust to any new working environment.

- Chapter 8 concludes the contributions of the thesis and draws the future directions towards a practical deep learning model that can handle the challenges in the real-world manufacturing environment.

Chapter 2

Literature Review

This chapter examines existing literature and related works for data-driven predictive maintenance. First, the conventional model-driven predictive maintenance is illustrated. Subsequently, the data-driven predictive maintenance and its general framework are presented. Then, a background on the prevailing data-driven approaches is provided to facilitate the concepts of the reviewed works in Section 2.3. After that, the existing literature for both fault diagnosis and fault prognosis tasks is presented in Sections 2.4 and 2.5 respectively. Last, the key approaches are summarized and the research gap is illustrated.

2.1 Model-driven Predictive Maintenance

Model-driven approaches heavily rely on domain expertise to model the physics of the monitored equipment/machine. Hence, it requires a strong theoretical understanding to model the machine's behavior and its detailed degradation process. Figure 2.1 shows the steps of the model-based approach to detect the incipient fault [22]. The model-based approach operates by observing the behavior of the current process and comparing it with a reference model provided by the system operation under nominal conditions. Subsequently, the difference between observed measurements and normal behavior is represented by a residual vector, which is employed to detect and isolate faults. The generated residuals are then examined to measure the probability of failure given the residual vector, then a decision-making criterion is adopted to determine the final decision about failure. The main advantage of these techniques is that they can detect unexpected faulty situations [9]. However, as the machine's complexity continues to evolve, it is becoming increasingly challenging to get accurate physical models, even with domain expertise.

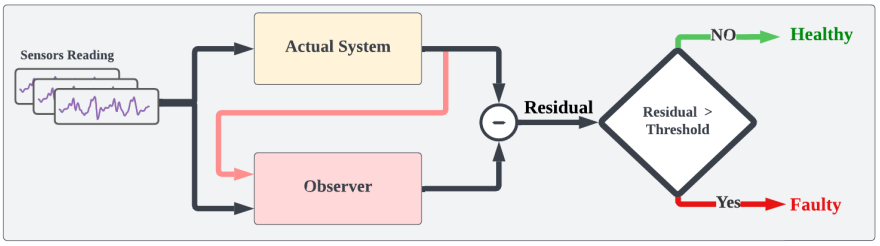


FIGURE 2.1: A model-based approach for fault detection

2.2 Data-driven Predictive Maintenance

In the era of Industry 4.0, the huge availability of sensory data coupled with the unprecedented advancement in data-driven approaches has evoked the dominance of data-driven predictive maintenance approaches. As illustrated in Fig 2.2, the life cycle of data-driven predictive maintenance can be represented by four main stages namely data acquisition, data preprocessing, development of the data-driven model, and deployment of the data-driven model. First, different types of sensors (i.e., vibration, temperature, pressure, etc. are deployed in the machine or some

sub-components. These sensors provide the required data to enable accurate monitoring of machine health. In the second stage, data preparation and preprocessing are conducted. This step involves smoothing the acquired signals, normalizing ranges across different sensory data, removing irregular sensor readings, and saving the data in a format that can be suitable for the subsequent steps. In the third stage, the data-driven model is developed to perform the key tasks for predictive maintenance namely fault diagnosis and fault prognosis tasks. The data-driven model can either be a conventional machine learning algorithm or a deep learning algorithm. Conventional machine learning requires more additional steps such as feature extraction and selection. In contrast, deep learning can automatically extract salient features that can be suitable for the target task. Development of any data-driven model usually involves many internal steps including selecting the suitable data-driven model, training and validating the model performance and testing the model on the target task. In the last stage of the PdM cycle, the model is deployed either on the edge device (i.e., the machine or the sub-component) or on a cloud system that can provide additional computational power.

This thesis mainly focuses on the third stage of developing the data-driven model. Particularly, the evolving complexity of industrial machines coupled with dynamic environments of real-world industry imposes serious challenges on the reliability of data-driven models and hinders their applicability under real-world scenarios.

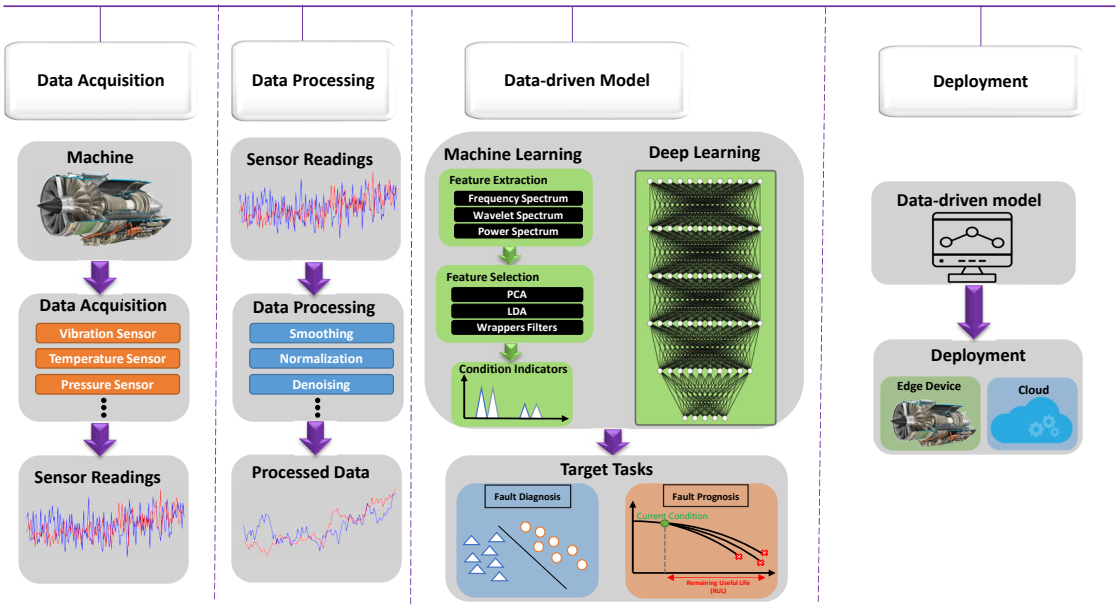


FIGURE 2.2: The framework of data-driven predictive maintenance

2.3 Background on Data-driven Approaches

In this section, we provide a detailed background for the data-driven approaches that have been utilized in different predictive maintenance tasks. Two main paradigms are illustrated namely deep learning and transfer learning. Different settings and categories of each paradigm will be illustrated in the following subsections.

2.3.1 Terminologies and Problem Settings

Data-driven approaches include different settings according to the target problem. In this section, we summarize the key terminologies that will be used in the current chapter and the next few chapters. For any given dataset, we define the following:

- *Domain*: $\mathcal{D} = \{\mathcal{X}, P(X)\}$ is represented by feature space $\mathcal{X}$, which is the space of all possible inputs, and the marginal probability distribution of data in the given feature space $P(X)$, with $X = x_1, \ldots, x_n \in \mathcal{X}$
- *Task*: $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$ is represented by the space of all possible labels $\mathcal{Y}$, and the predictive function that maps the input samples to the corresponding labels $f(\cdot) : \mathcal{X} \rightarrow \mathcal{Y}$
- *Deep Learning*: Given a training domain $\mathcal{D}_{train}$ and a training task $\mathcal{T}_{train}$ the main aim of the supervised learning task is to leverage the training data learn a predictive function $f(\cdot)$ that can generalize well on the testing data from domain $\mathcal{D}_{test}$ and with a task $\mathcal{T}_{test}$. For the supervised learning problem, the train and test data has similar domains $\mathcal{D}_{train} = \mathcal{D}_{test}$ and similar tasks $\mathcal{T}_{train} = \mathcal{T}_{test}$.
- *Transfer Learning*: Given a source domain $\mathcal{D}_S$ with task $\mathcal{T}_S$ and target domain D_T with task T_T, the main goal of transfer learning is to leverage the source knowledge to learn a predictive function $f(\cdot)$ that works well on the target domain, where either domains or tasks are different $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$ respectively.
- *Domain Adaptation*: A special sub-paradigm under transfer learning where there is a distribution shift between the source and target domains $\mathcal{D}_S \neq \mathcal{D}_T$ but the task is the common $\mathcal{T}_S = \mathcal{T}_T$. Domain adaptation settings can vary

TABLE 2.1: Different problem settings of data-driven approaches

Scenario	Source Data	Target Data	Distribution shift
Deep Learning	Fully labeled	Unavailable	✗
Supervised Domain Adaptation	Fully labeled	Labeled	✓
Semi-supervised Domain Adaptation	Full labels	Partially labeled & Unlabeled	✓
Unsupervised Domain Adaptation	Full labels	Unlabeled	✓
Domain Generalization	Full labels	Unavailable	✓

according to the amount of labeled data available on the target domain, which can be supervised, semi-supervised, or unsupervised.

- *Domain Generalization* A special sub-paradigm under transfer learning where the model is learned from multiple source domains $\{\mathcal{D}_S\}_{i=1}^{M}$, and supposed to generalize well to unseen target domain D_T.

The different problem settings are summarized in Table 2.1

2.3.2 Deep Learning and Prevailing Architectures

Deep learning is a form of machine learning that has made significant advances in recent years due to its increasing ability to solve complex problems effectively. Unlike conventional machine learning that requires hand-crafted features, deep learning can automatically extract salient features from complex signals, as shown in figure 2.2. Deep learning is composed of a multi-layer hierarchical structure, where each layer will pass its processed output to the subsequent layer. Different layer architectures have been leveraged in the literature [23] including fully connected neural network (FCNN), convolutional neural network (CNN), and recurrent neural network (RNN). Here we provide further details about each model architecture.

Fully Connected Neural Network

Fully connected networks (FCNNs) were first proposed in 1986 [24]. These networks are usually composed of the input layer, hidden layer, and output layer. Training MLP with multiple layers has been cumbersome for a long time. Recently, the modern activation layers for non-linearity, propagation, and advanced optimization have enabled the training of a deeper FCNN with more layers. The diagram of FCNN is shown in figure 2.3. Each layer is composed of multiple neurons, where

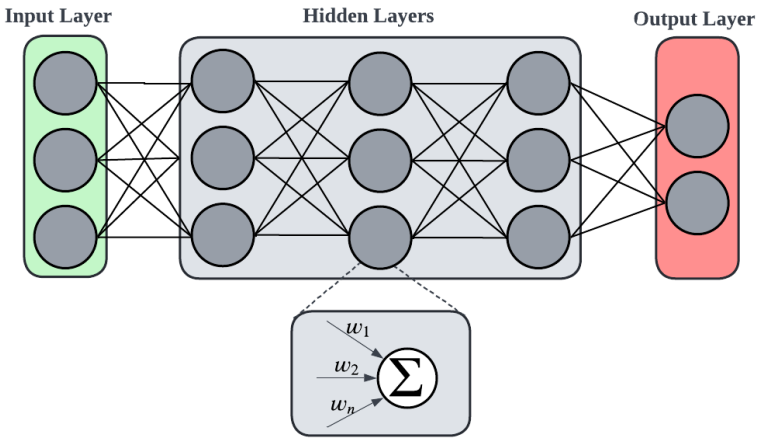


FIGURE 2.3: Fully Connected Neural Network Schematic Diagram

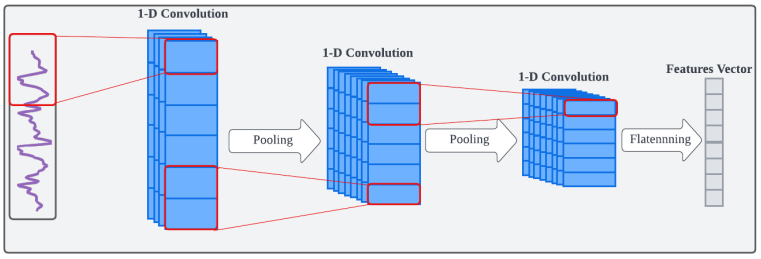


FIGURE 2.4: Schematic diagram of Convolutional Neural Network

each neuron is considered a single computation unit. In particular, given an input sample $\mathbf{x}_i \in \mathbb{R}^n$, the sum of weighted inputs will be estimated and a bias value will be added to each neuron. This can be formalized as follows:

$$o_j = \mathbf{w}_j^T \mathbf{x}_i \tag{2.1}$$

where $\mathbf{w}_j$ and o_j are the corresponding weights and output for neuron j. Subsequently, the output of each neuron is passed through a non-linear activation function. This process will be conducted for each layer and the output information will be passed to the next layer till the final output layer. Then the error function is applied between the predicted output of the neural network and the ground truth labels. Updating the neural network weights across multiple layers to minimize the error had also been challenging for a long time. To solve this issue, a back-propagation algorithm has been proposed to be able to train deeper neural networks efficiently [25].

Convolutional Neural Network (CNN) CNN is a deep learning architecture

that focuses on addressing non-structured data including signals and images. CNN has the ability to learn local features effectively. CNN has a long history as it has been first introduced by Lecun *et al.* in 1995. The reemerge of CNN happened in 2012 when it achieved outstanding performance in ImageNet competition [26]. The operation of CNN is mainly relying on the convolution theory in signal processing, where a local kernel function is applied to each part of the input data. Besides the convolution operation, the non-linear activation and pooling layers are key pillars in the CNN architecture. The CNN architecture is shown in figure 2.4. The information processing steps of the CNN network can be summarized as follows: 1) given an input signal or image, a single or multiple convolution kernels with limited length are applied locally to each part of the input; 2) the output of each convolutional kernel is represented in each feature map, where the number of feature maps is equal to the number of applied kernels; 3) a pooling operation is applied to the feature maps to reduce the dimensionality and select the prominent information from the feature map; 4) this process repeated for each convolutional layer until we reach to the final output feature maps; 5) the feature maps of the last layer are then flattened in one vector and passed to an FCNN layer to perform the task [26]. One key advantage of CNN over FCNN is the ability to address the local patterns while being shift-invariance. More specifically, the parameters of each kernel are shared across the whole input, enabling the representation of CNN to be equivariant [27]. The mathematical operation of the convolutional layer can be described as follows:

$$\phi = \sigma(\mathbf{x} \circledast \mathbf{k}) = \sigma(\sum_{a=0}^{m} \mathbf{x}_{i+a} k_a) \tag{2.2}$$

where ϕ is the output feature map for sample $\mathbf{x}$, $\mathbf{k}$ is the one dimensional convolutional filter with length m, and σ is a nonlinear activation function. Each convolutional layer usually followed a pooling layer to reduce the dimensionality.

Recurrent Neural Network (RNN) the recurrent neural network is a special type of NNs that specializes in modeling sequential patterns. It can store information over time, making them well-suited for modeling time series data. Unlike traditional neural networks, which process one input at a time, RNNs analyze a sequence of inputs concurrently. This enables RNNs to process data with temporal dynamics. RNNs have a wide range of applications, including speech recognition, machine translation, natural language processing, and market forecasting. The operation process of RNN is illustrated in figure 2.5. In particular, given an input

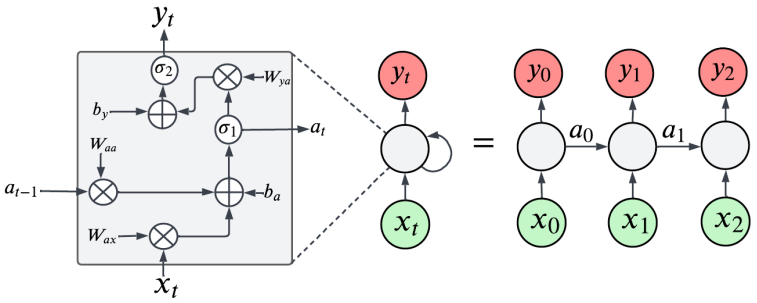


FIGURE 2.5: Schematic Diagram of recurrent neural network

sample with multiple time steps, RNN processes the input for each time step and passed the processed output to the next step. It is worth noting that the parameters are shared across the whole time steps, enabling fixed model complexity for any input length. Formally speaking, given a sample $x = x_1, x_2, \ldots, x_T$ with T time steps, for each time step input x_t the activation a_t and the output can be described as follows:

$$\begin{aligned} a_t &= \sigma_1(W_{aa}a_{t-1} + W_{ax} + b_a) \\ y_t &= \sigma_2(W_{ya}a_t + b_y) \end{aligned} \tag{2.3}$$

where W_* and b_* represent the shared RNN parameters, and σ_1, σ_2 are the non-linear activation functions.

2.3.3 Transfer Learning

Transfer learning (TL) aims to transfer knowledge from one source domain to a different but related target domain. TL can be leveraged to either reduce the amount of data or labels needed when adapting to the target domain. TL has been used for a wide range of applications, including natural language processing, computer vision, speech recognition, and time series applications [28]. Different categories of transfer learning exist according to whether the task is different $\mathcal{T}_S \neq \mathcal{T}_T$ or the domain is different $\mathcal{D}_S \neq \mathcal{D}_T$. In this thesis, we focus on the latter where a distribution shift exists between the source and target domains. Therefore, we focus on the following categories: (1) Domain Adaptation; (2) Domain Generalization. Domain Adaptation settings according to the label availability can be supervised, semi-supervised, and unsupervised, as shown in table 2.1. Here we provide more details about each category.

2.3.3.1 Supervised Domain Adaptation (SDA)

Supervised Domain Adaptation, also named finetuning, assumes access to labeled source data and very limited labeled target data. The procedure of supervised domain adaptation can be illustrated as follows. First, the model is pretrained on the labeled data from the source domain. Then, the weights of the source pretrained model are transferred to the target model. Last, the target model is fine-tuned with available labels from the target domain. One key parameter to tune is to decide the number of layers to be frozen and fine-tuned, which can vary according to the amount of labeled data available. Figure xx illustrates the supervised domain adaptation scenario. While promising, this approach can only work under the assumption that labeled data is available for the target domain. However, in reality, obtaining labeled data can be laborious and time-consuming how much layer to train and freeze depends on the amount of data available for the target domain.

2.3.3.2 Semi-supervised Domain Adaptation (SSDA)

Semi-supervised domain adaptation assumes access to both limited labels and massive unlabeled data from the target domain. Therefore, the proposed model usually aims to exploit both labeled and unlabeled data to improve the adaptation performance on the target domain [29].

2.3.3.3 Unsupervised Domain Adaptation (UDA)

Unsupervised Domain Adaptation is the most prevalent domain adaptation setting, where only unlabeled data are available for the target domain. While this setting is more challenging, it can be more practical as labeled data can be expensive and laborious for many applications. The mainstream of UDA algorithms is to address the domain shift problem by finding domain invariant feature representation. In particular, given a labeled source domain and unlabeled target domain, UDA algorithms are mainly aiming to find target domain features that are similar to the source features such that the source classifier can work well on the adapted target features. Formally speaking, given a feature extraction network $f_\theta : X \to Z$,

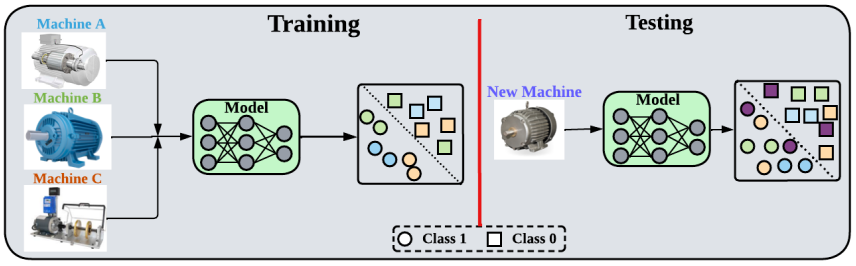


FIGURE 2.6: Domain Generalization scenario across different machines

which maps the input space to the feature space, the UDA algorithm mainly optimizes the feature extractor to minimize a domain alignment loss $\mathcal{L}_{\text{align}}$, aiming to mitigate the distribution shift between the source and target domains such that $P_s(f_\theta(x)) = P_t(f_\theta(x))$. The domain alignment loss can either be estimated from a statistical distance measure [30] or an adversarial discriminator network [31], which can be formalized as follows:

$$\mathcal{L}_{\text{align}} = \min_{f_\theta, h_\theta} \ell(Z_s, Z_t), \tag{2.4}$$

where ℓ can be a statistical distance or an adversarial loss. Concurrently, a classifier network h_θ is applied on top of the feature extractor network to map the encoded features to the corresponding class probabilities. Particularly, given the source domain features Z_s generated from the feature extractor, we can calculate the output probabilities $\mathbf{p}_s = h_\theta(Z_s))$. Thus, the source classification loss can be formalized as follows

$$\mathcal{L}^s_{\text{cls}} = -\mathbb{E}_{(\mathbf{x}_s, y_s)\sim P_s} \sum_{k=1}^{K} \mathbb{1}_{[y_s=k]} \log \mathbf{p}^k_s, \tag{2.5}$$

where $\mathbb{1}$ is the indicator function, which is set to be 1 when the condition is met, and set to 0 otherwise. Both the source classification loss $\mathcal{L}^s_{\text{cls}}$ and the domain alignment loss $\mathcal{L}_{\text{align}}$ are jointly optimized to mitigate the domain shift while learning the source classification task, which can be expressed as

$$\min_{f_\theta, h_\theta} \mathcal{L}^s_{\text{cls}} + \mathcal{L}_{\text{align}}. \tag{2.6}$$

2.3.3.4 Domain Generalization

Domain adaptation methods usually assume that unlabeled data from the target domain is readily available during the training process. Yet, with continuous changes in working conditions, access to data from new working conditions in advance can be infeasible. Therefore, domain generalization aims to address a more realistic scenario where the target data are not readily available during the training. The key idea behind domain generalization is to leverage the knowledge in multiple source domains to improve the generalization ability to the new unseen target domain. Formally speaking, in the domain generalization problem, we are given multiple source domains $(\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^M\}$, where M is the total number of source domains. Each domain $\mathcal{D}^i = \{\mathbf{x}_j^i, y_j^i\}_{j=1}^{N_i} \sim P_{XY}^i$ has N_i number of samples. The joint distributions vary among the multiple source domains, i.e., $P_{XY}^1 \neq P_{XY}^2 \ldots \neq P_{XY}^M$. The objective of the domain generalization is to learn a model $h : \mathcal{X} \rightarrow \mathcal{Y}$ based on all the source domains and generalize well to a new unseen target domain $D^{test} = \{\mathbf{x}_j^{test}, y_j^{test}\}_{j=1}^{N_{test}} \sim P_{XY}^{test}$, where $P_{XY}^{test} \neq P_{XY}^i \; for \; i \in \{1, \ldots, M\}$. It is worth emphasizing that we do not have any access to D^{test} during the training procedure. Figure 2.6 illustrates the scenario of domain generalization for predictive maintenance application.

2.4 Data-driven Fault Diagnosis

Data-driven fault diagnosis methods have great potential in many real-world industrial applications. For example, it can assist in monitoring machine health status, identifying root causes of failures, and making maintenance decisions.

2.4.1 Traditional Machine Learning for Fault Diagnosis

Various machine learning approaches have been employed for fault diagnosis during the past years including support vector machines (SVM), principle component analysis (PCA), artificial neural networks (ANN), K-nearest neighbors (KNN), and others. To apply these techniques, a typical procedure has been commonly used in the literature. First, deep data exploration and analysis are performed to extract relevant features, which require domain knowledge or specialized experts.

This step is followed by extracting low-dimensional features is extracted from the high-dimensional data using dimensional reduction techniques (e.g., PCA, etc.). Subsequently, a feature selection step is performed to select the most representative features which would be passed to the ML algorithm to perform the corresponding task. PCA has shown great potential in revealing hidden structures in the data and explaining its variability [32–34]. It has been used to ease manual feature extraction and select the most informative features. Last, various machine learning methods are used for the fault detection and classification step. For instance, ANN-based approaches have been employed for fault diagnosis problems [11, 35], where a nonlinear mapping of motor current signals is used to predict the damping coefficient and detect the faults. The Nonparametric K-NN approach has also been explored for the fault diagnosis problem [12, 36, 37]. Support vector machines (SVM) have been used extensively for fault diagnosis and exhibited significant improvement over all other approaches [10, 38–40]. In addition to the previous methods, other ML approaches have also been used in the fault diagnosis problems such as extreme-learning machines (ELM)[41] , linear discriminant analysis (LDA) [42], random forest (RF) [43], particle filter (PF) [44], and ensemble algorithms [45].

Despite the dense literature on traditional ML methods in fault diagnosis, they still need manual feature extraction and feature selection, which can be labor-intensive and problem-specific. For instance, among most ML approaches, the fault features are mainly comprised of the characteristic frequencies that correspond to specific motor speeds and feature geometry. However, the features of a characteristic frequency are so vulnerable to dynamic industrial environments. For example, if multiple faults occur concurrently, the characteristic frequency will not be representative of the faults as it encounters some changes due to complex processes. Moreover, the model can be trained under one working condition and tested in different environments; as a result, the extracted features during the training phase will be less representative in the testing phase.

2.4.2 Deep Learning for Fault Diagnosis

During the past few years, deep learning, with its ability to automatically extract salient features, has emerged as a promising alternative to traditional ML methods

in many applications including computer vision, speech recognition, and natural language processing [27]. Recently, deep learning has also been applied to fault diagnosis. More specifically, Convolutional neural networks (CNNs) have been widely adopted for the fault diagnosis problem. For instance, Chen *et al.* were the first to employ CNN for fault diagnosis problems; however, they only used CNN in the classification stage while relying on manual feature engineering to represent the input features [46]. To remove the burden of manual feature extraction, Janssens *et al.* leveraged CNN to learn representative features from raw input data [47]. Similarly, Wen *et al.* converted the time signals into two-dimensional signals and leveraged CNN to automatically extract features [48]. Following the same direction, [49] the raw time series data has been transformed into spectrograms before using a fully convolutional neural network (FCNN) to perform the fault diagnosis task. In [50], a CNN with an adaptive learning rate has been applied for fault classification problems. Xia *et al.* proposed a feature fusion approach by merging temporal and spatial information in the dataset to classify the faults accurately [51]. Zhang *et al.,* proposed another CNN with multi-scale kernels to extract features at different resolutions [52].

In addition to CNN, autoencoder-based approaches (AE) have also been used for fault diagnosis problems [14, 53–55]. Autoencoders can automatically extract salient features in an unsupervised manner. Particularly, an encoder network first maps the input signal to latent feature representation while a decoder network tries to reconstruct the input signal from the latent features [56]. For example, Jia *et al.,* developed a deep neural network (DNN) based autoencoder to extract features from spectral data [57]. Mao *et al.* combined an autoencoder with an extreme learning machine for fault classification [58]. While Guo *et al.* used a stacked denoising autoencoder (SDA) to handle environmental noise in the fault diagnosis task [59]. Similarly, in [60], a stacked autoencoder is augmented with compressed sensing to reduce the amount of measured data and automatically extract features in a transform domain.

On the other hand, recurrent neural networks (RNN) have shown widely acclaimed performance in modeling dynamic systems for many applications such as natural language processing (e.g., language modeling, neural machine translation, and speech recognition), and time series modeling and forecasting. RNNs have been also applied for the fault diagnosis problem. For example, Abed *et al.* extracted

discrete wavelet features and passed them to the RNN to perform the fault classification task of the bearing machine [15]. While RNNs suffer from vanishing gradient problems, long short-term memory (LSTM) is a strong variant of RNN that can handle gradient vanishing and explosion problems. Pan *et al.,* combined the LSTM network with one dimensional CNN layer and a fully connected layer for the classification of bearing faults [61]. While Jiang *et al.* used stacked LSTM units to have better feature representation and achieve accurate fault classification [62]. Other approaches have been also adapted for fault diagnosis tasks such as deep belief networks (DBN), restricted Boltzmann machines (RBM), deep Boltzmann machines (DBM), extreme learning machines, etc [63].

Despite the wide adoption of deep learning for fault diagnosis tasks, it is only applicable under certain conditions. First, deep learning requires access to a vast amount of labeled data for model training. However, the prolonged time that machines take to fail, coupled with the difficulty of annotating time series data limits the availability of labeled data. Second, deep learning assumes that the training data and testing data are collected from the same working condition. Such assumptions can not hold under dynamic and highly varying environments. Therefore, there is a need for a more practical way that can handle the variability of working environments under the scarcity of labeled data.

2.4.3 Transfer Learning for Fault Diagnosis

2.4.3.1 Supervised Domain Adaptation for Fault Diagnosis

SDA also known as finetuning assume access to a large amount of labeled data from the source domain and limited labels from the target domain. Only few approaches tried to improve the target domain performance with only few labeled data available on the fault diagnosis application. For instance, Shao *et al.* first converted the signal data to a time-frequency graph to train the model on the source domain. subsequently, the pretrained network is tuned on the target domain with few labeled data available [64]. Kim *et al.* have introduced a novel approach to freeze only the output-sensitive parameters in the source network while preventing over-fitting on the few labeled target data [65].

2.4.3.2 Semi-supervised Domain Adaptation for Fault Diagnosis

Unlike the supervised domain adaptation approach, the semi-supervised approach assumes access to few labeled and massive unlabeled data from the target domain. This approach can be more realistic as unlabeled data can be attainable for the fault diagnosis task. However, little attention has been to SSDA in fault diagnosis applications. For instance, Bin *et al.* first developed a CNN network to extract features from both source and target domains. Subsequently, a regularization term has been introduced to reduce the domain's discrepancy coupled with target pseudo labels to further improve the class separability of the target domain. Another approach has leveraged the virtual adversarial training coupled with batch-norm maximization to improve the model robustness on the target data[66].

2.4.3.3 Unsupervised Domain Adaptation for Fault Diagnosis

Unlike both SDA and SSDA, Unsupervised domain adaptation (UDA) is the most prevalent paradigm of domain adaptation in fault diagnosis applications. UDA has achieved noticeable success in addressing distribution shift problems under the lack of labeled data for visual applications [67]. The UDA scenario assumes access to labeled data from a source domain (i.e., publicly available or simulated data with cheap labeling efforts), and unlabelled data from a target domain (i.e., the domain of interest) that is sampled from *different but related* distribution. The main goal of UDA is to leverage the labeled source domain to improve the performance of the unlabeled target domain. Recently, unsupervised domain adaptation has been widely explored for the fault diagnosis task to address the variability of the working conditions under the lack of labeled data [68]. The scope of this section will be limited to the Deep UDA methods that mainly build UDA methods on top of deep learning models. The existing UDA methods for fault diagnosis can be classified into two mainstream approaches namely the instance re-weighting approach and the domain-invariant features approach.

On one hand, the instance re-weighting approach tries to attend more to the source samples that are more related to the target domain while giving less attention to the source samples that are distinct from the target domain [69]. One way to automatically calculate the importance weights of the source samples is to leverage statistical distance measures. Particularly, a statistical distance measure is first

used to calculate the distance between the source and target samples. Then, higher weights will be given to samples with smaller distances and lower weights will be given to the distant samples. For instance, Zhang *et al.* employed the maximum mean discrepancy (MMD) measure to measure the discrepancy between source and target samples [70]. Differently, in [71], the signed-rank test and the Chi-square test have been used as similarity measures between the source and target domains. Despite the effectiveness of these approaches, their efficacy tends to decrease with the larger discrepancy between the source and target domains.

On the other hand, obtaining domain invariant features is a prevailing paradigm for addressing the domain shift problem. The key idea of this approach is to map the target domain samples to a new feature representation that is invariant from the source feature representation. By doing so, the classifier trained on the source samples can work well on the newly adapted target sample. This approach can be further categorized into three main approaches [72].

- Discrepancy-based methods: These methods leverage the popular domain discrepancy metric *Maximum Mean Discrepancy* (MMD) [30] to measure the discrepancy between the source and target distributions. Subsequently, the deep learning model is trained to minimize the MMD distance between the source and target domains to obtain domain-invariant feature representation. The first attempt in this area has been realized by Lu *et al.* who utilized the MMD distance to align the source and target features in addition to using a weight regularization loss to enforce representative features [73]. Li *et al.* leveraged 1D CNN to automatically extract features from the frequency domain input and employed MMD with a distance metric to align the source and target domains [74]. In [75], a two-phase approach that has been developed, the 1D CNN model is first pretrained on the source domain; then, the model is fine-tuned to minimize the MMD distance between the source and target domains. Another approach used a sparse filtering approach coupled with high-order Kullback-Leibler (KL) divergence to find domain invariant features in an unsupervised manner [76].

- Adversarial-based methods: Inspired by the generative adversarial networks (GANs) [77], these methods rely on adversarial training to align the source and target domains. Particularly, given a feature extraction network (maps

the input signals to the feature space) and a domain-classifier network (classifies between the source and target features). The main objective of adversarial domain adaptation is to optimize the feature extractor network to produce target features that can be invariant from the source features while training the domain classifier network to distinguish between the source and target features. The feature extraction network aims to make the source and target features indistinguishable from the domain classifier network. At this stage, a model that works well on the source features should generalize well to the newly adapted target features. Several approaches have realized adversarial domain adaptation for fault diagnosis. For instance, Guo *et al.* have leveraged adversarial training coupled with MMD distance to adapt between different machines [78]. Chen *et al.* proposed a domain adversarial training technique using the standard GAN loss with inverted labels to address the domain shift problem across different working conditions [79]. Cheng *et al.* addressed the instability of standard GAN loss by using Wesserestein distance during the adversarial training process to address the distribution shift between different working conditions [80]. Jiao *et al.* employed dual-domain classifiers during the adversarial training to ensure class separability while finding domain-invariant features [81]. Song *et al.*, integrated adversarial training with a novel retraining strategy to perform well across different working conditions [82]. A different set of methods have leveraged GANs network to first generate source-like data or target-like data, then train the deep learning using the newly generated data to improve the performance on the target domain [31, 83].

- Reconstruction-based methods: The key idea of this approach is using a normal autoencoder to reconstruct target samples from the source samples or vice-versa. By doing so, the autoencoder network can reach an intermediate feature representation that can be shared between the source and target domains. For instance, Wen *et al.* proposed a sparse autoencoder to extract features in an unsupervised manner while minimizing MMD between the extracted features to align the source and target domains [84]. Differently, Pang *et al.* leveraged stacked denoising autoencoder coupled with manifold learning to align both the marginal distribution and the conditional distribution across different working conditions [85].

The above approaches have achieved wide success and paved the way for deep learning models to handle domain shift problems across different machines or working conditions. Nevertheless, all existing methods work under single source *single* target settings (1S1T). As such, they need to re-train a new model for each working condition/target domain, which is not a viable solution in practice. This approach can have limited scalability under highly varying working environments, hindering the applicability of these approaches under highly dynamic environments.

2.4.3.4 Domain Generalization for Fault Diagnosis

While domain adaptation has shown wide success in addressing the distribution shift problem under dynamic environments, domain adaptation approaches assume access to unlabeled data from the target domain (i.e., working conditions) during the training stage. Such approaches cannot handle online prediction situations, where the model is required to act on the fly without retraining. Besides, in many real applications, data from target working conditions may not be readily available beforehand. The above limitations urge for a more generalizable data-driven approach that can handle new unseen working environments with no prior data available. Recently, a promising paradigm named Domain Generalization (DG) has emerged to improve the generalization ability of the data-driven approach to new domains with no prior data. A considerable amount of DG approaches have existed in Computer Vision Applications. Nevertheless, little attention has been paid to Domain Generalization for fault diagnosis tasks. Very recently, few works have realized domain generalization for fault diagnosis problems. For instance, Li *et al.* proposed a gradient reversal layer and a domain augmentation to improve generalization for machinery fault diagnosis [86]. While Liao *et al.* leveraged Wasserstein's generative adversarial network with gradient penalty to learn the invariant features to generalize well to new working conditions [87]. Differently, Zhang *et al.* relied on salience maps to remove superficial features and to improve the performance of the fault diagnosis task [88]. Han *et al.* leveraged adversarial learning and triplet loss to improve generalization performance on unseen target domains [89].

Most of these approaches enforce invariance between the global distribution between domains to improve the generalization performance. However, in fault classification tasks, the global distribution of each working condition encloses multiple

fine-grained distributions for each class. Furthermore, aligning the global distribution between different working environments does not obligate the alignment of these fine-grained class distributions within each domain. Consequently, the fine-grained class distributions can still be misaligned even if the global distributions are well-matched, yielding poor cross-domain performance for the trained model.

2.5 Data-driven Fault Prognosis

One key task in prognostics and health management (PHM) is the reliable prediction of *the remaining useful life* (RUL) of equipment. With accurate RUL estimation, industries can have predictive maintenance planning and thus prevent catastrophic failures or faults from happening [90]. Data-driven approaches have emerged for predicting the RUL of machines with increasing data availability in the digital era.

2.5.1 Deep Learning for Prognosis

Various traditional machine learning models have been used to estimate the RUL including the hidden Markov model [91], artificial neural network [92], and support vector machines [93]. However, these approaches suffer from extensive efforts for feature engineering. Deep learning with automatic feature extraction ability has achieved wide success in many fields, including computer vision, natural language processing, and speech recognition [27]. Various deep learning methods have recently been proposed for fault prognosis tasks. Yet, the majority of methods either used convolutional neural networks (CNNs) or recurrent neural networks (RNNs). Convolutional neural network (CNN) has widely acclaimed performance for RUL prediction. For instance, Li *et al.,* proposed a CNN with 1-D filters to extract features from input sensor data for RUL prediction and also used the window-time approach to prepare data samples for improved feature extraction [94]. Yang *et al.* developed a two-stage approach by using a CNN network to inspect the fault points and another CNN to estimate the RUL [95]. Zhu *et al.,* proposed a multi-scale CNN to extract features and predict the degradation of bearing [96]. Zhang *et al.,* combined multi-layer perception (MLP) and CNN to extract features from vibration data and predict the health index of the machine [97]. As shown

in the above studies, CNN-based methods have achieved outstanding performance for RUL prediction. However, they are still limited when dealing with the sequence data as they ignore the temporal dependency among data points in a given input sequence. Therefore, it is motivated to explicitly handle the temporal dependency of sequence data for RUL prediction. Recurrent neural networks (RNN) have been shown wide success in modeling dynamic systems and learning temporal dependency in the data. In particular, Long Short-Term Memory (LSTM), is a special type of recurrent model that can model the dynamics of sequences by introducing long short-term memory cells [98]. LSTM has become increasingly popular for RUL prediction. For instance, Zheng *et al.,* have used two layers of the LSTM network to predict the RUL of turbofan engines [99]. Huang *et al.,* employed stacked-bidirectional LSTM with auxiliary inputs to model sensor data under multiple operating conditions [100]. Miao *et al.* designed a deep LSTM framework to jointly perform degradation assessment and RUL prediction [101]. Other recent approaches have combined LSTM networks with CNN networks for RUL prediction. Al-Dulaimi *et al.,* proposed a two-parallel path approach with one for LSTM and one for CNN [102], while Liu *et al.,* combined CNN with LSTM in a series manner, by feeding the output convolutional features to a Bi-directional LSTM network aiming to improve the latent representation of the input sequence [103]. In addition to CNN and LSTM-based methods, other deep learning algorithms have also been developed for RUL prediction. For instance, Min *et al.,* presented denoising autoencoder-based deep neural networks (DNNs) with a two-stage approach to estimate the RUL of bearings [104]. Ma *et al.,* used a coupling autoencoder model on multimodal sensor data to perform the fault diagnosis task [105]. Encoder-decoder networks have been employed for health index prediction and RUL estimation [106, 107]. On the other hand, a deep belief network (DBN) has also been proposed together with ensemble techniques for RUL prediction [108].

While the aforementioned approaches have performed well on the RUL prediction tasks, they are not flawless for the following reasons. First, most of these approaches are only relying on a single supervised objective of a single-valued RUL label to extract the features and predict the RUL values. This can make the learned features very specific to the training data, reducing its generalization to the test data. Second, these approaches mainly attend to the latest information, while giving less importance to the historical data. Therefore, there is a need for a more general approach that can focus on the relative information in the long sequences.

2.5.2 Transfer Learning for Fault Prognosis

The above Deep learning approaches only work under two main assumptions. First, they assume the availability of a large amount of labeled data. This may not be attainable for many real applications due to the prolonged failure time of complex machines. Second, they usually assume training data and testing data are drawn from the same distribution. Yet, this assumption may not hold in many dynamic environments, where the variability of working environments is inevitable. Therefore, there is a need for more realistic approaches that work under dynamic environments with less labeled data. Domain Adaptation (DA) is a promising paradigm that can address the distribution shift problem with scarce labels. While dozens of works have addressed domain adaptation fault classification problems, only a few approaches have realized domain adaptation for remaining useful life estimation. For instance, Zhang *et al.* proposed a transfer learning approach for the RUL problem, where they trained the model on the source dataset and fine-tuned the model on target working conditions [109]. Yet, they assumed accessibility to labeled data for the target domain, which cannot hold for real-world scenarios. Recently, Costa *et al.* proposed a deep domain adaptation (DDA) method for the RUL prediction problem using unlabeled target domain data. The DDA applied the LSTM network to extract features and the reverse gradient approach to alleviate the domain shift problem [110]. Zhang *et al.*, developed a transfer learning approach to an incomplete target dataset based on a deep regularization technique [111]. Very recently, Siahpour *et al.* have used adversarial domain adaptation with consistency regularization to address the domain shift problem for RUL [112].

Nevertheless, most of these approaches leverage adversarial training to find domain-invariant features between the source and target domains. But simply enforcing the target features to be similar to the source with no constraints may remove useful target-specific information in the target domain, i.e., the mutual information between the target data and the target extracted feature. This can limit the performance of domain adaptation for the RUL prediction task

2.6 Summary

This chapter reviewed the existing literature on data-driven fault diagnosis and prognosis. Sections 2.4 and 2.5 surveyed different approaches for fault diagnosis including conventional machine learning, deep learning, and transfer learning. Particularly, while deep learning has outstanding performance for the fault classification task, it can only work under two main assumptions: (1) availability of a massive amount of labeled data; (2) stationarity of working environments. Yet, these assumptions may not hold in many practical situations. Domain adaptation can tackle these issues by transferring knowledge from rich-labeled domains to different unlabeled domains. Various domain adaptation techniques for fault diagnosis and prognosis have been investigated, including supervised domain adaptation, semi-supervised domain adaptation, and unsupervised domain adaptation, in Sections **??** and 2.5.2 respectively. Despite the efficacy of domain adaptation paradigm approaches, it mainly assumes access to unlabeled data from tasks of interest beforehand. This assumption limits the applicability of domain adaptation in a highly varying environment that requires on-the-fly predictions. Therefore, Section 2.4.3.4 reviewed a more challenging yet practical domain generalization approach, which can generalize to unseen domains with no prior data available. Table 2.2 summarizes the related works and their key limitations.

