Data-driven Fault Detection and Diagnosis Frameworks for HVAC Systems using Probabilistic and Deep Generative Models

Viet Tra

A Thesis
in
The Concordia Institute
for
Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Information and Systems Engineering) at
Concordia University
Montréal, Québec, Canada

October 2024

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Viet Tra**

Entitled: **Data-driven Fault Detection and Diagnosis Frameworks for HVAC Systems using Probabilistic and Deep Generative Models**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality. Signed by the Final Examining Committee:

______________________ Chair
Dr. Ghazanfarah Hafeez

______________________ External Examiner
Dr. Carl Chalmers

______________________ Examiner
Dr. Zachary Patterson

______________________ Examiner
Dr. Arash Mohammadi

______________________ Examiner
Dr. Bruno Lee

______________________ Supervisor
Dr. Nizar Bouguila

______________________ Supervisor
Dr. Manar Amayri

Approved by ______________________
Dr. Farnoosh Naderkhani, GPD

__________ 2024 ______________________
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Data-driven Fault Detection and Diagnosis Frameworks for HVAC Systems using Probabilistic and Deep Generative Models

Viet Tra, Ph.D.
Concordia University, 2024

The thesis tackles major challenges in developing fault detection and diagnosis (FDD) models for heating, ventilation, and air conditioning (HVAC) systems. HVAC systems are crucial for ensuring thermal and air comfort in buildings, but they frequently face inefficiencies due to inadequate maintenance, component wear and tear, and control issues, leading to considerable energy waste. The research highlights several key challenges, such as data contamination, lack of labeled data, and high dimensionality, which limit the effectiveness of traditional FDD methods. To overcome these challenges, the thesis introduces innovative approaches: a supervised multiclass version of the deep autoencoding Gaussian mixture model (DAGMM) that improves outlier detection by leveraging label information; a neural network-based approach for mixtures of probabilistic principal component analyzers (NN-MPPCA) with a robust loss function for enhanced anomaly detection; a new framework combining variational autoencoders (VAE) with NN-MPPCA to handle high-dimensional data and incomplete datasets; and the adaptive adversarial autoencoder (AdaAAE), which refines anomaly detection with deep support vector data description (DSVDD). Comprehensive validation against cutting-edge algorithms highlights the superior performance of the proposed methods, leading to more efficient and reliable HVAC systems.

Acknowledgments

I would like to express my deepest gratitude to my main supervisor, Dr. Nizar Bouguila, and my co-supervisor, Dr. Manar Amayri, for their invaluable support, guidance, and encouragement throughout my PhD program. Their academic advising and generous funding have been instrumental in the completion of this thesis. Without their mentorship, this work would not have been possible.

Dr. Nizar Bouguila has consistently provided understanding and support, especially during the difficult periods when I had to balance my PhD work with family responsibilities while my son was an infant. His empathy and thoughtful guidance have been invaluable, helping me to focus on my research with confidence and easing my journey significantly.

I am profoundly grateful to my wife, Nhi Nhi, for her unwavering support, both financially and emotionally, during this journey. Her patience and understanding have been a constant source of strength and motivation.

Lastly, I want to thank my son, Henry, for being the greatest source of happiness and inspiration in our family. His presence is a constant reminder of why this work is important and motivates me to continue growing and striving for a brighter future.

To all those mentioned and many others who have supported me in various ways, I extend my heartfelt thanks.

Contribution of Authors

This thesis comprises chapters that encapsulate published papers, in which I am the first and corresponding author. My primary contributions across these papers include the conceptualization, design, execution of experiments, and drafting of the manuscripts. These works were conducted under the guidance and supervision of Dr. Nizar Bouguila and Dr. Manar Amayri, who served as my main and co-supervisor, respectively.

Dr. Nizar Bouguila provided invaluable expertise and direction, assisting in idea generation through the provision of key references and offering detailed feedback on proposed methods. His comments on drafts and revisions were essential in ensuring that the manuscripts met the highest standards of quality.

Dr. Manar Amayri contributed valuable insights throughout the drafting and revision process, helping to align the research with practical applications and suggesting ideas to enhance the real-world relevance of the findings. Her feedback played an important role in shaping the research outcomes.

Each co-author's contributions were instrumental to the development of these works, which together form the foundation of this thesis.

Contents

List of Figures

List of Tables

List of Acronyms

AAE	Adversarial Autoencoder
AdaAAE	Adaptive Adversarial Autoencoder
AdaBoost	Adaptive Boosting
AD	Anomaly Detection
AE	Autoencoder
AF	Actuator Faults
AHU	Air Handling Unit
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
CF	Controller Faults
DAGMM	Deep Autoencoding Gaussian Mixture Model
DAE	Deep Autoencoder
DNN	Deep Neural Networks
DSVDD	Deep Support Vector Data Description
DV-MPPCA	Deep Variational Mixture of Probabilistic Principal Component Analyzers
ELE	Elliptical Envelope
ELBO	Evidence Lower Bound

EM	Expectation-Maximization
EF	Equipment Faults
ERS	Energy Resource Station
FDD	Fault Detection and Diagnosis
GAN	Generative Adversarial Network
GMM	Gaussian Mixture Model
HVAC	Heating, Ventilation, and Air Conditioning
IF	Isolation Forest
KDE	Kernel Density Estimation
KNN	K-Nearest Neighbors
LR	Logistic Regression
MLN	Multilayer Neural Network
MLP	Multi-Layer Perceptron
MPPCA	Mixtures of Probabilistic Principal Component Analyzers
NF	Normalization Flows
NN-MPPCA	Neural Network-Based Mixtures of Probabilistic Principal Component Analyzers
OC-SVM	One-Class Support Vector Machine
PCA	Principal Component Analysis
RF	Random Forest
RP	Research Project
SAE	Stacked Autoencoder

SL	Severity Level
SMOTE	Synthetic Minority Over-Sampling Technique
S-DAGMM	Supervised Multiclass Deep Autoencoding Gaussian Mixture Model
SVDD	Support Vector Data Description
SVM	Support Vector Machine
THO	Temperature Humidity Output
TSO	Temperature Sensor Output
VAE	Variational Autoencoder

Chapter 1

Introduction

In this chapter, we provide a comprehensive overview of the motivations, challenges, potential approaches, and key contributions of the research. Section 1.1 discusses the motivations behind the study, highlighting the importance of optimizing energy consumption and improving comfort in HVAC systems. Section 1.2 addresses the challenges faced in developing fault detection and diagnosis models, focusing on issues such as data contamination, scarcity of labeled data, and high dimensionality. Section 1.3 presents potential approaches and main contributions, detailing the innovative methods proposed to overcome these challenges. Finally, Section 1.4 outlines the structure of the thesis, guiding the reader through the subsequent chapters.

1.1 Overview and Motivations

Nowadays, heating, ventilation, and air-conditioning (HVAC) systems are widespread in both commercial and residential buildings. They play a crucial role in maintaining thermal and air comfort within these structures. In order to ensure optimal thermal conditions and fresh air, HVAC systems require a significant amount of energy. Statistics reveal that HVAC systems contribute to 57% of energy consumption in U.S. buildings and over 40% of the global total B. Li, Cheng, Zhang, Cui, Cai (2021); Yan, Chong, Mo (2020). Despite their essential function, a considerable portion of this energy is wasted due to inadequate maintenance, component deterioration, and control failures. Without timely intervention, minor issues in components can escalate, ultimately leading to system breakdown. This not only results in energy wastage but also causes discomfort in commercial buildings. While scheduled maintenance can mitigate catastrophic failures and extend the system's operational reliability, the associated costs, including tools and expert services, are high. Moreover, unexpected failures may occur during maintenance periods. To prevent the adverse consequences of unexpected disruptions, it is imperative to detect early signs of faults in building HVAC systems using accurate fault detection and diagnosis (FDD) techniques Tra, Amayri, Bouguila (2022a, 2022b); Yan, Huang, Shen, Ji (2020); Yan, Su, Huang, Mo (2020).

The substantial energy consumption by HVAC systems presents both a challenge and an opportunity for improvement. Faults in HVAC components can lead to significant energy losses and reduced comfort levels. These issues are exacerbated by underqualified maintenance, component degradation, and installation failures. Consequently, there is a pressing need for efficient energy management and accurate fault detection and diagnosis (FDD) to mitigate energy wastage and discomfort caused by system failures. The emergence of machine learning and the availability of extensive historical data have positioned data-driven approaches as the most promising solution for FDD in HVAC systems. These approaches offer flexibility and adaptability, making them suitable for complex, nonlinear systems without requiring extensive expert knowledge or detailed mathematical models. By leveraging machine learning techniques, data-driven FDD methods can detect and diagnose faults early, minimizing their impact on energy usage and building comfort.