TABLE 2.2: Summary of related works and their key limitations

Category	Approaches	Methods	Limitations
Deep Learning	Convolutional Neural Networks	[46–52, 75, 96, 97, 113]	Vast amount of labeled data Limited with long sequences Prone to distribution shift
	Autoencoders	[14, 53–55, 57, 58, 73, 104, 114, 115]	Vast amount of labeled data Limited with long sequences Prone to distribution shift
	Recurrent Neural Networks	[15, 61, 62, 98, 99, 101–103, 107, 116, 117]	Vast amount of labeled data Computational Complexity Prone to distribution shift
Domain Adaptation	Instance Re-weighting	[69–71]	Degrade with large domain shifts
	Discrepancy Based Approach	[30, 73–76]	Computational Complexity Limited scalability Ignore Temporal Dependency
	Adversarial-Based Approach	[78, 80–82, 118]	Limited scalability Ignore Temporal Dependency
	Generative Approach	[31, 83–85]	Unstable training Limited scalability Computational Complexity eliminate domain-specifc features
Domain Generalization	Adversarial Approach	[86–89]	Prone to Class-conditional shift

The next few chapters will address the aforementioned unsolved challenges towards realizing reliable and realistic data-driven predictive maintenance. Starting with the fault prognosis task, Chapter 3 aims to tackle the limited feature representation and the disregard of the historical information over long sequences by proposing a dual-objective attention-based sequence-to-sequence approach. In Chapter 4, we address the lack of labeled data and variability of working conditions by proposing a novel adversarial domain adaptation for remaining useful life estimation. Moving to the fault diagnosis task, Chapter 5 tackles the domain shift problem under the lack of labeled data for the fault diagnosis task by proposing a novel self-supervised autoregressive domain adaptation approach. Chapter 6 addresses the limited scalability problem of the existing domain adaptation approaches by proposing a single-source multiple-target domain adaptation approach that can handle multiple working conditions concurrently. Last, in Chapter 7, we address the generalization to unknown working conditions with no prior data available by proposing a novel contrastive domain generalization approach that can generalize to new unseen working conditions.

Part I

Fault Prognosis Task

Prognosis originated from the Greek word *Prognostikos* which means foreseeing or forecasting. Fault prognosis is the process of predicting the potential degradation or failure of certain equipment or machine. By providing prior failure warning, fault prognosis enables remedial maintenance in advance. One key task in Prognostics is the reliable prediction of *remaining useful life* (RUL) of equipment. With accurate RUL estimation, industries can have prior maintenance planning and thus prevent catastrophic failures [119]. Albeit critical, precise prediction of the remaining useful life is becoming increasingly challenging especially with the evolving complexity of industrial machines.

Chapter 3

Attention-Based Sequence to Sequence Model for Machine Remaining Useful Life Prediction

In recent years, with the surge of computational power and the data volume, deep learning, with its hierarchical representative power and its ability to extract salient features, has shown strong potential for the RUL prediction task. Various deep learning methods, including convolutional neural networks (CNN) and recurrent neural networks (RNN), have been developed for the RUL prediction task [120, 121]. However, CNN-based approaches may not be able to capture long-range sequential dependencies in sensory data, resulting in a limited capability for RUL prediction. Meanwhile, recurrent neural networks including conventional RNNs, gated RNNs, and long short-term memory (LSTM) were developed to capture the temporal dependency among time series data [99, 100, 122–125]. However, LSTM-based methods only focus on the latest sequence information when mapping the whole input sequence into a fixed-length vector representation. As a result, they tend to lose relevant and important historical information when dealing with very long sequences [126]. In addition, all the aforementioned approaches including both CNN and LSTM-based methods only used a single objective, i.e., minimizing the mean square error (MSE) between the predicted and true values to train the model.

The work in this chapter has been published in Neurocomputing 466 (2021): 58-68.

I argue that using a single objective can limit the generalization performance of the model on unseen test data [127, 128].

To tackle the above limitations, this chapter presents a dual-objective sequence-to-sequence model named ATS2S for accurate RUL prediction. First, to address the shortage of LSTM with long sequences, attention-based decoding is proposed to focus on the important parts of the input sequence (instead of the latest information in LSTM) that can maximize the decoding performance without losing relevant information. Besides, the last hidden state of the decoder with the encoder's hidden features is integrated as a comprehensive *dual-latent feature representation* for the RUL predictor. Second, inspired by the success of auxiliary tasks in improving the generalization performance [127, 128] in computer vision applications, a novel auxiliary task is designed to further improve the prediction capability on unseen test data. Particularly, given the current input sequence, the model is trained to reconstruct the *future* input sequence in an unsupervised manner. Concurrently, a supervised mean square error (MSE) loss is adopted between the true RUL labels and the predicted ones.

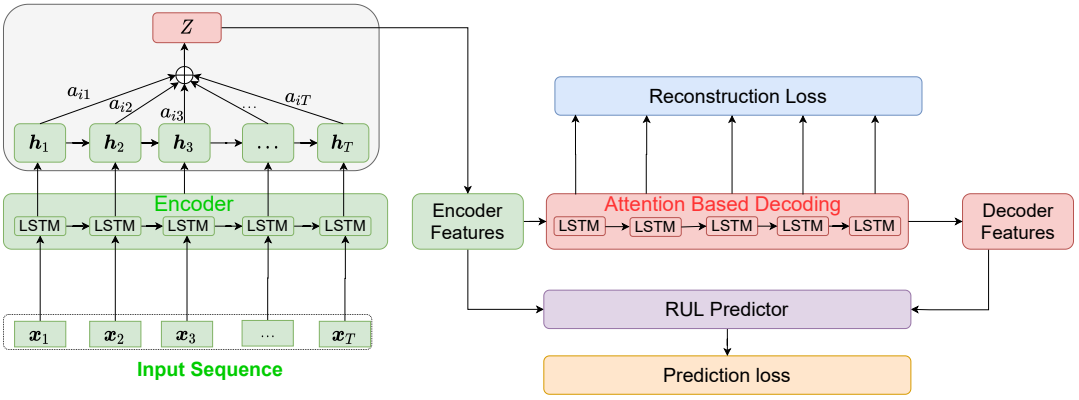


FIGURE 3.1: Attention-based sequence-to-sequence model for RUL prediction

Overall, this chapter provides the following contributions:

- A novel auxiliary task is proposed to predict the future sequence from the current input sequence, empowering the model predictive capabilities.
- An *attention mechanism* is built on top of the encoder-decoder network to handle the long sequences. As such, our model can focus on the most relevant information of the input sequences for the RUL prediction task.

- A *dual-latent feature representation* is introduced via integrating the encoder features and decoder hidden states to capture rich semantic information in the data for RUL prediction.
- Extensive experiments are conducted on four benchmark datasets to evaluate our proposed approach. The results show that the proposed approach can significantly improve RUL prediction over 13 state-of-the-art methods.

3.1 Attention Based sequence to sequence model with Auxiliary Task

This section provides a detailed explanation of the proposed model. Particularly, the LSTM encoder model and the attention-based decoding are first illustrated. Subsequently, the novel reconstruction-based auxiliary task and the RUL predictor model are described. Last, the joint optimization of the multi-objective loss is presented.

3.1.1 Overview of ATS2S

The proposed ATS2S is composed of three main components, namely, encoder, decoder, and RUL predictor, as shown in Fig. 3.1. First, the encoder maps the whole input sequence into a sequence of hidden states. Unlike conventional encoder-decoder models that compress all the input information into a single fixed-length vector (i.e., the encoder's last hidden state), we design an *attention layer* to select the hidden states that are relevant and important for the decoding. Then, we pass the weighted sum of the encoder's hidden states (i.e., attention outputs) as *encoder features* to the decoder. The decoder is then trained to forecast the *next* input sequence given the current input sequence, in order to give our model more *predictive power*. Finally, the RUL prediction network (a fully connected neural network) takes *dual-latent feature representation* via integrating both the encoder and decoder's hidden states/features for RUL prediction. The predictor maps from the feature dimension space to a single value, i.e., predicted RUL.

Note that our ATS2S method jointly optimizes the RUL prediction loss, which is the difference between the predicted RUL label and ground-truth label, as well as the reconstruction loss, which is the difference between the predicted and actual sequence. Next, we will introduce each of the three components of ATS2S in detail.

3.1.2 LSTM Based Encoder

In order to model the input dynamics of sensor signals, we employ the LSTM model as our backbone architecture in the sequence-to-sequence model. Given an input sample $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) \in \mathbb{R}^{n \times T}$, $\mathbf{x}_t \in \mathbb{R}^n$ is n-dimensional input vector at each time step t ($1 \leq t \leq T$) from n sensors. At each time step t, LSTM takes the input vector $\mathbf{x}_t$ and *previous* hidden state $\mathbf{h}_{t-1}$ to produce *current* hidden state $\mathbf{h}_t$, current long term memory cell $\mathbf{c}_t$ and output $\mathbf{o}_t$, as shown in Fig.3.2. The following equations demonstrate the detailed process in the LSTM cell.

$$\mathbf{i}_t = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i), \tag{3.1}$$

$$\mathbf{f}_t = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f), \tag{3.2}$$

$$\mathbf{o}_t = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o), \tag{3.3}$$

$$\mathbf{g}_t = \delta(W_g \mathbf{x}_t + U_g \mathbf{h}_{t-1} + \mathbf{b}_g), \tag{3.4}$$

$$\mathbf{c}_t = \mathbf{o}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t, \tag{3.5}$$

$$\mathbf{h}_t = \mathbf{f}_t \odot \delta(\mathbf{c}_t), \tag{3.6}$$

where σ is nonlinear *sigmoid* function, $\odot$ is an element-wise multiplication operator, $\mathbf{W}_* \in \mathbb{R}^{n \times p}$ (i.e., $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$ and $\mathbf{W}_g$) are the model parameters that map from input dimension n to hidden dimension p, $\mathbf{U}_* \in \mathbb{R}^{p \times p}$ map from the previous hidden state to the current hidden state, and $\mathbf{b}_* \in \mathbb{R}^p$ are bias vectors. It is worth noting that the parameters are shared across all the time steps.

The Encoder model f_{enc} takes the input sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ and produces a sequence of hidden states $(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T)$ and a sequence of cell states $(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_T)$ in Equation (3.7).

$$[(\mathbf{h}_1, \ldots, \mathbf{h}_T), (\mathbf{c}_1, \ldots, \mathbf{c}_T)] = f_{enc}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T; \boldsymbol{\theta}_{enc}), \tag{3.7}$$

where $\boldsymbol{\theta}_{enc} = [\mathbf{W}_{enc}, \mathbf{U}_{enc}, \mathbf{b}_{enc}]$ are the parameters of the Encoder model.

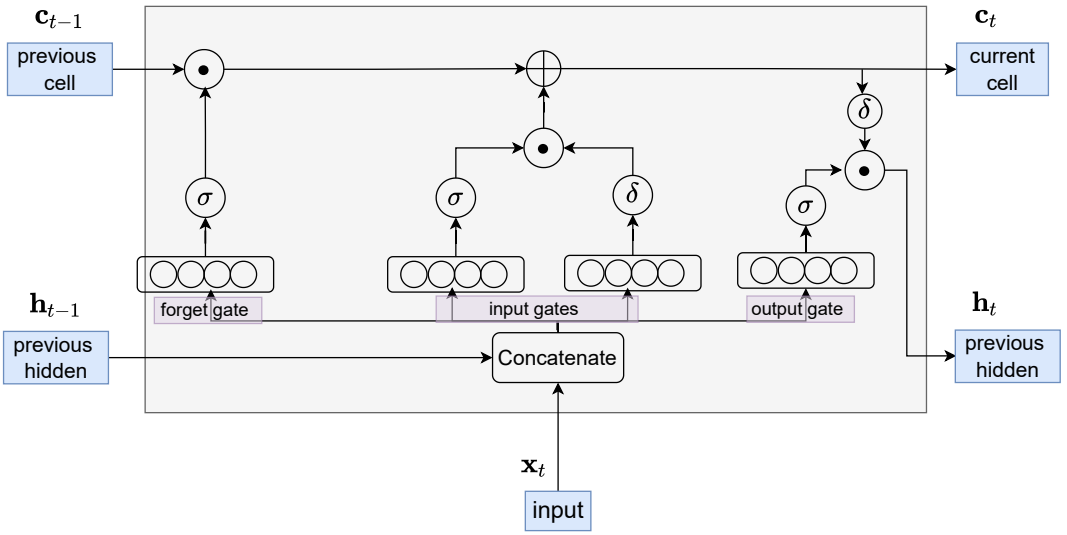


FIGURE 3.2: Structure of LSTM cell

3.1.3 Decoding

The main idea of attention is inspired by human visual systems where humans can focus on the relevant part of a scene and ignore irrelevant parts. Similarly, we design an attention mechanism in our sequence-to-sequence model for the whole sequence of hidden states. In particular, we focus on *all the important* hidden states of the encoder for decoding, while the standard sequence-to-sequence model relies solely on the *last* hidden state and thus loses valuable information.

3.1.3.1 Calculation of attention weights

For the decoding process, we employ the hidden states from both the encoder and decoder to produce the attention weights. Note that each decoding time step will have different attention weights. For instance, in the decoding time step i, the attention weights $\boldsymbol{a_i} = [a_{i1}, a_{i2}, \ldots, a_{iT}]$ can be calculated by the attention module $f_{attn}(\cdot)$, which can be expressed as

$$\boldsymbol{a_i} = f_{attn}(\mathbf{s}_{i-1}, \mathbf{H}), \tag{3.8}$$

where $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T] \in \mathbb{R}^{p \times T}$ represents the encoder hidden states, and $\boldsymbol{s}_{i-1} \in \mathbb{R}^p$ is the previous decoder hidden state which is initialized by the last encoder hidden state at the very beginning of the decoding process. Here, p is the

dimension of each hidden state and T is the total number of time steps for one sample. Fig. 3.3 shows the detailed structure of the attention module $f_{attn}(\cdot)$. Specifically, the decoder hidden state $\boldsymbol{s}_{i-1}$ will be concatenated with each encoder hidden state and then passed through a fully connected layer $FC : \mathcal{R}^{2p} \rightarrow \mathcal{R}$. The outputs of the fully connected layer will be fed into a softmax layer which produces the final attention weights $\boldsymbol{a_i}$.

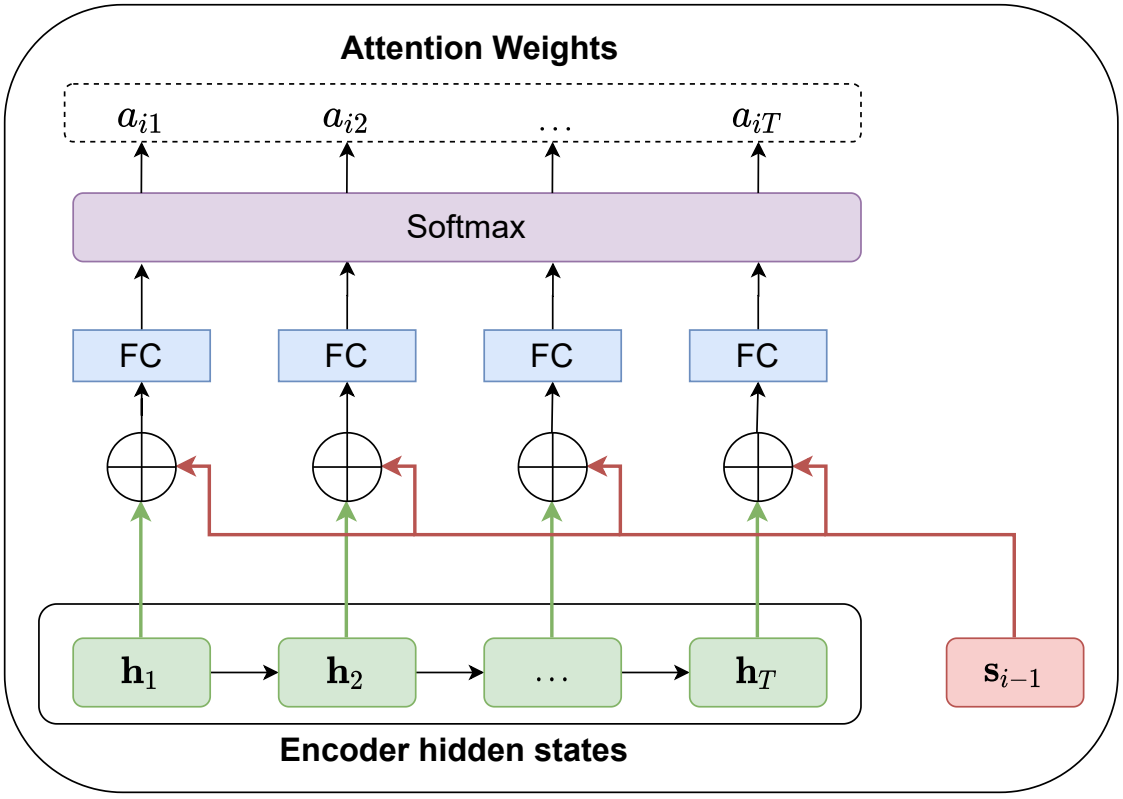


FIGURE 3.3: The detailed structure of the attention module.

3.1.3.2 Attention based decoding

For each time step i in the decoding process, we employ the attention weights $\boldsymbol{a}_i$ and the encoder hidden states $\mathbf{H}$ to calculate the context vector $\boldsymbol{z}_i$ as follows:

$$\boldsymbol{z}_i = \sum_{j=1}^{T} a_{ij} \boldsymbol{h}_j, \tag{3.9}$$

where $\mathbf{Z} = [\boldsymbol{z}_i, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_{i+T-1}]$ is a collection of context vectors for all the time steps. For the i-th time step, the context vector $\boldsymbol{z}_i$ will be concatenated with the

current input $\hat{\boldsymbol{y}}_i$ which is the prediction of the previous step. Then, the concatenated vector and the previous hidden state $\boldsymbol{s}_{i-1}$ will be passed through the decoder cell, which can be formalized as:

$$\boldsymbol{s}_i = f_{dec}((\hat{\boldsymbol{y}}_i, \mathbf{z}_i, \mathbf{s}_{i-1}); \boldsymbol{\theta}_{dec}). \tag{3.10}$$

where θ_{dec} represents the parameters of the decoder network. Then, we map from $\mathbf{s}_i$ to the next step of the target $\hat{\mathbf{y}}_{i+1}$ in Equation (3.11):

$$\hat{\mathbf{y}}_{i+1} = f_{FC}(\mathbf{s}_i; \boldsymbol{\theta}_{FC}), \tag{3.11}$$

where f_{FC} is a fully connected layer that maps from the hidden dimension to the output dimension, and θ_{FC} represents the parameters of the fully connected network. It is worth noting that we pass the output of the last time step as the next input. Hence, the decoder is trained to predict the future step given the current input which can be valuable for the RUL prediction.

3.1.4 RUL Predictor

The objective of the RUL predictor is to accurately predict the corresponding RUL value for each input sequence (sensor signals). We first integrate the last hidden state of the decoder with the encoder's hidden features, as a comprehensive *dual-latent feature representation*, and then design a function that maps the dual-latent feature representation to a single RUL value. We denote the RUL predictor as $f_{pred} : \mathbb{R}^D \rightarrow \mathbb{R}$ in Equation (3.12), where D is the dimension of dual-latent feature representation.

$$\widehat{RUL} = f_{pred}((\mathbf{h}_T, \mathbf{s}_T); \boldsymbol{\theta}_{pred}), \tag{3.12}$$

where $\widehat{RUL} \in \mathbb{R}$ is the predicted label, $\mathbf{h}_T$ and $\mathbf{s}_T$ are the features of encoder and decoder respectively. Fig. 3.4 shows the diagram of the RUL predictor, which is a multi-layer feed-forward network followed by a non-linear activation function (i.e., ReLU).

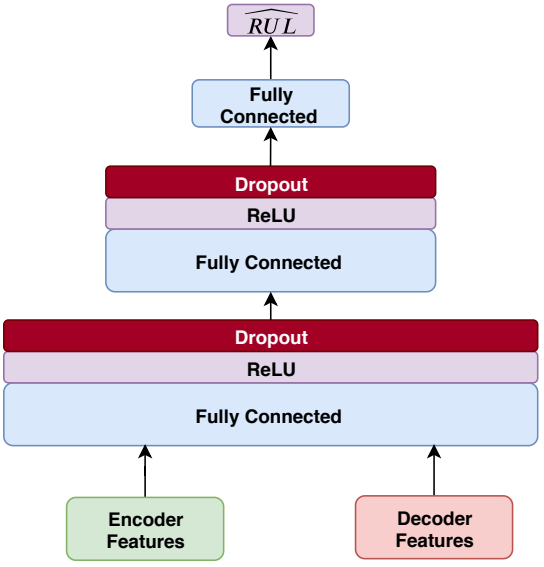


FIGURE 3.4: The architecture of RUL predictor network

3.1.5 Multi-objective Optimization

3.1.5.1 Reconstruction Loss

In our ATS2S, we aim to forecast the next input sequence given the current input sequence so that our model has predictive power. Fig. 3.5 shows the detailed process of the forecasting-based reconstruction loss. Given a predicted sequence by the decoder $\hat{\mathbf{Y}}_i = (\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \cdots, \hat{\mathbf{y}}_T) \in \mathbb{R}^{n \times T}$, and a target sequence $\mathbf{Y}_i = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_T) \in \mathbb{R}^{n \times T}$ where $\mathbf{y}_t = \mathbf{x}_{t+1} \in \mathbb{R}^n$, T is the length of the sequence, and n is the number of sensors. We define the reconstruction loss as the mean square error between the target output and predicted output. Equation (3.13) shows the formulation of the reconstruction loss.

$$L_{rec}(\theta) = \frac{1}{N} \sum_{i=1}^{N} ||\hat{\mathbf{Y}}_i - \mathbf{Y}_i||_2^2, \tag{3.13}$$

where θ is the model parameters, and N is the total number of samples.

3.1.5.2 RUL Prediction Loss

The RUL prediction loss is defined as the mean square error between the true RUL label and the predicted RUL label for each input sequence. The RUL loss can be

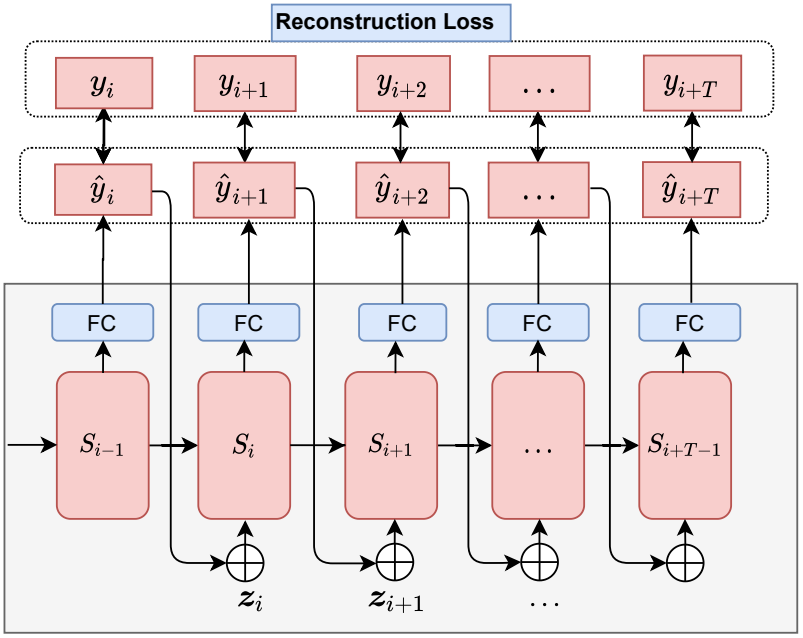


FIGURE 3.5: Details of the forecasting-based reconstruction task

defined as follows:

$$L_{rul}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(\widehat{RUL}_i - RUL_i)^2 \tag{3.14}$$

where $\widehat{RUL}_i$ is the predicted label and RUL_i is the true label.

3.1.5.3 Joint Loss

The proposed model aims to optimize both reconstruction and prediction losses concurrently. We argue that jointly optimizing both losses can not only provide a good and rich latent representation but also improve the accuracy of RUL prediction. The joint loss can be formulated as follows

$$L(\theta) = \alpha L_{rec}(\theta) + L_{rul}(\theta), \tag{3.15}$$

where α is a tunable parameter to control the contribution of the reconstruction loss. It can control the contribution from reconstruction loss while maintaining the prediction loss (the major loss for RUL prediction).

3.2 Experiments and Results

3.2.1 Experimental Data

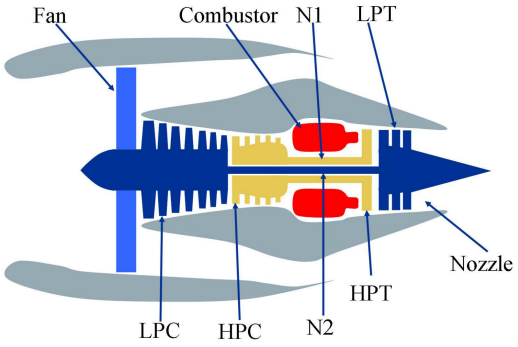


FIGURE 3.6: Diagram of the engines in C-MAPSS data [1].

We evaluate our proposed ATS2S method on C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) data [1]. C-MAPSS data describes the degradation process of aircraft engines as shown in Fig. 3.6. It consists of four benchmark datasets with different numbers of training/testing engines, operating conditions, and fault types. The details about these four datasets are summarized in Table 3.1.

TABLE 3.1: Properties of C-MAPSS Dataset

Dataset	FD001	FD002	FD003	FD004
# Training engines	100	260	100	249
# Testing engines	100	259	100	248
# Operating conditions	1	6	1	6
# Fault types	1	1	2	2

3.2.1.1 Sensor Data Selection

Twenty-one sensors are deployed in different locations of the engine to measure temperature, pressure, and speed. To select the most informative sensors, we visualize the sensor readings of randomly selected engines. Fig. 3.7 and Fig. 3.8 show the sensor readings from FD001 and FD002 subsets respectively. Clearly, some sensors are almost constant during the whole degradation, which can hinder the model from correctly modeling the deterioration process. While the informative

sensors are those sensors that can show a clear degradation trend from run to failure. Therefore, 14 sensors, namely sensors S2, S3, S4, S7, S8, S9, S11, S12, S13, S14, S15, S17, S20, and S21, are used for RUL prediction. We used the same set of sensors for the FD003 data set as it follows the same degradation patterns as FD001. A similar procedure has been done for FD002 and FD004. Eventually, we adopt 9 sensors [100], namely sensors 3, 4, 9, 11, 14, 15, 17, 20, and 21 for RUL prediction on FD002 and FD004.

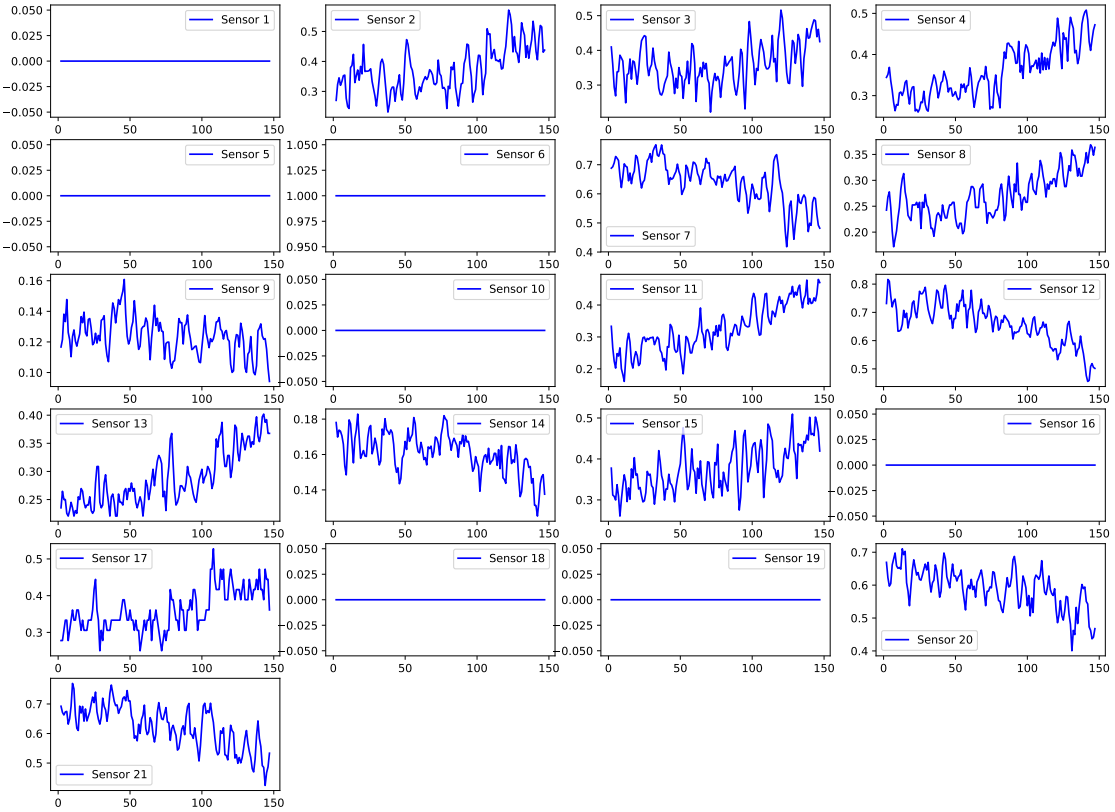


FIGURE 3.7: The readings of 21 sensors for a randomly selected engine in the FD001 dataset

3.2.1.2 Data Segmentation and Processing

We follow the sliding window method [129, 130] for data segmentation. Fig. 3.9 shows the process of data segmentation with a sliding window, where W is the window size, n is the number of sensors, and s is the shifting size. Given that the total number of cycles is T, the RULs for the first and second windows/samples are thus $T - W$ and $T - W - s$, respectively. In our experiments, W and s are set to be 30 and 1, respectively.

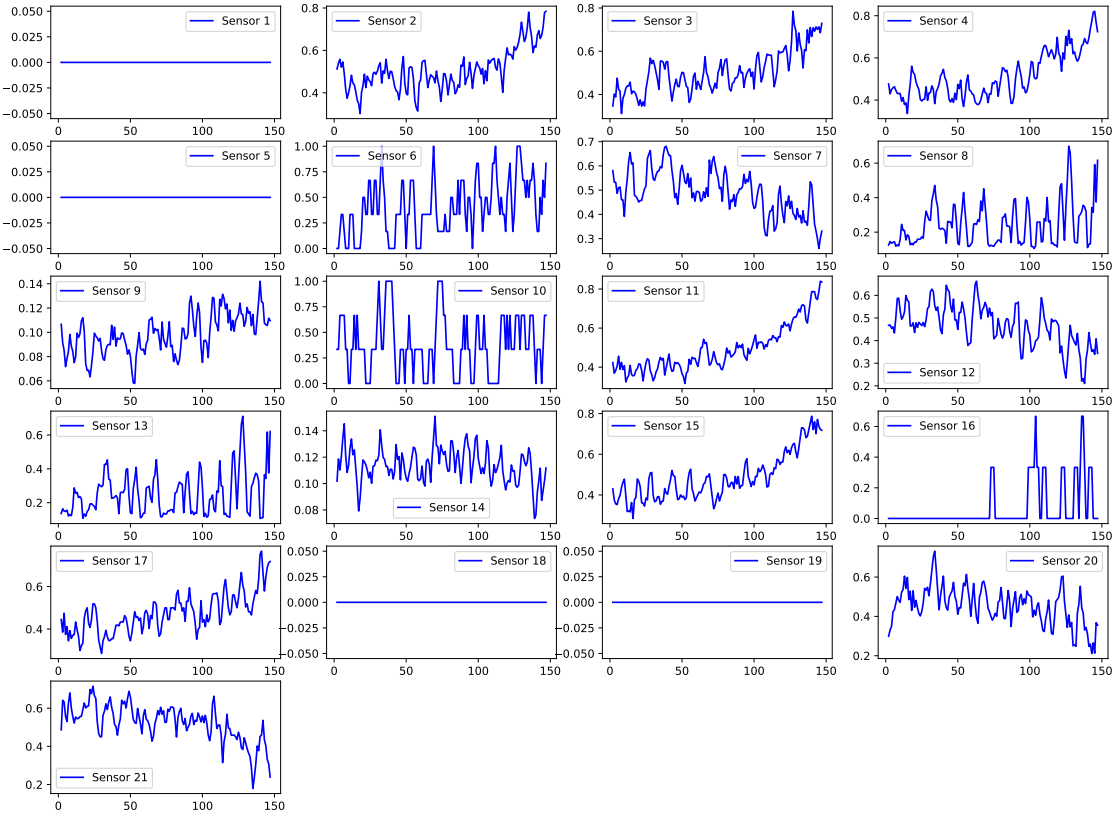


FIGURE 3.8: The readings of 21 sensors for a randomly selected engine in the FD002 dataset

Moreover, we adopt the piece-wise linear degradation model [100] for the RUL labels. In case a sample has an RUL value greater than a pre-defined threshold, we reset the RUL value as the threshold for this sample. In particular, we follow the previous studies [100] and set the threshold as 125 for FD001/FD003 and 130 for FD002/FD004.

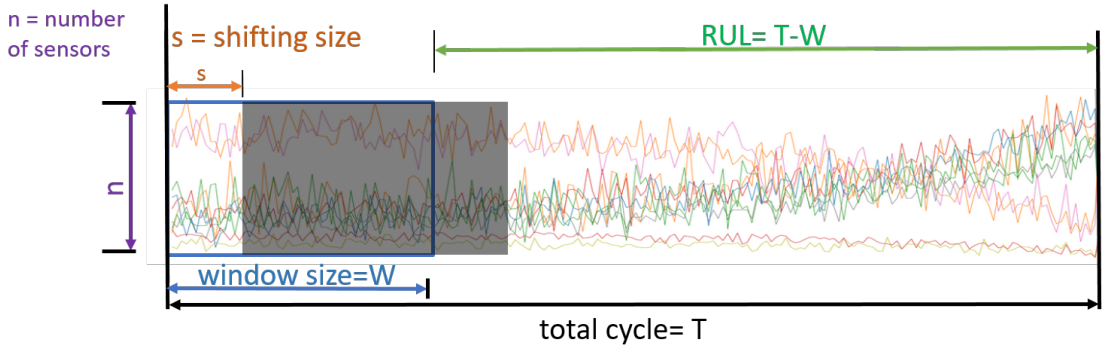


FIGURE 3.9: Data segmentation using the sliding window for RUL prediction

3.2.1.3 Data Normalization

The prognostic problem of real systems involves different types of sensors and different operating conditions. Directly feeding the raw sensor readings with high variance to the machine learning models may hinder the learning process and affect the model performance. To remedy this issue, we use min-max normalization for each sensor to restrict the values within $[0, 1]$. For datasets with multiple working conditions, we normalize the sensor readings with respect to their corresponding working condition. In particular, we first group the sensors by their corresponding working conditions, then we apply normalization to each cluster independently. To formulate the scaling function, let a vector $\mathbf{Q}_{rm}$ contain all the data points of the r-th sensor under m-th working conditions. The normalized vector $\hat{\mathbf{Q}}_{rm}$ is calculated as follows:

$$\hat{\mathbf{Q}}_{rm} = \frac{\mathbf{Q}_{rm} - min(\mathbf{Q}_{rm})}{max(\mathbf{Q}_{rm}) - min(\mathbf{Q}_{rm})}. \tag{3.16}$$

3.2.2 Experimental Settings and Evaluation Metrics

3.2.2.1 Experimental Settings

Our architecture is composed of three main parts, namely, the encoder network, decoder network, and RUL predictor network. Both encoder and decoder networks rely on the LSTM model. To reconstruct the next input sample, the decoder network is followed by a single layer fully connected (FC) network to map from the hidden dimension to the output dimension. The attention mechanism is implemented by two FC networks, i.e., one network computes the attention weights with a dimension of $n \times 30$, while the other network generates a weighted sum of the encoder hidden states using attention weights. Finally, the RUL predictor network consists of three FC layers, and each layer is followed by a rectified linear unit (ReLU) to increase complexity. Adam optimizer is used to optimize the overall model with a learning rate of $3e-4$. Moreover, a dropout regularization algorithm is employed to relieve the over-fitting problem. Table 3.2 summarizes all the hyper-parameters in our ATS2S model. All the experiments have been conducted using PyTorch 1.7 on NVIDIA GeForce RTX 2080 Ti GPU.

TABLE 3.2: Hyper-parameters of the proposed approach

Hyper-parameters	Range
Batch size	{10}
Learning rate	{0.0003}
Training epochs	{10, 20}
Dropout rate	{0.2, 0.5}
Sequence length	{30}
α	{1}
Number of layers (Encoder and Decoder)	{1}
Number of hidden units (Encoder and Decoder)	{18, 32}
Number of layers (Attention Model)	{2}
Number of hidden units (Attention Model)	L1{30}, L2{9, 14}
Number of layers (RUL predictor)	{2, 3}
Number of hidden units (RUL predictor)	L1{18,32}, L2{18,1}, L3{1}

3.2.2.2 Performance Metrics

We employ two standard metrics, namely, root mean square error (RMSE) and the Score, to evaluate the performance of various methods for RUL prediction. RMSE is defined following Equation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\widehat{RUL}_i - RUL_i)^2} \tag{3.17}$$

where $\widehat{RUL}_i$ and RUL_i are the predicted RUL and true RUL respectively, and N is the total number of samples. For machine prognosis and RUL prediction, late prediction of RUL (e.g., the predicted RUL is longer than the actual RUL) can lead to catastrophic consequences compared to early prediction. However, RMSE is not able to distinguish between early and late predictions. Hence, it requires an asymmetric evaluation function to give a larger penalty for overestimation. To address this issue, a score metric has been developed, which was firstly proposed by the PHM community during the 2008 PHM data challenge competition [1]. Recently, much-related research has adopted the scoring metric to evaluate the

performance of a model on the RUL prediction task [100], [130]. The scoring metric can be formalized as follows:

$$Score = \begin{cases} \sum_{i=1}^{N}(e^{-\frac{error_i}{13}} - 1), & \text{if } (error_i \leq 0) \\ \\ \sum_{i=1}^{N}(e^{\frac{error_i}{10}} - 1), & \text{if } (error_i > 0) \end{cases} \tag{3.18}$$

where $error_i = (\widehat{RUL}_i - RUL_i)$ is the difference between the predicted value (i.e. $\widehat{RUL}_i$) and the true value (i.e. RUL_i.)

3.2.3 Comparison Against State-of-the-arts

TABLE 3.3: Comparison among various methods in terms of RMSE and Score

Category		RMSE				Score			
	Method	FD001	FD002	FD003	FD004	FD001	FD002	FD003	FD004
Traditional ML	SVM [129, 130]	40.72	52.99	46.32	59.96	7703	316483	22542	141122
	RF [129]	17.91	29.59	20.27	31.12	480	70457	711	46568
	GB [129, 130]	15.67	29.09	16.84	29.01	474	87280	577	17818
CNN methods	2D CNN [129]	18.45	30.29	19.82	29.16	1287	13570	1596	7886
	1D CNN [120]	**12.61**	22.36	12.64	23.31	274	10412	284	12466
LSTM methods	D-LSTM [99]	16.14	24.49	16.81	28.17	338	4450	852	5550
	LSTMBS [116]	14.89	26.86	15.11	27.11	481	7982	493	5200
	BLSTM [100]	N/A	25.11	N/A	26.61	N/A	4793	N/A	4971
Ensemble methods	MODBNE [130]	15.04	25.05	12.51	28.66	334	5585	422	6558
Encoder-decoder methods	BiLSTM-ED [107]	14.74	22.07	17.48	23.49	273	3099	574	3202
Hybrid CNN-LSTM methods	CNN-LSTM [131]	14.4	27.23	14.32	26.69	290	9869	316	6594
	BLCNN [132]	13.18	19.09	16.76	20.97	302	1558	381	3859
	HDNN [102]	13.02	15.24	12.22	18.16	245	1282	288	1527
Proposed	**ATS2S**	12.63	**14.65**	**11.44**	**16.66**	**243**	**876**	**263**	**1074**
IMP		**-0.15%**	**3.87%**	**6.4%**	**8.3%**	**0.82%**	**31.6%**	**8.7%**	**29.7%**

In this section, to comprehensively evaluate our proposed ATS2S method, we compare it against 13 state-of-the-art methods, which can be classified into 6 categories as follows.

- Traditional machine learning (ML) methods. Three shallow models are employed in the comparison, including support vector machine (SVM) [130], random forest (RF) [130], and gradient boosting (GB) [130].
- CNN-based methods. A 2D CNN network was used in [129] to predict the RUL for turbofan engines, while Li *et. al,* used 1D CNN with multiple channels for RUL prediction [120].

- LSTM-based methods. A standard LSTM network [99] and a bi-directional LSTM [100] were developed for RUL prediction. In [116], LSTM is augmented with a bootstrap algorithm to predict the RUL values.
- Ensemble methods. A deep belief network (DBN) is used together with ensemble techniques for the RUL prediction task [130].
- Hybrid CNN-LSTM-based methods. A combination of CNN and LSTM models has been used for RUL prediction. CNN and LSTM can be cascaded in a sequential manner, e.g., CNN-LSTM [131] put CNN in the first stage, while BLCNN [132] reversed the order. In addition, HDNN [102] combined both the features from CNN and LSTM to generate the final predictions.
- Encoder-decoder-based methods. BiLSTM-ED [107] first extracts the health index and then estimates the health index curves using a linear regression model. Finally, it uses curve-similarity matching to estimate the RUL.

Table 3.3 shows the comparison among the above methods for RUL prediction. Note that the highest score in each column is in **bold**, while the second-best score is underlined. We have used the same datasets and experimental settings of the compared approaches to ensure a fair comparison. Hence, in Table 3.3, we have directly reported their published results.

We can observe that our proposed ATS2S outperforms all the other methods consistently, except that it achieves a comparable RMSE with 1D CNN [120] on the FD001 dataset. In particular, our ATS2S achieves significant improvement over the state-of-the-art on FD002 and FD004, which are two complex datasets with multiple working conditions and thus indicate more practical scenarios. For example, ATS2S is able to achieve improvements over the second-best performer on FD004 by 8.3% and 29.7% in terms of RMSE and Score, respectively. Such improvements on FD002 and FD004 demonstrate that ATS2S has clear advantages over the competing methods to handle complex datasets. In addition, compared with the RMSE metric, our ATS2S achieves even better improvements in terms of the Score metric, indicating that we can better address the issue of late predictions.

To further evaluate the complexity of our proposed model, we have compared it with some state-of-the-art methods for RUL prediction, i.e., BLSTM, HDNN,

BLCNN, BiLSTM-ED, and D-LSTM, in terms of the number of model parameters. The results are shown in Table3.4. It can be clearly seen that our model requires fewer parameters, which indicates its efficiency. In a nutshell, our ATS2S outperforms existing state-of-the-art for RUL prediction in terms of RMSE and Score without requiring an additional computational burden.

To further show the efficacy of our proposed approach, we have visualized the predicted RUL against the true RUL for test engines among four different datasets, as shown in Fig.3.10 It is worth noting that we have sorted the RUL values of the test engines in descending order for clearer visualization. It can be clearly seen that our predicted RUL values are well aligned with the true RUL values for all four datasets.

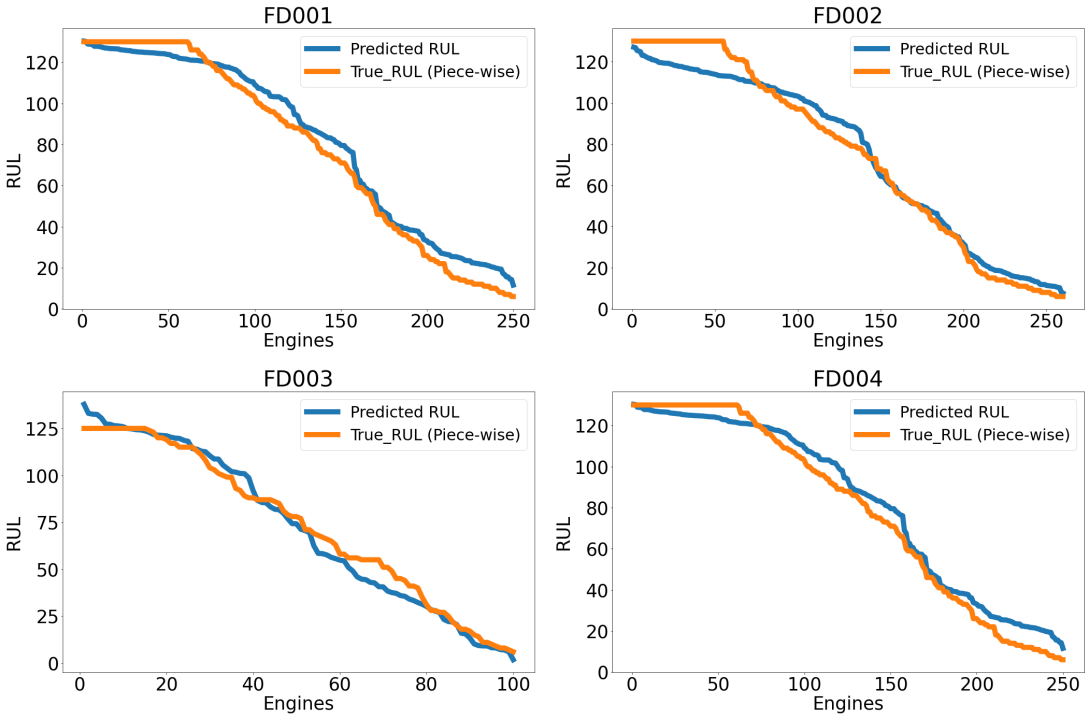


FIGURE 3.10: Comparison between predicted RULs of the proposed model and the actual RULs. Each point represents a test engine and its corresponding RUL. The test engines are sorted in descending order based on their RUL values.

TABLE 3.4: Comparison of the number of parameters between the proposed method and some state-of-the-art methods.

Model	ATS2S	BLSTM	HDNN	BLCNN	BiLSTM-ED	D-LSTM
Number of model Parameters	**13628**	29053	54766	16196	345600	14865

3.2.4 Model Analysis

3.2.4.1 Ablation Study

In this section, we disentangle the contribution of each part of the ATS2S model. In addition to the ATS2S model, we further derive three variants: (1) Basic sequence to sequence model without reconstruction or attention, (2) Basic model with reconstruction, (3) Basic model with attention. Fig. 3.11 shows the comparison between these 3 variants and the proposed ATS2S model. Based on the comparison shown in Fig. 3.11, we can further draw two conclusions.

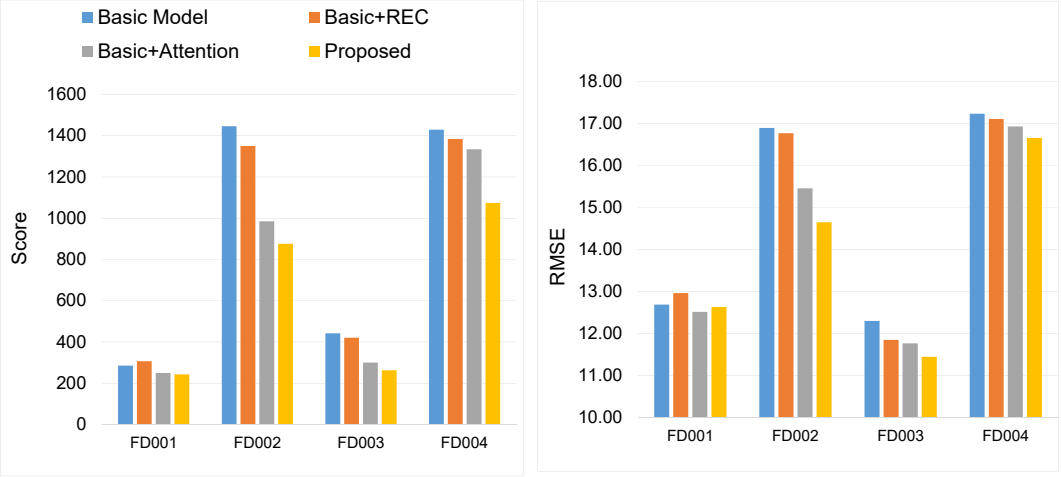


FIGURE 3.11: Ablation study for the proposed ATS2S method

First, our proposed ATS2S model with both attention mechanism and reconstruction architecture achieves the best performance over 4 datasets in terms of both metrics, showing that it is indeed more effective for RUL prediction than the basic sequence-to-sequence model. This demonstrates that learning from the most relevant information from long signals by attention mechanism, not just focusing on the latest information, as well as enabling predictive power and capturing temporal dependencies by reconstruction architecture, are critical for improving RUL prediction.

Second, the model with attention mechanism outperforms the model with reconstruction architecture, indicating that the attention mechanism has a larger impact than the reconstruction task in our ATS2S model. Without the attention mechanism, we squash the whole input sequence into a single hidden vector (i.e., the *last* hidden state of the encoder). Instead, the attention mechanism can consider

all the hidden states with different weights and help to learn better comprehensive *dual-latent feature representation* from both the encoder and decoder for RUL prediction.

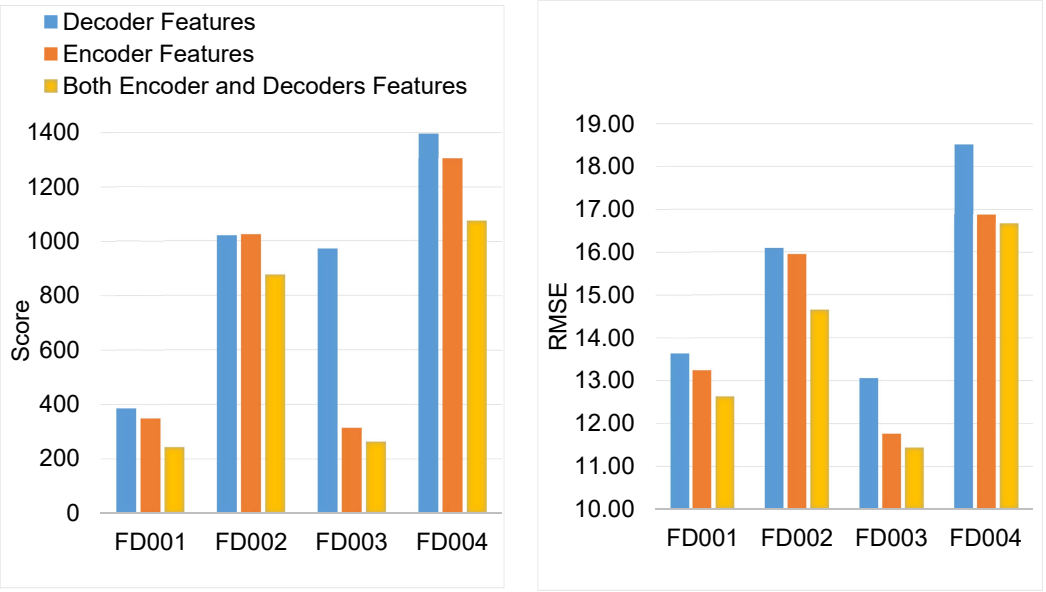


FIGURE 3.12: Study of feature importance of the proposed method

3.2.4.2 Feature Importance Analysis

As shown in Fig.3.4, we use the *dual-latent feature representation* to integrate features from both the encoder and decoder for RUL prediction. To study the importance of the features used in our ATS2S, we conduct experiments using three different feature sets, namely, encoder features (i.e., encoder hidden states), decoder features, and integrated features, i.e., encoder-decoder features (dual-latent feature representation). Fig. 3.12 shows the detailed model performance with three different feature sets. We can observe that dual-latent feature representation achieves the best performance overall in four data subsets consistently, indicating the importance of a comprehensive representation with rich semantics from both encoder and decoder features.

3.2.4.3 Sensitivity Analysis

As shown in Equation (3.15), the parameter α controls the contribution of reconstruction loss in the final joint loss. In this section, we perform the sensitivity

analysis for this parameter α. Fig. 3.13 shows the performance of the ATS2S model across four datasets with different values for α. Overall, it can be clearly observed that equal contribution from both reconstruction and prediction loss (i.e., $\alpha = 1$) achieves the best performance, demonstrating that both of them are critical for accurate RUL predictions.

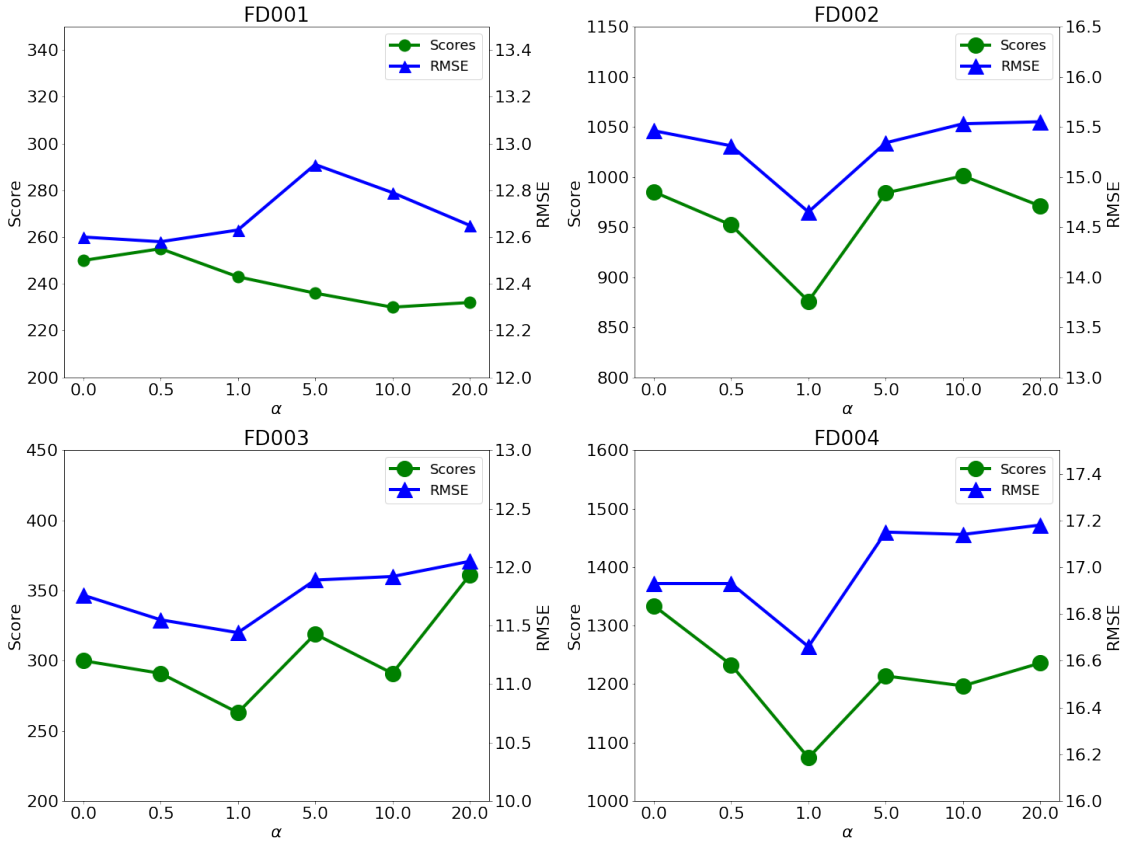


FIGURE 3.13: Sensitivity analysis of reconstruction weight

3.2.4.4 Attention weights

To demonstrate our model's capability of capturing long-term dependencies, we have visualized the attention weights among different time steps. Fig. 3.14 shows the attention weights of a randomly selected sample at one decoding time step. It can be found that the model pays more attention to previous time steps. This indicates that the attention mechanism helps the model to capture the long-term dependencies of the data.

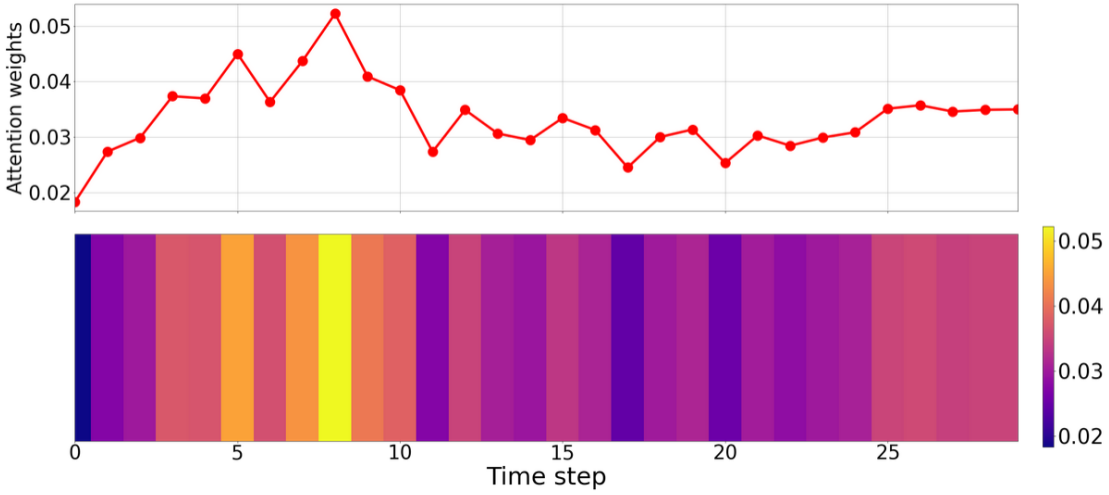


FIGURE 3.14: Illustration of attention weights for a randomly selected sample.

3.3 Summary

In this work, we presented a novel attention-based sequence-to-sequence model ATS2S to accurately predict equipment RUL. In particular, we designed a novel framework that learns to reconstruct the next sequence and predict the RUL labels concurrently. In addition, we showed that our attention mechanism can better capture all the relevant historical information from long sensor sequences compared to the standard LSTM approach that focuses only on the latest information only. Finally, our *dual-latent feature representation* which integrates both the encoder and decoder features is very effective for RUL prediction. Our extensive experimental results demonstrate that our proposed ATS2S significantly outperforms 13 state-of-the-art for RUL predictions across 4 benchmark datasets consistently. One limitation of the work in this chapter is the assumption that a large amount of labeled data are always available. This assumption may not be feasible for many real applications. As annotating time series data can be very labor intensive even for experts. Therefore, the next chapter will be addressing the lack of labeled data under variable working conditions.

Chapter 4

Contrastive Adversarial Domain Adaptation for Machine Remaining Useful Life Prediction

The development of ATS2S has paved a way for deep learning models to accurately estimate the remaining useful life of machines. Nevertheless, ATS2S only works well under two main assumptions: (1) training and testing data are collected under the same operating condition; (2) rich-labeled data are available for the RUL prediction task. These assumptions can be impractical for many real-world applications for the following reasons. First, the collection of labeled data (failures) is expensive. For some complex and critical machines, running to failure can be costly and cause catastrophic consequences [133, 134]. Second, the labeled data may only be available under a specific working condition, which can be leveraged to build a model for RUL prediction. However, when the working condition changes, the previously trained model often cannot work well, due to the distinct data distributions for different working conditions [67, 133, 135]. With the aforementioned problems, the RUL prediction for scarce-labeled machines/working conditions can be very challenging. Therefore, there is an urgent need for a prognostic model that is able to estimate the RUL of new working conditions with no labeled data

The work in this chapter has been published in 2020 IEEE International Conference on Prognostics and Health Management (ICPHM), 2020, and in IEEE Transactions on Industrial Informatics, vol. 17, no. 8, pp. 5239-5249, Aug. 2021, and

available. Domain adaptation (DA) enables knowledge transfer from a rich-labeled domain to a different but related scarce-labeled domain [67], providing a promising solution for this problem. Most of the existing DA algorithms are designed for image-related tasks [136]. Recently, some approaches extended DA for fault diagnosis problems (classification problems) to classify faults among different machines or working conditions [137, 138]. However, less attention has been paid to domain adaptation for the RUL prediction which is a typical time-series regression problem.

To mitigate this gap, this chapter presents a novel contrastive adversarial domain adaptation (CADA) approach for machine RUL prediction across different working conditions. CADA aims to transfer the knowledge learned from one working condition to solve the RUL prediction problem in another working condition. Generally, adversarial adaptation approaches aim to find a feature representation of the target domain that can be invariant from the source domain. Existing deep feature extractors with their large complexity can find the arbitrary transformation of the target domain that can be similar to the source. However, only finding domain invariant features does not guarantee good performance on the target domain [139, 140]. Specifically, forcing target domain features to be similar to source domain features with no constraints can remove the target-specific information, i.e., the mutual information between the target data and the target extracted features, which could hinder the model performance. To handle this issue, inspired by noise contrastive estimation (NCE), we propose a novel approach that leverages the InfoNCE loss [141] to preserve the structure of the target domain features during the domain adaptation process. We jointly optimize the target feature extractor to minimize both the domain adaptation loss and the InfoNCE loss. Specifically, the domain adaptation loss guides the target feature extractor to produce source-like features, and the InfoNCE loss preserves the target-specific features by maximizing the mutual information between the target input data and the target features. Maximizing the mutual information between the input space and the feature space can preserve the intrinsic structure of the target data during the domain alignment process, which can boost the performance of domain adaptation. We have performed extensive experiments to verify the performance of the proposed CADA method on machine RUL prediction across different working conditions.

The main contributions of this chapter are summarized as follows:

- A novel contrastive domain adaptation approach is proposed for challenging yet practical machine RUL prediction. This approach successfully transfers knowledge for RUL prediction from one condition (distribution/domain) to another.
- A novel solution based on the InfoNCE loss is designed to learn the invariant representation and preserve the original structure for the target domain. As such, satisfactory performance for RUL prediction can be achieved.

4.1 Contrastive Adversarial Domain Adaptation

4.1.1 Problem Formulation and Notations

To clearly formulate the problem, we introduce the basic standard notations of domain adaptation [67]. Let a domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$, where $\mathcal{X}$ is the feature space, $X \in \mathcal{X}$, and $P(X)$ is the marginal distribution of data in this feature space. Given a labeled source domain $\mathcal{D}_S = \{\mathcal{X}_S, P_S(X)\}$ and unlabeled target domain $\mathcal{D}_T = \{\mathcal{X}_T, P_T(X)\}$, the unsupervised domain adaptation problem aims to transfer knowledge from the labeled source to improve the performance on the unlabeled target. In our problem, $\mathcal{D}_S$ and $\mathcal{D}_T$ are both multivariate time-series data of aircraft engines under different working/fault conditions. Particularly, we have labeled data from aircraft engines with a specific working/fault condition, and we aim to improve the RUL prediction of unlabeled data with different working/fault conditions. We denote the source domain $\mathcal{D}_S = \{X_S^i, y_S^i\}_{i=1}^{n_S}$, with n_S the total number of samples, where $X_S^i \in \mathbb{R}^{M \times K}$ is the input source sample with M sensors and K time steps, $y_S^i \in \mathbb{R}$ is the corresponding RUL label. Similarly, the unlabeled target domain $\mathcal{D}_T = \{X_T^j\}_{j=1}^{n_T}$, where $X_T^j \in \mathbb{R}^{M \times K}$ and n_T is the number of target domain samples.

4.1.2 Overview

Domain adaptation for multivariate time-series regression can be a very challenging task. Therefore, only few works have been presented for RUL estimation problems across domains [134]. In this paper, we develop a novel contrastive adversarial

domain adaptation (CADA) approach for machine RUL prediction. Specifically, the proposed CADA can find domain invariant representations of the target domain data while preserving their intrinsic structure which is crucial to achieving satisfactory performance in the target domain.

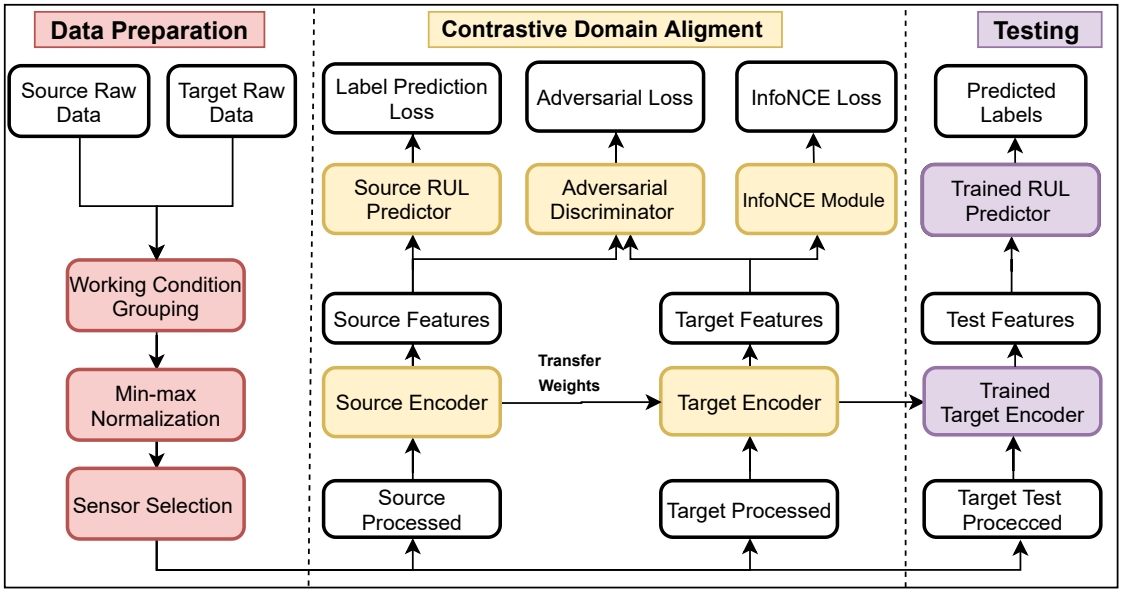


FIGURE 4.1: The proposed CADA approach is composed of three main stages: data preparation, contrastive domain alignment, and testing on the unlabeled target data.

Fig. 4.1 shows the overall framework that presents the detailed steps of the learning procedure of the CADA model. The first stage involves data preparation for both source and target domains. In the second stage, the source and target features are extracted by the source and target encoders respectively. Given the target features, the target encoder E_T is updated to optimize both the adversarial loss and the InfoNCE loss. In the last stage, the trained target feature extractor and the trained source RUL predictor are combined to predict the RULs for the target domain data. We will provide a detailed explanation of each module in the following subsections.

4.1.3 Supervised Pre-training on the Source Domain

In this section, we will present our approach that models the dynamics of multivariate time series and automatically extracts salient features. In addition, we

will provide details about the RUL prediction network that maps from the latest features to the RUL.

4.1.3.1 Recurrent Multivariate Modeling

Recurrent-based approaches are widely adopted for modeling temporal dependencies of time-series data. But RNNs often suffer from the problem of vanishing gradient with long-term sequences [142]. Alternatively, the LSTM which is a strong variant of RNN can handle long-term dependencies and tackle the vanishing gradient problem. In this work, we design a very deep bi-directional LSTM network with 5 successive layers for automatic and representative feature extraction. The LSTM feature extractor represents the multivariate time series to a single-vector hidden representation as shown in Fig. 4.2. Specifically, the LSTM network can

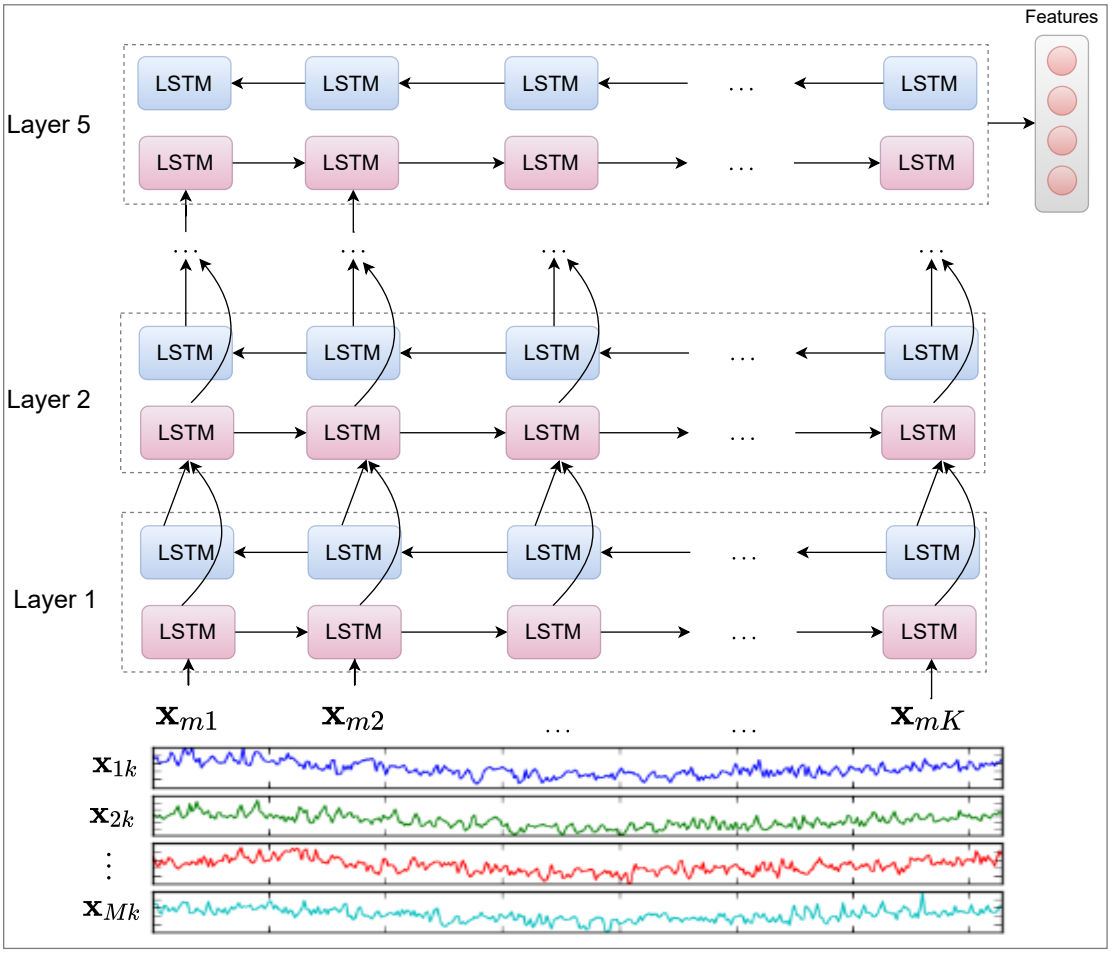


FIGURE 4.2: Deep BiLSTM feature extractor

be represented as multiple sequential feed-forward layers. The transition function between these layers is a key function to model the temporal dependency along with the data, which can be formulated as follows:

$$\mathbf{h}_k, \mathbf{c}_k = H_{cell}(\mathbf{x}_k, \mathbf{h}_{k-1}, \mathbf{c}_{k-1}) \tag{4.1}$$

where H_{cell} receives the current input $\mathbf{x}_k$, the previous hidden $\mathbf{h}_{k-1}$, and the previous memory cell $\mathbf{c}_{k-1}$. A detailed explanation of the process of an LSTM cell can be found in 3.1.2

4.1.3.2 RUL Prediction Network

Given the extracted features from the LSTM feature extractor $\mathbf{f}_S = E_S(X_S)$. The RUL predictor is a multi-layer network $R : \mathbb{R}^d \to \mathbb{R}$ that maps the latent features into the corresponding RUL value. The RUL predictor R and the feature extractor E_S are trained in an end-to-end manner using the mean square error loss between the predicted RULs and the true RULs, which can be formalized as follows:

$$\mathcal{L}_{mse} = \frac{1}{n_S} \sum_{i=1}^{n_S} (\hat{y}_S^{(i)} - y_S^{(i)})^2 \tag{4.2}$$

where $\hat{y}_S = R(E_S(X_S)$ is the predicted RUL label, y_S is the ground-truth RUL values, and n_S is the number of source samples.

4.1.4 Contrastive Adversarial Domain Alignment

The contrastive adversarial adaption module consists of a domain discriminator D and the InfoNCE module as shown in Fig. 4.3. Firstly, the weights of the trained source feature extractor are adopted to initialize the target feature extractor. The output features from both the source and target domains are fed into an adversarial discriminator network to minimize the discrepancy. Concurrently, the target features are fed into the InfoNCE loss module to preserve the target-specific features during the alignment process. In particular, the InfoNCE loss maximizes the mutual information between the target domain inputs and the

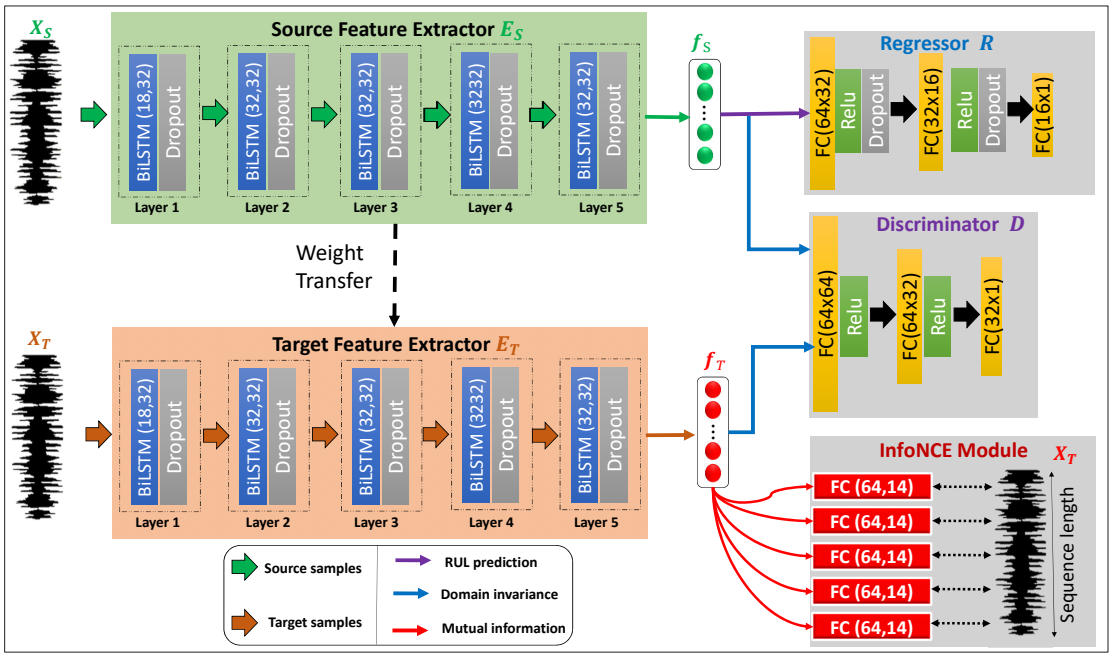


FIGURE 4.3: Proposed CADA Approach

target domain features to preserve task-specific information. Algorithm 1. shows the formal procedure of our contrastive adversarial domain adaptation approach. While the domain discriminator encourages the source and target features to be invariant, the contrastive estimation module preserves the target-specific features bia maximizing the mutual information between the input space and feature space of the target data.

4.1.4.1 Adversarial Adaptation Module

Let E_S and R_S be the source-trained LSTM feature extractor and the RUL predictor respectively. To predict the RUL labels of the unlabeled target domain data, we can naively initialize our target model (i.e., E_T and R_T) with pre-trained source models. However, due to the large discrepancy among the data from different working/fault conditions, the model can fail to predict RUL accurately. To tackle this domain discrepancy problem, we adversarially train the LSTM feature extractor against a domain discriminator network to minimize the distribution differences between the source features and the target features. Specifically, the domain discriminator network D is trained to discern between the source and target features. Concurrently, we train the target feature extractor E_T to produce target features such that the domain discriminator network cannot distinguish them from the

Algorithm 1: Contrastive Adversarial Domain Adaptation

Input: Source domain: $\mathcal{D}_S = \{X_S^i, y_S^i\}_{i=1}^{n_S}$
Target domain:$\mathcal{D}_T = \{X_T^i\}_{i=1}^{n_T}$
Output: Trained target encoder E_T
$E_S \leftarrow$ Trained source encoder
$E_T \leftarrow$ Initialize with E_s parameters
$D \leftarrow$ Domain Discriminator
for *number of iterations* **do**

1. Sample mini-batch of m source samples $X_S \sim P_S$
2. Sample mini-batch of m target samples $X_T \sim P_T$
3. Extract source features: $\mathbf{f}_S = E_S(X_S)$
4. Extract target features: $\mathbf{f}_T = E_T(X_T)$
5. Feed $\mathbf{f}_S$ and $\mathbf{f}_T$ to D
6. Compute adversarial loss $\mathcal{L}_{adv}$ by Eq. 4.3
7. Update D by $\mathcal{L}_{adv}$
8. Compute InfoNCE loss $\mathcal{L}_{\text{InfoNCE}}$ based on Algorithm 2
9. Update E_T by $\mathcal{L} = \mathcal{L}_E + \lambda \mathcal{L}_{\text{InfoNCE}}$

end

source features. The adversarial training between the discriminator network D and the target E_T can be expressed as follows:

$$\begin{aligned}\min_{E_T} \max_{D} \mathcal{L}_{adv} =& \mathbb{E}_{X_s \sim P_S}\big[\log D(E_S(X_S))\big] \\ &+ \mathbb{E}_{X_T \sim P_T}\big[\log(1 - D(E_T(X_T)))\big]. \end{aligned} \tag{4.3}$$

where X_S and X_T are the sources and target samples respectively. The target feature extractor E_T is updated to minimize $\mathcal{L}_{adv}$, and the discriminator network D is adversarially trained to maximize $\mathcal{L}_{adv}$. Eventually, the trained target feature extractor E_T will be able to extract features $\mathbf{f}_T$ that have minimum discrepancy from the source features.

4.1.4.2 Contrastive Estimation Module

Adversarial domain adaptation can successfully find target domain features that are invariant from the source features. However, it can remove task-specific information

Algorithm 2: Contrastive Loss

Input: $X_T = \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}$, $\mathbf{f}_T = E_T(X_T)$
Output: Contrastive Loss $\mathcal{L}_{\text{InfoNCE}}$
$\Theta^k \leftarrow$ Linear layer at timestep k
for *K timesteps* **do**

1. $\mathbf{q}_k \leftarrow \Theta^k \mathbf{f}_T$
2. Apply $\phi_k(\mathbf{x}_k, \mathbf{q}_k^\intercal)$ as in Eq. 4.5
3. Compute $\mathcal{L}_{\text{InfoNCE}}$ using Eq. 4.6

end
return $\mathcal{L}_{InfoNCE}$

from the target features to minimize the adversarial loss, which can deteriorate the performance on the target domain – even with perfect domain alignment. Hence, it is required to preserve target-specific features during the domain alignment task. To achieve that, we rely on InfoNCE loss [141] to maximize the mutual information between the encoded representations of the target domain and the original inputs, as shown in the Algorithm 2. Given a sample $X_T \sim \mathcal{X}_T$, where $X_T \in \mathcal{R}^{M \times K}$, we apply the target encoder E_T on X_T to obtain its corresponding feature representation $\mathbf{f}_T = E_T(X_T)$. To model the mutual information between $\mathbf{x}_k$, and $\mathbf{f}_T$, following the previous studies [143], we define a density ratio function ϕ_k at each time step, which is formalized as follows.

$$\phi_k(\mathbf{x}_k; \mathbf{f}_T) \propto \frac{p(\mathbf{x}_k|\mathbf{f}_T)}{p(\mathbf{x}_k)} \tag{4.4}$$

By maximizing the mutual between the latent target features $\mathbf{f}_T$ and the input $\mathbf{x}_k$, we can preserve the common latent variables between the target features $\mathbf{f}_T$ and the input $\mathbf{x}_k$. To compute ϕ_k, the latent features $\mathbf{f}_T$ and the input $\mathbf{x}_k$ should be mapped to the same dimension. To achieve that, we use a fully connected network $\Theta : \mathbb{R}^d \rightarrow \mathbb{R}^M$ that maps feature dimension d to input dimension M. Thereafter, the density ratio ϕ_k is estimated by a dot product between the transformed features $\mathbf{q}_k = \Theta^k(\mathbf{f}_T)$ and the original input $\mathbf{x}_k$, which can be compactly represented as follows:

$$\phi_k(\mathbf{x}_k, \mathbf{f}_T) = \mathbf{x}_k^\intercal \mathbf{q}_k \tag{4.5}$$

where $\Theta^k = \{\theta_1, ..., \theta_M\}$ are the weights of a fully connected layer at time step k. Note that Θ^k is different among the time steps. To maximize the density ratio

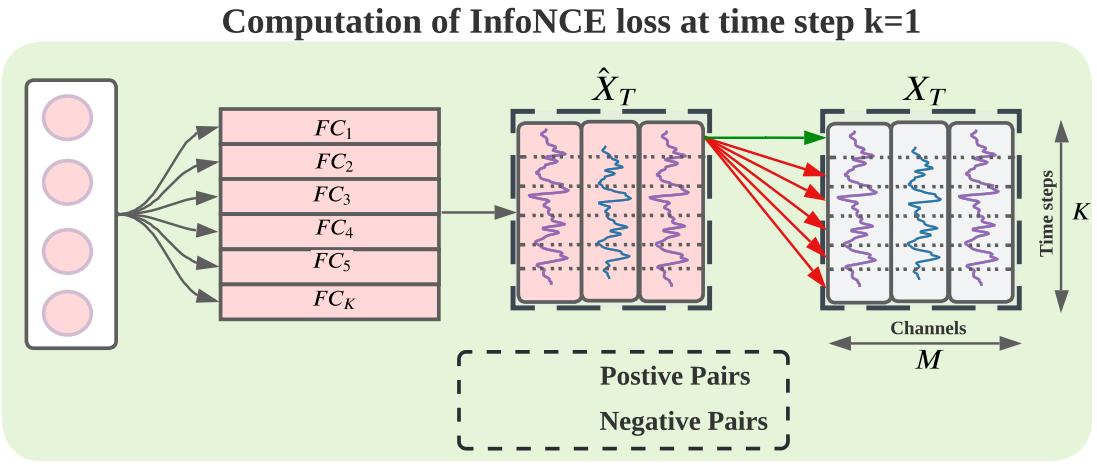


FIGURE 4.4: Computation of InfoNCE loss at time step k=1

function, we jointly optimize the target feature extractor E_T and the fully connected layers Θ using the contrastive estimation loss. The InfoNCE loss maximizes the mutual information by contrasting the positive and negative samples. Fig. 4.4 illustrates the positive and negative samples for time step k = 1. The overall InfoNCE loss can be formulated as

$$\min_{E_T,\Theta} \mathcal{L}_{\text{InfoNCE}} = -\mathop{\mathbb{E}}_{X_T}\left[log\frac{e^{\phi_k(\mathbf{x}_k,\mathbf{f}_T)}}{\sum_{\mathbf{x}_j\in X_T} e^{\phi_k(\mathbf{x}_j,\mathbf{f}_T)}}\right] \tag{4.6}$$

The optimal probability of the NCE loss $p(d = k|X_T, \mathbf{f}_T)$ can be formulated as:

$$p\left(d=k|X_T,\mathbf{f}_T\right) = \frac{p\left(\mathbf{x}_k|\mathbf{f}_T\right)\prod_{l\neq k} p\left(\mathbf{x}_l\right)}{\sum_{j=1}^{K} p\left(\mathbf{x}_j|\mathbf{f}_T\right)\prod_{l\neq j} p\left(\mathbf{x}_l\right)} \tag{4.7}$$

$$= \frac{\frac{p(\mathbf{x}_k|\ \mathbf{f}_T)}{p(\mathbf{x}_k)}}{\sum_{j=1}^{K}\frac{p(\mathbf{x}_j|\ \mathbf{f}_T)}{p(\mathbf{x}_j)}} \tag{4.8}$$

By substituting Eq. 4.6 into the above equations, we can formalize the mutual information in terms of the InfoNCE loss $\mathcal{L}_{\text{InfoNCE}}$, detailed derivation can be found in [143]. The resulting formula can be written as:

$$I(\mathbf{x}_k, \mathbf{f}_T) = log(K) - \mathcal{L}_{\text{InfoNCE}} \tag{4.9}$$

where $I(\cdot)$ represents the mutual information between $\mathbf{x}_k$ and $\mathbf{f}_T$. Clearly, minimizing InfoNCE loss maximizes the lower bound of $I(\mathbf{x}_k, \mathbf{f}_T)$, which in turn maximizes the mutual information.

4.1.4.3 Overall Loss Function

In this chapter, the adversarial adaptation loss and contrastive estimation loss are jointly optimized in an end-to-end manner. The total domain alignment loss can be summarized as follows:

$$\begin{aligned}
&\min_{E_T,\Theta}\max_{D} V\left(D, E_T, \Theta\right) \\
&\qquad = \mathcal{L}_{adv} + \lambda \mathcal{L}_{\text{InfoNCE}} \\
&\qquad = \mathbb{E}_{X_S \sim p_S}\left[\log D\left(\mathbf{f}_S\right)\right] + \\
&\mathbb{E}_{X_T \sim p_T}\left[\log\left(1 - D\left(\mathbf{f}_T\right)\right) - \lambda \log \frac{e^{\phi_k}\left(\mathbf{x}_k, \mathbf{f}_T\right)}{\sum_{\mathbf{x}_j \in X_T} e^{\phi_k\left(\mathbf{x}_j, \mathbf{f}_T\right)}}\right]
\end{aligned} \tag{4.10}$$

where $\mathcal{L}_{adv}$ is the adversarial loss, $\mathcal{L}_{NCE}$ is the contrastive estimation loss, and λ is a weight parameter that controls the proportion of the learning domain invariant features and preserving task-specific information.