1.2 Challenges and Objectives

Despite the potential of data-driven FDD methods, several challenges remain, which this research aims to address through specific objectives:

(1) **Data Contamination:** The development of fault detection and diagnosis (FDD) models faces challenges from data contamination caused by outliers, which are observations that deviate significantly from the majority of the dataset. In HVAC systems, outliers can occur due to recording errors, noise, faulty measurements, or missing values. These outliers adversely affect the reliability and robustness of FDD models. As a result, it is essential to identify and remove outliers to maintain data integrity before constructing dependable machine learning models. Additionally, it is important to create robust learning models that can withstand datasets with missing values.

(2) **Scarcity of Labeled Data:** Acquiring labeled fault data is often difficult, expensive, and time-consuming, as it typically requires expert intervention to identify, categorize, and label fault conditions. In many cases, faults are rare events, leading to an inherent imbalance in the dataset, where normal operating conditions are well-represented, but fault data is scarce. This scarcity makes it challenging to train accurate and robust machine learning models, as these models typically require substantial labeled data to learn patterns effectively and make reliable predictions. To address this challenge, researchers often explore unsupervised learning techniques and pretraining methods, such as autoencoders, to leverage the available unlabeled data. However, these approaches may introduce their own set of challenges and complexities, requiring careful consideration in their application.

(3) **High Dimensionality:** The challenge of high dimensionality in data is a significant obstacle in constructing effective fault detection and diagnosis (FDD) models. Traditional shallow FDD methods typically perform well with low-dimensional data but struggle with high-dimensional data due to the curse of dimensionality. This curse manifests in several ways, including increased data sparsity, which makes instances more likely to be perceived as rare events, thus raising the false alarm rate. High-dimensional data also poses computational challenges, as it demands more resources and increases the risk of overfitting to noise rather than capturing meaningful patterns. Generally, HVAC systems consist of numerous variables and parameters that must be monitored, resulting in high-dimensional data. To address these issues, there is a need for compression networks that can handle high-dimensional data by transforming it into a more manageable form.

By addressing these challenges and objectives, the research aims to contribute to the development of more efficient and reliable HVAC systems, ultimately leading to reduced energy consumption and enhanced comfort in buildings.

1.3 Potential Approaches and Main Contributions

Anomaly detection (AD) has gained significant attention recently due to its ability to address key challenges in this research. AD algorithms excel at identifying outliers that deviate from typical data patterns, with outliers located in low-probability regions while most data resides in high-probability areas. AD algorithms are also flexible for fault detection tasks, as faults, like anomalies, often show distinct patterns different from normal operations. By learning normal patterns from data collected from well-functioning devices, these algorithms can identify and flag atypical states, such as unforeseen faults. Notably, anomaly detection algorithms operate in an unsupervised manner, allowing them to be trained using only unlabeled data collected via remote sensors installed on the device.

With the rapid growth of probabilistic and deep generative models, density estimation-based methods have become essential in anomaly detection applications Pang, Shen, Cao, Hengel (2021); Ruff et al. (2021). Traditional shallow probabilistic models, such as the Gaussian mixture model (GMM), kernel density estimation (KDE), and histogram estimator, are known for their simplicity and reliable performance. Meanwhile, deep generative models like the variational autoencoder (VAE), generative adversarial network (GAN), and normalization flows (NF) are highly effective, particularly for high-dimensional data. This project aims to leverage the strengths of both approaches by developing solutions that integrate representative models from each into a unified framework, ultimately achieving superior performance in anomaly detection tasks.

This research addresses significant challenges in developing fault detection and diagnosis (FDD) models for HVAC systems, particularly the issues raised in Section 1.2. The key contributions of this study are summarized as follows:

(1) We introduced a supervised multiclass version of the deep autoencoding Gaussian mixture model (DAGMM) that effectively detects outliers in multiclass datasets by leveraging label information from the training data. We also propose utilizing the encoding layers of DAGMM's compression network as a pre-trained network for deep neural networks (DNNs), improving the DNN's capability in handling classification tasks. Additionally, we conducted a comprehensive comparative study with state-of-the-art algorithms to validate the proposed methods, demonstrating superior performance.

(2) We introduced several key contributions to improve the training and effectiveness of mixtures of probabilistic principal component analyzers (MPPCA) models for anomaly detection. First, we developed a neural network-based approach, NN-MPPCA, which employs a multilayer neural network to model a mixture of probabilistic principal component analyzers, with parameter optimization achieved through back-propagation. Second, we designed a robust loss function for NN-MPPCA that incorporates a penalty component to address the singularity issues encountered during the inversion of covariance matrices, ensuring convergence to the optimal solution. Finally, we validated the proposed

method using data from a 90-ton centrifugal water-cooled chiller, demonstrating its superiority over existing state-of-the-art outlier detection algorithms through extensive experimental testing.

(3) We introduced a new framework, deep variational mixture of probabilistic principal component analyzers (DV-MPPCA), for effective anomaly detection. Key contributions include the integration of a neural network-based MPPCA (NN-MPPCA) with a variational autoencoder (VAE) to handle high-dimensional data. A modified evidence lower bound (ELBO) loss function enhances the framework's ability to work with incomplete datasets. The framework's total loss function ensures convergence to the global optimum by jointly training the VAE and NN-MPPCA components. Extensive case studies using data from the ASHRAE Research Project 1312 demonstrate the superior performance of DV-MPPCA compared to existing models in detecting faults in HVAC systems.

(4) We introduced the adaptive adversarial autoencoder (AdaAAE), a novel deep anomaly detection method that enhances the adversarial autoencoder (AAE) framework with deep support vector data description (DSVDD) to improve the compactness of the latent code distribution. The training process of AdaAAE is meticulously crafted to ensure that all components converge simultaneously, utilizing an iterative procedure to update the prior distribution. The approach also includes a novel method for accurately estimating the anomaly threshold without prior knowledge of the anomaly ratio in test samples, by calculating the radius of the latent distribution at the end of training. Experiments using various seasonal datasets from an air handling unit system, developed under the ASHRAE Research Project 1312 (RP-1312), demonstrate the framework's effectiveness and superiority over existing state-of-the-art models.

1.4 Thesis Structure

This section presents a concise overview of the content in each chapter of the thesis:

Chapter 1 introduces the thesis by outlining the motivations, objectives, and challenges associated with fault detection and diagnosis in HVAC systems. It also highlights the key contributions and significance of the research.

Chapter 2 covers the literature review and describes the experimental datasets used throughout the research, including the ASHRAE RP-1043 chiller dataset and the ASHRAE RP-1312 air handling unit dataset.

Chapter 3 details the development of the supervised multiclass deep autoencoding Gaussian mixture model (DAGMM) for detecting outliers in building chiller datasets, and presents the experimental results demonstrating its effectiveness.

Chapter 4 discusses the unsupervised outlier detection approach using neural network-based mixtures of probabilistic principal component analyzers (NN-MPPCA) for building chiller fault diagnosis, focusing on model development and validation.

Chapter 5 introduces the deep variational mixture of probabilistic principal component analyzers (DV-MPPCA) framework, emphasizing its application in unsupervised fault detection for air handling unit systems.

Chapter 6 presents the adaptive adversarial autoencoder (AdaAAE) for unsupervised AHU fault detection, highlighting advancements in latent code description.

Chapter 7 concludes the thesis by summarizing the key findings and contributions, and outlines potential future research directions to further enhance the robustness and effectiveness of anomaly detection models in real-world applications.

Chapter 2

Literature Review and Experimental Datasets

2.1 Overview of Fault Detection and Diagnosis

HVAC systems are responsible for a significant portion of energy consumption in buildings, and undetected faults can lead to increased energy usage, higher operational costs, and reduced system lifespan. Additionally, faults in HVAC systems can result in poor indoor air quality and discomfort for occupants. Effective FDD helps in identifying and rectifying issues at an early stage, minimizing downtime and maintenance costs, and enhancing overall system reliability and sustainability. FDD approaches can be roughly divided into three different categories: rule-based, model-based, and data-driven methods. The first family of approaches, rule-based Xiao, Zheng, Wang (2011), detects faults based on a set of rules established by an expert's knowledge and experience. The second category, model-based S. Li Wen (2014); X. Zhao, Yang, Li (2012), focuses on developing a mathematical model reliable enough to mirror the real physical process. This category of approaches requires high complexity and time for model establishment, especially for large-scale, nonlinear systems like HVAC. The final group of approaches, data-driven, is capable of working without constructing a complex model, but by learning fault patterns with machine learning techniques.

Compared to rule-based J. Liu et al. (2020); Xiao et al. (2011), and model-based FDD S. Li Wen (2014); X. Zhao et al. (2012) approaches, data-driven approaches have been preferred during last few decades for a number of reasons. First, data-driven B. Li et al. (2021); Mirnaghi Haghighat (2020); Tra et al. (2022a) approaches are simple to implement without the need of expert's knowledge as rule-based approaches. In addition, unlike model-based approaches, data-driven approaches do not require high complexity and time for model establishment. Data-driven approaches only need sufficient system historical data to generalize FDD tasks. Second, with the flourishing growth of machine learning and deep learning techniques, data-driven approaches have proven their flexibility to adapt for different FDD tasks, i.e., supervised FDD, unsupervised FDD, or semi-supervised FDD. For these reasons, data-driven FDD approaches are now the most favored not only for complex, nonlinear HVAC systems as air handling unit (AHU) H. Zhang, Li, Li, Zhang, Peng (2021); H. Zhang, Li, Wei, Zhang (2022) or chiller systems B. Li et al. (2021); Yan, Su, et al. (2020) but for other systems in general Tra, Duong, Kim (2019); Tra, Kim, Khan, Kim (2017).

2.2 Fundamentals of Anomaly Detection

Anomaly detection (AD) is a critical process in identifying unusual patterns or deviations from the norm within a dataset. These techniques are essential for detecting outliers, which are data points that differ significantly from the majority, often indicating errors or novel phenomena. In the context of fault detection, AD techniques can be particularly effective as they enable the identification of abnormal system behavior that may signify underlying faults. During last few decades, a numerous number of AD approaches have been proposed to resolve various AD applications. These approaches can be grouped into three main categories: (1) one-class classification, (2) reconstruction-based, (3) density estimation-based Ruff et al. (2021).

One-class classification or single-class classification El-Yaniv Nisenson (2006); Moya, Koch, Hostetler (1993) is a prominent technique that has been exhaustively exploited for AD approaches. The feature of this technique is to use a discriminative model to learn a decision boundary of the normal data distribution. The location of any unseen instance with respect to the decision boundary decides its anomalous status (i.e., an anomaly instance or a regular instance). Among one-classification methods, kernel-based OC-SVM Schölkopf, Platt, Shawe-Taylor, Smola, Williamson (2001) and SVDD Tax Duin (2004) are the most famous for AD applications.

Reconstruction-based methods are among the earliest and most common neural network-based AD approaches Aggelis, Mpalaskas, Matikas (2013). Their objective is to optimize a model to well-reconstruct normal data instances. As a result, an unseen instance that is poorly reconstructed by the learned model should be detected as an anomaly. Principal component analysis (PCA)-based Huang et al. (2006); Shyu, Chen, Sarinnapakorn, Chang (2003), and autoencoder (AE)-based Principi, Vesperini, Squartini, Piazza (2017) approaches are two notable examples of reconstruction-based AD methods. While PCA-based AD methods use a linear transformation matrix to encode and decode data, AE-based AD methods use neural networks as nonlinear encoder and nonlinear decoder.

Density estimation-based methodologies rely on probabilistic models Ruff et al. (2021); Xu et al. (2018); Yairi et al. (2017); Zong et al. (2018). In these approaches, the task of anomaly detection entails employing probabilistic models to accurately capture the characteristics of historical training data. Subsequently, the anomaly score for a test instance is derived by evaluating the negative log-likelihood of the instance with respect to the fitted probabilistic model.

In this thesis, the primary anomaly detection approach employed is the density estimation-based methodologies. This approach is highly effective for the type of data encountered in HVAC systems, where identifying low-probability regions is crucial for detecting anomalies. By modeling the data distribution using probabilistic models, we can assess the likelihood of each data point and classify those that fall within low-probability regions as anomalies. This probabilistic framework provides a robust method for detecting subtle deviations from normal operation, which is essential for accurate anomaly detection in complex systems like HVAC.

2.3 HVAC Validation Data

In this thesis, we employ two prominent experimental datasets to evaluate the performance of our proposed fault detection and diagnosis models: the chiller dataset from the ASHRAE RP-1043 project and the air handling unit (AHU) dataset from the ASHRAE RP-1312 project. These datasets are well-regarded in the field of building system performance and provide a comprehensive basis for testing FDD models in HVAC systems.

2.3.1 Typical Chiller Faults and Data Description

The ASHRAE RP-1043 chiller dataset consists of detailed operational data from a 90-ton centrifugal water-cooled chiller system, capturing various fault conditions and normal operating scenarios M. Comstock Braun (2002). This dataset is instrumental in validating the effectiveness of our models, as it encompasses a wide range of chiller operating conditions and fault types, enabling a robust assessment of model performance in detecting and diagnosing chiller faults.

The schematic of a typical, four-component, single-stage vapor compression chiller is shown in Fig. 2.1. The four main components of a typical chiller include (1) compressor, (2) condenser, (3) expansion valve, and (4) evaporator. Condenser operation at high pressure (high temperature) and evaporator operation at low pressure (low temperature) are enabled for heat rejection and absorption.

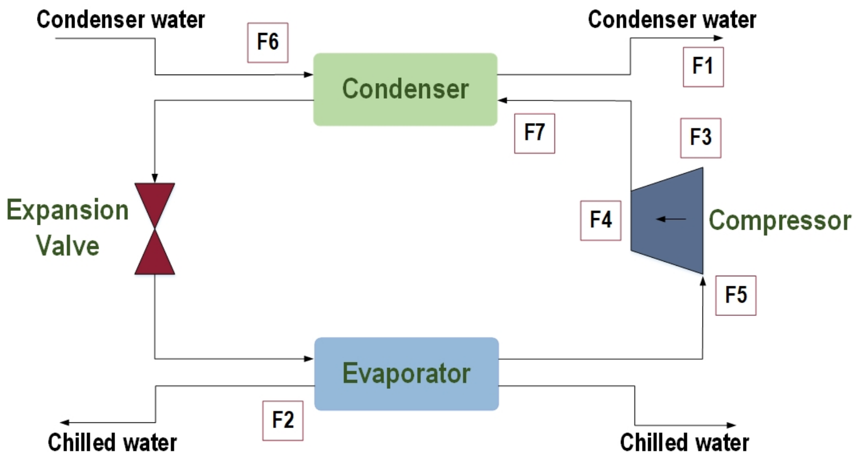

Figure 2.1: Internal structure of the 90-ton chiller.

A total of seven process fault types were introduced into the chiller of the RP-1043 project. Each fault type was recorded at four severity levels (ranging from the lowest SL1 to the highest SL4) defined based on the divergent degree of the chiller's important parameters from the normal. The details of seven fault types and severity levels are defined in Table 2.1. Typical faults were introduced into the 90-ton centrifugal water-cooled chiller as follows: condenser and evaporator's water flow rate was intentionally reduced to simulate reduced condenser water flow rate (ReduCF) and reduced evaporator water flow rate (ReduEF) faults; increasing refrigerant charge and reducing refrigerant charge to trigger refrigerant overcharge (RefOver) and refrigerant leakage (RefLeak) faults; excess oil (ExcsOil) fault was injected by charging more oil than normal; condensed fouling (ConFoul) fault and non-condensable in refrigerant (NonCon) fault were introduced by plugging tubes into the condenser, and by adding nitrogen to the refrigerant, respectively. While RefLeak, RefOver, and ExcsOil faults affect the chiller at the system level (i.e., system-level faults), the other faults including ReduCF, ReduEF, ConFoul, and NonCon are regarded as element-level faults since these faults symptomize a certain location of the chiller.

Table 2.1: Description of chiller fault types and severity levels.