4.2 Experiments and Results

To evaluate the performance of our approach, the C-MAPSS [1] benchmark dataset has been employed. The C-MAPSS dataset is mainly composed of four different subsets, namely FD001, FD002, FD003, and FD004, which differ in terms of working conditions, fault modes, and life spans. Further details about the C-MAPSS dataset and its preprocessing techniques can be found in Chapter 3. In the cross-domain problem, labeled data from the source working condition and unlabeled data from the target working condition are used for the model training. Consider the scenario of FD001→FD002 as an example, we intend to transfer the knowledge from the labeled samples in FD001 to the unlabeled target samples in FD002. Thus, to ensure the same feature space during the knowledge transfer, we only select the common sensors among source and target domains that are the most informative. In our experiments, we selected the following sensors: S2, S3, S4,

S7, S8, S9, S11, S12, S13, S14, S15, S17, S20, and S21. A detailed explanation of sensor selection criteria can be found in Subsection 3.2.1.1.

4.2.1 Experimental Settings

The CADA approach consists of five main models: Source feature extractor (E_S), target feature extractor (E_T), RUL predictor (R), domain discriminator (D), and InfoNCE module. The detailed structure of each model has been shown in Fig. 4.3. Specifically, the source and target feature extractors are deep BiLSTM networks with 5 layers, where each layer has 32 neurons. The Discriminator is composed of three fully connected (FC) layers with 64, 32, and 1 hidden neuron. The RUL predictor also consists of three FC layers, i.e., hidden layer 1 with 32 neurons, hidden layer 2 with 16 neurons, and an output layer with a single neuron. Each layer is followed by a nonlinear activation function called rectified linear unit (ReLU) and the dropout regularization technique to relieve the over-fitting problem. The detailed architecture of the RUL predictor is shown in Fig. 4.5. All experiments have been conducted using PyTorch 1.7 on NVIDIA GeForce RTX 2080 Ti GPU.

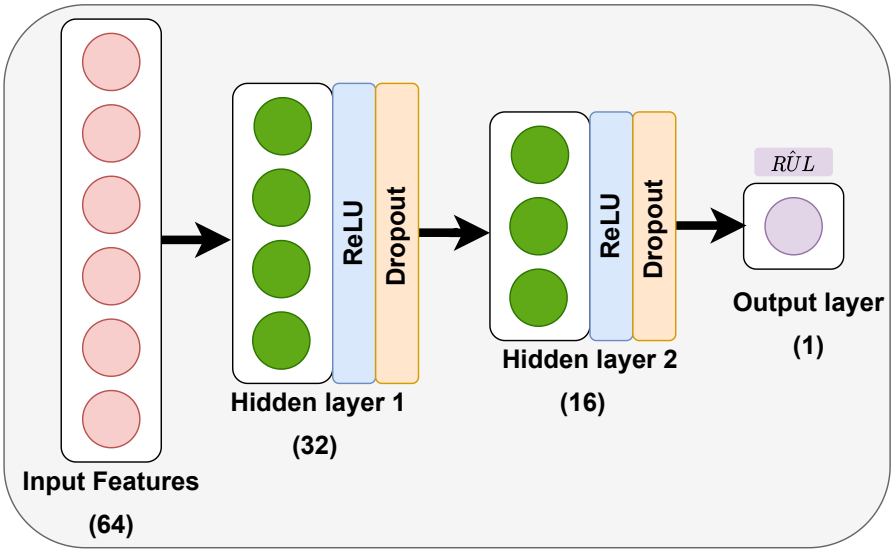


FIGURE 4.5: Detailed architecture of the RUL predictor network.

To train the CADA model, a mini-batch training with a batch size of 256 is adopted. To reduce overfitting, dropout regularization is also used across the whole structure and the dropout ratio is set to 0.5. Adam optimizer has been used to minimize the joint loss with a learning rate of 0.5e-4 for the feature extractor and the domain

discriminator. As the InfoNCE module is trained from scratch during the alignment process, a larger learning rate of 1e-2 is leveraged. The training epochs range from 20 to 150 epochs. The weight of the InfoNCE loss λ can vary across different cross-domain scenarios and later its effect on the prediction performance will be shown through a sensitivity analysis.

To quantify the performance of models, two evaluation metrics have been adopted, i.e., root mean square error (RMSE) and score metric, as in [100, 134]. The RMSE metric is defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)}, \tag{4.11}$$

where $\hat{y}_i$ and y_i represent the predicted RUL and ground-truth RUL respectively.

The RMSE metric treats the early and late RUL predictions equally. For prognostic applications, late RUL prediction can be more harmful to the systems. To handle this issue, the scoring metric is used to impose a bitter penalty for late RUL predictions. It can be formalized as follows:

$$Score = \begin{cases} \frac{1}{N}\sum_{i=1}^{N}(e^{\frac{\hat{y}_i - y_i}{13}} - 1), & \text{if } (\hat{y}_i < y_i) \\ \frac{1}{N}\sum_{i=1}^{N}(e^{\frac{\hat{y}_i - y_i}{10}} - 1), & \text{if } (\hat{y}_i > y_i) \end{cases} \tag{4.12}$$

4.2.2 Comparison with State-of-the-art Methods

To evaluate our approach in cross-domain scenarios, we train the model using a labeled source domain (e.g, FD001) and evaluate it on an unlabeled target domain (e.g., FD002, FD003, or FD004). As we have 4 sub-datasets (i.e., domains), we thus have 12 cross-domain scenarios. In this paper, we implement five state-of-the-art approaches as follows. In addition, we report the average performance (i.e., RMSE and Score) over 5 consecutive runs with different random seeds.

- Correlation alignment (CORAL)[144]: CORAL minimizes the covariance shift between the source and target features to align the distribution.

TABLE 4.1: Comparison of the proposed method against state-of-the-art approaches

Metric	RMSE					
Method	CORAL [144]	WDGRL [145]	DDC [146]	ADDA [147]	RULDDA [134]	CADA
FD001→FD002	22.85	21.46	44.05	31.26	24.08	**19.52**
FD001→FD003	44.21	71.7	39.62	57.09	43.08	**39.58**
FD001→FD004	50.03	57.24	44.35	56.66	45.7	**31.23**
FD002→FD001	24.43	15.24	46.96	19.73	23.91	**13.88**
FD002→FD003	42.66	41.45	39.87	37.22	47.26	**33.53**
FD002→FD004	52.12	37.62	43.99	37.64	45.17	**33.71**
FD003→FD001	40.33	36.05	39.95	40.41	27.15	**19.54**
FD003→FD002	56.67	40.11	44.07	42.53	30.42	**19.33**
FD003→FD004	38.16	29.98	47.46	31.88	31.82	**20.61**
FD004→FD001	51.44	42.01	41.55	37.81	32.37	**20.10**
FD004→FD002	31.61	35.88	43.99	36.67	27.54	**18.5**
FD004→FD003	30.44	18.18	44.47	23.59	23.31	**14.49**
Metric	Score					
FD001→FD002	2798	33160	5958	4865	2684	**2122**
FD001→FD003	56991	15936	288061	32472	10259	**8415**
FD001→FD004	52053	86139	156224	68859	26981	**11577**
FD002→FD001	3590	157672	640	689	2430	**351**
FD002→FD003	23071	19053	62823	11029	12756	**5213**
FD002→FD004	62852	52372	44872	16856	25738	**15106**
FD003→FD001	4581	18307	25826	32451	2931	**1451**
FD003→FD002	73026	32112	1012978	459911	6754	**5257**
FD003→FD004	11407	296061	275665	82520	5775	**3219**
FD004→FD001	154842	45394	162100	43794	13377	**1840**
FD004→FD002	38095	38221	179243	23822	4937	**4460**
FD004→FD003	6919	77977	1623	1117	1679	**682**

- Deep domain confusion (DDC)[146]: DDC employs a distance metric called MMD to confuse the source and target features.

- Wasserstein distance guided representation learning (WDGRL)[145]: WDGRL employs a neural network to measure the empirical Wasserstein distance while utilizing the feature extractor network to minimize this distance between the source and target domain.

- Adversarial discriminative domain adaption (ADDA)[147]: ADDA uses a typical GAN loss to find target domain features that can be similar to the source features.

- Deep domain adaptation (DDARUL)[134]: In DDARUL, an LSTM feature extractor is trained to confuse the source and target domains, while a domain classifier network is trained to classify between the source and target features.

Table 4.1 shows the experimental results. the CADA outperforms all the competing approaches across the 12 cross-domain scenarios in terms of both RMSE and Score. In addition, we observe that knowledge transfer between simple and complex datasets is challenging due to the large domain shift, yet our CADA can successfully align the two distant domains. For example, FD001 and FD004 are the simplest and most complex data subsets respectively. As shown in Table4.1, simply forcing the features to be similar among these two datasets can significantly harm the performance. Overall, we achieve significant improvement over the second-best approach (underlined) in each scenario with an average of more than 21% and 38% for RMSE and Score respectively. Our domain adaptation strategy can preserve task-specific information, leading to the superior performance of our proposed CADA.

4.2.3 Model Ablation Study

Here, we perform our ablation study to verify the contribution of individual components in our CADA approach. We derive two variants of CADA, namely, "Source-Only" and "w/o InfoNCE". In particular, "Source-Only" refers to the non-adapted version of our model, whereas the "w/o InfoNCE" is our adversarial adaptation approach without using the contrastive estimation loss.

Table 4.2 shows the comparison between the CADA and its two variants. We observe that the "Source-Only" has the worst performance, indicating the big gap between the source and target domain data distributions. The proposed CADA method outperforms the one without the InfoNCE loss in most cases, which signifies the effectiveness of the InfoNCE loss on domain adaptation based RUL prediction.

TABLE 4.2: Ablation study of the proposed approach

Metric	RMSE			Score		
Method	Source-Only	w/o InfoNCE	CADA	Source-Only	w/o InfoNCE	CADA
FD001→FD002	20.62	20.48	**19.52**	5448	4600	**2122**
FD001→FD003	55.09	**39.33**	39.58	31062	11866	**8415**
FD001→FD004	36.81	**31.19**	31.23	20786	11713	**11577**
FD002→FD001	15.29	**13.82**	13.88	543	**342**	351
FD002→FD003	35.46	33.65	**33.53**	5339	5350	**5213**
FD002→FD004	37.66	33.82	**33.71**	19807	**15070**	15106
FD003→FD001	39.03	24.66	**19.54**	5700	6469	**1451**
FD003→FD002	46.11	24.84	**19.33**	72405	35036	**5257**
FD003→FD004	31.44	21.94	**20.61**	40772	8873	**3219**
FD004→FD001	37.90	26.34	**20.10**	99597	14985	**1840**
FD004→FD002	32.98	28.73	**18.50**	62345	48726	**4460**
FD004→FD003	19.47	**14.38**	14.49	2470	793	**682**

TABLE 4.3: The values of λ and the number of LSTM layers for different scenarios.

Scenario	λ	Number of Layers
FD001→FD002	0.2	5
FD001→FD003	0.2	3
FD001→FD004	0.001	5
FD002→FD001	0.001	5
FD002→FD003	0.001	5
FD002→FD004	0.001	5
FD003→FD001	0.5	5
FD003→FD002	0.5	5
FD003→FD004	0.2	3
FD004→FD001	0.2	3
FD004→FD002	0.2	5
FD004→FD003	0.001	1

4.2.4 Sensitivity Analysis

4.2.4.1 Coefficient of the InfoNCE loss

In this section, we investigate the sensitivity of the proposed CADA with respect to the coefficient of the InfoNCE loss λ. We have conducted experiments with λ varying from 0.001 to 1.0 for the 12 cross-domain scenarios. The results are shown in Fig. 4.6. It can be found that different scenarios may require different λ to boost the performance. Table 4.3 summarizes the selected λ values for the 12 cross-domain scenarios in experiments.

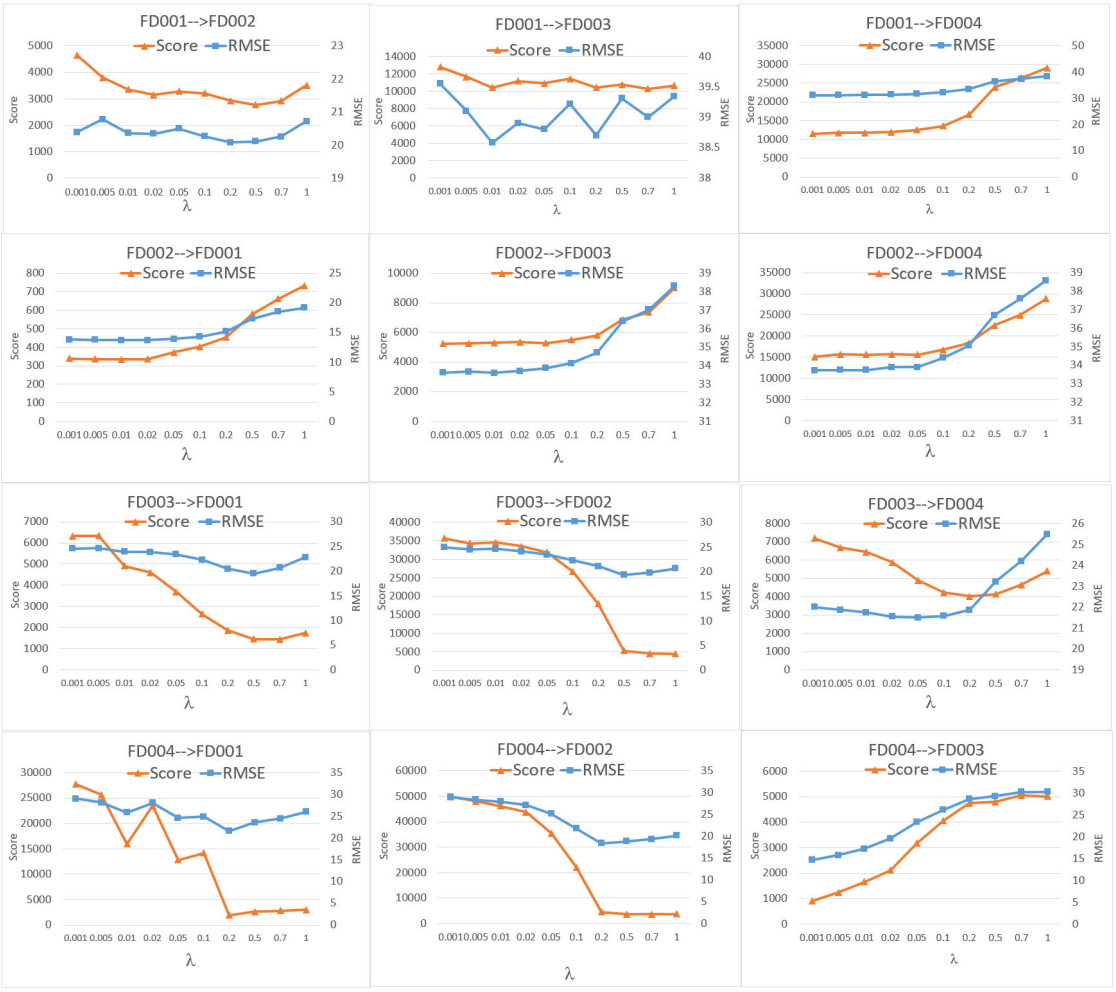


FIGURE 4.6: The experimental results with different λ values for 12 cross-domain scenarios

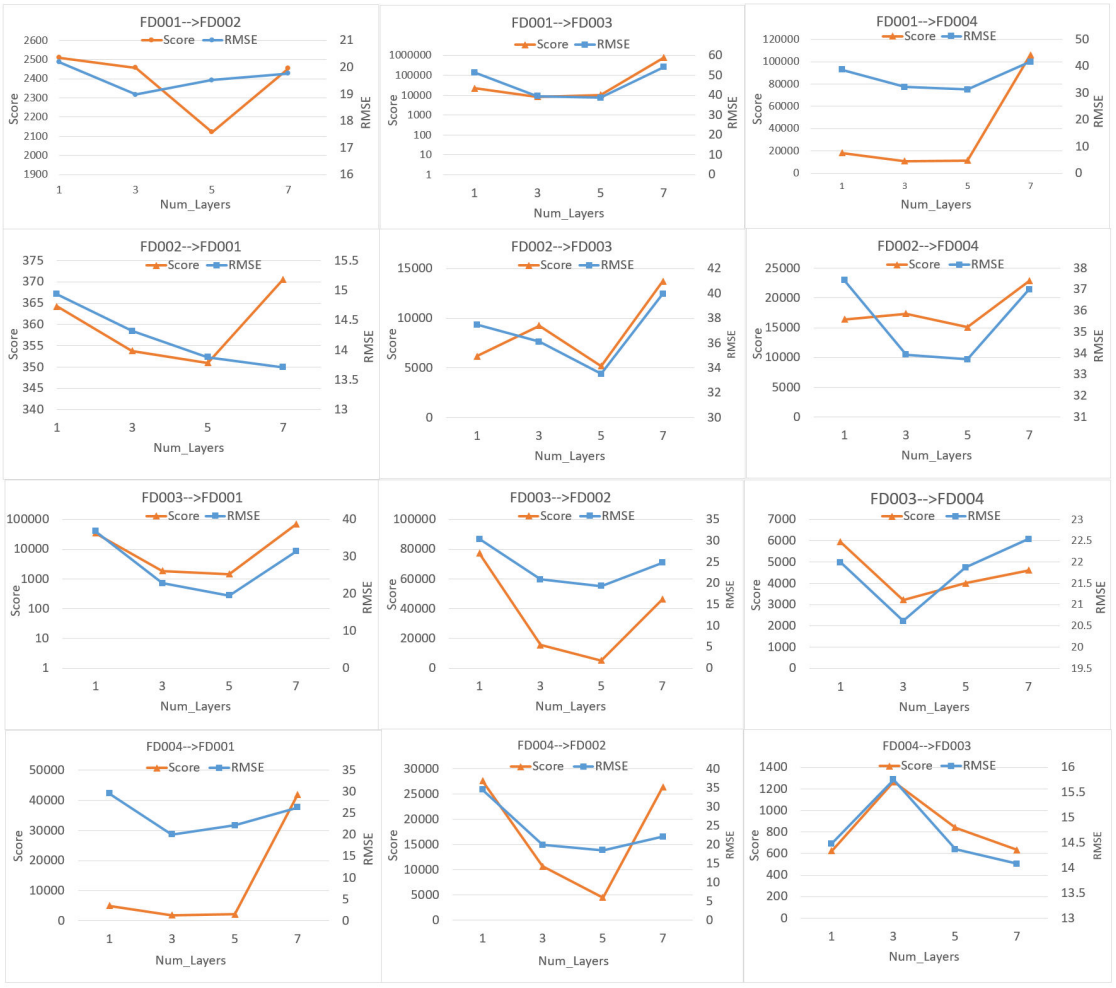


FIGURE 4.7: The experimental results with different numbers of LSTM layers for 12 cross-domain scenarios

4.2.4.2 The number of LSTM Layers

Another important hyperparameter for the proposed method is the number of LSTM layers. We have investigated the model performance with different numbers of LSTM layers, i.e., 1, 3, 5, and 7, in order to find a balance between the model performance and the training time. Fig.4.7 shows the experimental results. We can find that the proposed method with 5 layers can achieve the best performance in most of the scenarios. However, some scenarios require fewer layers to obtain a better or comparable performance. For example, for the scenario FD004→FD003, the method with 7 LSTM layers performs the best. However, the performance of the method with 1 LSTM layer is comparable to the best performance but much more efficient. In this case, using a single LSTM layer is more reasonable when considering the balance between the performance and the efficiency of the

algorithm. Table 4.3 shows the selected number of LSTM layers for each cross-domain scenario.

4.3 Summary

This chapter presented a novel contrastive adversarial domain adaptation (CADA) approach to automatically adapt the features between different working conditions while preserving domain-specific information for machine RUL prediction tasks. Extensive experiments have been conducted to verify the effectiveness of the CADA method. The experimental results show that the proposed CADA method significantly outperforms all the state-of-the-art. Moreover, the conducted ablation study demonstrates the effectiveness of the InfoNCE loss when performing domain adaptation. One limitation of the work on this chapter is that the current InfoNCE loss is only applicable to multivariate time-series problems, limiting its extension to univariate time-series applications. Chapters 3 and 4 have focused on the fault prognosis task, in the subsequent chapters, we will address the deep learning limitations for the fault diagnosis task under real-world environments.

Part II

Fault Diagnosis Task

Fault diagnosis is the process of detecting, isolating, and identifying the current degradation of a specific component or machine based on physical changes. Fault diagnosis can encompass three basic tasks:

- Fault detection: Discover the presence of faults by detecting anomaly behaviors of the monitored systems.
- Fault isolation: Pinpoint the exact location of faults or identify failing components or subsystems.
- Fault identification: Determine the degradation level of the machine component and the fault extent.

Chapter 5

Self-supervised Autoregressive Domain Adaptation for Machine Fault Diagnosis

In the previous chapters, we have addressed robustness and domain shift problems for the RUL prediction task via developing the ATS2S approach and CADA approach respectively. In this chapter, we aim to address the variability of working conditions under the scarcity of labeled data for fault diagnosis tasks. Several attempts have been developed to address the variability of working conditions with no labeled data available for diagnosis tasks. Several attempts have been made to address the domain shift problem in data-driven fault diagnosis [68, 74, 83, 148]. One key scheme for domain adaptation is to find domain invariant features that can be transferable between the source and target domains such that the model trained on the source domain can generalize well to the target domain. Most of these approaches aim to find domain invariant features by adversarially training a feature extractor network to deceive a domain discriminator, inspired by Generative Adversarial Networks (GANs) [77]. However, the discriminator networks in such approaches are mainly designed for image applications and are incapable of regarding the temporal dependency of time series sensory data. Furthermore, adversarial training schemes are mainly aiming to align the global distributions

The work in this chapter has been accepted for publication in IEEE Transactions on Neural Networks and Learning Systems, 2022

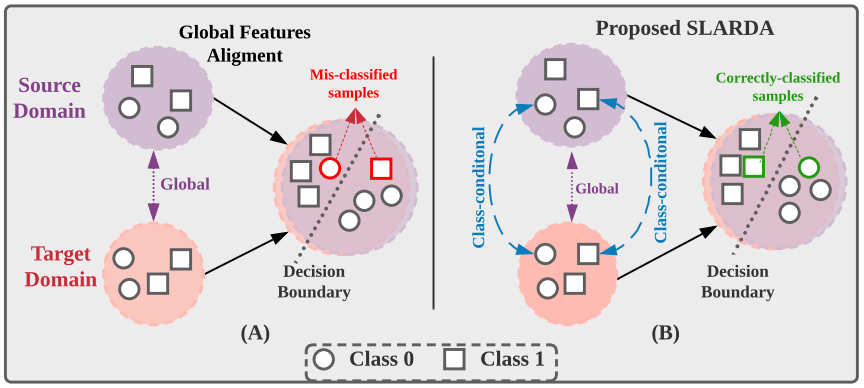
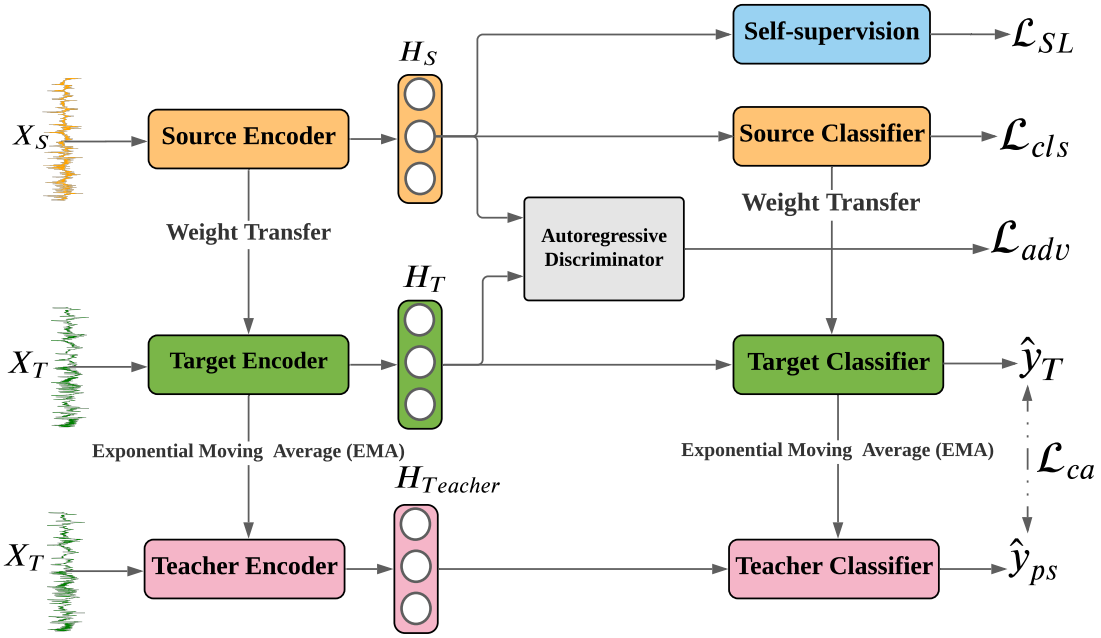


FIGURE 5.1: Illustration of different domain alignment approaches. (A) The global distributions of the source and target domains are aligned, but the classes are misclassified between the source and target. (B) In our proposed approach, both global feature alignment and class-conditional alignment are considered during the adaptation process to align the domains in the feature and class levels.

between different domains, disregarding the fine-grained class distribution within each domain. As a result, classes among different working environments can be misaligned, even with perfect matching of global distributions, leading to sub-optimal performance, as shown in Fig. 5.1

To address all the aforementioned limitations, we propose a novel **S**e**L**f-supervised **A**uto**R**egressive **D**omain **A**daptation (SLARDA) framework to boost the performance of time series UDA. First, unlike existing approaches that utilize self-supervised learning for unsupervised representation learning [149, 150], we design a self-supervised pretraining approach to improve the transferability and generalization of the learned features in the source domain. With the lack of an ImageNet-like dataset for time series pretraining, we are the first to propose self-supervised pretraining as a strong alternative for time series domain adaptation. Second, to incorporate the temporal dependency of time series data during feature alignment, we propose a novel autoregressive domain adaptation approach. Particularly, an autoregressive domain discriminator is developed to consider the temporal dimension when classifying between the source and target features, which helps the feature extractor to learn better features. Last, to mitigate the class-conditional shift between the source and target domains, a teacher-based approach with confident pseudo labels is proposed to guide the target model and correctly align the fine-grained source and target classes.



The o

FIGURE 5.2: Overall framework of the proposed SLARDA.

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

- self-supervised pretraining approach is developed for the source domain via a contrastive predictive loss to improve the representation learning and transferability of the learned features. To the best of our knowledge, we are the first to propose self-supervised pretraining for time series domain adaptation.
- To consider the temporal dependency among source and target features during domain alignment, an autoregressive domain discriminator is designed.
- An ensemble teacher model confident pseudo labeling is developed to generate reliable pseudo labels in the target domain for domain alignment, which can mitigate the class-conditional shift between the source and target domains.

5.1 Self-supervised Autoregressive Domain Adaptation

5.1.1 Problem Formulation

In this chapter, we address the problem of UDA for time series data. Given a labelled source domain $\mathcal{D}_{\mathrm{S}} = \{X_S^i, \boldsymbol{y}_S^i\}_{i=1}^{n_S}$ with n_S samples, and an unlabeled target domain $\mathcal{D}_{\mathrm{T}} = \{X_T^j\}_{j=1}^{n_T}$, with n_T samples. The source and target domains are sampled from different distributions $P_S(X)$ and $P_T(X)$ respectively, where $P_S(X) \neq P_T(X)$. The samples of the source and target domains can be either uni-variate or multi-variate time series. Formally, we have input source sample $X_S^i \in \mathbb{R}^{M \times K}$ with M channels and K time steps, and its corresponding label $\boldsymbol{y}_S^i \in \mathbb{R}^C$, where C is the number of classes. Our main goal is to design a predictive model that can accurately predict the label $\boldsymbol{y}_T^i$ of the unlabeled target sample $X_T^i \in \mathbb{R}^{M \times K}$.

5.1.2 Overview of SLARDA

Fig. 5.2 shows the proposed SLARDA framework, which is composed of three main components: (1) a self-supervised pretraining module to improve the transferability of the learned source features; (2) an autoregressive discriminator model to explicitly consider the temporal dependency among the source and target features during domain alignment; (3) a class-conditional alignment module to address the class-conditional shift and adapt the fine-grained distribution of different categories for the unlabeled target domain. We will elaborate on each component in more detail in the following subsections.

5.1.3 Self-supervised Learning for Source Pretraining

Most of the existing UDA approaches initialize the target domain model by a supervised pre-trained model on the labeled source domain. We argue that the learned representation from supervised objectives tends to be more specific towards a single domain and may have limited transferability to out-of-distribution domains.

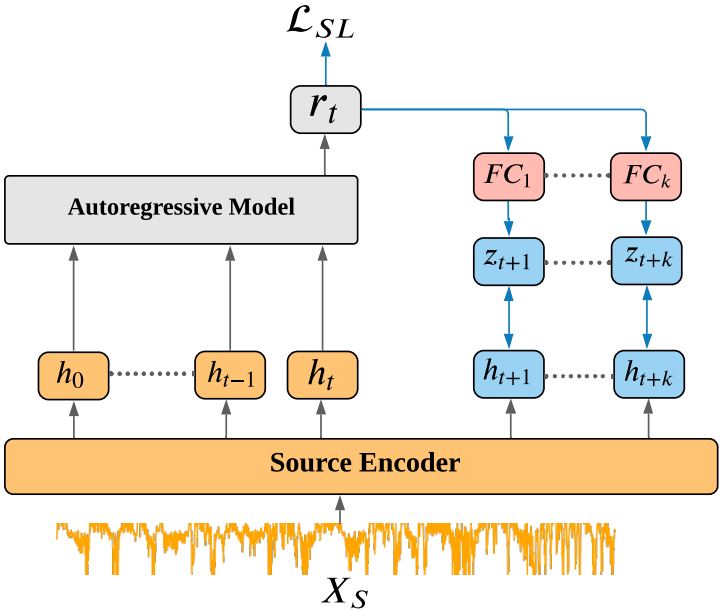


FIGURE 5.3: Self-supervised learning in the source domain.

Inspired by [149], we propose a novel self-supervised auxiliary task to improve the transferability of the learned representations in the source domain. Specifically, given the encoded latent features, we pick a time step t and train the model to predict the future time steps given the past ones, as shown in Fig. 5.3. Thus, the model will learn more general features that encompass the shared information among multiple time steps.

To map the input data into a latent space, we first design a 1D-CNN encoder model. Then, we leverage an autoregressive model to summarize the latent features into a context vector. Formally, given the output latent features from the encoder $E_{\leq t} = \{\boldsymbol{h_0}, \ldots, \boldsymbol{h_t}\}$, they are fed into an autoregressive model to obtain the context vector $\boldsymbol{r_t}$. Subsequently, we pass the context vector to a parameterized fully connected mapping layer $P2C_k$ to predict the future latent feature $\boldsymbol{z_{t+k}} = FC_k(\boldsymbol{r_t})$. To measure the similarity between $\boldsymbol{h_{t+k}}$ and $\boldsymbol{z_{t+k}}$, we leverage a dot product similarity measure between the predicted vector and the true latent future. The similarity matching function can be formulated as follows:

$$\phi_k(\boldsymbol{h_{t+k}}, \boldsymbol{z_{t+k}}) = \exp\left(\boldsymbol{h}_{t+k}^{\intercal} \boldsymbol{z_{t+k}}\right), \tag{5.1}$$

where ϕ_k is a log bi-linear model. Here, we jointly optimize the encoder model, the autoregressive model, and the log bi-linear model via the contrastive objective to

maximize the similarity between the predicted future $\boldsymbol{z_{t+k}}$ and its corresponding true future latent feature $\boldsymbol{h_{t+k}}$. While the true latent feature changes during the training, the predicted vector varies correspondingly to preserve their relationship and stabilize the training process.

This auxiliary task of predicting the future time steps via self-supervised learning helps to better model the temporal dependency of the input samples and produce more transferable features from the source domain. We formulate the problem as a binary classification problem between positive and negative samples. In our case, the future latent of the same sample is considered as a positive pair while the future latent of all other samples in the mini-batch are considered negative pairs. This can be formalized as follows:

$$\mathcal{L}_{\text{SL}} = -\mathop{\mathbb{E}}_{H_b}\left[\log \frac{\phi_k\left(\boldsymbol{h_{t+k}}, FC_k(\boldsymbol{r_t})\right)}{\sum_{\boldsymbol{h_j}\in H_b}\phi_k\left(\boldsymbol{h_j}, FC_k(\boldsymbol{r_t})\right)}\right], \tag{5.2}$$

where H_b represents a mini-batch of samples.

We design the aforementioned self-supervised loss to optimize the source encoder E_S on the source domain data. Concurrently, we train the encoder model E_S to perform well on the main classification task via cross-entropy loss on the labeled source domain data, shown as follows:

$$\mathcal{L}_{\text{cls}} = -\mathbb{E}_{X_S\sim P_S}[\boldsymbol{y}_S^{\intercal}\log(C_S(E_S(X_S)))]. \tag{5.3}$$

Finally, we jointly train the source encoder E_S with the self-supervised task along with the supervised objective to produce more transferable features as follows:

$$\min_{E_S}\ \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{SL}}. \tag{5.4}$$

5.1.4 Autoregressive Domain Adaptation

Adversarial domain adaptation has achieved remarkable performance for visual applications. However, the design of discriminator networks in existing methods does not consider temporal dependency in the feature space of the time series data, resulting in a limited performance for domain alignment. To address this critical issue, we propose an autoregressive domain discriminator to exhibit the

temporal dynamic behavior of time series data during domain alignment, as shown in Fig. 5.4.

The autoregressive discriminator D_{AR} consists of two main components. First, an autoregressive network f_{AR} that encodes the temporal dependencies among both source and target features into vector representations, shown as follows:

$$f_{AR}(\boldsymbol{h_0}, \ldots, \boldsymbol{h_K}) = p(\boldsymbol{h_K} \mid \boldsymbol{h_{<K}}), \tag{5.5}$$

where $p(\boldsymbol{h_K} \mid \boldsymbol{h_{<K}})$ is the conditional distribution among different time steps of the sequential features. Second, a binary classification network f_D is applied to the summarised feature vectors to classify between the source and target features. Thus, the autoregressive discriminator can be represented as $D_{AR} = f_D(f_{AR}(\cdot))$. A detailed explanation of the autoregressive discriminator and its architecture are discussed in Section 5.2.2. To align the source and target domains, we first freeze the self-supervised pre-trained source model and transfer its weights to the target model. Then, we adversarially train the autoregressive domain discriminator against the target model to produce domain-invariant features. The autoregressive discriminator is optimized to discern between the source and target features, which can be formalized as:

$$\begin{aligned} \min_{D_{AR}} \mathcal{L}_{\mathrm{D}} = &- \mathbb{E}_{X_S \sim P_S}\big[\log D_{AR}(H_S)\big] \\ &- \mathbb{E}_{X_T \sim P_T}\big[\log(1 - D_{AR}(H_T))\big], \end{aligned} \tag{5.6}$$

where $E_S = E_S(X_S)$ and $E_T = E_T(X_T)$ are the temporal output features from the source and target encoders respectively, and D_{AR} represents the autoregressive discriminator network. Concurrently, we train the target encoder to confuse the discriminator by mapping the target features to be similar to the source ones. The target encoder loss can be formalized as:

$$\min_{E_T} \mathcal{L}_{\mathrm{adv}} = \mathbb{E}_{X_T \sim P_T}\big[\log(1 - D_{AR}(H_T))\big]. \tag{5.7}$$

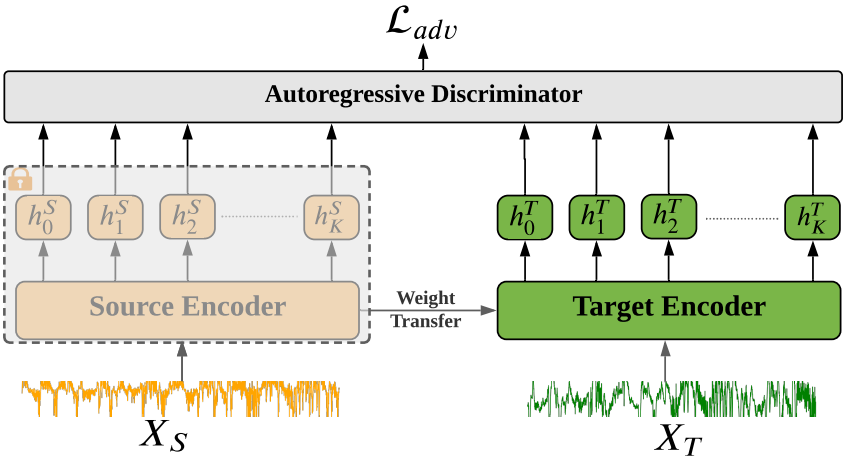


FIGURE 5.4: Autoregressive discriminator.

5.1.5 Class-conditional Alignment via Teacher Model

Autoregressive domain adaptation can successfully align the marginal distribution of the source and target temporal features. However, it can still misalign the different classes among source and target domains due to class-conditional shifts. To overcome this issue, we develop a teacher-based confident pseudo-labeling approach to adapt the fine-grained distribution of different categories among the source and target domains.

5.1.5.1 Teacher Model

Inspired by the mean teacher for semi-supervised learning [151], we design an ensemble teacher model f_ψ to produce robust pseudo labels for the unlabeled target

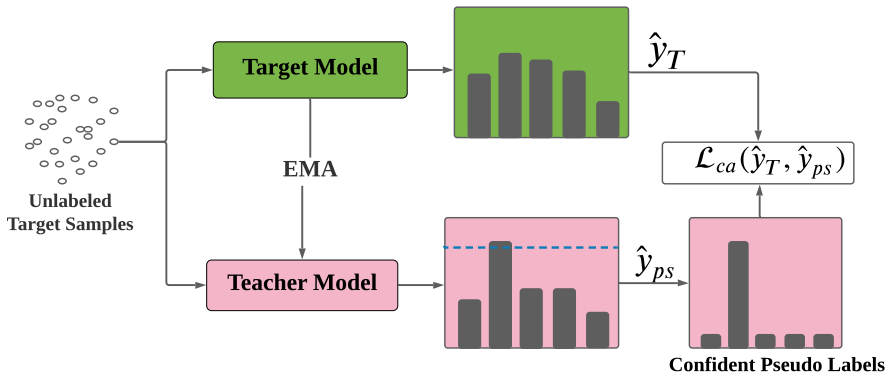
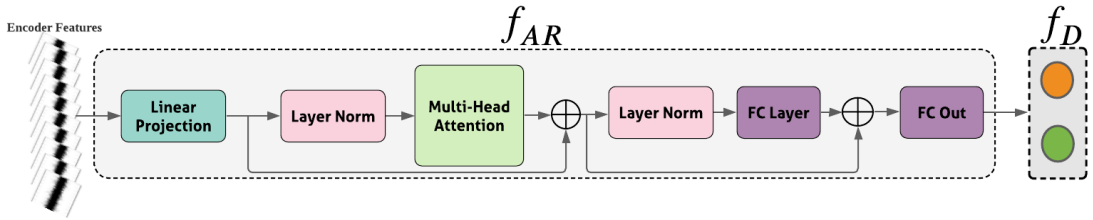
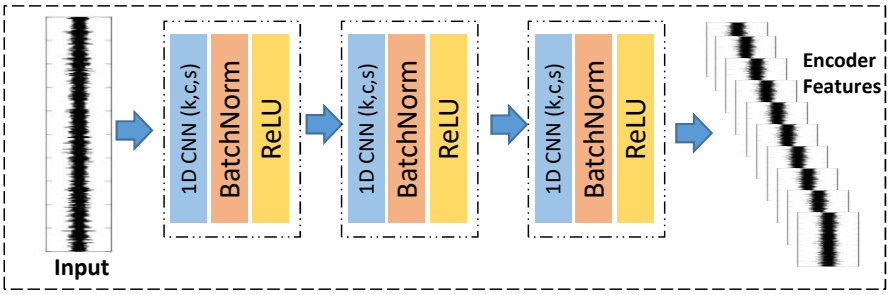


FIGURE 5.5: Class-conditional alignment via teacher model.



The a

FIGURE 5.6: Architecture of the autoregressive discriminator.



The a

FIGURE 5.7: Architecture of feature extraction network.

domain, as shown in Fig. 5.5. We obtain the weights of the teacher model $\mathcal{W}_\psi$ by applying the exponential moving average (EMA) over the target model parameters $\mathcal{W}_{\theta_T}$ across successive training steps. The momentum updates of the teacher model parameters can be represented as follows:

$$\mathcal{W}_\psi = \alpha \mathcal{W}_\psi + (1 - \alpha)\mathcal{W}_{\theta_T}, \tag{5.8}$$

where α is a momentum parameter that controls the speed of the weight updates of the teacher model. Given the teacher model f_ψ, we obtain the output predictions as follows:

$$\boldsymbol{p}_\psi = f_\psi(X_T), \tag{5.9}$$

$$\hat{\boldsymbol{y}}_\psi = softmax(\boldsymbol{p}_\psi), \tag{5.10}$$

where $\boldsymbol{p}_\psi$ is the output predictions of the teacher model, and $\hat{\boldsymbol{y}}_\psi$ are the corresponding probabilities.

5.1.5.2 Confident Pseudo Labels

To further refine the predicted labels of the teacher model, we only preserve the confident labels that are above a predefined confidence threshold ζ. This can be formalized as follows:

$$\hat{y}_{\text{ps}} = \hat{\boldsymbol{y}}_{\psi}[max(\boldsymbol{p}_{\psi}) > \zeta], \tag{5.11}$$

where $\hat{\boldsymbol{y}}_{\text{ps}}$ are the retained confident pseudo labels. To align the class-conditional distribution, we leverage the obtained confident pseudo labels to train the target model by a cross-entropy loss:

$$\mathcal{L}_{\text{ca}} = -\mathbb{E}_{X_T \sim P_T}\Big[\sum_{k=1}^{K} \mathbb{1}_{[y_{ps}=k]} \log(\hat{\boldsymbol{y}}_T^k)\Big], \tag{5.12}$$

where $\mathcal{L}_{\text{ca}}$ is the class-conditional alignment loss, and $\hat{\boldsymbol{y}}_T = C_T(E_T(X_T))$ are the predicted labels by the target classifier C_T.

5.1.6 Overall Objective Function

In our approach, we jointly optimize the target encoder E_T to minimize both the autoregressive domain adaptation loss and class-conditional alignment loss in an end-to-end learning manner. Our overall objective can be formalized as follows:

$$\begin{aligned} \mathcal{L}_{\text{overall}} &= \mathcal{L}_{\text{adv}} + \lambda \mathcal{L}_{\text{ca}} \\ &= \min_{E_T} \mathbb{E}_{X_T \sim P_T}\Big[\log(1 - D_{AR}(E_T(X_T))) \\ &\quad - \lambda \hat{\boldsymbol{y}}_{ps}^{\intercal} \log(C_T(E_T(X_T)))\Big], \end{aligned} \tag{5.13}$$

where λ is the weight of the class-conditional loss. Algorithm 3shows the detailed procedures of our autoregressive adaptation approach.