Fault type	**Description**	**Severity Level 1**	**Severity Level 2**	**Severity Level 3**	**Severity Level 4**
F1	Reduced Condenser Water Flow	10% reduced in flow	20% reduced in flow	30% reduced in flow	40% reduced in flow
F2	Reduced Evaporator Water Flow	10% reduced in flow	20% reduced in flow	30% reduced in flow	40% reduced in flow
F3	Refrigerant Leak	10% reduced in charge	20% reduced in charge	30% reduced in charge	40% reduced in charge
F4	Refrigerant Overcharge	10% increase in charge	20% increase in charge	30% increase in charge	40% increase in charge
F5	Excess Oil	14% increase in charge	32% increase in charge	50% increase in charge	68% increase in charge
F6	Condenser Fouling	12% reduced in tubes	20% reduced in tubes	30% reduced in tubes	45% reduced in tubes
F7	Non-condensable in Refrigerant	1% by volume Nitrogen	2% by volume Nitrogen	3% by volume Nitrogen	5% by volume Nitrogen

The healthiness status of the RP-1043 chiller system is defined by a record of 65 attributes including in/out water temperature from the condenser, evaporator, oil feed, etc., that were recorded in every 10-second interval. For each fault at each severity level, an approximate number of 5191 samples were collected. For 8 different classes (i.e., normal and seven typical chiller faults) of each severity level, the total number of samples collected is 5191*8=41,528 (i.e., both steady samples and transient samples).

The health status of the RP-1043 chiller system is defined by a data instance consisting of 65 attributes, such as the in/out water temperature from the condenser, evaporator, and oil feed, which were recorded every 10 seconds. Approximately 5,191 instances were collected for each fault at each severity level. With 8 different classes (including normal and seven typical chiller faults) for each severity level, the total number of samples collected is 41,528 (comprising both steady and transient samples).

2.3.2 Typical AHU Faults and Data Description

AHU is the key system for maintaining comfortable and healthy air quality inside commercial buildings. The AHU dataset from the ASHRAE RP-1312 project includes extensive data on air handling units, which are critical components in HVAC systems. The dataset provides valuable insights into the behavior of AHUs under different operational conditions, including fault scenarios. By leveraging this dataset, we can thoroughly evaluate our models' ability to detect faults in AHU operations, ensuring their applicability and reliability in real-world settings.

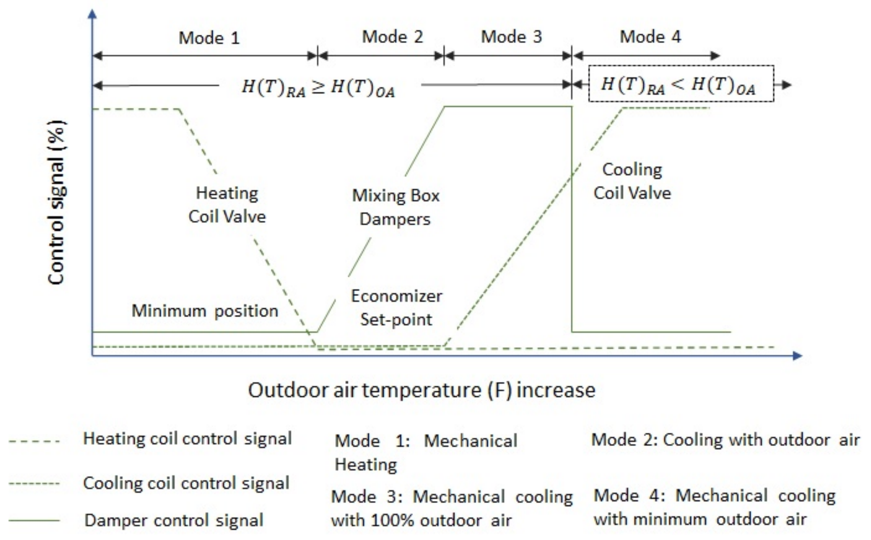
Mode 1
Mode 2
Mode 3
Mode 4
$H(T)_{RA} \geq H(T)_{OA}$
$H(T)_{RA} < H(T)_{OA}$
Control signal (%)
Heating Coil Valve
Mixing Box Dampers
Cooling Coil Valve
Economizer Set-point
Minimum position
Outdoor air temperature (F) increase
Heating coil control signal
Cooling coil control signal
Damper control signal
Mode 1: Mechanical Heating
Mode 2: Cooling with outdoor air
Mode 3: Mechanical cooling with 100% outdoor air
Mode 4: Mechanical cooling with minimum outdoor air

Figure 2.2: Four operating modes of an AHU system. An AHU system switches between these modes in accordance with the temperature and humidity of the seasonal outdoor air.

Depending on the temperature and humidity of the seasonal outdoor air, AHU operation switches between four different modes (Fig. 2.2). The first mode of AHU is the mechanical heating mode (Mode 1) of AHU triggered when the outdoor air temperature is low. To maintain the supply air temperature at the heating set point, the outdoor air damper is kept at its minimum position to prevent the ingress of outdoor cold air. The cooling coil valve is closed and the heating coil valve is appropriately controlled to regulate the supply air temperature. When the outdoor air temperature increases, AHU transits to the free cooling mode (Mode 2). In this mode, AHU closes the heating and cooling valves and controls the air damper to receive the outdoor air for regulating the supply air temperature around the cooling set point. If maximum outdoor air can not regulate the supply air to meet the cooling set point, AHU turns to the economizer cooling mode (Mode 3). In this mode, the cooling valve is modulated to regulate the supply air temperature. When the

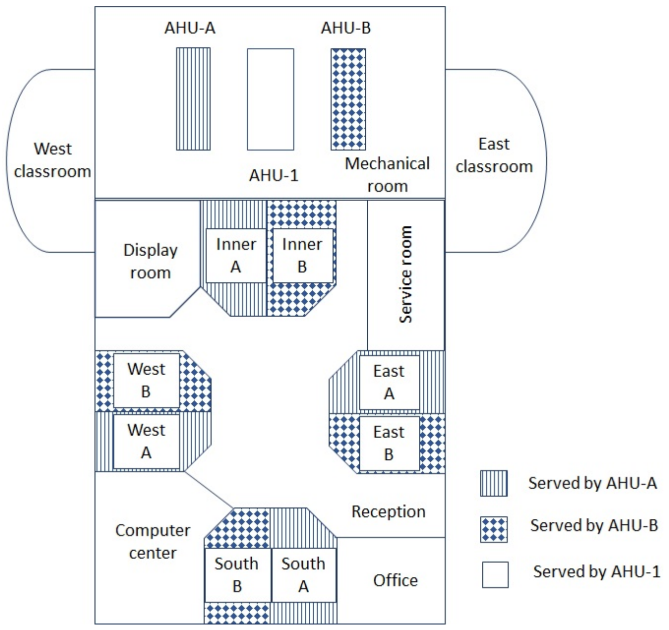

Figure 2.3: Layout of the test facility. AHU-A and AHU-B are identical, and each AHU serves four areas. The other areas are served by AHU-1.

outdoor air temperature is significantly high to meet the economizer set point, AHU turns to the mechanical cooling mode (Mode 4). At this stage, the outdoor air damper is kept at a minimum position to prevent the ingress of high-temperature outdoor air. The cooling valve is modulated to ensure the supply air temperature meets the cooling set point.

To study AHU behaviour and faults, ASHRAE's RP-1312 established a single-duct dual-fan variable air volume AHU system at the energy resource station (ERS) test facility, as detailed in Wen Li (2011). This setup included three AHUs, as seen in Fig. 2.3. AHU-1 served common areas, while AHU-A and AHU-B managed different zones. During the experiments, common component faults were introduced into AHU-A, and the data for these typical faults were recorded over 24 hours at one-minute intervals. Meanwhile, AHU-B operated under standard conditions. It's worth noting that AHU-A and AHU-B were identical in design, and the experiments were conducted under real-world conditions, with the AHU system operating from 6:00 AM to 6:00 PM and switched off from 6:00 PM to 6:00 AM. Seasonal variations significantly influence the occurrence of AHU faults. Depending on the time of year and prevailing environmental conditions, particular types of faults tend to manifest more frequently. In our investigation, we have discerned that during the Summer season, eight common AHU faults tend to manifest, while in Winter, there are typically six distinct faults observed. Furthermore, in Spring, eight typical faults are commonly encountered, please

Table 2.2: Common Faults in Summer Conditions

Category	Device	No.	Fault Description
EF	Duct	F1	AHU duct leak after supply fan
	Duct	F2	AHU duct leak before supply fan
AF	Damper	F3	OA damper stuck (fully close)
	Damper	F4	OA damper leak (55% open)
	Damper	F5	EA damper stuck (fully close)
	Valve	F6	Cooling coil valve stuck (fully open)
CF	Valve controller	F7	Cooling coil valve control unstable
	Valve controller	F8	Cooling coil valve reverse action
EF: equipment faults, AF: actuator faults, CF: controller faults.			

Table 2.3: Common Faults in Winter Conditions

Category	Device	No.	Fault Description
EF	Coil	F1	Heating coil fouling
	Coil	F2	Heating coil reduce capacity
AF	Damper	F3	OA damper stuck (fully close)
	Damper	F4	OA damper leak (62% open)
	Damper	F5	EA damper stuck (fully close)
	Valve	F6	Cooling coil valve stuck (partially open - 20%)
EF: equipment faults, AF: actuator faults.			

refer to Tables 2.2 - 2.4.

Table 2.4: Common Faults in Spring Conditions

Category	Device	No.	Fault Description
EF	Filter	F1	Air filter blockage (25%)
	Fan	F2	Return fan complete failure
AF	Damper	F3	OA damper stuck (fully closed)
	Damper	F4	EA damper stuck (fully closed)
	Valve	F5	Cooling coil valve stuck (fully closed)
CF	Damper controller	F6	Mixed air damper unstable
	Valve controller	F7	Sequence of heating and cooling unstable
	Fan controller	F8	Supply fan control unstable

EF: equipment faults, AF: actuator faults, CF: controller faults.

Chapter 3

Outlier Detection Via Multiclass Deep Autoencoding Gaussian Mixture Model for Building Chiller Diagnosis

3.1 Introduction

High energy consumption is a significant concern for heating, ventilation, and air conditioning (HVAC) systems in modern times Mirnaghi Haghighat (2020); Y. Zhao, Li, Zhang, Zhang (2019). Statistics reveal that HVAC systems account for approximately 33% of the total energy consumption in commercial buildings in the USA. In tropical countries with hot and humid climates, such as Singapore, this percentage can soar to 60% or more B. Li et al. (2021). Apart from essential usage, several contributing factors exacerbate this issue, including underqualified maintenance, underperforming components, and installation and control failures. Given that chillers constitute the largest share of HVAC system energy consumption in constructions, comprising 35-40% of total building energy consumption M. C. Comstock, Braun, Groll (2002), they present a prime opportunity for enhancing energy efficiency. Besides the obvious discomfort caused by chiller faults, the consequential energy wastage demands attention. Addressing this problem necessitates the implementation of accurate fault detection/diagnosis (FDD) techniques.

In general, reliable diagnosis is achievable with data-driven approaches, given the assumptions that normal and fault data are available at a large number and the data set is not contaminated by outliers.

However, HVAC systems in general, and chiller systems specifically, rely on numerous sensors to monitor parameters like temperature, humidity, pressure, etc. Over time, these sensors may encounter issues like noise, drift, or even malfunction, resulting in inaccurate readings, which makes the presence of outliers in the recorded data unavoidable. Additionally, human actions, like adjustments to setpoints or manual overrides, can introduce anomalies into the system. These outliers and anomalies are distinct from the majority of the data and negatively impact the reliability of the fault diagnosis model. As a result, identifying and removing these undesirable entities is essential before constructing a reliable machine learning model. Typically, an outlier or anomaly is considered an instance that falls within the low-probability regions of the data distribution, making them detectable by anomaly detection (AD) algorithms.

Despite the recent advancements in density estimation-based anomaly detection methods Su et al. (2019); Zong et al. (2018), creating a reliable anomaly detection system without human supervision remains a significant challenge, particularly when dealing with high-dimensional data. As the dimensionality of input data increases, the probability of an input sample being recognized as a rare event also grows. In commercial buildings, HVAC systems typically involve a large number of variables and parameters that need to be monitored, resulting in high-dimensional data. This significantly complicates fault detection and diagnosis. To tackle this challenge, a commonly used approach to address this challenge involves a two-step process: first, dimensionality reduction is performed, followed by density estimation in the reduced-dimensional space Candès, Li, Ma, Wright (2011). The problem with this method is that suboptimal performance is likely to happen due to the unawareness of the first step in regard to the second one. In addition, there is a possibility that key information for anomaly detection can be lost during the dimensionality reduction step. Therefore, a joint optimization for both dimensionality reduction and density estimation is crucial to this process. A deep learning framework called Deep Autoencoding Gaussian Mixture Mode (DAGMM) that addresses the aforementioned challenges has been proposed by Zong et al. Zong et al. (2018). A compression network and an estimation one are the core of DAGMM construction. Dimensionality reduction is first conducted by an autoencoder in the compression network. This compression network thereby prepares low-dimensional representations for the estimation network by concatenating reduced low-dimensional features from encoding and reconstruction error from decoding. The estimation network acts as GMM. It inputs the compression network's output and returns the mixture membership prediction of each sample. Based on that result, GMM parameters are estimated next, which facilitates input samples' statistical energy calculation. The sample's statistical energy value models the probability that we can observe the sample. If the sample has low statistical energy, there is a high probability that the sample follows the GMM (i.e., an inlier sample). Conversely, if the sample has high statistical energy, then it does not follow the GMM (i.e., an outlier sample). With the reconstruction error from the compression network and sample statistical energy from the estimation network minimized, a joint optimization for both dimensionality reduction and density estimation is possible.

However, DAGMM was originally designed as an unsupervised model that is mainly used for anomaly detection and some fault detection applications. For multiclass data like chiller data with different types of faults, the original version of DAGMM is ineffective for outlier detection. In this study, we propose a supervised multiclass version of DAGMM (S-DAGMM) that can work well on multiclass data. This supervised version undergoes the training phase in which labeled training data is first divided into sub-datasets, according to the data samples' labels. Each sub-dataset is used to train one individual DAGMM so that the number of trained individual DAGMMs is proportional to the number of data classes. DAGMMs are then used to detect outliers of their according sub-datasets based on sample statistical energy values. Clean sub-datasets are obtained after removing outliers and then are grouped to form the clean labeled training data for a chiller diagnosis model. During the online phase, new samples (unlabeled or labeled samples) are checked for outliers by trained DAGMMs and a voting scheme. Votes are increased by 1 if one of the trained DAGMMs identifies the sample as an outlier. After being checked by all DAGMMs, if the number of votes is larger than a pre-defined value, the sample is considered an outlier and vice versa.

Recently, deep learning techniques have witnessed a lot of successful applications in many areas, including natural language processing, computer vision, and robotics Tra et al. (2017); Tra, Nguyen, Kim, Kim (2021). End-to-end classification models have displayed a promising competence to extract features automatically during the requisite training period. Therefore, this study uses a deep neural network (DNN) as the chiller diagnostic model. Like other supervised classifiers, the high classification accuracy of DNN-based chiller FDD relies on the sufficiency of the labeled training data. In practice, however, while unlabeled data can be handily collected by remote sensors, the cost to obtain labeled fault data is expensive. There are two effective approaches to handle this obstacle. The first approach is to generate artificial training samples of minority classes using data augmentation methods such as the synthetic minority over-sampling technique (SMOTE) Tra et al. (2019). The upgraded version of SMOTE is called PCA-SMOTE in which the principal component analysis (PCA) technique is first used to extract principal features from high-dimensional original features before applying SMOTE to generate synthetic minority samples Fan, Cui, Han, Lu (2019);

Z. Zhou et al. (2021). Another effective method is to employ the concept of function approximator from neural networks like generative adversarial networks (GANs) to scale up the minority dataset. The advantage of these methods is that they can generate an unlimited number of artificial data samples from a small number of initial minority seeds. However, generated artificial data samples are often distributed close to the seeds in the feature space, so if the number of initial minority seeds is too small, the generated samples will not be more likely to fully cover the inherent data space of the minority class in testing online data.

The second approach is to take advantage of unlabeled data by using unsupervised layer-wise pretraining. One of the most used pre-trained networks is a stacked autoencoder (SAE) Tra et al. (2021). Since a pre-trained network can learn the data distribution from the unlabeled data, it can prevent DNN from getting stuck on local optima. By pretraining, moreover, sophisticated and abstract features with hierarchical structures can be efficiently learned because the technique offers layer-by-layer, high-level feature extraction from lower-level ones. Instead of using SAE, this research work utilizes the encoding layers of DAGMM's compression network to pre-train the DNN in order to prevent it from being trapped in local optimum due to random initialization. The hidden layers of the DNN classifier are first trained successively in a layer-wise, unsupervised learning strategy. After a soft-max output layer is added, the whole DNN is fine-tuned in a supervised manner with the labeled data. The studies in Zong et al. (2018) have shown that with regularization brought by the estimation network, the autoencoder is more capable of escaping from less attractive local optima. In addition, DAGMM's autoencoder reconstruction error is significantly lower than that of an autoencoder without the regularization part from the estimation network.