Algorithm 3: Autoregressive Domain Adaptation

Input: Source domain: $\mathcal{D}_S = \{X_S^i, y_S^i\}_{i=1}^{n_S}$
Target domain:$\mathcal{D}_T = \{X_T^i\}_{i=1}^{n_T}$
Output: Trained target encoder E_T
$E_S \leftarrow$ Pre-trained source encoder
$E_T \leftarrow$ Initialize with E_s parameters
$f_\psi \leftarrow$ Teacher model
$D_{AR} \leftarrow$ Autoregressive Domain Discriminator
for *number of iterations* **do**

1. Sample mini-batch of m source samples $X_S \sim P_S$
2. Sample mini-batch of m target samples $X_T \sim P_T$
3. Extract source features: $H_S = E_S(X_S)$
4. Extract target features: $H_T = E_T(X_T)$
5. Feed H_S and H_T to D_{AR}
6. Assign labels of ones to H_S and zeros to H_T
7. Compute discriminator loss $\mathcal{L}_D$ by Eq. 4.3
8. Update D_{AR} by $\mathcal{L}_D$
9. Invert the labels of H_T
10. Compute $\mathcal{L}_{adv}$ with the inverted labels by Eq. 5.7
11. Pass X_T to the Teacher model f_ψ
12. Obtain the confident pseudo labels by Eq. 5.11
13. Compute the class conditional loss $\mathcal{L}_{CA}$ by Eq. 5.12
14. Update E_T using both $\mathcal{L}_{adv}$ and $\mathcal{L}_{CA}$ via Eq. 5.13

end

5.1.7 Testing on the target domain

In the testing phase, we only use the pretrained target encoder E_T and target classifier C_T while ablating both the transformer model and the autoregressive network, ensuring consistency of the backbone network when evaluating against other UDA algorithms. Given the test data from the target domain, the encoder model E_T will extract the target-adapted features. Subsequently, the target classifier C_T will predict corresponding class predictions.

$$\hat{\mathbf{p}}_{test} = \mathbb{E}_{X_{test} \sim P_{test}} \big[\sigma(C_T(E_T(X_{test})))\big], \tag{5.14}$$

$$\hat{y}_{test} = argmax(\hat{\mathbf{p}}_{test}), \tag{5.15}$$

where $\sigma(a)_i = \frac{e^{a_i}}{\sum_{j=1}^{k} e^{a_j}}$ represents the the softmax function, $\hat{\mathbf{p}}_{test}$ is the output probability vector, and $\hat{y}_{test}$ is the predicted label.

5.2 Experimental Setup

5.2.1 Paderborn Dataset

We evaluate our SLARDA on the Paderborn Fault Diagnosis dataset, which was collected using the modular rig tester as shown in Fig. **??** [152]. The tester consists of several components: (1) an electric motor, (2) a torque-measurement shaft, (3) a rolling bearing test module, (4) a flywheel, and (5) a load motor. More details about the modular tester for data collection can be found in [152]. In this dataset, 32 experiments for rolling bearing elements were conducted to collect 3 types of data, namely, undamaged bearing data, artificially damaged bearing data, and real damaged bearing data. In particular, the bearing data in each experiment has 20 files and each file was collected for 4 seconds with a sampling rate of 64 kHz.

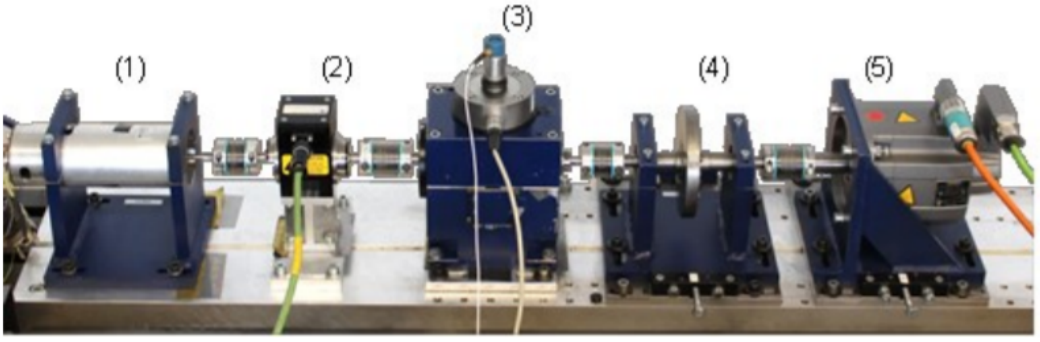


FIGURE 5.8: Modular test rig for collecting the Paderborn dataset [2]

To generate the data samples, we also used overlapping sliding windows to segment the time series data, where we set the window size as 5120, as in [153]. As mentioned above, the Paderborn dataset has 3 classes - 1 normal class (undamaged) and 2 faulty classes including inner faults and outer faults, which can be caused by

either artificial or real damages. In this chapter, we focused on the faults from real damages and generated 4900, 6200, and 6200 samples for normal class, inner faults, and outer faults, respectively.

In addition, Paderborn bearing data was collected under 4 different working conditions, denoted as P1, P2, P3, and P4. Table 5.1 shows the parameter settings (i.e., rotational speed, load torque, and radial force) for each working condition.

TABLE 5.1: Different Working Conditions

Working Condition	Rotational Speed [rpm]	Load Torque [Nm]	Radial Force [N]
P1	900	0.7	1000
P2	1500	0.1	1000
P3	1500	0.7	400
P4	1500	0.7	1000

The second dataset was generated by the KAT data center at Paderborn University with a sampling rate of 64 kHz [154]. The test rig of the Paderborn dataset is shown in Fig 5.8. The damages were generated using both artificial and natural ways. More specifically, an electric discharge machine (EDM), a drilling, and an electric engraving were used to manually produce the artificial faults. While the natural damages were caused by using accelerated run-to-failure tests. The data collection process for both types of damages, i.e., artificial and real, was exposed under working conditions with different operating parameters such as loading torque, rotational speed, and radial force. In total, the Paderborn datasets were collected under 6 different operating conditions including 3 conditions with artificial damages (denoted as domains I, J, and K) and 3 conditions with real damages (denoted as domains L, M, and N). Table 7.2 demonstrates the detailed specifications of each working condition. For example, the loading torque varies from 0.1 to 0.7 Nm and the radial force varies from 400 to 1000 N, while the rotational speed is fixed at 1500 RPM. Each operating condition (i.e., domain) contains three classes, namely, healthy class, inner fault (IF) class, and outer fault (OF) class. To prepare the data samples for the Paderborn dataset, we adopted sliding windows with a fixed length of 5,120 and a shifting size of 4,096 [20]. As such, we generated 12,340 for each artificial domain (i.e., I, J, and K) and 13,640 samples for each real domain (i.e., L, M, and N) respectively.

TABLE 5.2: Parameter setting for the CNN encoder and the autoregressive feature extractor.

Parameters	MFD Dataset
Encoder model:	
# of Layers	5
# of Channels (c)	8
Kernel size (k)	32
# stride (s)	2
Transformer (Adaptation):	
FC Layer	128
Input Channels	8
# of Layers	4
# Num of Heads	4
GRU (Pretraining):	
Hidden Dimension	64
Input Dimension	8
# of Layers	1

5.2.2 Model Architectures

Our algorithm has two main models, namely the feature extractor model and the autoregressive discriminator model. We provide further details about the architecture of each model in the following subsections.

5.2.2.1 Feature Extractor

We adopt the 1D-CNN architecture to extract features, as shown in Fig.5.7.Table 5.2 shows the detailed encoder parameters for the Paderborn dataset. We adopted the commonly used architecture in the literature for each application. Particularly, we used a 5-layer 1D-CNN with a kernel size of 32.

5.2.2.2 Autoregressive Discriminator

We employ the transformer model [162] to model the temporal dependency among time steps for both source and target domains. The transformer model uses self-attention, which has an advantage over recurrent neural networks in terms of efficiency and speed [163]. The model architecture is shown in Fig. 5.6. First, a linear

TABLE 5.3: Experimental results on Fault Diagnosis dataset Among 12 cross-domain scenario (Accuracy %).

Method	Source Only	Deep Coral [155]	DAN [156]	WDGRL [157]	MDDA [158]	HoMM [159]	CDAN [160]	DIRT [161]	SLARDA
P1→P2	25.70	38.05	50.86	40.67	38.15	46.78	52.95	47.21	**84.38**
P1→P3	36.18	47.07	53.57	51.70	48.65	45.47	61.38	54.13	**75.70**
P1→P4	25.81	45.37	56.30	52.02	49.14	51.28	53.55	51.46	**96.04**
P2→P1	36.62	41.30	38.86	51.37	35.35	41.15	31.64	45.71	**86.60**
P2→P3	71.74	66.98	65.16	72.56	72.28	75.19	74.25	**85.91**	79.47
P2→P4	**99.89**	92.63	98.82	94.89	97.79	98.43	99.66	98.26	99.68
P3→P1	32.26	36.92	26.13	52.73	23.56	34.17	55.20	31.06	**75.59**
P3→P2	90.91	82.31	91.09	67.73	85.53	84.97	91.98	**99.28**	90.10
P3→P4	93.81	81.60	87.97	76.74	81.61	83.35	93.14	**99.14**	92.94
P4→P1	38.09	42.80	45.31	51.28	39.60	44.82	42.08	45.64	**91.17**
P4→P2	98.90	96.29	98.27	97.98	**99.42**	98.99	98.71	99.23	97.40
P4→P3	78.23	69.48	69.71	65.79	70.86	75.43	72.90	**84.66**	80.69
Average	60.68	61.73	65.17	64.62	61.83	65.00	68.95	70.14	**87.48**
P-Value	1.6E-07	9.8E-07	1.1E-04	2.6E-07	1.9E-04	4.9E-07	1.8E-04	2.3E-04	-

projection layer is utilized to map from the input dimension to the hidden dimension of the transformer model. Then, layer normalization is applied to the input features. After that, a multi-head self-attention is employed for the normalized features. Table 5.2 shows the detailed parameters for the autoregressive discriminator. As each dataset has different characteristics, we adopt different parameters for each dataset.

5.2.2.3 Autoregressive Network (Pretraining)

In our pretraining step, we leverage Gated Recurrent Network (GRU) to summarize the latent features into a context vector. Particularly, we used a single-layer GRU network for all the datasets, while input and hidden dimensions vary according to each dataset. Table 5.2 illustrates the detailed architectures of the GRU network on each dataset.

5.2.3 Implementation Details

In our experiments, we use labeled data from the source domain and unlabeled data from the target domain, following the standard protocol of unsupervised domain adaptation [160, 161]. All experiments have been conducted using PyTorch 1.7 on NVIDIA GeForce RTX 2080 Ti GPU. We use a batch size of 512. We adopt Adam optimizer with a learning rate of 1e-4, and a weight decay of 3e-4, as in

[164]. For the teacher model, the conditional alignment weight λ is set to 0.005, the momentum of updating the teacher model α is set to 0.996, and the confidence threshold ζ for pseudo labels is set to 0.9. For all the datasets, we randomly split the data into 60% for training, 20% for validation, and 20% for testing. We report the mean value of 5 consecutive runs with different random seeds.

5.3 Results and Discussions

5.3.1 Baselines

To evaluate the performance of the proposed SLARDA, we have compared it against multiple strong baseline methods. As most of the state-of-the-art approaches are implemented for image-related datasets, we re-implemented 9 state-of-the-art methods to fit our time-series datasets. Additionally, to promote fair evaluation, we adopt our backbone architecture which works well on time series for all the baseline methods. In particular, we compare our SLARDA with the following state-of-the-art methods: Deep Adaptation Networks (**DAN**) [156], Wasserstein Distance Guided Representation Learning (**WDGRL**) [157], **Deep CORAL** [155], Minimum Discrepancy Domain Adaptation (**MDDA**) [158], **HoMM** [159], Domain Adversarial Neural Networks (**DANN**) [165], Conditional Adversarial Domain Adaptation (**CDAN**) [160], and Virtual Adversarial Domain Adaptation (**VADA**) [161]. It is worth noting that some baselines failed to outperform Source Only on some datasets as they are not specifically designed for time-series data. Hence, we only reported the methods that outperform the Source Only for each dataset. In Table 5.3, the best performance is **bolded** while the second best is underlined.

5.3.2 Cross-domain Performance

The Paderborn dataset has four different working conditions, denoted as H, I, J, and K. Table. 5.3 shows the results of the 12 cross-condition scenarios. Similarly, our proposed approach outperforms baselines in 6 out of 12 cross-domain scenarios with an average improvement of 17.34% over the second-best method, i.e., VADA.

The SLARDA outperforms the benchmark methods on the challenging transfer tasks with large domain shifts, e.g., P1→P2, P1→P3, and P1→P4.

5.3.3 Statistical Significance

We performed a comparative analysis of the statistical significance of our SLARDA approach against all the other baselines. Specifically, we leveraged Wilcoxon signed-rank test to measure the P-Value of our SLARDA against other baseline methods [38]. Tables 5.3show the P-value of our SLARDA against other baselines. Clearly, for all the baseline methods, our SLARDA achieves P-value <0.05 and is significantly better than other approaches on all the datasets with a 95% confidence level.

5.3.4 Ablation Study and Sensitivity Analysis

5.3.4.1 Ablation Study

To show the contribution of each component in our proposed method, we conduct an ablation study on the Paderborn dataset. The model variants are defined as follows:

- **SLARDA(w/o SL)**: we replace the self-supervised pretraining with conventional supervised pretraining.
- **SLARDA(w/o AR)**: we replace the autoregressive domain discriminator with a conventional fully connected discriminator network trained with standard GAN loss.
- **SLARDA(w/o Teacher)**: we remove the conditional alignment component from the SLARDA model.
- **SLARDA(full)**: we include all the model's components.

Fig. 5.9 shows the average results of different variants for the 12 cross-domain scenarios. It can be seen that removing self-supervised (SL) pretraining can be detrimental to the performance with more than 8% degradation. This is because

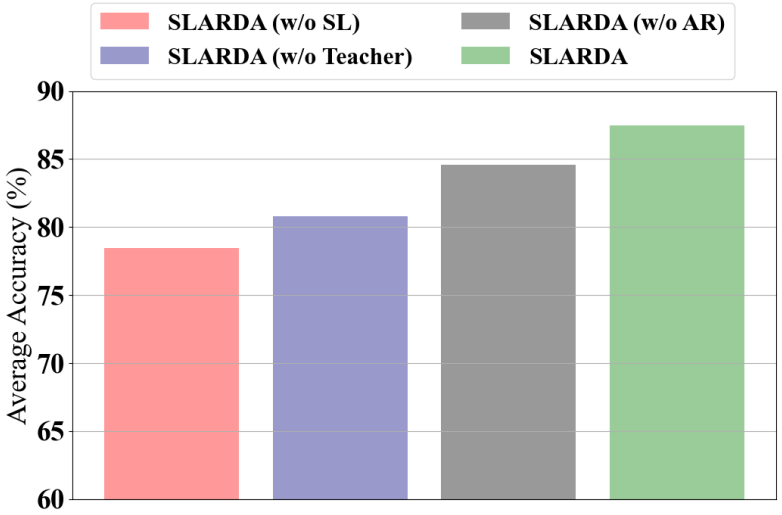


FIGURE 5.9: Ablation Study on the Paderborn dataset.

removing SL can reduce the feature's transferability between domains, which can also affect the efficacy of our remaining modules (i.e., AR and Teacher). Similarly, removing the class-conditional alignment (i.e., Teacher) also has a significant impact on the model performance. Last, adding the autoregressive component by addressing the temporal features can improve the overall performance by about 3%. To sum up, this ablation clearly shows the effectiveness of each component in our SLARDA model.

TABLE 5.4: Shows the total training time of each approach on the Fault Diagnosis dataset (Seconds)

Method	DAN	Deep Coral	HoMM	MMDA	DANN	CDAN	WDGRL	DIRT	SLARDA
Computational Time	1,125	1,411	1,793	1,427	1,467	1,862	2,736	2,929	1,765

5.3.4.2 Sensitivity Analysis of the class conditional loss

There are some key parameters in the proposed approach, which may have a significant impact on model performance. One of the key parameters is λ in Eq. (12), which indicates the contribution of the class-conditional loss. Here, we investigate the impact of this key parameter on model performance. We conduct experiments on the Paderborn dataset and report the average performance of 12 cross-domain scenarios. We vary the weight parameter λ from 0.0001 to 1. Fig. 5.10 shows the results of our proposed SLARDA with different values of λ. Gradually increasing λ improves the performance of our SLARDA. Yet, over-weighting

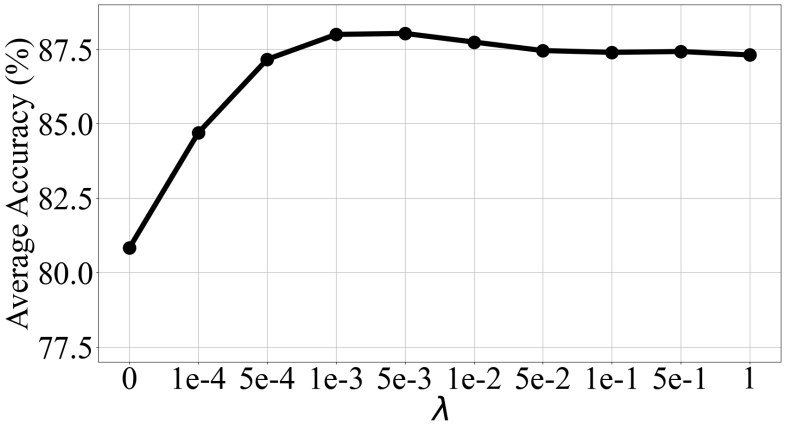


FIGURE 5.10: Sensitivity analysis of class-conditional loss in Eq. (12).

the class-conditional loss deteriorates the performance as the predicted pseudo labels can still be noisy. In a nutshell, our SLARDA approach performs best with λ values between 0.001 and 0.005.

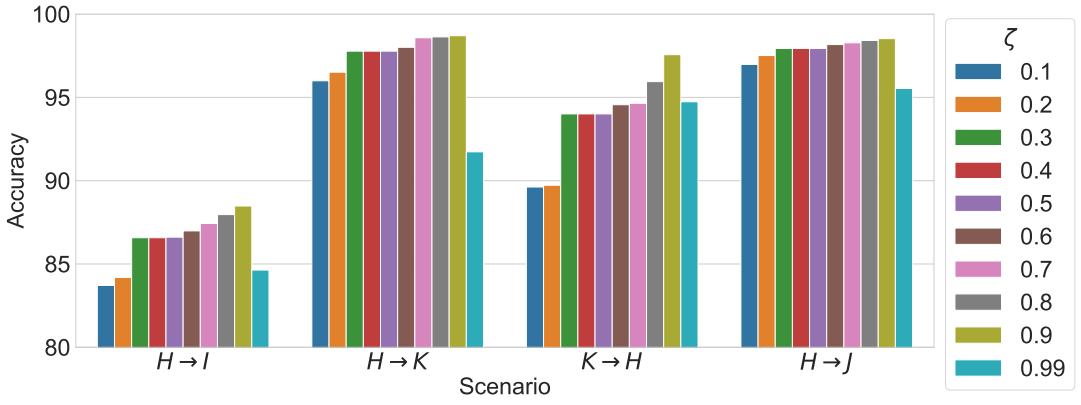


FIGURE 5.11: Sensitivity analysis of confidence threshold parameter ζ

.

5.3.4.3 Sensitivity Analysis of the Confidence Threshold

We conducted a sensitivity analysis experiment to measure the sensitivity of our approach to the confidence threshold parameter. Fig. 5.11 shows the evaluation performance on four randomly selected cross-domain scenarios for the Paderborn Dataset. We varied the confidence threshold from 0.1 to 0.99 and reported the corresponding performance. Clearly, lower values of the confidence threshold can degrade the generalization performance across domains as noisy pseudo labels can be utilized to train the target model. In comparison, higher confidence thresholds

consistently yield better performance across the four experimented cross-domain scenarios. However, a very large confidence threshold, e.g., 0.99, can deteriorate the performance on cross-domain scenarios, as we may not be able to find a sufficient amount of pseudo labels that satisfy this large threshold.

5.3.5 Computational Complexity

To evaluate the time complexity of our proposed approach against other baseline methods, we calculated the total running time over all the cross-domain scenarios on the Fault Diagnosis dataset, as shown in Table 5.4. Generally, discrepancy-based approaches (i.e., DAN, Deep Coral, HoMM, and MMDA) have lower computational complexity, when compared to adversarial-based methods. Among all the adversarial-based methods, our SLARDA approach has the second-lowest computational cost with a total computational time of 1,765 seconds.

5.4 Summary

In this chapter, we realized a domain adaptation algorithm that can generalize to different working with no labeled data for fault diagnosis tasks. As time-series sensory reading is predominant for fault diagnosis tasks, the proposed SLARDA is designed to explicitly consider the temporal relationship during both the feature learning step and domain alignment step. Moreover, we demonstrated that providing confident pseudo labels can successfully address the class-conditional shift between classes of different working environments. By successfully faults from different unlabeled working environments, SLARDA is addressing the second and third objectives of our thesis. Albeit promising, SLARDA can only work under single source *single* target settings (1S1T). As a result, a new model needs to be trained independently for each new working condition. This can be an unscalable approach under highly varying environments. In the next chapter, we explore a more scalable approach that can handle multiple working conditions concurrently.

Chapter 6

Adversarial Multiple Target Domain Adaptation for Fault Diagnosis

In the previous chapter, SLARDA has shown outstanding performance in diagnosing faults from different unlabeled working conditions. However, SLARDA can only work under single-source single-target settings, which can only handle a single working condition at a time. Similarly, earlier approaches attempted to address the domain shift problem to transfer the model from the source domain (one working condition) to the target domain (different working conditions) in intelligent fault diagnosis and prognosis problems [74, 83, 118, 166–168]. However, all existing methods work under single source *single* target settings (1S1T), which *is not feasible as the working conditions can be varying* to satisfy different manufacturing needs. As such, if the target domain has changed, we need to train a new model independently as shown in Fig. 6.1, which is not a viable solution in practice. On the other hand, naïvely merging multiple target domains into a single target will not work either, as data from multiple target domains typically have different data distributions and unique data characteristics.

In this chapter, we build upon the work done by *Tzeng et al.*[147], who proposed an adversarial domain adaptation approach with (1S1T) to obtain domain invariant features for image-related problems. We extend this work in two directions. First,

The work in this chapter has been published in IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-11, 2021

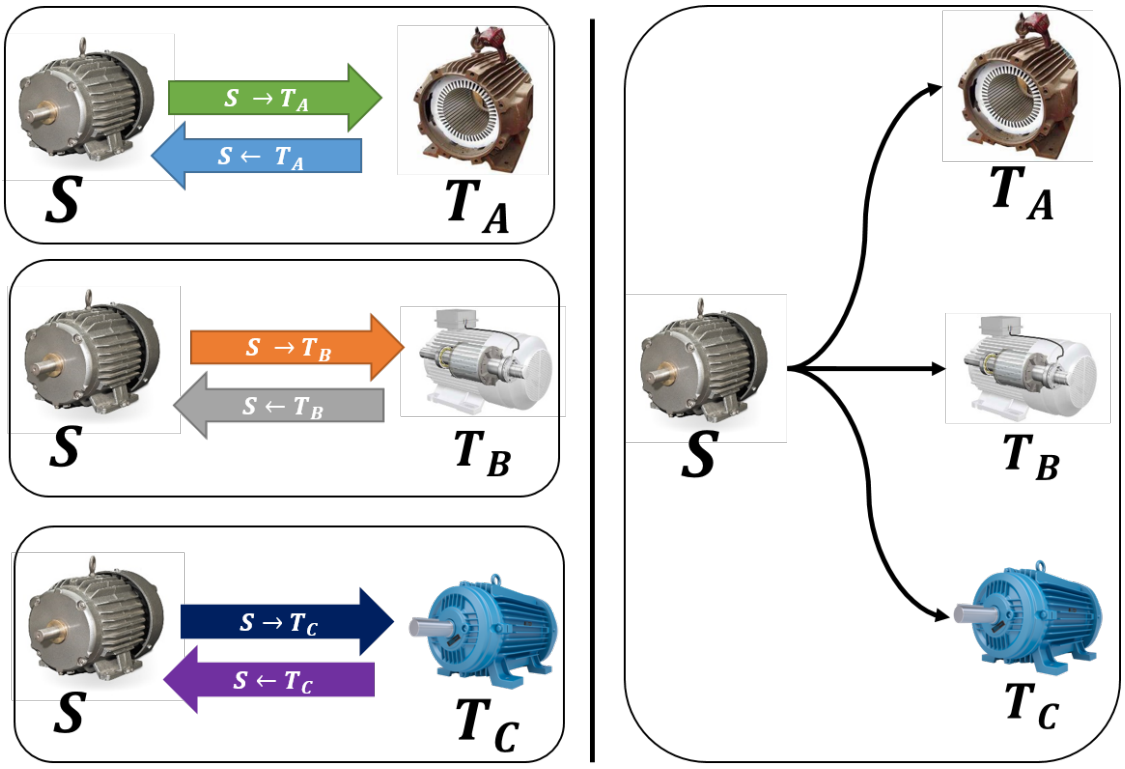


FIGURE 6.1: Existing approaches vs our scalable multi-target approach

we realize the adversarial domain approach for time series data. Second, we tackle a more challenging and practical domain adaptation problem under the single source and multiple targets (1SmT) setting for fault diagnosis purposes. For instance, we assume that a machine can work under four different loads, i.e., A, B, C, and D. Some data have been collected to train a fault diagnostic model when the machine is working under load A. In our 1SmT setting, the model can adapt to multiple different loads concurrently, i.e., B, C, and D. We propose a novel deep learning architecture for adversarial unsupervised domain adaptation for the 1SmT problem. As shown in Fig. 6.2, we first train the source feature extractor to obtain class discriminative features using the labeled source domain. Then, the target feature extractors are initialized by the weights of the source feature extractor and thus inherit the class-discriminative property. On the other hand, a discriminator network is trained to distinguish between the source and multiple target features. To obtain domain invariant features among different targets, we adversarially update multiple target feature extractors to generate features that can be indistinguishable for the discriminator. During testing, our scalable model can take any of the target domains and generate source-like features, where the trained source classifier can generalize well to any of the targets.

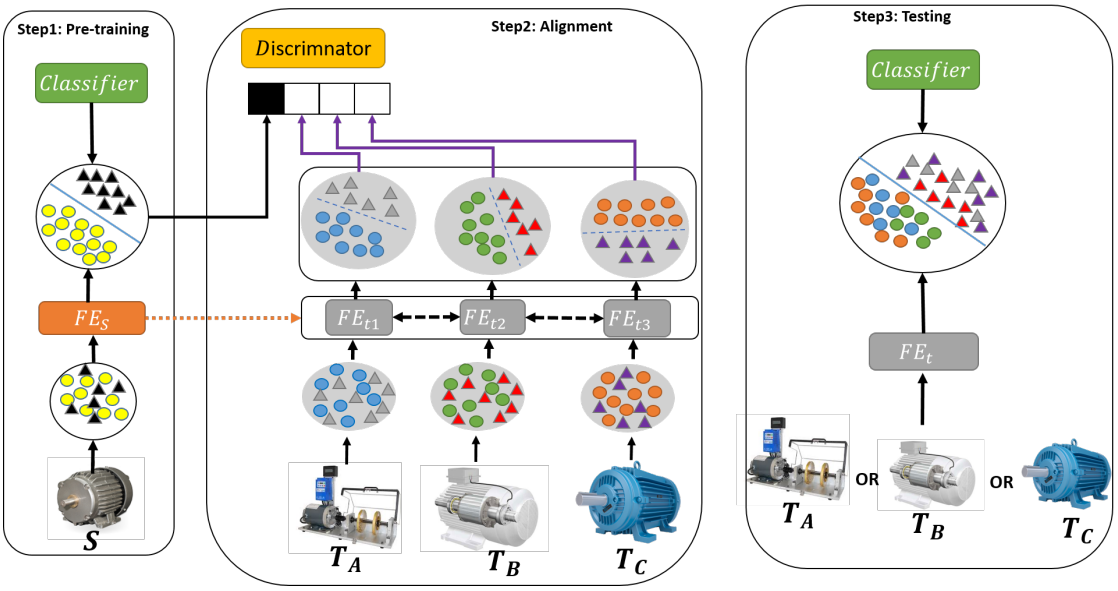


FIGURE 6.2: Proposed adversarial multiple target domain adaptation for fault diagnosis.

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

- We formulate a more realistic 1SmT problem that is particularly used for real-world fault diagnostic problems.
- We propose a novel multiple adversarial domain adaptation (AMDA) method, which designs a deep learning architecture for adversarial unsupervised domain adaptation to address the 1SmT problem. To the best of our knowledge, it is the first attempt in this area.
- We addressed the limited scalability of existing approaches by proposing a general model that can generalize to multiple targets concurrently.
- Extensive experimental results demonstrate that our proposed AMDA model can generalize to multiple target domains simultaneously and achieve significantly better results than the state-of-the-art methods consistently.

6.1 Adversarial Multiple-Target Domain Adaptation

In this section, we first present our problem formulation for single-source multiple targets (1SmT) and then provide technical details on addressing the 1SmT problem with an application to time series data for fault diagnosis. The proposed framework as shown in Fig. 6.3 is composed of three main architectures, namely, Feature Extractor E, Classifier C, and Discriminator $CW4$. Specifically, we used E to construct single source feature extractor E_s, and multiple target feature extractors $E_{t(N)}$ with tied weights.

Different from existing approaches, we tie the weights of the feature extractors of the multiple target domains, inspired by multi-task learning [169]. This enables a single feature extractor to generalize to multiple target domains during the testing stage. In addition, it helps to reduce the capacity of the model and acts as a regularizer to avoid overfitting. To this end, unlike all existing approaches, which can generalize to a single target at a time, our model can be more scalable and have a generalization ability that can handle multiple targets concurrently.

In general, our proposed method contains three main steps: (1) supervised learning using source domain labels; (2) adversarial adaptation of N target domains to single-source domain; (3) testing the domain adapted model on all N target domains. The goal of this paper is to construct a network that can find a shared latent space between the source and multiple target domains, such that the discrepancy between the source domain and target domains is minimized. As such, the model can be better generalized to multiple target domains concurrently. In the following subsections, we will explain each step in more detail.

6.1.1 Problem Formulation

The domain adaptation involves a domain $\mathcal{D}$ and task $\mathcal{T}$ [170], where the domain $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and marginal distribution $P(\mathbf{x})$, where $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$, $\mathbf{x} \in \mathcal{X}$, where $\mathbf{x}$ is the data sample. Correspondingly, the task $\mathcal{T}$ consists of two components: a label space $\mathcal{Y}$ and mapping function $f(\mathbf{x})$,

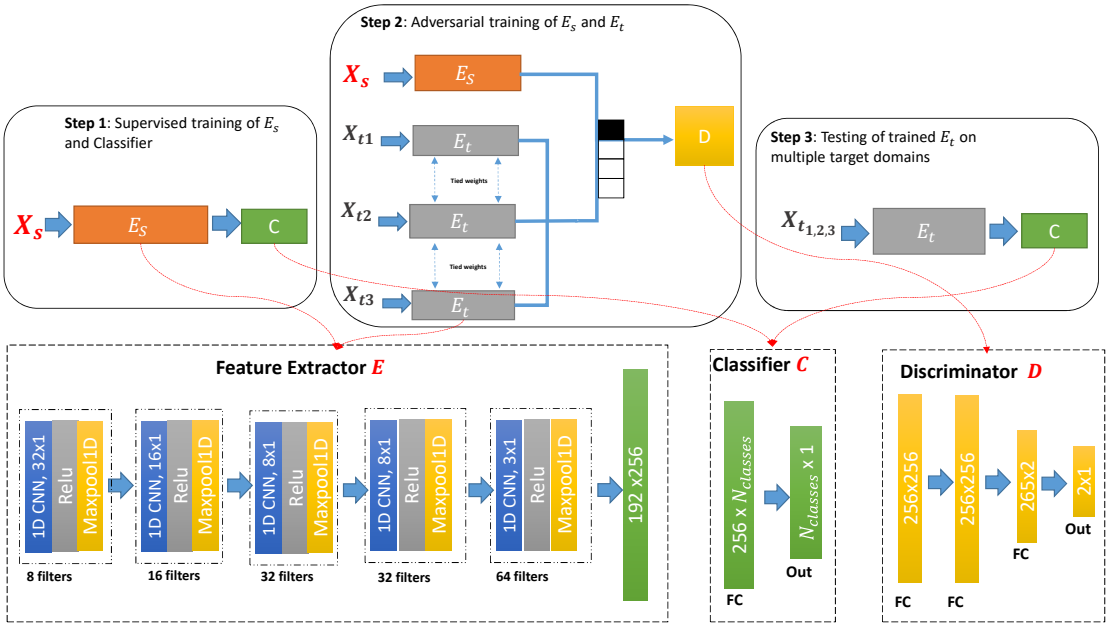


FIGURE 6.3: Adversarial single source multiple target domain adaptation (AMDA) model for fault diagnosis with three main architectures: feature extractors (e.g., E_s for source domain and E_t for target domains), classifier C, and discriminator $CW4$.

where $\mathcal{T} = \{\mathcal{Y}, f(\mathbf{x})\}$.

Our 1SmT problem can be formulated as follows:

- We have a *labelled* single source domain $\mathcal{D}_s = \{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$ of n_s samples, where $\mathbf{x}_s^i \in \mathcal{X}_s$ is the data sample and $y_s^i \in \mathcal{Y}_s$ is the corresponding label. Similarly, we have *unlabeled* multiple target domains $\{\mathcal{D}_{t(1)}, \ldots, \mathcal{D}_{t(N)}\}$, where N is the number of target domains and $\mathcal{D}_{t(j)} = \{\mathbf{x}_{t(j)}^i\}_{i=1}^{n_t}$ represents the total samples of domain j. More specifically, $\mathbf{x}_{t(j)}^i \in \mathcal{X}_{t(j)}$ is the i^{th} sample of the target domain j, where $\mathcal{X}_{t(j)}$ is feature space and n_t is the number of unlabeled samples for the corresponding target domain.

- The feature space of the single source and multiple target domains is the same, i.e., $\mathcal{X}_s = \mathcal{X}_{t(1)} = \mathcal{X}_{t(2)} = ... = \mathcal{X}_{t(N)}$, where N is the number of target domains.

- The marginal distribution between the source domain and target domains is different due to variation on multiple target domains (e.g. with different working conditions), i.e., $P_s(\mathbf{x}) \neq P_{t(j)}(\mathbf{x})$ $(j = 1, 2, ..., N)$. In addition,

marginal distributions among different target domains are also different, i.e., $P_{t(j)}(\mathbf{x}) \neq P_{t(k)}(\mathbf{x})$, where $j \neq k$.

- Label space of the single source domain and multiple target domains is the same, i.e. $\mathcal{Y}_s = \mathcal{Y}_{t(1)} = \mathcal{Y}_{t(2)} = ... = \mathcal{Y}_{t(N)}$

6.1.2 Supervised Learning with Labelled Source Domain Data

Our first step employs the labelled source domain data $\mathcal{D}_s = \{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$, where $y_s^i \in \{1, ..., k\}$ and k is the number of classes, to learn a feature extractor E_s and classifier C in a supervised learning manner by minimizing the cross-entropy loss between the predicted labels and ground-truth labels which is shown in the following equation.

$$L_{ce} = -\frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{1}\{y_s^i = j\} \log C(E_s(\mathbf{x}_s^i)) \tag{6.1}$$

Where L_{ce} is the cross-entropy loss, $y_i \in \mathcal{Y}_s$, and $\mathbf{1}$ is an indicator function that returns 1 when the argument is true.

The parameters of the feature extractor will be used in the next step for two purposes: 1) initialize the target domain feature extractors $E_{t(N)}$ to be inherently class discriminative, and 2) be used as a reference model during adversarial training. Algorithm 1 provides the pseudo-code, including the details of training source feature extractor E_s under the supervision of source domain labels, by employing $\mathcal{D}_s$ to learn the parameters of E_s that can minimize the classification loss in Eq.6.1.

6.1.3 Adversarial Training of Multiple Target Feature Extractors

The key idea of adversarial training is based on the min-max game between the target feature extractor and the domain discriminator. More specifically, the discriminator network is trained to distinguish between the source and target features,

Algorithm 4: Supervised Learning Using Labelled Data from Source Domain

Input: Single source domain: $\mathcal{D}_s = \{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$, and batch size is m
Output: Trained source feature extractor E_s and classifier C
$E_s \leftarrow$ Convolutional neural network
$C \leftarrow$ Fully connected neural network
for *number of samples* **do**

1. $\mathbf{X}_s \leftarrow \{\mathbf{x}_s^1, \ldots, \mathbf{x}_s^m\}$, mini-batch of source samples
2. $Y_s \leftarrow \{y_s^1, \ldots, y_s^m\}$, mini-batch of source labels
3. $Preds \leftarrow C(E_s(\mathbf{X}_s))$
4. Train E_s and C using Eq. 1
5. Update the weights using *Adam* optimizer

end

while the target feature extractor is trained to maximize the discriminator loss by producing target features that are invariant from the source domain features [77]. Hence, the classifier trained on the source domain features can generalize well on the target domain features. Nevertheless, this approach can generalize well to only a single domain at a time and for any change, in the target or the source domain, you need to train a new model independently. As such, to handle k working conditions you need to train k different models which is not a viable solution. In our work, we propose a scalable model that can handle multiple working conditions concurrently. We find a new shared feature representation among the multiple target domains that can be invariant from the source domain features in one training phase. Thus, the trained source classifier can generalize to the domain invariant features of the target domains. To do so, we tie the weights of all the target feature extractors during the training phase. As a result, we can use the common weights of target feature extractors to map any of the target domains to be invariant from the source domain features. In this section, we provide the detailed training process of our proposed approach

Our key idea is to provide an efficient framework to handle N target domains in one training phase, by training a discriminator against N target feature extractors simultaneously. Particularly, we pass $\{\mathbf{X}_{t(1)}, \ldots, \mathbf{X}_{t(N)}\}$ to N feature extractors with *tied weights* to produce $\{\mathbf{h}_{t(1)}, \ldots, \mathbf{h}_{t(N)}\}$. Then the discriminator network D will perform domain classification between the source domain features h_s and the target domain features. However, initially, the target domains features (e.g.,

$\{\mathbf{h}_{t(1)}, \ldots, \mathbf{h}_{t(N)}\}$) are very distinguishable from source domain features (e.g., h_s). Thus the discriminator loss can vanish and limit the domain alignment process. To tackle this issue, the discriminator is trained every N iteration of training target feature extractors. Hence, the discriminator can push the N target feature extractors to map all the target domains to shared latent space, where the discrepancy between the source domain and these N target domains is minimized. The discriminator and multiple target feature extractors are trained with generative adversarial networks (GAN) loss [**?**]. In particular, the discriminator is trained using a logistic function by assigning 1 to the source domain data, and 0 to the data in N target domains. The discriminator classifies each input sample and decides whether it belongs to the source domain or the target domains, under standard supervised learning fashion, where the loss is denoted as $\mathcal{L}_D$.

$$\begin{aligned}\min_D \mathcal{L}_D = &- \mathbb{E}_{\mathbf{x}_s \sim P_s}\big[\log D(E_s(\mathbf{x}_s))\big] \\ &- \sum_{j=1}^{N} \mathbb{E}_{\mathbf{x}_{t(j)} \sim P_{t(j)}}\big[\log(1 - D(E_{t(j)}(\mathbf{x}_{t(j)})))\big]. \end{aligned} \tag{6.2}$$

Where $\mathbf{x}_s$ is source domain sample , $\mathbf{x}_{t(j)}$ are the target domains samples with $(1 \leq j \leq N)$.

The objective function of the target feature extractors is defined as follows:

$$\min_{E_{t(1)}, \ldots E_{t(N)}} \mathcal{L}_E = -\sum_{j=1}^{N} \mathbb{E}_{\mathbf{x}_{t(j)} \sim P_{t(j)}}\big[\log D(E_{t(j)}(\mathbf{x}_{t(j)}))\big]. \tag{6.3}$$

where $E_{t(i)}$ is the feature extractor for the i^{th} target domain $(1 \leq i \leq N)$. By minimizing the loss function $\mathcal{L}_E$, the target feature extractors will map the target domain features to a shared latent space where the discrepancy between the centroid of all target distributions and source domain distribution is minimized.

Detailed steps for the fine-tuning phase are presented in Algorithm 2, where the parameters of $E_{t(N)}$ are derived such that the output features are domain invariant and class discriminative. Adversarial training is employed between N target feature extractors with tied layers and discriminator $CW4$ to minimize $\mathcal{L}_C W4$ and $\mathcal{L}_E$.

Algorithm 5: Adversarial Training for Multiple Targets

Input: Single source domain : $\mathcal{D}_s = \{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$, Multiple target domains: $\{\mathcal{D}_{t(1)}, \ldots, \mathcal{D}_{t(N)}\}$, where with $\mathcal{D}_{t(j)} = \{\mathbf{x}_t^j\}_{i=1}^{n_t}$, N is number of target domains, and m is the batch size.

Output: Trained multiple target feature extractors $E_{t(1)}, \ldots, E_{t(N)}$

$E_s \leftarrow$ Pretrained source feature extractor

$E_{t(N)} \leftarrow$ Initialize with source parameters E_s

$D \leftarrow$ Discriminator network

for *number of iterations* **do**

1. Sample mini-batch of m source samples $\mathbf{X}_s \sim P_s$
2. Sample mini-batch of m from each target domain: $\{\mathbf{X}_{t(1)}, \ldots, \mathbf{X}_{t(N)}\} \sim \{P_{t(1)}, \ldots P_{t(N)}\}$
3. Extract source domain features: $E_s(\mathbf{X}_s)$
4. Extract features from N target domains concurrently: $\{E_{t(1)}(\mathbf{X}_{t(1)}), \ldots, E_{t(N)}(\mathbf{X}_{t(N)})\}$
5. Update D by Eq. 2 `// Train Discriminator`

for *M steps* **do** `// Train` E_t `M times`

6. Extract features from N target domains: $\{E_{t(1)}(\mathbf{X}_{t(1)}), \ldots, E_{t(N)}(\mathbf{X}_{t(N)})\}$
7. Update the target feature extractor E_t by Eq. 3

end

end

6.1.4 Testing on the Target Domain

To justify our contribution by formulating the DA problem as 1SmT, we test the trained E_t to samples from any of N target domains, and then pass the output features to the pre-trained classifier C to predict the class of the corresponding sample. Eq. 6.4 shows the usage of softmax to compute the probability of each class given the input instance from any target domains:

$$p(y_i = k|C) = \frac{\exp(C_k(\mathbf{F}_t))}{\sum_{k'} \exp(C_{k'}(\mathbf{F}_t))} \tag{6.4}$$

where F_t is latent representation of the corresponding target domain, $C_{k'}(\cdot)$ denotes the output of k_{th} class resulted from softmax.