The proposed fault diagnosis framework for the chiller is schematically illustrated in Fig. 3.1. First, S-DAGMM is used to detect and remove outliers in both labeled and unlabeled chiller data. Next, the unlabeled data are utilized to train DAGMM which plays a role as a pretrained network for DNN. After fine-tuning using the labeled data, the DNN classifier is used as the fault diagnostic model in the online application. The DNN classifier inputs real-time data and outputs the corresponding fault type.

In summary, this chapter presents the following contributions:

(1) The supervised multiclass version of DAGMM is proposed. This method has outstanding performance in detecting outliers of multiclass data by profiling the label information of training data.

(2) The encoding layers of the DAGMM's compression network are suggested as a pre-trained network for the DNN-based diagnostic model. The encoding layers of DAGMM can compress the input well

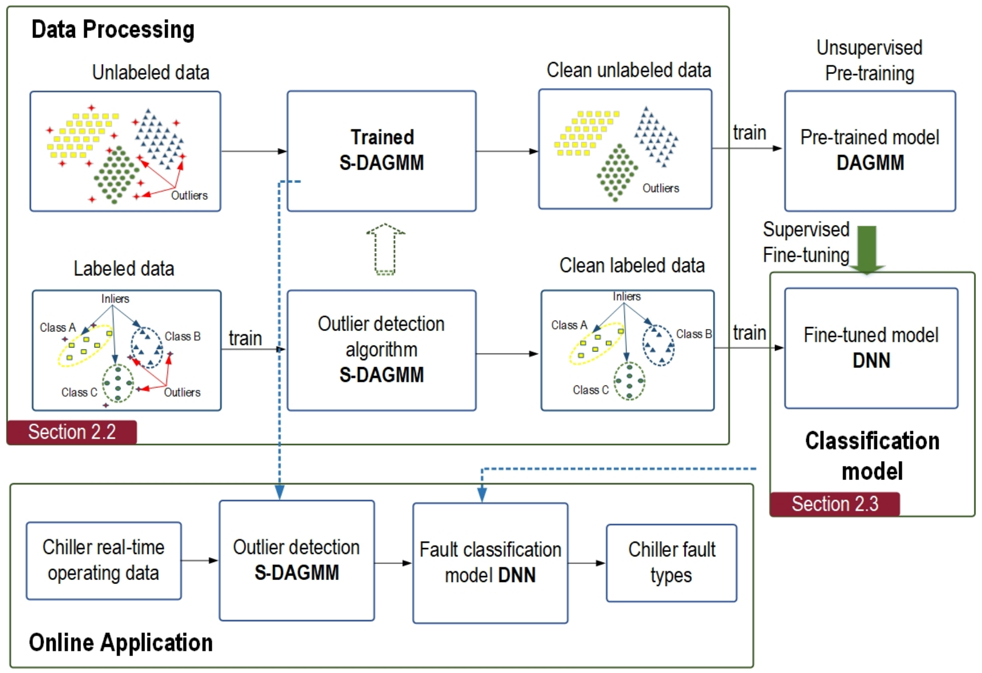

Figure 3.1: Graphical diagram of the proposed chiller fault diagnosis framework.

during end-to-end training so that their compressed representation can help DNN resolve generic classification tasks better.

(3) The comprehensive comparative study with state-of-the-art algorithms to validate the effectiveness of the proposed methods.

The following sections of this chapter are arranged as: Section 3.2 describes the supervised multiclass version of DAGMM and the DNN pretraining procedure. Experimental results to validate the proposed method's performance are shown in Section 3.3. Section 3.4 summarizes the conclusions.

3.2 Methodologies

3.2.1 Deep Autoencoding Gaussian Mixture Model

As previously described, DAGMM has two central parts: a compression network and an estimation network, which are depicted in Fig. 3.2. The estimation network inputs the representations returned by the dimensionality reduction process in the compression network and then outputs the statistical energy in the GMM framework.

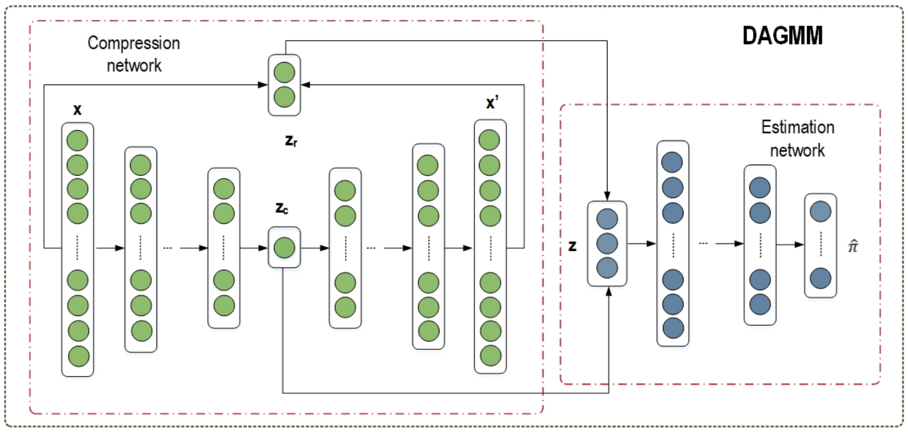

Figure 3.2: The pictorial diagram of Deep Autoencoding Gaussian Mixture Model.

There are two sources of features provided by the compression network's output: 1) the deep autoencoder's reduced low-dimensional representations; and 2) reconstruction error features. In this study, a sample $\mathbf{x}$ is the vector of chiller features. Each feature is a sensor measurement from a chiller. The low-dimensional representation $\mathbf{z}$ can be computed from the sample $\mathbf{x}$ as follows:

$$\mathbf{z}_c = h(\mathbf{x}; \theta_e), \quad \mathbf{x}' = g(\mathbf{z}_c; \theta_d), \tag{3.1}$$

$$\mathbf{z}_r = f(\mathbf{x}; \mathbf{x}'), \quad \mathbf{z} = [\mathbf{z}_c, \mathbf{z}_r]. \tag{3.2}$$

where $\mathbf{z}_c$ is the reduced low-dimensional representation learned by the deep autoencoder; $\mathbf{z}_r$ represents reconstruction error features (i.e., relative Euclidean distance, absolute Euclidean distance, cosine similarity, etc.); θ_e and θ_d are the parameters of the deep autoencoder; $\mathbf{x}^{'}$ is the reconstructed counterpart of $\mathbf{x}$; $h(\cdot)$ is the encoding function; $g(\cdot)$ is the decoding function; $f(\cdot)$ is the reconstruction error feature calculation function. Afterward, the estimation network is fed with the computed $\mathbf{z}$.

Given the low-dimensional representations computed by the compression network, the process follows with the density estimation performed by the GMM framework. The parameters of GMM and the sample statistical energy are then estimated by the estimation network with unknown mixture component distribution ϕ, mixture means μ, and mixture covariance Σ in the training phase. This is achievable through a multi-layer neural network that predicts each sample's mixture membership as follows:

$$\mathbf{p} = \text{MLP}(\mathbf{z}, \theta_m), \quad \hat{\gamma} = \text{softmax}(\mathbf{p}). \tag{3.3}$$

where $\mathbf{z}$ is the low-dimensional representation; K is the number of mixture components; $\hat{\boldsymbol{\gamma}}$ is a K-dimensional vector utilized for the prediction of soft mixture component membership; $\mathbf{p}$ is the multi-layer network's output; and θ_m is the multi-layer network's parameter.

The learning of GMM can be taken further with a N-sample batch and its membership prediction as follows:

$$\hat{\phi}_k = \frac{1}{N}\sum_{i=1}^{N}\hat{\gamma}_{ik}, \quad \hat{\boldsymbol{\mu}}_k = \frac{\sum_{i=1}^{N}\hat{\gamma}_{ik}\mathbf{z_i}}{\sum_{i=1}^{N}\hat{\gamma}_{ik}}, \quad \hat{\mathbf{\Sigma}}_k = \frac{\sum_{i=1}^{N}\hat{\gamma}_{ik}(\mathbf{z_i}-\hat{\boldsymbol{\mu}}_k)(\mathbf{z_i}-\hat{\boldsymbol{\mu}}_k)^T}{\sum_{i=1}^{N}\hat{\gamma}_{ik}}. \tag{3.4}$$

where $\hat{\boldsymbol{\gamma}}_i$ is the low-dimensional representation $\mathbf{z_i}$'s membership prediction; $\hat{\phi}_k$, $\hat{\boldsymbol{\mu}}_k$, $\hat{\mathbf{\Sigma}}_k$ are the k^{th} mixture component's weight, mean and covariance, respectively.