6.2 Experiments

In this section, we evaluate the performance of our proposed AMDA model on fault diagnosis that needs to classify machine-bearing health status into either normal or different classes of faults.

6.2.1 Implementation Details

In our model, we employed a five-layer 1D CNN as a feature extractor and used a wide input kernel for longer dependencies. A fully connected neural network with a softmax layer was used for fault classification, while a two-layer fully connected network was used to discriminate between the source domain and target domain data. Fig.6.3 illustrates the detailed implementation of both the feature extractor and classifier. The learning rate of the feature extractor and discriminator is set to be 1e-4, which is small enough to avoid overshooting valley or minimum in the error surface and thus yields the maximum generalization accuracy.

6.2.2 Case 1: Case Western Reserve University Dataset

6.2.2.1 Dataset Description

The Case Western Reserve University (CWRU) dataset is a widely adopted dataset for rolling bearing elements [171]. Fig **??** shows the test rig for the CWRU dataset. Accelerometer sensors were deployed at both the drive-end and fan-end of the housing motors. Vibration signals were collected with a 12 kHz sampling rate under eight different operating conditions. Particularly, we have four different operating conditions with different loading torques collected from the drive-end, denoted as domain CW1, CW2, CW3, and CW4. For each operating condition, there is one healthy state and three faulty states, namely, inner fault (IF), outer fault (OF), and bearing fault (BF). Each faulty state has three levels of severity with dimensions of 7, 14, and 21 mils. In total, we have 10 classes with 1 healthy class and 9 faulty classes. Table 7.1shows a detailed description of the CWRU dataset. To prepare the data for our experiments, we partitioned the sensor readings into smaller samples using sliding windows with a fixed length of 4,096 and a shifting step of 290, which

is widely used in previous studies [172]. Overall, we can generate 4,000 samples for each domain.

TABLE 6.1: CWRU bearing dataset description [3]

Working Condition	Loading Torque	Fault Type	Fault Size (inches)
CW1	0 hp	Normal, IF, OF, BF	0, 007,0.014, 0.021
CW2	1 hp	Normal, IF, OF, BF	0, 007,0.014, 0.021
CW3	2 hp	Normal, IF, OF, BF	0, 007,0.014, 0.021
CW4	3 hp	Normal, IF, OF, BF	0, 007,0.014, 0.021

6.2.2.2 Experimental Results

We denote four working conditions as CW1, CW2, CW3, and CW4, which correspond to load 0, 1, 2, and 3 respectively. To comprehensively evaluate the performance of our proposed AMDA model, we conducted 12 cross-domain experiments as shown in Fig. 6.4. For the first 3 experiments (CW1→CW2, CW1→CW3, CW1→CW4), we used working condition CW1 as the source domain, and CW2, CW3, and CW4 as multiple target domains to learn the feature extractors, classifier, and discriminator. Then, we tested the learned feature extractor on each target domain CW2, CW3, and CW4 to generate the results for CW1→CW2, CW1→CW3, and CW1→CW4, respectively. Similarly, we also used CW2, CW3, and CW4 as our source domains for cross-domain experiments.

Fig. 6.4 shows the performance of our proposed AMDA model over 12 cross-domain experiments. Note that without DA in Fig. 6.4 refers to our AMDA model without the discriminator, i.e., directly using the source feature extractor for the target domain. Overall, our AMDA achieves an average accuracy of 99.13% over 12 experiments, which is 6.04% higher than without DA. These results demonstrate the effectiveness of domain adaptation in our model for cross-domain fault diagnosis. Note that we use a 1-layer classifier C (see Fig. 6.3) in this work. Our empirical test demonstrates that if we use more layers for classifier C, the AMDA without DA will perform even worse (i.e., the gap between AMDA with and without DA becomes larger) due to the general issue of overfitting.

In addition, there are some *easy* transfer cases, such as CW1→CW2 and CW2→CW1 scenarios, for which without DA can achieve an accuracy of 96.02% and 97.18%

as shown in Fig. 6.4. Meanwhile, CW4→CW1 and CW4→CW2 scenarios are *hard* to transfer, with the performance of 89.97% and 86.24% respectively. With our proposed AMDA model, we can achieve improvement for both *easy* and *hard* transfer cases, e.g., 3.33% for CW1→CW2 and 11.34% for CW4→CW2. Hence, AMDA can play a more important role and achieve better performance when domain discrepancies become larger and harder to transfer.

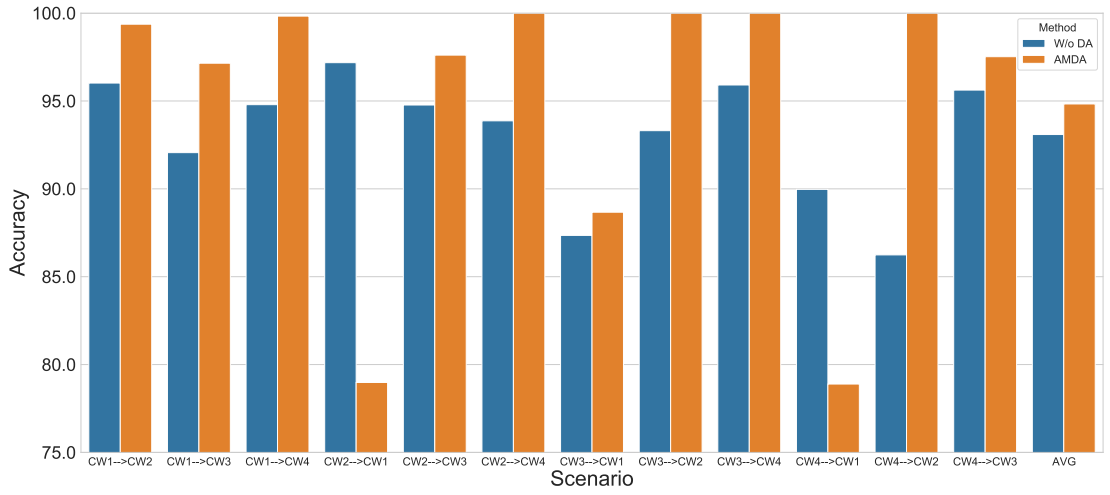


FIGURE 6.4: Evaluation of AMDA with and without domain adaptation on the CWRU dataset using 12 cross-domain scenarios

6.2.2.3 Comparison to Domain adaptation baselines

To demonstrate the superiority of the proposed AMDA, we implemented four domain adaptation baselines: Transfer Component Analysis (TCA) [173], Joint Distribution Adaptation (JDA) [174], Correlation Alignment (CORAL) [175], Deep Domain Confusion (DDC) [146], Deep Maximum Mean Discrepancy (MMD) [176], and Deep CORAL [177]

Table 6.2 shows the results of different domain adaptation techniques using the CWRU dataset. It can be found that the DDC achieves the best performance among baselines with an overall accuracy of 96.25%. The proposed AMDA outperforms all the baseline techniques on 12 domain adaptation scenarios with an overall accuracy of 99.13%, which indicates the effectiveness of the proposed AMDA for this domain adaptation task.

TABLE 6.2: Evaluation of AMDA on CWRU dataset against domain adaptation baselines using 12 cross-domain scenarios

	Shallow			Deep			
Scenario	CORAL	TCA	JDA	DDC	Deep MMD	Deep CORAL	**AMDA**
CW1→CW2	53.73	64.06	71.35	95.62	97.27	88.73	**99.35**
CW1→CW3	49.29	64.4	66.25	98.42	90.60	87.13	**99.70**
CW1→CW4	49.21	76.94	82.23	95.04	94.69	97.52	**99.52**
CW2→CW1	79.74	66.94	67.69	95.56	96.23	97.58	**98.56**
CW2→CW3	74.72	75.92	73.68	98.33	98.88	98.75	**99.95**
CW2→CW4	78.76	82.96	83.76	99.06	97.90	98.38	**99.31**
CW3→CW1	71.41	56.06	54.49	95.83	94.60	94.54	**99.10**
CW3→CW2	62.55	67.34	66.10	97.17	96.63	96.04	**98.62**
CW3→CW4	62.19	30.4	60.32	97.29	93.6	97.21	**99.65**
CW4→CW1	75.48	74.86	75.86	86.42	95.25	96.10	**98.27**
CW4→CW2	73.17	44.79	80.25	96.62	95.50	96.52	**97.58**
CW4→CW3	68.25	70.05	70.61	99.62	99.06	98.19	**99.97**
AVG	66.55	64.56	71.05	96.25	95.85	95.56	**99.13**

TABLE 6.3: Comparison with related works on 6 Transfer Scenarios

Method	**CW1→CW2**	**CW1→CW3**	**CW2→CW1**	**CW2→CW3**	**CW3→CW1**	**CW3→CW2**	**AVG**
DNN	82.2	92.6	72.3	77.0	76.9	77.0	79.60
WDCNN	99.2	91.0	95.1	91.5	78.1	85.1	90.00
TICNN	99.1	90.7	97.4	98.8	89.2	97.6	95.47
AMDA	**99.4**	**99.7**	**98.6**	**99.9**	**99.1**	**98.6**	**99.21**

6.2.2.4 Comparison to State-of-the-arts

To better evaluate the performance of our proposed AMDA model, we have also conducted experiments to compare it with 3 different state-of-the-art baselines, which are summarized as follows.

- The first approach is fault diagnosis using a deep neural network (DNN) [178], which consists of pre-training the stacked autoencoder in an unsupervised manner and fine-tuning the network under the supervision of source labels.

- The second approach is a 5-layer CNN with a wide input kernel which was demonstrated to achieve high accuracy (WDCNN) [179].
- The last method is transfer inference with a convolutional neural network (CNN) with a 6-layer CNN and introduces dropout in the first input layer. Additionally, ensemble learning has been employed to stabilize the performance of their model [180].

Table 6.3shows the performance comparison between the proposed AMDA model with three state-of-the-art methods. For these three competing methods, they only reported their results on 6 cross-domain experiments. Therefore, we also conducted the same cross-domain experiments for a fair evaluation.

We observe that our proposed AMDA method achieves better results than the three existing methods consistently. Note that almost all the methods have achieved good results for *easy* transfer cases (e.g., CW1→CW2), however, they fail to achieve good results in more challenging tasks with high domain discrepancies (e.g., C→CW1). Nevertheless, with well-designed adversarial domain adaptation, our AMDA model can achieve significant improvements over all the state-of-the-art methods. Furthermore, this excellent performance is achieved under the challenging settings of 1SmT by adapting multiple targets simultaneously in one training phase, in comparison with only one single target at a time for all the competing methods.

6.2.3 Case 2: Paderborn Bearing Dataset

6.2.3.1 Dataset Description

The Paderborn Bearing Dataset[1] is generated from sensor readings that are deployed under four different operating conditions, denoted as P1, P2, P3, and P4. Each operating condition refers to different operating parameters, including rotational speed, load torque, and radial force [2]. In our experiments, each operating condition is considered as one domain, while all the domains have the same fault classes i.e., healthy, inner-bearing damage, and outer-bearing damage. Eventually, we can perform 12 cross-condition scenarios for domain adaptation. A detailed explanation of this dataset has been covered in Chapter 5.

[1] `https://mb.uni-paderborn.de/en/kat/main-research/datacenter/bearing-datacenter/data-sets-and-download`

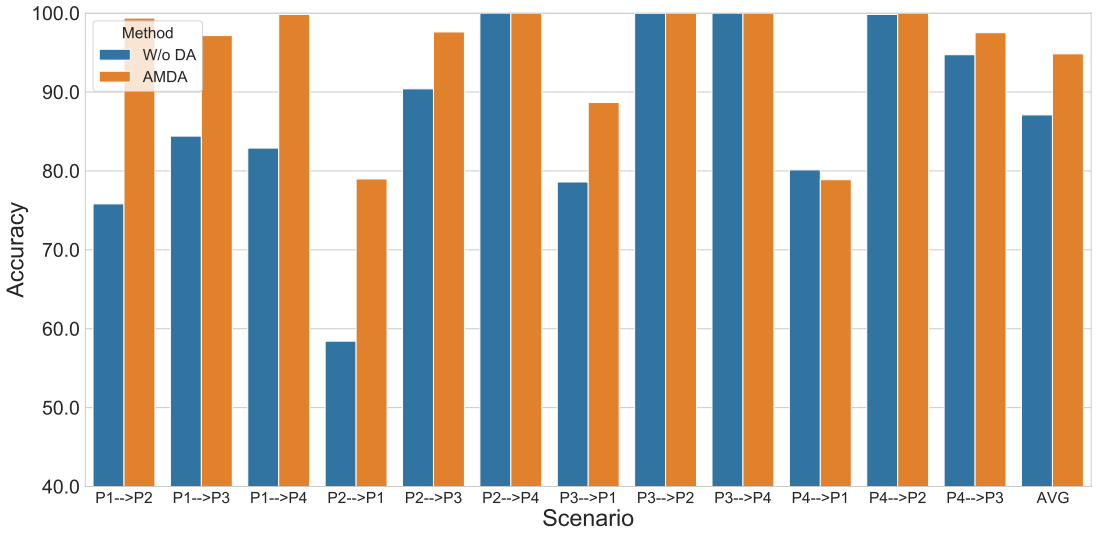


FIGURE 6.5: Evaluation of AMDA with and without domain adaptation on the Paderborn dataset using 12 cross-domain scenarios

TABLE 6.4: Evaluation of AMDA on the KAt dataset against domain adaptation baselines using 12 cross-domain scenarios

	Shallow			Deep			
Scenario	CORAL	TCA	JDA	DDC	Deep MMD	Deep CORAL	**AMDA**
P1→P2	55.77	42.06	65.21	49.77	81.39	84.06	**99.37**
P1→P3	66.24	68.47	70.60	60.33	84.16	87.03	**97.15**
P1→P4	56.25	45.36	64.90	59.31	91.04	88.80	**99.83**
P2→P1	43.42	53.00	80.07	59.14	81.18	**80.65**	78.98
P2→P3	79.81	80.56	74.05	**97.84**	97.83	90.17	97.61
P2→P4	87.26	93.42	82.03	99.80	99.98	99.99	**100.00**
P3→P1	50.51	63.36	85.26	**89.14**	81.63	83.22	88.67
P3→P2	88.91	92.26	87.10	94.94	99.67	99.98	**100.00**
P3→P4	87.46	90.62	82.50	99.69	99.97	99.99	**100.00**
P4→P1	34.81	51.59	74.89	**86.07**	89.14	80.44	78.89
P4→P2	94.11	96.96	91.14	99.87	99.66	100.00	**100.00**
P4→P3	77.68	84.45	74.88	97.62	97.72	**98.50**	97.52
AVG	68.52	71.84	77.72	82.79	91.95	91.07	**94.83**

TABLE 6.5: Comparison with state-of-the-art methods

Method	P2→P3	P2→P4	P3→P2	P3→P4	P4→P2	P4→P3	AVG
ACDIN	79.43	78.73	85.07	90.53	79.53	75.60	81.48
WDCNN	72.33	94.70	69.33	69.77	93.67	70.27	78.35
Alexnet	78.87	98.47	65.93	66.20	96.03	74.07	79.93
Resnet	71.33	96.67	64.53	67.23	92.73	72.60	77.52
ICN	80.67	96.97	70.23	70.67	94.27	79.50	82.05
AMDA	**97.61**	**100.00**	**100.00**	**100.00**	**100.00**	**97.52**	**99.19**

6.2.3.2 Experimental Results

We also conducted 12 cross-domain experiments on the Paderborn dataset to validate the performance of our proposed AMDA model. For example, we employed the working condition P1 as the source domain and P2, P3, and P4 as multiple target domains to generate the results for cross-domain tasks P1→P2, P1→P3 and P1→P4.

Fig. 6.5 shows the evaluation results of our AMDA model with and without domain adaptation (without DA). Over 12 cross-domain tasks, AMDA achieves an average accuracy of 94.83%, which is 7.73% higher than without DA. Once again, it demonstrates that the designed domain adaptation technique in the AMDA model is effective for cross-domain fault diagnosis.

As shown in Fig. 6.5, without DA achieves relatively low performance for the 6 tasks involving the working condition E (i.e., P1→P2, P1→P3, P1→P4, P2→P1, P3→P1 and P4→P1) with an average accuracy of 76.71%. This indicates that changing the rotational speed would cause a more significant domain shift than changing load torque or radial force, leading to a large domain discrepancy between E and the other 3 domains F, G, and H. However, our AMDA can perform very well for this 6 *hard* transfer tasks - it achieves an average accuracy of 90.48%, with a significant improvement of 13.77% over without DA.

6.2.3.3 Comparison with Domain Adaptation Baselines

Here, we compare the proposed AMDA method with the same domain adaptation baselines, i.e., TCA, JDA, CORAL, DDC, Deep MMD, and Deep CORAL. Table 6.4 presents the comprehensive evaluation of various methods across 12 different transfer tasks. In over 8 out of 12 cross-domain tasks, our AMDA method performs better than the implemented baselines. Overall, AMDA achieves the highest average accuracy of 94.83% as shown in Table 6.4, which is 3.76% higher than the second-best method, i.e., Deep CORAL.

6.2.3.4 Comparison with the state-of-the-art

The authors [153] reported the performance of 5 deep learning-based methods on the Paderborn dataset using 6 cross-domain scenarios. These 5 state-of-the-art methods include ACDIN [181], WDCNN [179], AlexNet [26], ResNet [182], and ICN [153]. In particular, AlexNet and ResNet, which are famous convolutional architectures for image classification, were applied for fault diagnosis [153]. Meanwhile, the other three methods are recently proposed for fault diagnosis. For example, ACDIN [181] refers to a deep inception network with atrous convolution. The inception part in ACDIN concatenates multiple filters of different sizes to support different resolutions, while atrous convolution is a dilated filter to support a wider input field. WDCNN [179] implements five 1-dimensional convolutional layers with a wide input kernel. ICN [153] is an inception-based capsule network for fault diagnosis, where the capsule network [183] is employed to capture the correlation between different features, and inception is used to extract features on different resolutions.

For a fair comparison, we selected the same cross-domain scenarios for AMDA and the 5 state-of-the-art methods. Table 6.5 shows the performance of various methods over 6 transfer tasks on the Paderborn dataset. Overall, our AMDA significantly surpasses the 5 competing approaches with an average accuracy of 97.52%, which is 15.47% higher than ICN (the second-best method).

6.2.4 Case 3: Self-collected Dataset

6.2.4.1 Dataset Description

We collected an additional dataset based on the drive train dynamic simulator (DDS) platform [184] for further verification. The sampling rate of the vibration signal is 5120Hz. This dataset encompasses three different working conditions, denoted as D1, D2, and D3. Table 6.6 shows the corresponding loading torque for each working condition. Each working condition consists of one normal class and three types of faults, i.e., inner-race (IR), outer-race (OR), and ball-crack (BC). We also use sliding windows with overlaps to segment the data, while the window size and the step size are the same as the CWRU dataset.

TABLE 6.6: Different Working Conditions for the self-collected dataset

Working Condition	Loading Torque	Fault Type
D1	0 Nm	Normal, IR, OR, BC
D2	7.2 Nm	Normal, IR, OR, BC
D3	14.4 Nm	Normal, IR, OR, BC

TABLE 6.7: Comparison Against Domain Adaptation Baselines

	Shallow			Deep			
SCenario	CORAL	TCA	JDA	DDC	Deep MMD	Deep CORAL	**AMDA**
D1→D2	44.95	74.30	71.96	83.81	87.4	89.45	**92.42**
D1→D3	60.37	49.61	48.19	72.41	68.34	68.01	**73.04**
D2→D1	50.48	87.52	75.03	90.25	80.97	87.49	**93.15**
D2→D3	49.95	50.19	56.79	57.45	55.13	61.91	**74.6**
D3→D1	59.42	56.37	50.22	69.28	59.16	65.20	**94.17**
D3→D2	42.13	58.67	57.06	77.56	66.96	68.84	**93.44**
AVG	51.22	62.78	59.875	75.13	69.66	73.48	**86.80**

6.2.4.2 Experimental Results

We denote three working conditions as D1, D2, and D3, which correspond to load 0 Nm, 7.2 Nm, and 14.4 Nm respectively. Thus, six cross-domain experiments

for our proposed method with and without DA have been performed as shown in Fig. 6.6. The results are consistent with our previous evaluation that DA can significantly improve the performance of fault diagnosis. Specifically, the proposed AMDA achieves an average accuracy of 86.80%, which is 11.50% higher than that without DA. This further indicates the effectiveness of the proposed method for cross-domain fault diagnosis.

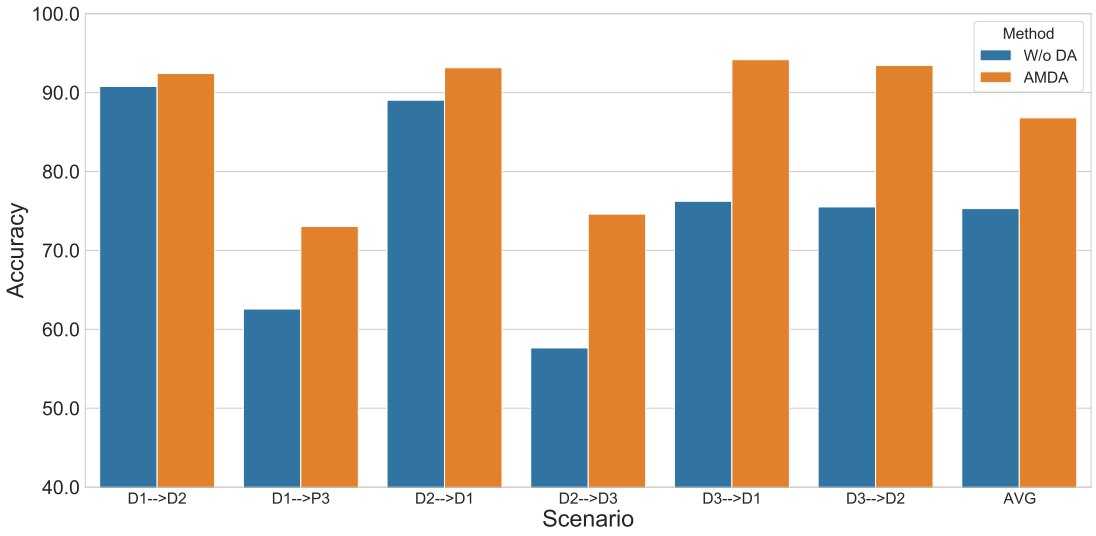


FIGURE 6.6: Evaluation of AMDA with and without domain adaptation on a self-collected dataset.

6.2.4.3 Comparison with Domain Adaptation Baselines

Similar to the previous evaluation, we compare against well-known approaches for DA, including conventional DA methods (i.e., TCA, JDA, and CORAL) and deep DA methods (i.e., DDC, Deep MMD, and Deep CORAL). The results for the six cross-domain experiments are demonstrated in Table6.7. Due to the relatively large gap (load variation) between domains, the performances of all the approaches degrade to some extent. Consistently, our AMDA method outperforms the benchmark approaches in all six cross-domain scenarios.

6.2.5 Evaluation of Proposed 1SmT Setting

In this section, we compare the 1SmT setting and 1S1T setting on the Paderborn dataset in terms of generalization and time efficiency. For 1S1T, we selected the

DDC method as it is the best baseline as shown in both Tables 6.2 and 6.4. In addition to 1SmT and 1S1T settings, we further constructed the 1SmxT settings by mixing N target domains as a single target domain. We also ran DDC and our AMDA under the 1SmxT setting and included their results for comparison.

Table 6.8 illustrates the accuracy of AMDA and DDC under different settings. The column E_source in Table 6.8 means that E is used as the source domain and F, G, and H are target domains (similarly for columns F_source, G_source, and H_source). Our AMDA (1SmT) outperforms AMDA (1SmxT) by 3.38% and also significantly outperforms DDC under both 1S1T and 1SmxT settings. For DDC itself, mixing the target domains, i.e., DDC (1SmxT), leads to performance deterioration of 6.84% compared to DDC (1S1T).

TABLE 6.8: Accuracy (%) of AMDA and DDC under different settings

	P1_source	**P2_source**	**P3_source**	**P4_source**	**AVG**
AMDA (1SmT)	98.78	92.20	96.22	92.14	**94.83**
AMDA (1S1T)	97.94	95.35	96.33	98.81	**97.11**
AMDA (1SmxT)	93.66	92.13	92.96	87.05	91.45
DDC (1SmxT)	45.78	80.94	92.70	84.37	75.95
DDC (1S1T)	56.47	85.59	94.59	94.52	82.79

TABLE 6.9: Training Time (sec) of different approaches under 1smT and 1S1T settings

Model	Total Time
AMDA (1SmT)	712.07
AMDA (1S1T)	1781.12
DDC (1SmT)	3848.23
DDC(1S1T)	5915.65
DANN (1SmT)	**644.92**
DANN(1S1T)	964.63

In addition, we can observe that AMDA (1S1T)–which is also our implementation–achieves higher accuracy than AMDA (1SmT). However, AMDA (1SmT) has higher scalability than AMDA (1S1T) and can generalize well to multiple target domains. In particular, AMDA (1SmT) can significantly reduce the model training compared with AMDA (1S1T), as shown in Table 6.9. Therefore, our proposed AMDA (1SmT) is more suitable than AMDA (1S1T) for practical scenarios.

6.3 Summary

In this chapter, a novel adversarial multiple-target domain adaptation (AMDA) approach has been introduced. The proposed AMDA provides a more scalable solution for dynamic working environments via generalizing to multiple working conditions concurrently. The adversarial training approach has been leveraged to find domain invariant representation for the multiple target domains. The efficacy of AMDA has been evaluated on two public datasets and one self-collected dataset. The experimental results show the clear capability of AMDA to generalize well to multiple, unlabeled, and different working conditions, addressing the fourth objective of our thesis. Albeit scalable, the AMDA work assumes access to unlabeled data from the target working conditions. However, with continuous changes in the working condition, access to data from the new working condition in advance can be infeasible. Therefore, in the next chapter, we aim to address the domain generalization problem to handle the new unseen working conditions with no prior data available.

Chapter 7

Conditional Contrastive Domain Generalization for Fault Diagnosis

The previous chapters have successfully realized various domain adaptation approaches to address the lack of labeling problem under variable working conditions, which either generalize to a single working condition such as CADA and SLARDA, or multiple working conditions such as AMDA. However, these approaches assume access to unlabeled data from the target working condition beforehand. Such assumption may not hold in many practical scenarios for the following reasons. First, access to the target domain data during the training phase may not be attainable when encountering a new target domain with no prior data. Besides, under DA settings, a new model needs to be trained independently for each new target domain, which can be time-consuming and not scalable solutions for dynamic working environments. Hence, there is an urgent need for a more realistic fault diagnosis model that can generalize to new unseen domains without prior data.

To tackle this issue, Domain Generalization (DG), a more practical yet challenging scenario, leverages data from multiple source domains to generalize well to new unseen domains. Few studies have investigated domain generalization for fault diagnosis [86, 87, 89, 185]. A predominant approach is to leverage adversarial learning to minimize the discrepancy of the marginal distribution between multiple source domains. Yet, in fault classification tasks, the marginal distribution of the data can inherently encompass multi-modal class structures. Thus, only aligning

The work in this chapter has been published in IEEE Transactions on Instrumentation and Measurement, 2022

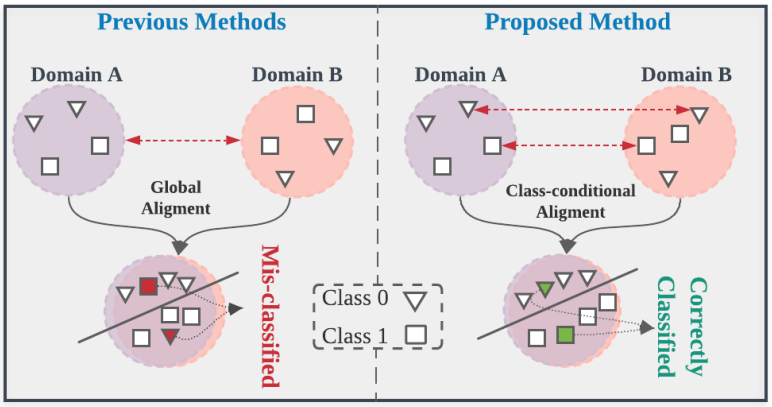


FIGURE 7.1: (Best viewed in colors). Comparison between previous and proposed methods in distribution matching. **Left**: Aligning the marginal distribution without considering fine-grained class distributions tends to misclassify samples among domains. **Right**: Our proposed approach with its class-conditional contrastive loss can successfully align the classes among different domains.

the marginal distribution without considering the fine-grained class distribution within each domain tends to fail in some challenging scenarios [186], as shown in Fig. 7.1. Recently, contrastive learning has achieved widely acclaimed performance in representation learning for visual applications. The key idea is to learn feature representations such that similar samples (i.e., positive pairs) are pulled together while dissimilar samples (i.e., negative pairs) are pushed away [187]. However, most of the existing contrastive learning approaches heavily rely on domain-specific augmentations to construct the positive pairs, which can be laborious and require extensive expert knowledge.

In this chapter, a novel conditional contrastive domain generalization approach (CCDG) is proposed for fault diagnosis. Particularly, instead of relying on laborious domain-specific augmentations, we leverage the variability among multi-domains (i.e., working conditions) to define more representative and realistic positive and negative pairs. Additionally, to realize class-conditional invariance, we maximize the mutual information between the prediction scores of the same classes among different domains while minimizing mutual information among different classes.

The main contributions of this paper are summarized as follows.

1. We propose a novel domain generalization approach to tackle a more realistic yet challenging task in real-world fault diagnosis. It does not require any data from the target domain during training.

2. We design a new conditional contrastive loss across realistic pairs from multi-source domains to find the domain invariant class representation, leading to robust and discriminative representations for domain generalization.

3. We provide a theoretical derivation of the mutual information lower-bound of our conditional contrastive loss on multi-domain data.

4. We extensively evaluate our proposed CCDG approach on two rotating machinery datasets. The experimental results clearly show the efficacy of our approach in generalizing to new unseen domains with no prior data.

7.1 Condition Contrastive Domain Generalization

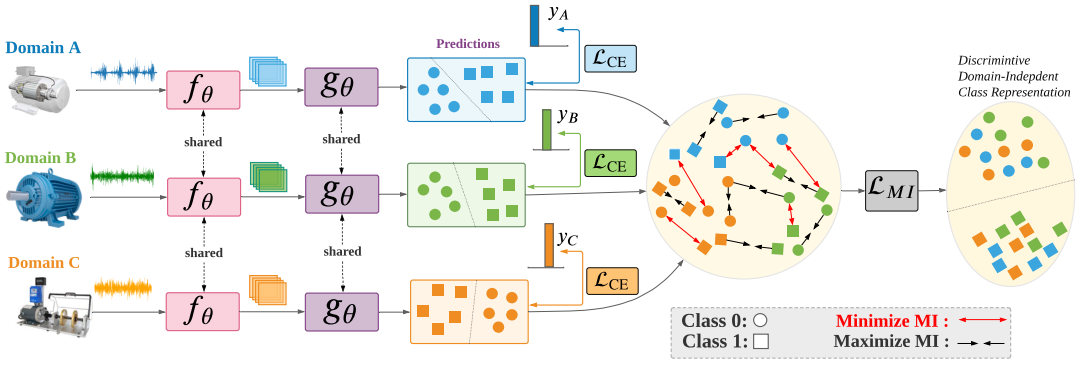


FIGURE 7.2: The framework of the proposed conditional contrastive domain generalization (CCDG). First, we pass the data from different domains through a shared network f_θ (i.e., feature extractor) to find the feature representation of the data. Second, a shared prediction network g_θ (i.e., predictor) converts the features to prediction scores, and we calculate the task-specific cross-entropy loss $\mathcal{L}_{CE}$ between the predicted labels and true labels. Third, we minimize a novel conditional contrastive loss $\mathcal{L}_{MI}$ to maximize both the intra-class similarity and inter-class separability. We train both the feature extractor f_θ and the predictor g_θ by optimizing $\mathcal{L}_{CE}$ and $\mathcal{L}_{MI}$ together (Best viewed in colors).

In this section, we first formulate the domain generalization problem. Then, an overview of our CCDG approach is presented. Subsequently, we describe each loss component of our CCDG approach in detail. Last, we provide an information-theoretic perspective for our class-conditional contrastive loss.

7.1.1 Problem Formulation

In this section, we provide a mathematical formulation for the domain generalization and domain invariance problems.

7.1.1.1 Domain Generalization Problem

Let $\mathcal{X}$ denote the feature space and $\mathcal{Y}$ denote the label space, a domain is represented by the joint distribution P_{XY} defined on $\mathcal{X} \times \mathcal{Y}$. In the domain generalization problem, we are given multiple source domains $(\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^M\}$, where M is the total number of source domains. Each domain $\mathcal{D}^i = \{\mathbf{x}_j^i,\ y_j^i\}_{j=1}^{N_i} \sim P_{XY}^i$ has N_i number of samples. The joint distributions vary among the multiple source domains, i.e., $P_{XY}^1 \neq P_{XY}^2 \ldots \neq P_{XY}^M$. The objective of the domain generalization is to learn a model $h : \mathcal{X} \rightarrow \mathcal{Y}$ based on all the source domains and generalize well to a new unseen target domain $D^{test} = \{\mathbf{x}_j^{test}, y_j^{test}\}_{j=1}^{N_{test}} \sim P_{XY}^{test}$, where $P_{XY}^{test} \neq P_{XY}^i \; for \; i \in \{1, \ldots, M\}$. It is worth emphasizing that we do not have any access to D^{test} during the training procedure.

7.1.1.2 Domain Invariance Problem

Most of existing domain generalization approaches assume that only marginal distribution changes among domains, i.e., $P_1^S(X) \neq P_2^S(X) \ldots \neq P_M^S(X)$, while the conditional distribution $P(Y|X)$ remains stable. Therefore, most of the existing approaches address the domain generalization problem via finding domain invariant representation on the feature space such that $P_1^S(f_\theta(X)) = P_2^S(f_\theta(X)) \cdots = P_M^S(f_\theta(X))$, where $f_\theta : \mathcal{X} \rightarrow \mathcal{F}$ is a deep neural network that maps from the space of the raw input time series signals (i.e., $\mathcal{X}$) to the vectorized feature space (i.e., $\mathcal{F}$).

In reality, $P(Y|X)$ may also be unstable among the domains. Therefore, only finding domain invariant features can be an insufficient condition to perfectly align the domains.

7.1.2 Overview of CCDG

In this work, we propose a novel conditional contrastive domain generalization (CCDG) approach that aligns the class-conditional distributions among multiple source domains to boost the generalization performance on unseen target domains. Fig. 7.2 shows the overall framework of our proposed CCDG method. First, we design a shared feature extractor f_θ for feature learning and a shared predictor g_θ for fault diagnosis. Second, we calculate the task-specific cross-entropy loss $\mathcal{L}_{CE}$ between the predicted labels and true labels. Third, we propose a novel conditional contrastive loss $\mathcal{L}_{MI}$ to maximize the mutual information among the same classes from different domains while minimizing the mutual information among different classes. Eventually, we train both the feature extractor f_θ and the predictor g_θ by optimizing the task-specific loss $\mathcal{L}_{CE}$ and the conditional contrastive loss $\mathcal{L}_{MI}$ concurrently in an end-to-end manner. In the next subsections, we will introduce these two losses, i.e., $\mathcal{L}_{CE}$ and $\mathcal{L}_{MI}$, in detail.

7.1.3 Task-specific Learning

The first step in our CCDG is to optimize the model performance on the corresponding task by leveraging the known class labels in the training data. The task-specific cross-entropy loss is formalized in the Equation 7.1 below.

$$\mathcal{L}_{CE} = -\frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}\sum_{j=1}^{N_i}\left[\ell(h_\theta(\mathbf{x}_j^i), y_j^i))\right], \tag{7.1}$$

where $\mathbf{x}_j^i$ refers to the j^{th} sample from the domain D^i, y_j^i is the class label of $\mathbf{x}_j^i$, and $h_\theta(\mathbf{x}_j^i) = g_\theta(f_\theta(\mathbf{x}_j^i))$ is a function that maps the raw inputs to the class predictions. Given C classes, both $\hat{y}_j^i$ and y_j^i are C-dimensional vectors, where $\hat{y}_j^i = h_\theta(\mathbf{x}_j^i)$ and y_j^i represent the predicted and true class labels, respectively. Then, the cross-entropy loss between the predicted labels and the true ones $\ell(\hat{y}_j^i, y_j^i)$ can be represented as follows:

$$\ell(\hat{y}_j^i, y_j^i) = \sum_{c=1}^{C} y(c)_j^i \times \log \hat{y}(c)_j^i. \tag{7.2}$$

Here, $y(c)_j^i$ is 1 if this sample $\mathbf{x}_j^i$ is from class c and 0 otherwise, and $\hat{y}(r)_j^i$ is the predicted score for the class c.

7.1.4 Class-conditional Contrastive Loss

The variability of working environments among the source domains can produce task-irrelevant features in each source domain. As a result, the class representation can be different from one domain to another. Such inconsistency of class representations among the multiple source domains can harm the generalization performance on new unseen target domains. Therefore, only optimizing the cross-entropy loss $\mathcal{L}_{CE}$ without considering the cross-domain relationship can hurt the generalization performance. To tackle this issue, we jointly optimize our class-conditional contrastive loss $\mathcal{L}_{MI}$. Particularly, we consider the cross-domain relationship via maximizing the mutual information between similar classes across different domains while removing the task-irrelevant information that is caused by the change in working environments. By doing so, our $\mathcal{L}_{MI}$ can capture the shared class information and obtain environment-independent class representation which can boost generalization performance on new unseen environments (i.e., domains).

Formally, given an anchor sample $\mathbf{x}_u \in B = \{b_1, b_2, \ldots, b_M\}$, where B is the set of batches from M source domains. We aim to maximize the mutual information between $\mathbf{x}_u$ and the corresponding positive samples in B while minimizing its similarity with the corresponding negative pairs in B. In particular, the positive samples of $\mathbf{x}_u$ are represented by all the samples in B that belong to the same class $pos(u) = \{\mathbf{x}_v \in B : y_u = y_v\}$. While the negative samples are the remaining samples in B that belong to different classes $neg(u) = \{\mathbf{x}_k \in B : y_u \neq y_k\}$. Our proposed approach is different from traditional approaches in two aspects. First, traditional approaches only align the global feature representation between different domains. As a result, samples that belong to different classes (i.e., $\mathbf{x}_u$ and $\mathbf{x}_k$) can be pulled together, despite the global distributions of different domains being well aligned. Differently, we consider a class-wise alignment where only samples

that belong to the same class are pulled together. Second, these approaches usually applied on the feature space (i.e., $f_\theta(\mathbf{x}_j^i)$. While our CCDG approach is applied on the class prediction level (i.e., $g_\theta(f_\theta(\mathbf{x}_j^i)))$ of the corresponding samples, which enables the model to find class-conditional invariant representation among multiple source domains.

The class-conditional contrastive loss of the sample $\mathbf{x}_u$, denoted as $\mathcal{L}_{MI}^u$. To formalize our class-conditional contrastive loss, we follow standard notations of contrastive learning in [188], which is represented as follows:

$$\begin{aligned}\mathcal{L}_{MI}^u &= \frac{-1}{|pos(u)|}\sum_{v\in pos(u)}\left(\log\frac{e^{(\sigma(\boldsymbol{h}_u,\boldsymbol{h}_v)/\tau)}}{\sum_{k\in neg(u)}e^{(\sigma(\boldsymbol{h}_u,\boldsymbol{h}_k)/\tau)}}\right)\\ &= \frac{-1}{|pos(u)|}\sum_{v\in pos(u)}\left(\log\underbrace{e^{(\sigma(\boldsymbol{h}_u,\boldsymbol{h}_v)/\tau)}}_{\text{positives}}\right.\\ &\qquad\left.-\log\sum_{k\in neg(u)}\underbrace{e^{(\sigma(\boldsymbol{h}_u,\boldsymbol{h}_k)/\tau)}}_{\text{negatives}}\right).\end{aligned} \tag{7.3}$$

Here, $|pos(u)|$ is cardinality of the positive sample set, $\sigma(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}^\intercal\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$ is the similarity scoring function for any given vectors $\mathbf{a},\mathbf{b}$, and τ is the temperature parameter to control the contrastive power. $\boldsymbol{h}_u = g_\theta \odot f_\theta(\mathbf{x}_u) \in \mathbb{R}^C$ are the logits of $\mathbf{x}_u$, $g_\theta \odot f_\theta$ represents the composition of the feature extractor and the predictor, and C is the dimension of the logits vector (i.e., the number of classes). The $\mathcal{L}_{MI}^u$ represents the contrastive loss for the anchor sample $\mathbf{x}_u$. Particularly, via optimizing $\mathcal{L}_{MI}^u$, we maximize the mutual information between the classifier prediction of the anchor sample $\boldsymbol{h}_u$ and all corresponding positive samples (i.e., samples that belong to the same class of the anchor sample $pos(u)$) across different domains, where $\boldsymbol{h}_v$ is sampled from the set of positive samples. Concurrently, we minimize the mutual information between the anchor sample and all the negative samples (i.e., all samples that belong to different classes from the anchor sample $neg(u)$.

The overall contrastive loss for all the samples is calculated as follows:

$$\mathcal{L}_{MI} = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}\sum_{u=1}^{N_i}\mathcal{L}_{MI}^u. \tag{7.4}$$

By minimizing $\mathcal{L}_{MI}$, we can maximize the lower bound of the mutual information among the positive samples [149]. A theoretical derivation that shows the mutual information lower-bound of the proposed approach can be found in Appendix B.

7.1.5 Overall Loss for Optimization

In our CCDG, both $\mathcal{L}_{MI}$ and $\mathcal{L}_{CE}$ are jointly optimized to improve the generalization performance on new unseen domains. Particularly, the objective of the cross-entropy loss is to improve the task-specific performance within each source domain. While the objective of the contrastive loss $\mathcal{L}_{MI}$ is to consider cross-domain relationships via finding domain-independent class representation, which can boost the generalization performance on unseen domains. The overall objective is then represented by a convex combination between them in Equation7.5.

$$\min_{f_\theta, g_\theta} \mathcal{L} = \alpha \mathcal{L}_{MI} + (1 - \alpha)\mathcal{L}_{CE}, \tag{7.5}$$

where α is the weight between the two losses. We use Adam [189] as the optimizer to minimize the overall objective and learn the feature extractor f_θ and the predictor network g_θ.

7.2 Experiments and Results

In this section, we first introduce the datasets and the setup of our experiments. Then, we present the evaluation results of our proposed CCDG method.

7.2.1 Datasets

We used two datasets to evaluate the performance of our CCDG approach.

- CWRU dataset: To better validate the domain generalization task, more working conditions have been included by the sensor readings from both the drive-end and fan-end. In total, 8 different working conditions have been

considered for our experiments, which are further illustrated in table 7.1. Detailed description CWRU dataset can be found in Chapter 6.

- Paderborn dataset: We included working environments with both artificial and real faults. Therefore, six different operating conditions were included. Table 7.2 illustrates the detailed specifications of each working condition. To prepare the data samples for the Paderborn dataset, we adopted sliding windows with a fixed length of 5,120 and a shifting size of 4,096 [20]. As such, we generated 12,340 for each artificial domain and 13,640 samples for each real domain respectively. Further details about Paderborn dataset can be found in Chapter 5.

TABLE 7.1: Details of CWRU bearing dataset.

Domain	Torque	Location	Fault Type	Fault Size (inches)
CW1	0 hp	Drive End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW2	1 hp	Drive End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW3	2 hp	Drive End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW4	3 hp	Drive End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW5	0 hp	Fan End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW6	1 hp	Fan End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW7	2 hp	Fan End	Normal, IF, OF, BF	0, 007, 0.014, 0.021
CW8	3 hp	Fan End	Normal, IF, OF, BF	0, 007, 0.014, 0.021

TABLE 7.2: Details for Paderborn dataset.

Domain	Damage Type	Load Torque [Nm]	Radial Force [N]
P1	Artificial	0.1	1000
P2	Artificial	0.7	400
P3	Artificial	0.7	1000
P4	Real	0.1	1000
P5	Real	0.7	400
P6	Real	0.7	1000

7.2.2 Experimental Setup

7.2.2.1 Baseline Methods

To show the efficacy of our proposed CCDG method, we first adapted state-of-the-art methods on visual domain generalization. Besides, we re-implemented domain generalization methods proposed for fault diagnosis tasks. To ensure fair evaluation, the same backbone architecture has been used for both our approach and the baseline methods. The compared baselines are shown as follows:

- Empirical Risk Minimization (**ERM**)[190]: it minimizes the sum of the empirical risk among the samples from all the domains.
- Maximum Mean Discrepancy (**MMD**)[176]: it minimizes the MMD distance between each pair of domains to improve the generalization performance.
- Deep Correlation Alignment (**Deep CORAL**)[175]: it aligns the co-variance matrices for each pair of domains.
- Conditional Domain Adversarial Networks (**CDANN**)[191]: it applies a separate domain classifier for each class to further improve the alignment performance.
- Beyond empirical risk minimization **Mixup** [192]: it generates new data samples via linear interpolation of samples and labels among random pair of domains.
- Self-supervised Contrastive Regularization for Domain Generalization (**SelfReg**) [193]: it applies stochastic weight averaging with inter-domain curriculum learning with contrastive regularization.
- A Hybrid Generalization Network for Intelligent Fault Diagnosis (**IEDGNet**) [89]: it leverages both extrinsic and intrinsic generalization objectives to regularize the discriminating performance of the deep neural network on fault diagnosis tasks.
- Domain Generalization for Rotating machinery (**DGRM**)[86]: it adversarially trains a feature extractor against a multi-class domain discriminator network to improve the generalization performance of rotating machinery.

TABLE 7.3: Details of hyper-parameter tuning setup.

Method	**Hyperparameter**	**Range**
MMD	Regularization λ	10^{-3} to 10^{-1}
Deep CORAL	Regularization λ	10^{-3} to 10^{-1}
DGRM, CDANN	Discriminator lr	10^{-5} to $10^{-3.5}$
	Discriminator weight decay	10^{-6} to 10^{-2}
	Discriminator Adam β_1	$\{0, 0.5\}$
	Discriminator steps	2^0 to 2^3
	Discriminator GP	10^{-2} to 10^1
	Adversarial regularization λ	10^{-2} to 10^2
Mixup	Beta shape parameter α	10^{-1} to 10^1
IEDGNet	adversarial regularization λ	10^{-2} to 10^2
	triplet loss weightβ	10^{-2}to10^2
CCDG	Temperature τ	10^{-1} to 10^0
	Contrastive weight α	10^{-1} to 10^0

7.2.2.2 Implementation Details

In our experiments, during the training phase, we split the data from multiple source domains into 80% for training and 20% for validation to monitor the model convergence on the source domains during training phase. To test the generalization performance, we iteratively leave one whole domain out for testing while using all the other domains for training, following the standard protocol of domain generalization [194]. It is worth highlighting that we neither use the testing data for the training nor the model selection. Instead, we select the model based on its performance on the validation sets of the source domains, which can be more practical and aligns with the domain generalization assumption that no test data are available during the training process. The accuracy score (i.e., the number of correctly classified test samples divided by the total number of test samples) is utilized to measure the diagnosis performance. All the experiments were conducted using PyTorch 1.7 on NVIDIA GeForce RTX 2080 Ti GPU. We implemented a 6-layer 1-D CNN to extract features from the raw input signals followed by a 4-layer fully connected network (FC) for fault classification. We used LeakyReLU as a non-linear activation layer and batch normalization to speed up the training process. To promote fair evaluation, the same network architecture is adopted for all the baseline methods.

To train the model, we fix the learning rate as 0.001, weight decay as $5e^{-5}$, and batch size as 32 throughout the whole experiment. To select the hyper-parameters, we ran 20 trials of hyper-parameter sweep for our approach and all baseline methods to ensure a fair evaluation. We sample the hyper-parameter values from a *uniform*

distribution within a specific range [194]. Detailed ranges for our approach and baseline methods can be found in Table 7.3. Note that the ERM method does not contain any tunable hyperparameters in this setup. For more practical model selection, we validate our model only on the held-out subsets of the source domains without access to any data from the target domain.

TABLE 7.4: Domain generalization results on CWRU bearing dataset (Accuracy %).

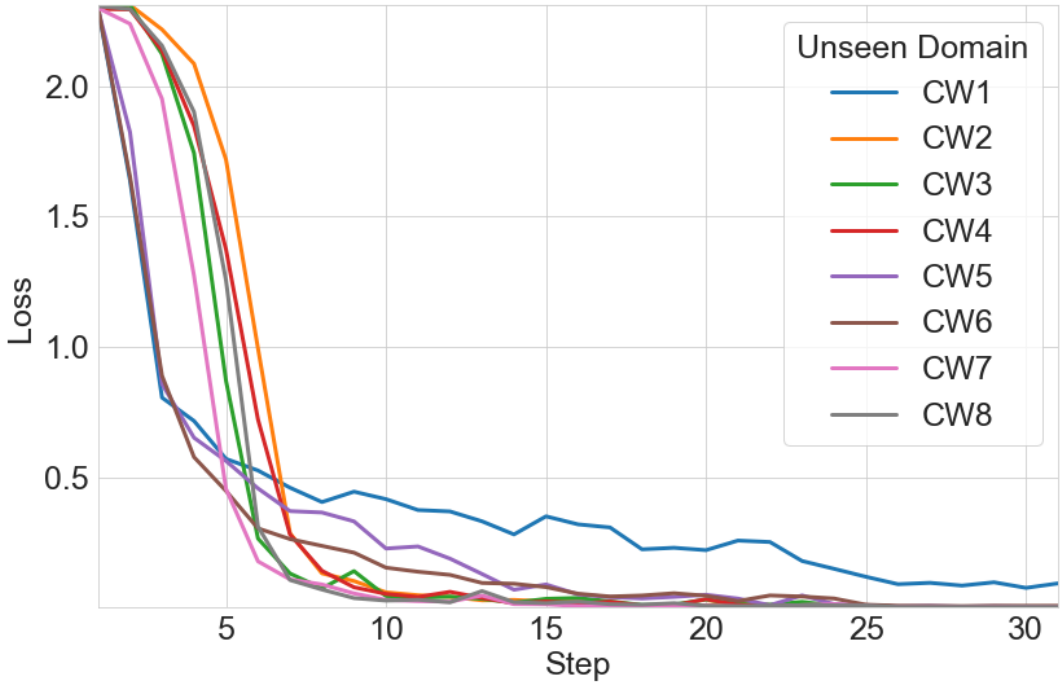
	Unseen Target Domains								
Method	CW1	CW1	CW3	CW4	CW5	CW6	CW7	CW8	Average
ERM	64.17 ± 9.2	94.58 ± 1.6	96.38 ± 2.9	70.58 ± 4.4	68.33 ± 7.7	83.46 ± 3.1	94.38 ± 4.8	86.38 ± 4.6	82.28
MMD	47.83 ± 9.2	57.17 ± 13.4	75.67 ± 3.7	66.67 ± 1.2	61.54 ± 16.0	73.17 ± 8.5	71.75 ± 10.8	54.17 ± 17.7	63.50
Deep CORAL	63.29 ± 15.1	70.96 ± 8.8	87.25 ± 4.4	72.13 ± 4.0	66.67 ± 3.5	71.71 ± 6.9	68.71 ± 3.8	76.71 ± 5.0	72.18
CDANN	64.08 ± 11.6	82.67 ± 4.9	90.13 ± 6.7	77.50 ± 5.4	72.13 ± 8.5	74.75 ± 3.6	89.21 ± 7.3	74.29 ± 11.3	78.09
Mixup	57.63 ± 7.9	83.92 ± 5.9	93.96 ± 4.7	75.79 ± 7.8	81.88 ± 8.3	**95.38 ± 2.7**	**98.13 ± 0.5**	88.96 ± 3.6	84.45
Self-Reg	62.83 ± 1.7	95.71 ± 3.0	93.54 ± 4.2	76.29 ± 0.6	82.71 ± 0.3	79.54 ± 20.3	89.42 ± 0.9	93.79 ± 2.4	84.23
IEDGNet	45.54 ± 11.5	68.88 ± 5.4	81 ± 6.3	54.67 ± 1.1	54.71 ± 9.2	67.17 ± 1.6	65.25 ± 5.3	73.58 ± 7.7	63.85
DGRM	55.29 ± 7.1	78.42 ± 15.4	91.63 ± 4.5	81.79 ± 3.4	67.58 ± 3.4	77.29 ± 1.1	85.08 ± 4.1	85.25 ± 4.2	77.79
CCDG	**76.79 ± 2.9**	**97.25 ± 1.2**	**99.83 ± 0.1**	**93.71 ± 1.5**	**89.25 ± 1.4**	92.04 ± 3.5	94.42 ± 3.5	**94.33 ± 1.4**	**92.20**

7.2.3 Experimental Results

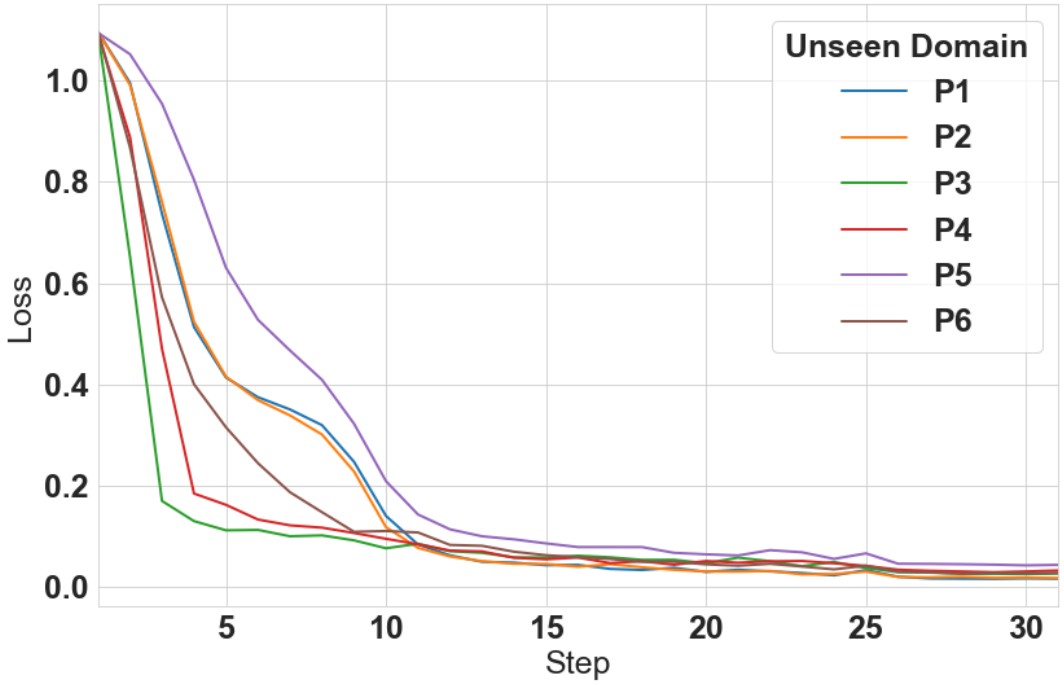
7.2.3.1 Results on CWRU Dataset

Table 7.4 shows the results of different approaches using the CWRU dataset. In particular, column A in Table 7.4 means that we use domain A as the unseen target domain and the other 7 domains as source domains. The last column shows the average performance for various methods over 8 target domains. Note that the best values on each target domain are highlighted in bold, while the second-best values are underlined. It can be found that our CCDG approach significantly outperforms all the baseline approaches. On average, the CCDG outperforms the second-best baseline (i.e., Mixup) by 7.75%.

Domains A and E have zero loading torque, while other domains have non-zero loading torque. Therefore, fault features in these two domains could be quite different from those in other domains, and it would be challenging to map other domains to A and E. This explains why various methods achieve unsatisfactory performance on CW1 and CW4. However, our CCDG approach, with its class-conditional invariance, can still perform the best in these two challenging domains. It outperforms ERM (second best) on domain CW1 by 12.62%, and outperforms Self-Reg (second best) on domain E by 6.54%, respectively.



(a) CWRU Dataset



(b) Paderborn Dataset

FIGURE 7.3: Convergence of training loss for different Unseen Target Domains on both CWRU and Paderborn Datasets.

TABLE 7.5: Domain generalization results on Paderborn dataset (Accuracy %).

	Unseen Target Domains						
Method	P1	P2	P3	P4	P5	P6	Average
ERM	83.23 ± 2.0	67.50 ± 0.2	88.36 ± 0.9	99.25 ± 0.2	74.78 ± 7.2	99.40 ± 0.4	85.42
MMD	77.69 ± 3.2	69.18 ± 0.4	81.04 ± 1.4	69.53 ± 17.5	65.74 ± 5.8	82.38 ± 8.0	74.26
Deep CORAL	80.58 ± 0.9	71.30 ± 2.4	79.27 ± 1.3	81.33 ± 0.4	67.23 ± 2.0	80.58 ± 0.2	76.71
CDANN	78.77 ± 1.7	71.89 ± 5.2	86.90 ± 0.1	92.16 ± 7.8	74.16 ± 13.0	94.90 ± 3.0	83.13
Mixup	80.96 ± 4.3	71.66 ± 5.3	78.26 ± 1.5	72.52 ± 1.1	55.22 ± 3.7	74.24 ± 2.1	72.14
Self-Reg	**85.41 ± 1.5**	73.01 ± 2.5	91.79 ± 1.7	97.58 ± 0.6	68.76 ± 0.3	98.99 ± 0.5	85.92
IEDGNet	64.17 ± 8.3	63.38 ± 4.7	80.23 ± 7.5	91.83 ± 0.3	66.14 ± 3.8	92.14 ± 3.9	76.31
DGRM	76.20 ± 1.8	73.28 ± 3.7	86.56 ± 0.5	88.60 ± 10.4	76.37 ± 11.1	88.16 ± 7.5	81.53
CCDG	80.46 ± 1.7	**73.29 ± 4.2**	**94.61 ± 1.3**	**99.87 ± 0.1**	**83.08 ± 4.2**	**99.88 ± 0.1**	**88.52**

7.2.3.2 Results on Paderborn Dataset

AppendixBTo further evaluates the superiority of our CCDG method, we have conducted additional experiments on the Paderborn dataset. Table 7.5 shows the performance of various methods on the Paderborn dataset. It can be observed that our CCDG approach outperforms the state-of-the-art methods on 5 out of 6 domains with an average accuracy of 88.52%. Moreover, it outperforms the second-best baseline (i.e., Self-Reg) by 2.60%. Notably, generalization to domains with small radial force (i.e., J and M) can be a very challenging task. However, our CCDG method with its class-conditional contrasting can still be the best in these two domains, demonstrating the robustness of our method against the large domain shifts. Additionally, unlike Self-Reg which only focuses on positive pairs, our CCDG approach aims to push away negative pairs that belong to different classes, which can improve the generalization performance. Moreover, methods that only align the marginal distribution like Deep CORAL and MMD perform poorly on the tough domains, i.e., J and M.

TABLE 7.6: Results on Paderborn dataset for single source domain generalization.

Scenario	P2→P1	P1→P2	P3→P4	P4→P3	P5→P6	P6→P5	Average
ERM	72.54 ± 0.7	65.45 ± 5.4	42.90 ± 12	50.50 ± 3.2	76.58 ± 2.6	87.16 ± 5.8	65.85
Self-Reg	**76.46 ± 2.8**	66.98 ± 1.1	34.72 ± 5.8	49.23 ± 3.0	77.13 ± 3.2	91.81 ± 1.2	66.06
CCDG	75.20 ± 2.1	**67.91 ± 3.1**	**44.49 ± 11**	**51.81 ± 1.8**	**78.18 ± 1.9**	**93.33 ± 6.6**	**68.49**

7.2.3.3 Single Source Domain Generalization

In this experiment, we measure the efficacy of our approach under extreme cases where only samples from a single source domain are available for training. Table

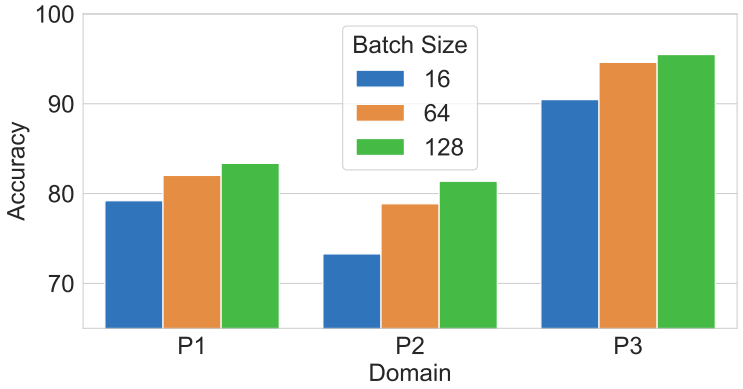


FIGURE 7.4: Effect of batch size on CCDG performance.

7.6 shows the results of the proposed approach versus the second-best performing methods (i.e., ERM and Self-Reg) on randomly selected scenarios from the Paderborn dataset. Overall, our CCDG performs best on 5 out of 6 cross-domain scenarios with an average improvement of 2.5%. While Self-Reg performs better than ERM, it can still underperform our CCDG approach. This is because that Self-Reg is mainly relying on image-specific augmentations, which may not be effective for fault diagnosis datasets. Besides Self-Reg can only align positive pairs (i.e., samples from the same class) without considering the negative pairs. Differently, our CCDG approach improves the performance by pushing away samples that belong to different classes within the same domain to have clear decision boundaries. Besides, the lower performance of our single-domain CCDG compared to multi-domain CDDG reflects the importance of having a realistic representation of variability among multiple domains.

7.2.3.4 Generalization to multiple unseen domains

In this experiment, we measure the generalization ability of our approach on multiple unseen target domains. Particularly, we have evaluated our approach on 6 randomly selected scenarios with multiple unseen target domains. Besides, to further show the efficacy of our approach, we compared it against SelfReg and ERM baselines (second and third best methods in this dataset). In our comparison, we reported the average performance on the multiple unseen domains. Our superior

generalization performance persists even with more challenging scenarios of multiple unseen domains. Moreover, our CCDG approach significantly outperforms both ERM and SelfReg baselines (strongest baselines) in all the scenarios.

TABLE 7.7: Generalization Performance of our CCDG approach on multiple unseen target domains on Paderborn Dataset (Accuracy %).

Scenario	Source Domains	Unseen Domains	ERM	SelfReg	CCDG
S1	P3, P4, P5, P6	P1, P2	75.68	77.95	**81.33**
S2	P1, P2, P5, P6	P3, P4	94.47	94.55	**97.65**
S3	P1, P2, P3, P4	P5, P6	85.49	82.61	**87.45**
S4	P2, P3, P5	P1,P4 , P6	78.25	87.39	**81.75**
S5	P4, P4, P5	P1, P2, P6	84.05	77.64	**86.40**
S6	P1, P2, P6	P3, P4, P5	85.23	82.38	**91.48**

7.2.4 Analysis

7.2.4.1 Training Convergence

To show the convergence of our proposed loss function, we have plotted the training convergence for each unseen target domain on both CWRU and Paderborn datasets, as shown in Fig. 7.3. For the CWRU dataset, the convergence is smooth for all the unseen domains while the highest rate of convergence happens with unseen domain G, as shown in Fig. 7.3a. Similarly, for Paderborn dataset, our training loss converges smoothly while the highest convergence rate is corresponding with unseen domain K, as shown in Fig. 7.3b.

7.2.4.2 Effect of batch size

We investigate the dependency of our CCDG on the number of negative samples within the min-batch. To do so, we report the performance of three different batch sizes on three unseen domains of Paderborn datasets, as shown in Fig. 7.4. It can be seen that our CCDG benefits from a larger batch size consistently among the three different domains. A possible explanation is that larger batch sizes help in providing more negative examples per batch, which can improve model convergence.

7.2.4.3 Parameter Sensitivity Analysis

In our CCDG, we have two main parameters, namely, the contrastive loss weight α and the contrasting temperature τ. To measure the model sensitivity to each of them, we fix one parameter while changing the other and vice versa. We plot the average performance over the 8 domains of the CWRU dataset, as shown in Fig. 7.5. First, we fixed the temperature value τ as 0.03 while varying the weighting factor α from 0.01 to 0.9. Notably, better generalization performance can be achieved with higher weight for the contrastive loss against the cross-entropy loss, while the best performance is achieved with α equals 0.7. However, the performance tends to degrade with α values larger than 0.7. This means that reducing the cross-entropy loss weight to lower values than 0.3 can hurt performance. To sum up, despite the contrastive loss can be more important than cross-entropy loss, the contribution of both losses is necessary for the best performance. Second, to measure the model dependency on the temperature parameter of the contrastive loss, we fixed α as 0.7 while varying the contrasting temperature τ from 0.01 to 0.9. It clearly shows that varying the temperature weight can change the performance from 82% at higher temperature values (i.e., 0.9) to 89% at lower temperature values (i.e., 0.03). In a nutshell, both α and τ can be of great importance to the model performance.

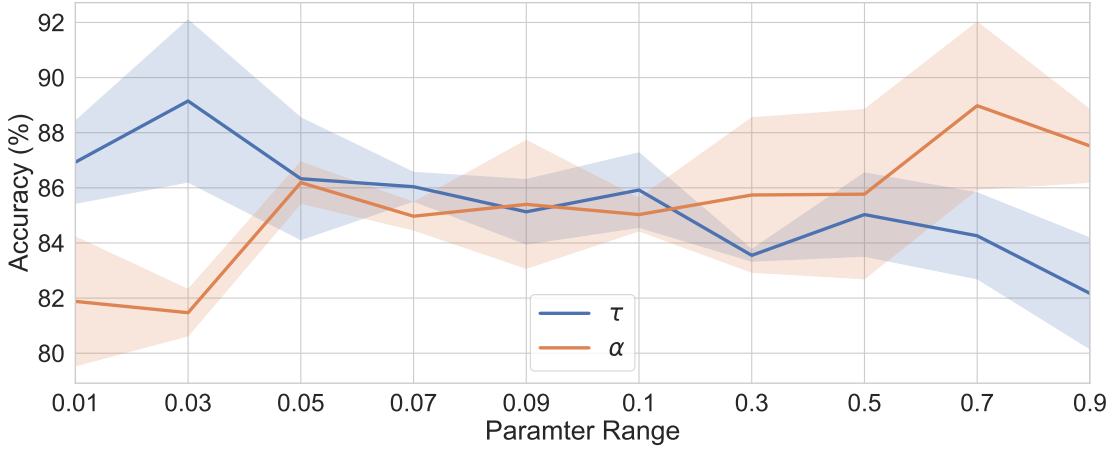
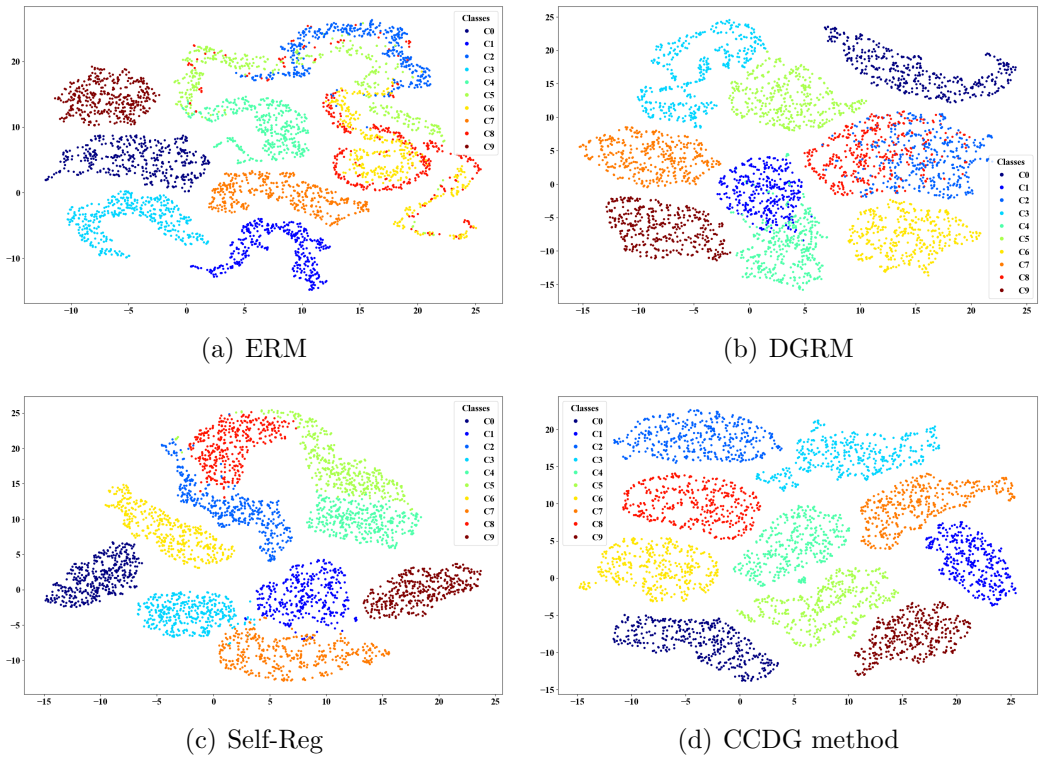


FIGURE 7.5: Sensitivity analysis for the CCDG parameters.



(a) ERM (b) DGRM

(c) Self-Reg (d) CCDG method

FIGURE 7.6: UMAP visualization results in unseen target domain D of the CWRU dataset for different methods.

7.2.4.4 Feature Visualization

To clearly show the effectiveness of the proposed CCDG, we visualize the learned features of our approach against three different baselines on unseen domain D of the CWRU dataset. We utilize the uniform manifold approximation and projection (UMAP) [195] to map the high dimensional features to a lower dimension as shown in Fig. 7.6.

Fig. 7.6a shows the learned features of ERM, which can be hardly separable, specifically for classes C2, C5, C6, and C8. Similarly, the visualization of DGRM and Self-Reg can still have less-separable classes such as C2, C5, and C8, as shown in Fig. 7.6b, 7.6c. Differently, our CCDG approach can separate different classes with clear boundaries, as illustrated in Fig. 7.6d.

The visualization results of CCDG suggest that it can learn a domain-independent class representation that can be transferable to new unseen domains.

7.3 Summary

In this chapter, we proposed a novel conditional contrastive domain generalization approach called CCDG that addresses a more challenging yet practical domain generalization problem for the real-world fault diagnosis of rolling machinery. Specifically, we leveraged data from multiple source domains to generalize to a new unseen target domain (i.e., an unseen working condition for fault diagnosis). By experimenting with two different datasets and comparing them against various solid baseline methods, we showed that achieving the conditional invariance across the class predictions of different source domains can significantly improve the generalization performance on unseen domains. The promising performance of our proposed CCDG method pushes towards more practical data-driven approaches that can work under challenging real-world environments.

Chapter 8

Conclusion and Future Work

The chapter will conclude the thesis by summarizing the key research aims, outlining the contributions, and discussing the limitations of this thesis while providing recommendations for future direction in the area of data-driven predictive maintenance.

8.1 Conclusion

The main purpose of this thesis is to develop robust and generalizable data-driven approaches that can work in the dynamic manufacturing environment, towards more applicable data-driven approaches in real-world industries. To fulfill this aim, four main research questions have been settled and successfully addressed. The research questions and their corresponding solutions are summarized as follows.

First, in Chapter 3 we investigated the first question of exploiting the historical information of sensory data to improve the robustness and generalization of deep learning models on the RUL prediction task. As reviewed in Chapter 2, the existing deep learning methods have only focused on the latest information to predict the RUL labels. Besides, most of the existing approaches only rely on a single supervised objective to perform the feature extraction step and RUL prediction step, limiting the generality of the learned features and degrading the performance on the testing data. Therefore, to address these issues, an Attention-based sequence-to-sequence model (ATS2S) has been developed with two main components: (1)

A novel reconstruction-based auxiliary task to improve the feature representation; (2) an attention-based decoding component to focus on the important historical time steps during the learning the procedure. The developed approach can prioritize important historical time steps while learning a more representative feature representation, resulting in a more robust and general model that outperforms state-of-the-art approaches.

The second question investigated the problem of designing adaptive deep learning models that can generalize to different working conditions with no labeled data available. In particular, Chapters 4 and 5 are mainly addressing these two questions for fault prognosis and fault diagnosis respectively. In Chapter 4, a novel CADA approach has been developed to improve the generalization of different unlabeled working conditions for RUL prediction tasks. The CADA approach aims to find domain invariant features while preserving the domain-specific information to adapt to the unlabeled working environment. The experimental results show that the CADA significantly outperforms state-of-the-art with over 21% and 38% improvements respectively in terms of two different evaluation metrics. On the other hand, in Chapter 5, a novel SLARDA approach has been proposed to improve cross-domain fault classification under the scarcity of labels. The SLARDA approach leveraged the temporal information to improve the features' transferability. In addition, to better adapt to the unlabeled working environment, an autoregressive discriminator network is developed to consider the temporal dimension during the alignment step. Last, a teacher model with robust pseudo labels is proposed to handle the conditional shift between different domains. The efficacy of the proposed SLARDA has been proved over state-of-the-art methods.

As both CADA and SLARDA can only generalize to a single unlabeled working condition at a time, the third question is investigating how can we improve the scalability of these approaches to multiple working conditions concurrently. To address this question, we developed a novel adversarial multiple target domain adaptation (AMDA) approach that leverages adversarial training to find a unified invariant representation for multiple unlabeled target domains, enabling a single model to generalize well to multiple working conditions. Extensive experimental results of three different datasets show that the proposed AMDA is capable of generalizing to multiple working environments concurrently.

All the above approaches in Chapter 4, 5, and 6 follow the domain adaptation problem setting that necessitates the existence of unlabeled data from the target working conditions beforehand. This can still be a tight assumption for many practical situations that require testing the model directly on new working conditions on-the-fly with no prior data available. Therefore, the final question is to scrutinize how can we can leverage historical information to generalize to new working conditions on-the-fly. To address this question, we developed a novel conditional contrastive approach (CCDG) to improve the model's generalization capability to new working environments on-the-fly. The CCDG approach can capture shareable class information and learn environment-independent class representation, enabling direct generalizing to new unseen working conditions. Extensive experiments showed the capability of the CCDG approach on handling new working environments with outstanding performance that significantly outperforms state-of-the-art approaches.

8.2 Future Work

In the final section of this thesis, we highlight the shortcoming of the proposed thesis while suggesting the potential future directions towards completely realistic data-driven approaches.

8.2.1 Few-shot Domain Adaptation

As unsupervised domain adaptation (UDA) assumes only unlabeled target data available, it fails to address situations where new classes emerge in the target domain. Therefore, to address this limitation, we leverage the few-shot domain adaptation scenario that assumes access to few-shots of labeled data from the target domain. By doing so, we are able to correctly classify the new emerging classes in the target domain that has not been seen in the source domain. Moreover, while UDA assumes abundant unlabeled data available, in reality, it can be challenging to obtain a vast amount of data and only a few samples can be available. Thus, the few-shot domain adaptation can also be leveraged to handle such situations.

8.2.2 Domain Adaptation Under Label Shift

The main objective of this thesis was to address the distributional shift caused by the variability of working conditions under scarce-labeled data regimes. Yet, most of the proposed approaches in this thesis work under the closed-set domain adaptation settings, where the label sets are assumed to be identical between the source and target domains. This assumption may not hold in many practical scenarios where new fault types can emerge in the target domain. In many real-world scenarios, the relationship between label sets (e.g, fault types) of the source and target domains can either be different or unknown. Therefore, exploring different categories of domain adaptation such as open set, partial, or Universal Domain Adaptation approaches can be more realistic and practical [196]. Specifically, in open set settings, it is assumed the source domain classes are subsets of the target domains. As a result, the model can encounter new classes in the target domain, which has to be identified as unknown. Partial DA settings assume that target classes are a subset of the source classes, therefore the model is required to disregard the additional source classes during the adaptation process. A more challenging yet practical scenario is the assumption that the relationship is fully unknown, and the model is required to handle such situations by correctly classifying the known classes while identifying the new classes as unknown classes. Figure 8.1 illustrates all the possible category shifts that can exist between the source and target label sets. This direction can open further possibilities for the realization of data-driven approaches in real-world scenarios.

8.2.3 Privacy-Preserving Domain Adaptation

The proposed approaches in this thesis always assume the possibility of sharing the source and target data. However, this may not be attainable in many practical scenarios due to privacy constraints. For instance, consider a situation where the source domain data belong to one entity while the target domain data belong to a different entity. Existing domain adaptation approaches necessitate the sharing of these data between different entities. However, sharing the data can badly violate privacy and confidentiality constraints, which is of great importance to many industrial entities. Moreover, In the big data era, sharing the whole data of an entity can add huge computational and storage overheads.

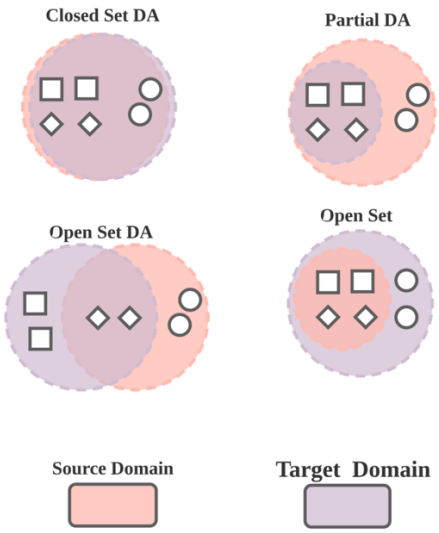


FIGURE 8.1: Different category shifts between the source and target domains. The closed Set DA assumes that source and target domains share the same set of classes. In Partial DA settings, the target classes are subsets of the source classes while in Open set settings the source classes are subsets of the target classes. In open Set DA, there are some source-private classes, target-private classes, and shared classes among both domains.

Federated Learning (FL), a promising paradigm that can train machine learning models while preserving data privacy has recently emerged [197]. Unlike conventional ways that transfer data between servers, FL transfers the parameters of the ML model securely and privately. Thus, coupling federated learning that can preserve data privacy with domain adaptation that can handle the domain shift problem can be a promising future direction [198]. Very recently some researchers have started to pay attention to this promising area, opening the possibilities for addressing the distribution shift problems under the data-privacy constraints [199].

8.2.4 Bayesian Deep Learning with Uncertainty Quantification

Data-driven approaches have widely acclaimed performance in the condition monitoring problem. Many sources of uncertainty are existing under the premise of real-world environments such as noisy sensory readings, imperfect measurements, and variability of working conditions. Thus, current deep learning models that only rely on point estimates can pose serious problems with uncertain predictions. Therefore, there is a need to quantify the uncertainty of deep learning models to be

able to provide how certain the model is. Bayesian deep learning approaches can have a strong potential to quantify uncertainty for data-driven predictive maintenance [200]. In standard deep learning, the weights are deterministic and the network learns by updating these weights to minimize the loss function. Differently, in Bayesian deep learning, the weights are no longer deterministic and it is considered random and is sampled from Gaussian distribution with parameters θ. Learning probability distribution over the weights of the neural network enables the network to have uncertainty estimate capability. Inspired by this and towards a more trusted data-driven fault diagnosis and prognosis, we aim to explore Bayesian deep learning approaches for fault diagnosis and prognosis to quantify the uncertainty of a deep neural network.

8.2.5 Explainable Deep Learning for Predictive Maintenance

Even with realizing robust and accurate deep learning models for predictive maintenance, the explainability of the black-box deep learning models can be a bottleneck for the applicability of deep learning in real-world applications. For sensitive applications such as the manufacturing industry, where unexpected behaviors not only cost economic losses but also cause human causalities, trusting a black-box predictor is not easily achievable. Therefore, there is an urgent need to realize explainable deep learning that can provide reasons behind the decisions and give descriptions to the end-users. Recently, there is a growing body of research field called Explainable Artificial Intelligence, which concerns about explainability of data-driven models for computer-vision applications [201]. Various methods have been developed for the explainability of data-driven models. For instance, a large body of research has targeted post-hoc methods by trying to explain fully trained models with an explainability layer [202]. Another set of works has focused on developing self-explainable models, these methods are named ante-hoc methods [203]. Nevertheless, little attention has been paid to applying XAI for predictive maintenance tasks. We believe realizing explainable and robust deep learning models can significantly boost the applicability of data-driven predictive maintenance.

8.3 Summary

The challenges and limitations of academic deep learning approaches when tested in a real manufacturing environment can be summarized as follows: (1) limitations of labeled data; (2) variability of working environments; (3) point estimates predictions without uncertainty quantification; (4) black-box predictions without precise explanations of the reasons behind decisions. Thus, towards addressing this gap, the proposed works in the thesis have addressed the first two points while the future works aim to address the other three points. In a nutshell, the thesis's contributions coupled with its future work can enable deep learning to be safely applied in practice.

Appendix A

Appendix for Chapter 5

A.1 Statistical Significance Test

we carried out a comparative analysis of our SLARDA approach and all other baselines based on the mean ranks of the accuracy values. We followed the procedure of Fawaz et al., to measure the statistical difference between different methods [204]. Particularly, we first used mean test accuracy over 10 runs to compare with the baseline methods. A Friedman test is used to reject the null hypothesis [205]. Then, a Wilcoxon signed-rank test is leveraged to measure the pair-wise statistical difference among the baseline methods [206]. Figure A.1 shows the Critical Difference (CD) diagram with the statistical comparison of our approach against all other methods. Methods that are not connected by a bold line have significantly different mean ranks with confidence level of 95%. Clearly, our SLARDA is significantly better than other approaches and ranked the highest compared to other baseline approaches.

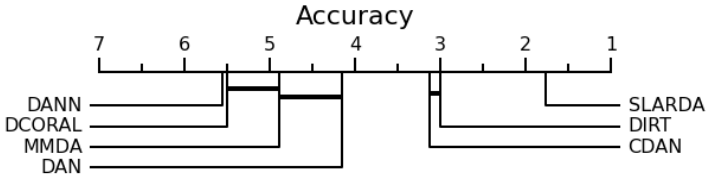


FIGURE A.1: Critical difference diagram that shows pair-wise statistical difference between our approach and baseline methods. Methods that are not connected by bold lines has significantly different mean ranks with confidence level of 95%

Appendix B

Appendix for Chapter 7

B.1 Connections to Mutual Information

Here, we provide the theoretical derivation of the lower bound of mutual information of our multi-domain class-conditional contrasting loss. Formally speaking, the optimal similarity score is proportional to the density ratio between the joint distribution and product of marginal distributions [149]. This can be formulated as follows:

$$\phi(a, b) \propto [\frac{p\,(a, b)}{p\,(a)\,p(b)}], \tag{B.1}$$

where $p\,(a, b)$ is the joint distribution of (a, b), and $p(a)p(b)$ is the product of marginal distributions.

In our approach, we aim to obtain the optimal contrastive loss $\mathcal{L}_{\mathrm{MI}}^{opt}$ for multi-domain setting. Particularly, for anchor $\boldsymbol{h}_u$, we contrast between the positive sample in the numerator, i.e., $(\boldsymbol{h}_u, \boldsymbol{h}_{\boldsymbol{v}})$ for $v \in pos(u)$, and all samples in the denominator, i.e., $(\boldsymbol{h}_u, \boldsymbol{h}_{\boldsymbol{i}})$ for $i \in \{0, \ldots, k\}$. It is assumed that the positive sample is at index $i = 0$, and the rest are negative samples, as in [207]. We substitute ϕ function from Eq. 5 into Eq. 6. Subsequently, we split the summation of the denominator into two parts: the term with $i = 0$, and the summation from $i = 1$ to k. Given that $\mathcal{L}_{\mathrm{MI}}^{opt}$ is the optimal contrastive loss, the mutual information between the anchor sample and negative sample, i.e. $(\boldsymbol{h}_u, \boldsymbol{h}_{\boldsymbol{i}})$ for $i > 0$, will be minimized. This means that the two samples can be considered as approximately independent

from each other and the ratio is $\frac{p(\boldsymbol{h_u},\boldsymbol{h_i})}{p(\boldsymbol{h_u})p(\boldsymbol{h_i})} \approx 1$ for $i > 0$ [149]. Last, given that $log(k)$ is independent from the summation, we take it out while replacing the remaining summation term by the mutual information notation (i.e., I). The full derivation can be viewed as follows:

$$
\begin{aligned}
\mathcal{L}_{\mathrm{MI}}^{opt} &= -\sum_{v \in pos(u)} \log \left[\frac{\phi\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)}{\sum_{i=0}^{k} \phi\left(\boldsymbol{h}_u, \boldsymbol{h_i}\right)} \right] \\
&= -\sum_{v \in pos(u)} \log \left[\frac{\frac{p(\boldsymbol{h}_u,\boldsymbol{h}_v)}{p(\boldsymbol{h}_u)p(\boldsymbol{h}_v)}}{\sum_{i=0}^{k} \frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}} \right] \\
&= -\sum_{v \in pos(u)} \log \left[\frac{\frac{p(\boldsymbol{h}_u,\boldsymbol{h}_v)}{p(\boldsymbol{h}_u)p(\boldsymbol{h}_v)}}{\frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}\Big|_{i=0} + \sum_{i=1}^{k} \frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}} \right] \\
&= \sum_{v \in pos(u)} \log \left[\frac{\frac{p(\boldsymbol{h}_u,\boldsymbol{h}_v)}{p(\boldsymbol{h}_u)p(\boldsymbol{h}_v)}}{\frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}\Big|_{i=0} + \sum_{i=1}^{k} \frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}} \right]^{-1} \\
&= \sum_{v \in pos(u)} \log \left[\frac{\frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}\Big|_{i=0} + \sum_{i=1}^{k} \frac{p(\boldsymbol{h}_u,\boldsymbol{h_i})}{p(\boldsymbol{h}_u)p(\boldsymbol{h_i})}}{\frac{p(\boldsymbol{h}_u,\boldsymbol{h}_v)}{p(\boldsymbol{h}_u)p(\boldsymbol{h}_v)}} \right] \\
&= \sum_{v \in pos(u)} \log \left[1 + \frac{p\left(\boldsymbol{h}_u\right) p\left(\boldsymbol{h}_v\right)}{p\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)} \sum_{i=1}^{k} \frac{p\left(\boldsymbol{h}_u, \boldsymbol{h_i}\right)}{p\left(\boldsymbol{h}_u\right) p\left(\boldsymbol{h_i}\right)} \right] \\
&\approx \sum_{v \in pos(u)} \log \left[1 + \frac{p\left(\boldsymbol{h}_u\right) p\left(\boldsymbol{h}_v\right)}{p\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)} k \right] \\
&\geq \log(k) - \sum_{v \in pos(u)} \log \left[\frac{p\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)}{p\left(\boldsymbol{h}_u\right) p\left(\boldsymbol{h}_v\right)} \right] \\
&\geq \log(k) - \sum_{v \in pos(u)} I\left(\boldsymbol{h}_u; \boldsymbol{h}_v\right).
\end{aligned}
\tag{B.2}
$$

With all the available positive samples $\sum_{v \in pos(u)} I\left(\boldsymbol{h}_u; \boldsymbol{h}_v\right) \geq \log(k) - \mathcal{L}_{\mathrm{MI}}^{opt}$, by minimizing $\mathcal{L}_{\mathrm{MI}}^{opt}\left(\boldsymbol{h}_u, \boldsymbol{h}_v\right)$, we can maximize the lower bound on the mutual information $I\left(\boldsymbol{h}_u; \boldsymbol{h}_v\right)$. Notably, as the number of negative samples k increases, the approximation can be more accurate.