Afterward, sample statistical energy can be calculated with the estimated parameters:

$$E(\mathbf{z}) = -\log\left(\sum_{k=1}^{K}\hat{\phi}_k\frac{\exp\left(-(1/2)(\mathbf{z}-\hat{\boldsymbol{\mu}}_k)^T\hat{\mathbf{\Sigma}}_k^{-1}(\mathbf{z}-\hat{\boldsymbol{\mu}}_k)\right)}{\sqrt{|2\pi\hat{\mathbf{\Sigma}}_k|}}\right). \tag{3.5}$$

where $|\cdot|$ indicating the matrix's determinant.

The construction of the DAGMM objective function is given as follows with a N-sample batch:

$$J(\theta_e,\theta_d,\theta_m) = \frac{1}{N}\sum_{i=1}^{N}L(\mathbf{x_i},\mathbf{x}_i^{'}) + \frac{\lambda_1}{N}\sum_{i=1}^{N}E(\mathbf{z_i}) + \lambda_2 P(\hat{\mathbf{\Sigma}}) \tag{3.6}$$

This objective function includes three components: 1) $L(\mathbf{x_i},\mathbf{x}_i^{'})$ as the loss function that characterizes the deep autoencoder's reconstruction error; 2) $E(\mathbf{z_i})$ as the component that models probabilities that an input sample can be observed; 3) $P(\hat{\mathbf{\Sigma}}) = \sum_{k=1}^{K}\sum_{j=1}^{d}1/\hat{\Sigma}_{kij}$ is the component utilized to penalize small values on diagonal entries in order to minimize the singularity problem that is inherent in DAGMM. Here, λ_1, and λ_2 are DAGMM's meta parameters and d is the dimension of the low-dimensional representations provided by the compression network.

3.2.2 Supervised Multiclass Deep Autoencoding Gaussian Mixture Model

DAGMM has shown outstanding performance for anomaly detection by distinguishing observations that have statistical energy values distinct from the majority of the data population. However, this method only works well for binary classification where anomalies are considered as the minority class. In the case of multi-class data such as the chiller data, which is being under investigation here, DAGMM is not capable of completely resolving the outlier detection problem. Since multiclass data often follow different distributions and an outlier can follow one of them, therefore, the DAGMM cannot distinguish a given outlier using data distributions based on its statistical energy. In this study, we have proposed the supervised multiclass version of DAGMM called the Supervised Multiclass Deep Autoencoding Gaussian Mixture Model (S-DAGMM). This method divides the multiclass outlier detection problem into small anomaly detection tasks by utilizing the label information of samples during the training process. Fig. 3.3 illustrates the proposed S-DAGMM.

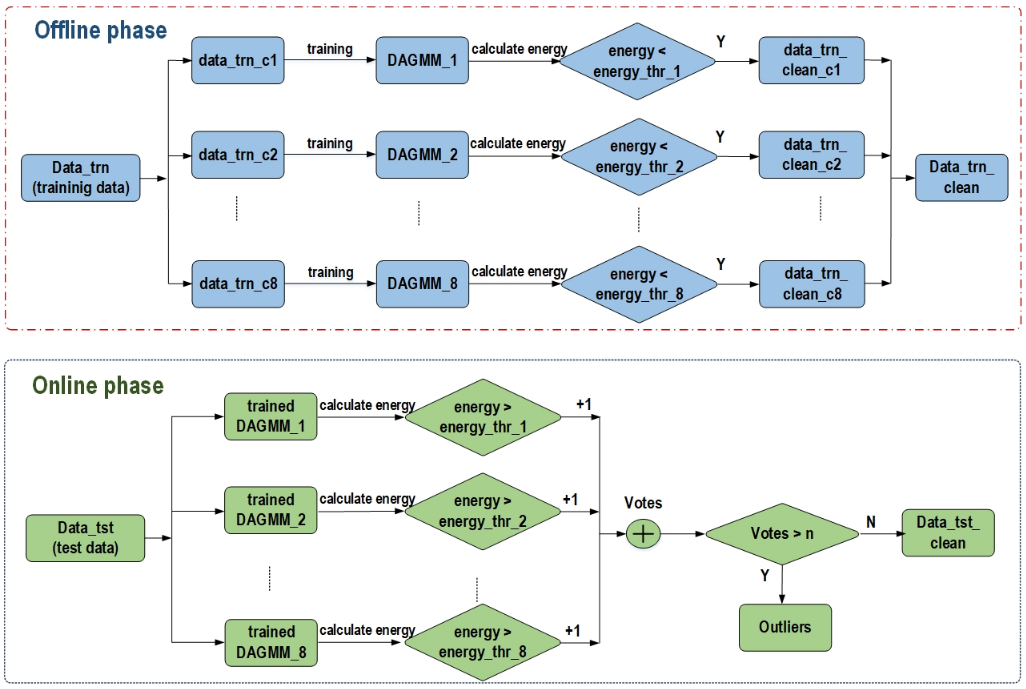

Figure 3.3: The illustrative diagram of Supervised Multiclass Deep Autoencoding Gaussian Mixture Model.

In the offline phase, labeled training data are first divided into sub-datasets, according to their labels. Each sub-dataset is used to train one DAGMM. The number of trained DAGMMs is proportional to the number of data classes. Each DAGMM is then used to detect outliers in the sub-dataset used to train that DAGMM based on the samples' statistical energy values. Clean sub-datasets are obtained after removing outliers and then grouped to create the clean labeled training data for the chiller diagnosis model. In the online phase, a new test sample (a labeled sample or an unlabeled sample) is checked (to be an outlier or not) by trained DAGMMs. The test sample's statistical energy is first computed by trained DAGMMs. Votes are incremented by 1 if one of the trained DAGMMs identifies the sample as an outlier. In the end, if the pre-defined value n is exceeded by the number of votes, the sample is then considered as an outlier (and vice versa). The voting scheme enforces reliability and ensures that S-DAGMM can detect efficiently outliers, even ambiguous ones.

3.2.3 Pretraining DNN classifier using DAGMM

The multiple hidden layers concept has been available since the early years of deep learning. This approach was initially disappointing because its performance was even worse than shallow networks. The reason behind this inferior performance is that conventional back-propagation with random initialization often causes the training process to get stuck in unoptimized local solutions. In order to deal with the existing limitations of DNN optimization, authors in Hinton, Osindero, Teh (2006) proposed unsupervised layer-wise pretraining. This technique is especially powerful in case the number of labeled training samples is not sufficient for DNN to generalize a classification task, while the number of unlabeled training samples is abundant and available. By pretraining, sophisticated and abstract features with hierarchical structures can be efficiently learned because the technique offers layer-by-layer, high-level feature extraction from lower-level ones.

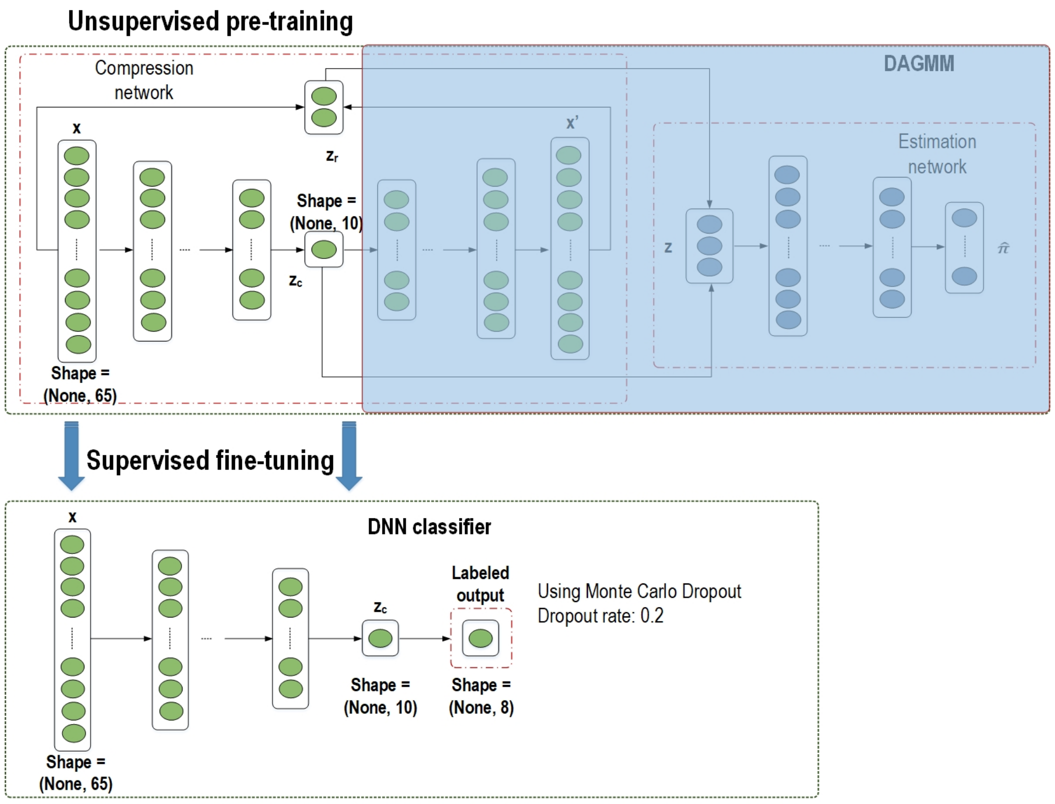

Figure 3.4: The pre-training and fine-tuning procedure of the DNN classifier.