List of Author's Awards, Patents, and Publications[1]

Awards

- **Finalist Paper Award**, "Adversarial Transfer Learning for Machine Remaining Useful Life Prediction," *2020 IEEE International Conference on Prognostics and Health Management (ICPHM).*

Journal Articles

- **Mohamed Ragab**, Zhenghua Chen, Min Wu, C. Kwoh, Ruqiang Yan, and Xiaoli Li. "Attention-based sequence to sequence model for machine remaining useful life prediction." Neurocomputing 466 (2021): 58-68.
- **Mohamed Ragab**, Zhenghua Chen, Min Wu, Chuan Sheng Foo, Chee Keong Kwoh, Ruqiang Yan, and Xiaoli Li. "Contrastive adversarial domain adaptation for machine remaining useful life prediction." IEEE Transactions on Industrial Informatics 17, no. 8 (2021): 5239-5249.
- **Mohamed Ragab**, Zhenghua Chen, Min Wu, Haoliang Li, C. Kwoh, Ruqiang Yan, and Xiaoli Li. "Adversarial Multiple-Target Domain Adaptation for Fault Classification." IEEE Transactions on Instrumentation and Measurement 70 (2021): 1-11.
- **Mohamed Ragab**, Zhenghua Chen, Wenyu Zhang, Emadeldeen Eldele, Min Wu, C. Kwoh, and Xiaoli Li. "Conditional Contrastive Domain Generalization for Fault Diagnosis

[1]The superscript * indicates joint first authors

- **Mohamed Ragab**, Emadeldeen Eldele, Zhenghua Chen, Min Wu, C. Kwoh, and Xiaoli Li. "Self-Supervised Autoregressive Domain Adaptation for Time Series Data." IEEE Transactions on Neural Networks and Learning Systems (2022).

- Xu, Qing, Zhenghua Chen, **Mohamed Ragab**, Chao Wang, Min Wu, and Xiaoli Li. "Contrastive Adversarial Knowledge Distillation for Deep Model Compression in Time-Series Regression Tasks." Neurocomputing (2021).

- Tiihonen, Armi, Sarah J. Cox-Vazquez, Qiaohao Liang, **Mohamed Ragab**, Zekun Ren, Noor Titan Putri Hartono, Zhe Liu, Shijing Sun, Cheng Zhou, Nathan C. Incandela, Jakkarin Limwongyut, Alex S. Moreland, Senthilnath Jayavelu, Guillermo C. Bazan and Tonio Buonassisi. "Predicting Antimicrobial Activity of Conjugated Oligoelectrolyte Molecules via Machine Learning." Journal of the American Chemical Society (2021).

Conference Proceedings

- Eldele, Emadeldeen, **Mohamed Ragab**, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. "Time-Series Representation Learning via Temporal and Contextual Contrasting." In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, 2352–59, 2021.

- **Mohamed Ragab**, Zhenghua Chen, Min Wu, Chee Keong Kwoh, and Xiaoli Li"Adversarial Transfer Learning for Machine Remaining Useful Life Prediction," *2020 IEEE International Conference on Prognostics and Health Management (ICPHM).*

- Zhang, Wenyu, **Mohamed Ragab**, and Ramon Sagara. "Robust domain-free domain generalization with class-aware alignment." In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2870-2874. IEEE, 2021.

- Jin, Chao, **Mohamed Ragab**, and Khin Mi Mi Aung. "Secure transfer learning for machine fault diagnosis under different operating conditions." In

International Conference on Provable Security, pp. 278-297. Springer, Cham, 2020.

- Yao, Zexi, Chao Jin, **Mohamed Ragab**, Khin Mi Mi Aung, and Xiaoli Li."DiagNet: Machine Fault Diagnosis Using Federated Transfer Learning in Low Data Regimes.", International Workshop on Trustable, Verifiable and Auditable Federated Learning in Conjunction with AAAI 2022

Bibliography

[1] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008. xxiii, 46, 50, 69

[2] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter 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. In *Proceedings of the European conference of the prognostics and health management society*, pages 05–08, 2016. xxiv, 92, 114

[3] G-Q. Jiang, P. Xie, X. Wang, M. Chen, and Q. He. Intelligent fault diagnosis of rotary machinery based on unsupervised multiscale representation learning. *Chinese Journal of Mechanical Engineering*, 30(6):1314–1324, 2017. xxvii, 111

[4] Jim Daily and Jeff Peterson. Predictive maintenance: How big data analysis can improve maintenance. In *Supply chain integration challenges in commercial aerospace*, pages 267–278. Springer, 2017. 1

[5] Fabio Arena, Mario Collotta, Liliana Luca, Marianna Ruggieri, and Francesco Gaetano Termine. Predictive maintenance in the automotive sector: A literature review. *Mathematical and Computational Applications*, 27 (1):2, 2022. 1

[6] Zaharah Allah Bukhsh, Aaqib Saeed, Irina Stipanovic, and Andre G Doree. Predictive maintenance using tree-based classification techniques: A case of railway switches. *Transportation Research Part C: Emerging Technologies*, 101:35–54, 2019. 2

[7] Michael Pecht and Jie Gu. Physics-of-failure-based prognostics for electronic products. *Transactions of the Institute of Measurement and Control*, 31(3-4): 309–322, 2009. 3

[8] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1):1–14, 2011. 3

[9] George J Vachtsevanos, Frank Lewis, Andrew Hess, and Biqing Wu. *Intelligent fault diagnosis and prognosis for engineering systems*, volume 456. Wiley Hoboken, 2006. 3, 12

[10] P. Konar and P. Chattopadhyay. Bearing fault detection of induction motor using wavelet and support vector machines (svms). *Applied Soft Computing*, 11(6):4203–4211, 2011. 3, 22

[11] Marco Cococcioni, Beatrice Lazzerini, and Sara Lioba Volpi. Robust diagnosis of rolling element bearings based on classification techniques. *IEEE Transactions on Industrial Informatics*, 9(4):2256–2263, 2012. 3, 22

[12] Mir Mohammad Ettefagh, Manizheh Ghaemi, and M Yazdanian Asr. Bearing fault diagnosis using hybrid genetic algorithm k-means clustering. In *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*, pages 84–89. IEEE, 2014. 3, 22

[13] Z. Chen, K. Gryllias, and W. Li. Intelligent fault diagnosis for rotary machinery using transferable convolutional neural network. *IEEE Transactions on Industrial Informatics*, pages 1–1, 2019. 3

[14] Han Liu, Jianzhong Zhou, Yang Zheng, Wei Jiang, and Yuncheng Zhang. Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders. *Isa Transactions*, 77:167–178, 2018. URL `https://academic.microsoft.com/paper/2801396593`. 3, 23, 32

[15] Wathiq Abed, Sanjay Sharma, Robert Sutton, and Amit Motwani. A robust bearing fault detection and diagnosis technique for brushless dc motors under non-stationary operating conditions. *Journal of Control, Automation and Electrical Systems*, 26(3):241–254, 2015. 3, 24, 32

[16] L Lueth, C Patsioura, D Williams, et al. The current state of data analytics usage in industrial companies. *Industrial Analytics, Digital Analytics Association Germany*, pages 38–49, 2016. 4

[17] Mohamed Ragab, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, Ruqiang Yan, and Xiaoli Li. Attention-based sequence to sequence model for machine remaining useful life prediction. *Neurocomputing*, 466:58–68, 2021. 10

[18] Mohamed Ragab, Zhenghua Chen, Min Wu, Chuan Sheng Foo, Chee Keong Kwoh, Ruqiang Yan, and Xiaoli Li. Contrastive adversarial domain adaptation for machine remaining useful life prediction. *IEEE Transactions on Industrial Informatics*, 17(8):5239–5249, 2020. 10

[19] Mohamed Ragab, Emadeldeen Eldele, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. Self-supervised autoregressive domain adaptation for time series data. *arXiv preprint arXiv:2111.14834*, 2021. 10

[20] Mohamed Ragab, Zhenghua Chen, Min Wu, Haoliang Li, Chee-Keong Kwoh, Ruqiang Yan, and Xiaoli Li. Adversarial multiple-target domain adaptation for fault classification. *IEEE Transactions on Instrumentation and Measurement*, 70:1–11, 2021. 10, 93, 131

[21] Mohamed Ragab, Zhenghua Chen, Wenyu Zhang, Emadeldeen Eldele, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. Conditional contrastive domain generalization for fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*, 71:1–12, 2022. 10

[22] Hasan Ferdowsi. *Model based fault diagnosis and prognosis of nonlinear systems*. Missouri University of Science and Technology, 2013. 12

[23] H. Venkateswara, S. Chakraborty, and S. Panchanathan. Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations. *IEEE Signal Processing Magazine*, 34(6):117–129, 2017. doi: 10.1109/MSP.2017.2740460. 15

[24] Hind Taud and JF Mas. Multilayer perceptron (mlp). In *Geomatic approaches for modeling land change scenarios*, pages 451–455. Springer, 2018. 15

[25] Raul Rojas. The backpropagation algorithm. In *Neural networks*, pages 149–182. Springer, 1996. 16

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 17, 117

[27] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*, volume 1. Citeseer, 2017. 17, 23, 29

[28] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020. 18

[29] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015. doi: 10.1109/MSP.2014.2347059. 19

[30] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2007. 20, 26, 32

[31] Taisheng Zheng, Lei Song, Jianxing Wang, Wei Teng, Xiaoli Xu, and Chao Ma. Data synthesis using dual discriminator conditional generative adversarial networks for imbalanced fault diagnosis of rolling bearings. *Measurement*, 158:107741, 2020. 20, 27, 32

[32] Xiaodong Zhang, Roger Xu, Chiman Kwan, Steven Y Liang, Qiulin Xie, and Leonard Haynes. An integrated approach to bearing fault diagnostics and prognostics. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 2750–2755. IEEE, 2005. 22

[33] Min Xia, Fanrang Kong, and Fei Hu. An approach for bearing fault diagnosis based on pca and multiple classifier fusion. In *2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*, volume 1, pages 321–325. IEEE, 2011.

[34] Jinane Harmouche, Claude Delpha, and Demba Diallo. A global approach for the classification of bearing faults conditions using spectral features. In *IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society*, pages 7352–7357. IEEE, 2013. 22

[35] Suyi Qian, Xiaoqiang Yang, Jie Huang, and Haitao Zhang. Application of new training method combined with feedforward artificial neural network for rolling bearing fault diagnosis. In *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–6. IEEE, 2016. 22

[36] Mir Mohammad Ettefagh, Manizheh Ghaemi, and M Yazdanian Asr. Bearing fault diagnosis using hybrid genetic algorithm k-means clustering. In *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*, pages 84–89. IEEE, 2014. 22

[37] David He, Ruoyu Li, Junda Zhu, and Mikhail Zade. Data mining based full ceramic bearing fault diagnostic system using ae sensors. *IEEE Transactions on Neural Networks*, 22(12):2022–2031, 2011. 22

[38] Shrinathan Esakimuthu Pandarakone, Yukio Mizuno, and Hisahide Nakamura. Distinct fault analysis of induction motor bearing using frequency spectrum determination and support vector machine. *IEEE Transactions on Industry Applications*, 53(3):3049–3056, 2017. 22

[39] Pankaj Malhotra, Vishnu Tv, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder. *arXiv preprint arXiv:1608.06154*, 2016.

[40] Abid Firas Ben, Zgarni Slaheddine, and Braham Ahmed. Bearing fault detection of induction motor using swpt and dag support vector machines. *IEEE Conference Proceedings*, 2016:1481, 2016. 22

[41] Qingbin Tong, Junci Cao, Baozhu Han, Xiaodong Zhang, Zhengwei Nie, Jiamin Wang, Yuyi Lin, and Weidong Zhang. A fault diagnosis approach for rolling element bearings based on rsgwpt-lcd bilayer screening and extreme learning machine. *IEEE Access*, 5:5515–5530, 2017. 22

[42] Xiaoli Zhao and Minping Jia. Fault diagnosis of rolling bearing based on feature reduction with global-local margin fisher analysis. *Neurocomputing*, 315:447–464, 2018. 22

[43] Ziwei Wang, Qinghua Zhang, Jianbin Xiong, Ming Xiao, Guoxi Sun, and Jun He. Fault diagnosis of a rolling bearing using wavelet packet denoising and random forests. *IEEE Sensors Journal*, 17(17):5581–5588, 2017. 22

[44] Bin Zhang, C Sconyers, C Byington, R Patrick, M E Orchard, and G Vachtsevanos. A probabilistic fault detection approach: Application to bearing fault detection. *IEEE Transactions on Industrial Electronics*, 58(5):2011–2018, 2011. URL `https://academic.microsoft.com/paper/2101946706`. 22

[45] Jing Tian, Michael H. Azarian, Michael Pecht, Gang Niu, and Chuan Li. An ensemble learning-based fault diagnosis method for rotating machinery. In *2017 Prognostics and System Health Management Conference (PHM-Harbin)*, pages 1–6, 2017. URL `https://academic.microsoft.com/paper/2765305400`. 22

[46] ZhiQiang Chen, Chuan Li, and René-Vinicio Sanchez. Gearbox fault identification and classification with convolutional neural networks. *Shock and Vibration*, 2015, 2015. 23, 32

[47] Olivier Janssens, Viktor Slavkovikj, Bram Vervisch, Kurt Stockman, Mia Loccufier, Steven Verstockt, Rik Van de Walle, and Sofie Van Hoecke. Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration*, 377:331–345, 2016. 23

[48] L. Wen, X. Li, L. Gao, and Y. Zhang. A new convolutional neural network-based data-driven fault diagnosis method. *IEEE Transactions on Industrial Electronics*, 65(7):5990–5998, July 2018. ISSN 0278-0046. doi: 10.1109/TIE.2017.2774777. 23

[49] Wendong Zhang, Fan Zhang, Wei Chen, Yongquan Jiang, and Dongli Song. Fault state recognition of rolling bearing based fully convolutional network. *Computing in Science and Engineering*, 21(5):55–63, 2019. 23

[50] Xiaojie Guo, Liang Chen, and Changqing Shen. Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, 93(93):490–502, 2016. URL `https://academic.microsoft.com/paper/2485614840`. 23

[51] Min Xia, Teng Li, Lin Xu, Lizhi Liu, and Clarence W. de Silva. Fault diagnosis for rotating machinery using multiple sensors and convolutional neural networks. *IEEE-ASME Transactions on Mechatronics*, 23(1):101–110, 2017. 23

[52] Zhuang Zilong and Qin Wei. Intelligent fault diagnosis of rolling bearing using one-dimensional multi-scale deep convolutional neural network based health state classification. In *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pages 1–6, 2018. 23, 32

[53] Shao Haidong, Jiang Hongkai, Li Xingqiu, and Wu Shuaipeng. Intelligent fault diagnosis of rolling bearing using deep wavelet auto-encoder with extreme learning machine. *Knowledge Based Systems*, 140:1–14, 2018. URL `https://academic.microsoft.com/paper/2765226309`. 23, 32

[54] Chuanhao Li, Wei Zhang, Gaoliang Peng, and Shaohui Liu. Bearing fault diagnosis using fully-connected winner-take-all autoencoder. *IEEE Access*, 6:6103–6115, 2018. URL `https://academic.microsoft.com/paper/2628062541`.

[55] Wentao Mao, Jianliang He, Yuan Li, and Yunju Yan. Bearing fault diagnosis with auto-encoder extreme learning machine: A comparative study:. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 231(8), 2017. URL `https://academic.microsoft.com/paper/2548257861`. 23, 32

[56] Chandrahas Mishra and D. L. Gupta. Deep machine learning and neural networks: An overview. *IAES International Journal of Artificial Intelligence*, 6(2):66–73, 2017. 23

[57] Feng Jia, Yaguo Lei, Jing Lin, Xin Zhou, and Na Lu. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, 72:303–315, 2016. URL `https://academic.microsoft.com/paper/2219903032`. 23, 32

[58] Wentao Mao, Jianliang He, Yuan Li, and Yunju Yan. Bearing fault diagnosis with auto-encoder extreme learning machine: A comparative study:. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 231(8), 2017. URL `https://academic.microsoft.com/paper/2548257861`. 23, 32

[59] Xiaojie Guo, Changqing Shen, and Liang Chen. Deep fault recognizer: An integrated model to denoise and extract features for fault diagnosis in rotating machinery. *Applied Sciences*, 7(1):41, 2016. URL `https://academic.microsoft.com/paper/2562639359`. 23

[60] J. Sun, C. Yan, and J. Wen. Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning. *IEEE Transactions on Instrumentation and Measurement*, 67(1):185–195, Jan 2018. ISSN 0018-9456. doi: 10.1109/TIM.2017.2759418. 23

[61] Honghu Pan, Xingxi He, Sai Tang, and Fanming Meng. An improved bearing fault diagnosis method using one-dimensional cnn and lstm. *Strojniski Vestnik/Journal of Mechanical Engineering*, 64, 2018. 24, 32

[62] Hongkai Jiang, Xingqiu Li, Haidong Shao, and Ke Zhao. Intelligent fault diagnosis of rolling bearings using an improved deep recurrent neural network. *Measurement Science and Technology*, 29(6):065107, 2018. 24, 32

[63] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X. Gao. Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115:213–237, 2019. URL `https://academic.microsoft.com/paper/2810292802`. 24

[64] Siyu Shao, Stephen McAleer, Ruqiang Yan, and Pierre Baldi. Highly accurate machine fault diagnosis using deep transfer learning. *IEEE Transactions on Industrial Informatics*, 15(4):2446–2455, 2019. doi: 10.1109/TII.2018.2864759. 24

[65] Hyunjae Kim and Byeng D. Youn. A new parameter repurposing method for parameter transfer with small dataset and its application in fault diagnosis of rolling element bearings. *IEEE Access*, 7:46917–46930, 2019. doi: 10.1109/ACCESS.2019.2906273. 24

[66] Xiaodong Wang, Feng Liu, and Dongdong Zhao. Cross-machine fault diagnosis with semi-supervised discriminative adversarial domain adaptation. *Sensors*, 20(13):3753, 2020. 25

[67] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010. ISSN 2326-3865. doi: 10.1109/TKDE.2009.191. 25, 59, 60, 61

[68] Xiang Li, Wei Zhang, and Qian Ding. Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks. *IEEE Transactions on Industrial Electronics*, 66(7):5525–5534, 2019. ISSN 02780046. doi: 10.1109/TIE.2018.2868023. 25, 81

[69] Rui Xia, Zhenchun Pan, and Feng Xu. Instance weighting for domain adaptation via trading off sample selection bias and variance. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden*, pages 13–19, 2018. 25, 32

[70] Zhongwei Zhang, Huaihai Chen, Shunming Li, and Zenghui An. Unsupervised domain adaptation via enhanced transfer joint matching for bearing fault diagnosis. *Measurement*, 165:108071, 2020. 26

[71] Fei Shen, Reza Langari, and Ruqiang Yan. Exploring sample/feature hybrid transfer for gear fault diagnosis under varying working conditions. *Journal of Computing and Information Science in Engineering*, 20(4), 2020. 26, 32

[72] Garrett Wilson and Diane J Cook. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(5):1–46, 2020. 26

[73] Weining Lu, Bin Liang, Yu Cheng, Deshan Meng, Jun Yang, and Tao Zhang. Deep model based domain adaptation for fault diagnosis. *IEEE Transactions on Industrial Electronics*, 64(3):2296–2305, 2017. 26, 32

[74] Xiang Li, Wei Zhang, and Qian Ding. A robust intelligent fault diagnosis method for rolling element bearings based on deep distance metric learning. *Neurocomputing*, 310:77–95, 2018. 26, 81, 101

[75] Bo Zhang, Wei Li, Xiao-Li Li, and See-Kiong Ng. Intelligent fault diagnosis under varying working conditions based on domain adaptive convolutional neural networks. *IEEE Access*, 6:66367–66384, 2018. 26, 32

[76] W. Qian, S. Li, and J. Wang. A new transfer learning method and its application on rotating machine fault diagnosis under variant working conditions. *IEEE Access*, 6:69907–69917, 2018. 26, 32

[77] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 26, 81, 107

[78] Liang Guo, Yaguo Lei, Saibo Xing, Tao Yan, and Naipeng Li. Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabeled data. *IEEE Transactions on Industrial Electronics*, 66(9):7316–7325, 2018. 27, 32

[79] Zhuyun Chen, Guolin He, Jipu Li, Yixiao Liao, Konstantinos Gryllias, and Weihua Li. Domain adversarial transfer network for cross-domain fault diagnosis of rotary machinery. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8702–8712, 2020. 27

[80] Cheng Cheng, Beitong Zhou, Guijun Ma, Dongrui Wu, and Ye Yuan. Wasserstein distance based deep adversarial transfer learning for intelligent fault diagnosis with unlabeled or insufficient labeled data. *Neurocomputing*, 409: 35–45, 2020. 27, 32

[81] Jinyang Jiao, Ming Zhao, and Jing Lin. Unsupervised Adversarial Adaptation Network for Intelligent Fault Diagnosis. *IEEE Transactions on Industrial Electronics*, 0046(c):1–1, 2019. ISSN 0278-0046. doi: 10.1109/tie.2019.2956366. 27

[82] Yan Song, Yibin Li, Lei Jia, and Meikang Qiu. Retraining Strategy based Domain Adaption Network for Intelligent Fault Diagnosis. *IEEE Transactions on Industrial Informatics*, 3203(c):1–1, 2019. ISSN 1551-3203. doi: 10.1109/tii.2019.2950667. 27, 32

[83] Xiang Li, Wei Zhang, and Qian Ding. Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks. *IEEE Transactions on Industrial Electronics*, 66(7):5525–5534, 2018. 27, 32, 81, 101

[84] Long Wen, Liang Gao, and Xinyu Li. A new deep transfer learning based on sparse auto-encoder for fault diagnosis. *IEEE Transactions on systems, man, and cybernetics: systems*, 49(1):136–144, 2017. 27

[85] Shan Pang and Xinyi Yang. A cross-domain stacked denoising autoencoders for rotating machinery fault diagnosis under different working conditions. *IEEE Access*, 7:77277–77292, 2019. 27, 32

[86] Xiang Li, Wei Zhang, Hui Ma, Zhong Luo, and Xu Li. Domain generalization in rotating machinery fault diagnostics using deep neural networks. *Neurocomputing*, 403:409–420, 2020. ISSN 18728286. doi: 10.1016/j.neucom.2020.05.014. 28, 32, 123, 132

[87] Yixiao Liao, Ruyi Huang, Jipu Li, Zhuyun Chen, and Weihua Li. Deep Semi-supervised Domain Generalization Network for Rotary Machinery Fault Diagnosis under Variable Speed. *IEEE Transactions on Instrumentation and Measurement*, 9456(c):1–1, 2020. ISSN 0018-9456. doi: 10.1109/tim.2020.2992829. 28, 123

[88] Wenyu Zhang, Mohamed Ragab, and Ramon Sagarna. Robust domain-free domain generalization with class-aware alignment. *arXiv preprint arXiv:2102.08897*, 2021. 28

[89] Te Han, Yan-Fu Li, and Min Qian. A hybrid generalization network for intelligent fault diagnosis of rotating machinery under unseen working conditions. *IEEE Transactions on Instrumentation and Measurement*, 2021. 28, 32, 123, 132

[90] JZ Sikorska, Melinda Hodkiewicz, and Lin Ma. Prognostic modelling options for remaining useful life estimation by industry. *Mechanical systems and signal processing*, 25(5):1803–1836, 2011. 29

[91] Gaige Chen, Jinglong Chen, Yanyang Zi, and Huihui Miao. Hyper-parameter optimization based nonlinear multistate deterioration modeling for deterioration level assessment and remaining useful life prognostics. *Reliability Engineering & System Safety*, 167:517–526, 2017. 29

[92] Jaouher Ben Ali, Brigitte Chebel-Morello, Lotfi Saidi, Simon Malinowski, and Farhat Fnaiech. Accurate bearing remaining useful life prediction based on weibull distribution and artificial neural network. *Mechanical Systems and Signal Processing*, 56:150–172, 2015. 29

[93] Chuang Sun, Zhousuo Zhang, Xue Luo, Ting Guo, Jinxiu Qu, and Bing Li. Support vector machine-based grassmann manifold distance for health monitoring of viscoelastic sandwich structure with material ageing. *Journal of Sound and Vibration*, 368:249–263, 2016. 29

[94] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018. 29

[95] Boyuan Yang, Ruonan Liu, and Enrico Zio. Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, 66(12):9521–9530, 2019. 29

[96] Jun Zhu, Nan Chen, and Weiwen Peng. Estimation of bearing remaining useful life based on multiscale convolutional neural network. *IEEE Transactions on Industrial Electronics*, 66(4):3208–3216, 2018. 29, 32

[97] Dingcheng Zhang, Edward Stewart, Jiaqi Ye, Mani Entezami, and Clive Roberts. Roller bearing degradation assessment based on a deep mlp convolution neural network considering outlier regions. *IEEE Transactions on Instrumentation and Measurement*, pages 1–1, 2019. 29, 32

[98] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 30, 32

[99] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 88–95, 2017. 30, 32, 37, 51, 52

[100] Cheng-Geng Huang, Hong-Zhong Huang, and Yan-Feng Li. A bidirectional lstm prognostics method under multiple operational conditions. *IEEE Transactions on Industrial Electronics*, 66(11):8792–8802, 2019. 30, 37, 47, 48, 51, 52, 71

[101] Huihui Miao, Bing Li, Chuang Sun, and Jie Liu. Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks. *IEEE Transactions on Industrial Informatics*, 15(9):5023–5032, 2019. 30, 32

[102] Ali Al-Dulaimi, Soheil Zabihi, Amir Asif, and Arash Mohammadi. A multimodal and hybrid deep neural network model for remaining useful life estimation. *Computers in Industry*, 108:186–196, 2019. 30, 51, 52

[103] Hui Liu, Zhenyu Liu, Weiqiang Jia, and Xianke Lin. A novel deep learning-based encoder-decoder model for remaining useful life prediction. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019. 30, 32

[104] Min Xia, Teng Li, Tongxin Shu, Jiafu Wan, Clarence W. de Silva, and Zhongren Wang. A two-stage approach for the remaining useful life prediction of bearings using deep neural networks. *IEEE Transactions on Industrial Informatics*, 15(6):3703–3711, 2019. 30, 32

[105] Meng Ma, Chuang Sun, and Xuefeng Chen. Deep coupling autoencoder for fault diagnosis with multimodal sensory data. *IEEE Transactions on Industrial Informatics*, 14(3):1137–1145, 2018. 30

[106] Pankaj Malhotra, Vishnu Tv, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder. *arXiv preprint arXiv:1608.06154*, 2016. 30

[107] Wennian Yu, Ii Yong Kim, and Chris Mechefske. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. *Mechanical Systems and Signal Processing*, 129:764–780, 2019. 30, 32, 51, 52

[108] Chong Zhang, Pin Lim, A. K. Qin, and Kay Chen Tan. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Transactions on Neural Networks*, 28(10):2306–2318, 2017. 30

[109] Ansi Zhang, Honglei Wang, Shaobo Li, Yuxin Cui, Zhonghao Liu, Guanci Yang, and Jianjun Hu. Transfer learning with deep recurrent neural networks for remaining useful life estimation. *Applied Sciences*, 8(12):2416, 2018. 31

[110] Paulo Roberto de Oliveira da Costa, Alp Akçay, Yingqian Zhang, and Uzay Kaymak. Remaining useful lifetime prediction via deep domain adaptation. *Reliability Engineering & System Safety*, 195:106682, 2020. 31

[111] Wei Zhang, Xiang Li, Hui Ma, Zhong Luo, and Xu Li. Transfer learning using deep representation regularization in remaining useful life prediction across operating conditions. *Reliability Engineering & System Safety*, 211: 107556, 2021. 31

[112] Shahin Siahpour, Xiang Li, and Jay Lee. A novel transfer learning approach in remaining useful life prediction for incomplete dataset. *IEEE Transactions on Instrumentation and Measurement*, 2022. 31

[113] Chen Lu, Zhenya Wang, and Bo Zhou. Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification. *Advanced Engineering Informatics*, 32:139–151, 2017. 32

[114] M. Ma, C. Sun, and X. Chen. Deep coupling autoencoder for fault diagnosis with multimodal sensory data. *IEEE Transactions on Industrial Informatics*, 14(3):1137–1145, March 2018. ISSN 1941-0050. doi: 10.1109/TII.2018.2793246. 32

[115] Z. Chen and W. Li. Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network. *IEEE Transactions on Instrumentation and Measurement*, 66(7):1693–1702, July 2017. ISSN 1557-9662. doi: 10.1109/TIM.2017.2669947. 32

[116] Yuan Liao, Linxuan Zhang, and Chongdang Liu. Uncertainty prediction of remaining useful life using long short-term memory network based on bootstrap method. In *2018 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–8, 2018. 32, 51, 52

[117] Biao Wang, Yaguo Lei, Tao Yan, Naipeng Li, and Liang Guo. Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery. *Neurocomputing*, 379:117–129, 2020. ISSN 0925-2312. 32

[118] Yan Song, Yibin Li, Lei Jia, and Meikang Qiu. Retraining strategy based domain adaption network for intelligent fault diagnosis. *IEEE Transactions on Industrial Informatics*, pages 1–1, 2019. 32, 101

[119] JZ Sikorska, Melinda Hodkiewicz, and Lin Ma. Prognostic modelling options for remaining useful life estimation by industry. *Mechanical systems and signal processing*, 25(5):1803–1836, 2011. 35

[120] Xiang Li, Qian Ding, and Jian-Qiao Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018. 37, 51, 52

[121] Boyuan Yang, Ruonan Liu, and Enrico Zio. Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, 66(12):9521–9530, 2019. 37

[122] Arnaz Malhi, Ruqiang Yan, and Robert X Gao. Prognosis of defect propagation based on recurrent neural networks. *IEEE Transactions on Instrumentation and Measurement*, 60(3):703–711, 2011. 37

[123] Zhenghua Chen, Min Wu, Rui Zhao, Feri Guretno, Ruqiang Yan, and Xiaoli Li. Machine remaining useful life prediction via an attention based deep learning approach. *IEEE Transactions on Industrial Electronics*, 2020.

[124] Huihui Miao, Bing Li, Chuang Sun, and Jie Liu. Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks. *IEEE Transactions on Industrial Informatics*, 15(9):5023–5032, 2019.

[125] Ji-Yan Wu, Min Wu, Zhenghua Chen, Xiao-Li Li, and Ruqiang Yan. Degradation-aware remaining useful life prediction with lstm autoencoder. *IEEE Transactions on Instrumentation and Measurement*, 70:1–10, 2021. 37

[126] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014. 37

[127] Trieu Trinh, Andrew Dai, Thang Luong, and Quoc Le. Learning longer-term dependencies in rnns with auxiliary losses. In *ICML 2018: Thirty-fifth International Conference on Machine Learning*, pages 4965–4974, 2018. 38

[128] Lei Le, Andrew Patterson, and Martha White. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. *Advances in neural information processing systems*, 31:107–117, 2018. 38

[129] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International Conference on Database Systems for Advanced Applications*, pages 214–228, 2016. 47, 51

[130] Chong Zhang, Pin Lim, A Kai Qin, and Kay Chen Tan. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE transactions on neural networks and learning systems*, 28(10):2306–2318, 2016. 47, 51, 52

[131] Zhenyu Wu, Shuyang Yu, Xinning Zhu, Yang Ji, and Michael Pecht. A weighted deep domain adaptation method for industrial fault prognostics according to prior distribution of complex working conditions. *IEEE Access*, 7:139802–139814, 2019. 51, 52

[132] Hui Liu, Zhenyu Liu, Weiqiang Jia, and Xianke Lin. A novel deep learning-based encoder-decoder model for remaining useful life prediction. In *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019. 51, 52

[133] W. Mao, J. He, and M. J. Zuo. Predicting remaining useful life of rolling bearings based on deep feature representation and transfer learning. *IEEE Transactions on Instrumentation and Measurement*, 69(4):1594–1608, April 2020. ISSN 1557-9662. doi: 10.1109/TIM.2019.2917735. 59

[134] Paulo Roberto de Oliveira da Costa, Alp Akçay, Yingqian Zhang, and Uzay Kaymak. Remaining useful lifetime prediction via deep domain adaptation. *Reliability Engineering & System Safety*, 195:106682, 2020. 59, 61, 71, 72, 73

[135] Liang Guo, Yaguo Lei, Saibo Xing, Tao Yan, and Naipeng Li. Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabeled data. *IEEE Transactions on Industrial Electronics*, 66(9):7316–7325, 2019. 59

[136] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018. 60

[137] Meng Ma, Chuang Sun, and Xuefeng Chen. Deep coupling autoencoder for fault diagnosis with multimodal sensory data. *IEEE Transactions on Industrial Informatics*, 14(3):1137–1145, 2018. 60

[138] Ruqiang Yan, Fei Shen, Chuang Sun, and Xuefeng Chen. Knowledge transfer for rotary machine fault diagnosis. *IEEE Sensors Journal*, 2019. 60

[139] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *ICLR 2018 : International Conference on Learning Representations 2018*, 2018. 60

[140] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoff Gordon. On learning invariant representation for domain adaptation. In *ICML 2019 : Thirty-sixth International Conference on Machine Learning*, pages 7523–7532, 2019. 60

[141] Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020. 60, 67

[142] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 63

[143] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 67, 68

[144] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer, 2017. 71, 72

[145] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 72

[146] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 72, 112

[147] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 72, 101

[148] Yan Song, Yibin Li, Lei Jia, and Meikang Qiu. Retraining Strategy based Domain Adaption Network for Intelligent Fault Diagnosis. *IEEE Transactions on Industrial Informatics*, 3203(c):1–1, 2019. ISSN 1551-3203. doi: 10.1109/tii.2019.2950667. 81

[149] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 82, 85, 130, 153, 154

[150] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 82

[151] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, volume 30, pages 1195–1204, 2017. 88

[152] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter 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. In *Proceedings of the European conference of the prognostics and health management society*, pages 05–08, 2016. 92

[153] Zhiyu Zhu, Gaoliang Peng, Yuanhang Chen, and Huijun Gao. A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis. *Neurocomputing*, 323:62–75, 2019. 92, 117

[154] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter 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. In *Proceedings of the European conference of the prognostics and health management society*, pages 05–08, 2016. 93

[155] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, pages 443–450. Springer, 2016. 95, 96

[156] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015. 95, 96

[157] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-second AAAI conference on artificial intelligence*, 2018. 95, 96

[158] Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotlagh, and Sridha Sridharan. On minimum discrepancy estimation for deep domain adaptation. *Domain Adaptation for Visual Understanding*, pages 81–94, 2020. 95, 96

[159] Chao Chen, Zhihang Fu, Zhihong Chen, Sheng Jin, Zhaowei Cheng, Xinyu Jin, and Xian-Sheng Hua. Homm: Higher-order moment matching for unsupervised domain adaptation. *AAAI*, 34(4):3422–3429, 2020. 95, 96

[160] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In *NIPS*, volume 31, pages 1640–1650, 2018. 95, 96

[161] Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *International Conference on Learning Representations*, 2018. 95, 96

[162] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 94

[163] Emadeldeen Eldele, Zhenghua Chen, Chengyu Liu, Min Wu, Chee-Keong Kwoh, Xiaoli Li, and Cuntai Guan. An attention-based deep learning approach for sleep stage classification with single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:809–818, 2021. 94

[164] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2352–2359, 2021. 96

[165] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):1–35, 2016. 96

[166] Y. Lei S. Xing T. Yan L. Guo and N. Li. Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabeled data. *IEEE Transactions on Industrial Electronics.*, 2018. 101

[167] Xiang Li, Wei Zhang, Qian Ding, and Xu Li. Diagnosing rotating machines with weakly supervised data using deep transfer learning. *IEEE Transactions on Industrial Informatics*, 16(3):1688–1697, 2020.

[168] Chao Chen, Fei Shen, Jiawen Xu, and Ruqiang Yan. Probabilistic latent semantic analysis based gear fault diagnosis under variable working conditions. *IEEE Transactions on Instrumentation and Measurement*, 2019. 101

[169] Ya Li, Xinmei Tian, Tongliang Liu, and Dacheng Tao. On better exploring and exploiting task relationships in multitask learning: Joint model and feature learning. *IEEE Transactions on Neural Networks*, 29(5):1975–1985, 2018. 104

[170] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010. ISSN 1041-4347. 104

[171] Wade A Smith and Robert B Randall. Rolling element bearing diagnostics using the case western reserve university data: A benchmark study. *Mechanical Systems and Signal Processing*, 64:100–131, 2015. 110

[172] Wei Zhang, Gaoliang Peng, Chuanhao Li, Yuanhang Chen, and Zhujun Zhang. A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals. *Sensors*, 17(2):425, 2017. 111

[173] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010. 112

[174] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200–2207, 2013. 112

[175] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer, 2017. 112, 132

[176] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2272–2281, 2017. 112, 132

[177] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*, pages 443–450, 2016. 112

[178] Feng Jia, Yaguo Lei, Jing Lin, Xin Zhou, and Na Lu. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, 72:303–315, 2016. 113

[179] Wei Zhang, Gaoliang Peng, Chuanhao Li, Yuanhang Chen, and Zhujun Zhang. A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals. *Sensors*, 17(2):425–425, 2017. 114, 117

[180] Wei Zhang, Chuanhao Li, Gaoliang Peng, Yuanhang Chen, and Zhujun Zhang. A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load. *Mechanical Systems and Signal Processing*, 100:439–453, 2018. 114

[181] Yuanhang Chen, Gaoliang Peng, Chaohao Xie, Wei Zhang, Chuanhao Li, and Shaohui Liu. Acdin: Bridging the gap between artificial and real bearing damages for bearing fault diagnosis. *Neurocomputing*, 294:61–71, 2018. 117

[182] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 117

[183] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017. 117

[184] Fei Shen, Reza Langari, and Ruqiang Yan. Transfer between multiple machine plants: A modified fast self-organizing feature map and two-order selective ensemble based fault diagnosis strategy. *Measurement*, 151:107155, 2020. 118

[185] Huailiang Zheng, Rixin Wang, Yuantao Yang, Yuqing Li, and Minqiang Xu. Intelligent Fault Identification Based on Multisource Domain Generalization

Towards Actual Diagnosis Scenario. *IEEE Transactions on Industrial Electronics*, 67(2):1293–1304, 2020. ISSN 15579948. doi: 10.1109/TIE.2019.2898619. 123

[186] Xiaolei Yu, Zhibin Zhao, Xingwu Zhang, Chuang Sun, Baogui Gong, Ruqiang Yan, and Xuefeng Chen. Conditional adversarial domain adaptation with discrimination embedding for locomotive fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*, 70:1–12, 2021. 124

[187] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 124

[188] Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pages 5628–5637. PMLR, 2019. 129

[189] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*, 2015. 130

[190] Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999. 132

[191] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639, 2018. 132

[192] Lang Huang, Chao Zhang, and Hongyang Zhang. Self-adaptive training: beyond empirical risk minimization. *Advances in Neural Information Processing Systems*, 33, 2020. 132

[193] Daehee Kim, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. *Proceedings of the IEEE international conference on computer vision [Accepted]*, 2021. 132

[194] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021. 133, 134

[195] Leland McInnes, John Healy, and James Melville. Umap: uniform manifold approximation and projection for dimension reduction. 2020. 140

[196] Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Universal domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2720–2729, 2019. 146

[197] Yong Cheng, Yang Liu, Tianjian Chen, and Qiang Yang. Federated learning for privacy-preserving ai. *Communications of the ACM*, 63(12):33–36, 2020. 147

[198] Olga Fink, Torbjørn Netland, and Stefan Feuerriegelc. Artificial intelligence across company borders. *Communications of the ACM*, 65(1):34–36, 2021. 147

[199] Wei Zhang and Xiang Li. Data privacy preserving federated transfer learning in machinery fault diagnostics using prior distributions. *Structural Health Monitoring*, page 14759217211029201, 2021. 147

[200] Hao Wang and Dit-Yan Yeung. Towards bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3395–3408, 2016. 148

[201] Luca Longo, Randy Goebel, Freddy Lecue, Peter Kieseberg, and Andreas Holzinger. Explainable artificial intelligence: Concepts, applications, research challenges and visions. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 1–16. Springer, 2020. 148

[202] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. The dangers of post-hoc interpretability: Unjustified counterfactual explanations. *arXiv preprint arXiv:1907.09294*, 2019. 148

[203] Or Biran and Courtenay Cotton. Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI)*, volume 8, pages 8–13, 2017. 148

[204] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019. 151

[205] Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940. 151

[206] Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer, 1992. 151

[207] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020. 153