In this study instead of using Stacked Autoencoder (SAE), DAGMM is used to pretrain DNN due to its friendliness to end-to-end training. The pretraining and supervised fine-tuning of the DNN classifier using DAGMM are illustrated in Fig. 3.4. First, DAGMM is pre-trained using abundant unlabeled training chiller data. Following the unsupervised pretraining, in order to fine-tune the weights and biases to the DNN classifier, the output layer with a size equal to the number of data categories is added to the top of the DAGMM's encoding layers. While the initialization of the DNN classifier's hidden layer weights is done with pre-trained encoding weights $\theta_e = \{\omega_e, b_e\}$, the output layer parameter $\{\omega_o, b_o\}$ can be initialized randomly. Afterward, the entire DNN is fine-tuned via backpropagation using labeled training data to achieve better weights. With regularization brought by the estimation network, the DAGMM's autoencoder is more capable of escaping from less attractive local optima Zong et al. (2018).

3.3 Experimental results

In this study, we utilized the ASHRAE RP-1043 chiller dataset, as detailed in Chapter 2, to validate both the proposed models and the baseline models. This dataset provides a comprehensive set of samples across various fault conditions and severity levels, enabling thorough evaluation and comparison. By leveraging this dataset, we aim to demonstrate the effectiveness and robustness of our models in accurately detecting and diagnosing faults within chiller systems.

3.3.1 The efficacy of S-DAGMM

For comprehensive validation, each severity level is evaluated individually. To mimic the practical scenario that outliers in chiller data are generated by errors made in the data recording process such as wrong measurements, or noises, for each dataset 20% of data is manually modified by randomly changing some features of each sample record to the value of 500 (i.e., 2 out of 65 original features). The order of the changed features in the records is not the same and is chosen at random. The reason for choosing a value of 500 is that except for some features with very high values (i.e., TSO, Shared Cond Tons, Cond Energy Balance, THO, etc.), most features have mean values in an approximate range from 14 to 270, so the value of 500 can be considered the wrong feature value of most features and should be detected with outlier detection algorithm. If we change features with lower values (i.e., 300), even if the outlier detection algorithms fail to detect simulated outliers, the classifiers still correctly classify these outliers. In contrast, if we change features with higher values (i.e., 1000), simulated outliers in this study will be easily detected by most outlier detection algorithms. Then, the comparison between algorithms will not be convincing when the output of the outlier detection algorithms is not too different. The idea of using transient fault data of the RP-1043 dataset as outliers has also been considered. However, the transient chiller fault data themselves are neither too divergent nor distinct from the steady fault data. Experimental results conducted in this study (i.e., Table 3.1) have shown that when the supervised classifiers like SVM, KNN, RF, etc. are tested with the processed data (after removing artificial outliers and containing both steady-state data and transient data), the accuracy rates reach to 99.8%. This means that with sufficient training data, supervised classifiers can generalize a classification task for both the online steady data and the transient data.

To demonstrate the effectiveness of the proposed supervised DAGMM (S-DAGMM) method, its performance is compared to those of the original unsupervised DAGMM and some state-of-the-art outlier detection algorithms such as one-class support vector machine (OC-SVM) Sadooghi Khadem (2018) and isolation forest (IF) F. T. Liu, Ting, Zhou (2012). The parameter that needs to be tuned in S-DAGMM is the pre-defined value of votes n. If the pre-defined value is set high, the final decision of S-DAGMM will be more certain (True Positive is high) since the decision is agreed upon by the majority of individual DAGMMs in S-DAGMM, but the probability of having misdetected outlier samples will increase (False Negative is high). On the contrary, if the pre-defined value is set low, more samples will be detected as outliers, but many of these detected outliers may be actually inliers (False Positive is high). In this study, the pre-defined value of votes is set so that the number of samples detected as outliers by S-DAGMM accounts for 20% of the original samples, equal to the proportion of outliers that were manually generated. For the experimental datasets, the pre-defined value is set to 5 (i.e., $n = 5$) to satisfy the above requirements.

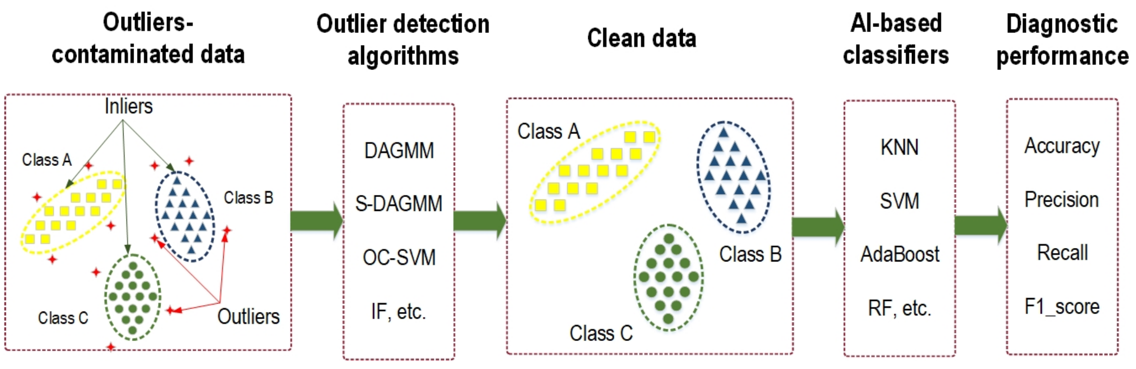

Figure 3.5: The experimental procedure to validate the performance of outlier detection algorithms.

As an indirect measurement for the comparison, base classifiers are utilized for fault diagnosis following the outlier detection algorithms processing. The procedure to indirectly validate the performance of outlier detection algorithms is illustrated in Fig. 3.5. To enforce the objectivity and reliability of the comparison, some notable and state-of-the-art algorithms are chosen as base classifiers, including KNN, SVM, RF, and adaptive boosting (AdaBoost). Diagnostic accuracy, which is the ratio of the number of correctly testing diagnosed samples over the total number of testing samples, is chosen as the performance evaluation metric. To reduce the variance and make results more credible, the grid search technique with 5-fold cross-validation is applied for base classifiers to find their best estimators. In the case of the RF and AdaBoost classifiers, a decision tree (DT) is used as a base estimator. Feature scaling is useful and applied for the KNN classifier.

For each of the four datasets, the number of samples in the training set and testing set is divided equally, so each contains 41,528 / 2 = 20,764 samples. After applying the outlier detection algorithms to remove outlying data samples (i.e., 20% of the dataset), the training set and testing set each contain 0.8 * 20,764 = 16,611 samples. Using these datasets to validate base classifiers, the classification accuracy rates are listed in Table 3.1. It can be perceived from Table 3.1 that diagnostic accuracy can be remarkably improved when sample outliers are detected and discarded from the chiller data. Without outlier detection applied, the testing diagnostic accuracy rates of KNN, SVM, RF, and AdaBoost of severity level 1 are 86.1%, 84.3%, 94.8%, and 63.0%, respectively. When the outlier detection is applied, the diagnostic accuracy rates of these base classifiers can be improved to 97.7%, 98.4%, 99.8%, and 67.1%, respectively. The accuracy rates of KNN, and SVM in cases without outlier detection applied (i.e., 86.1%, and 84.3%, respectively) proportionally reflect the percentage of artificial outliers in the analysis data (i.e., 20% in both training data and testing data). While DAGMM has approximate performance as those of the OC-SVM, and IF algorithms, S-DAGMM has a higher performance than the other methods with around 2%   5% diagnostic accuracy rates improvement, regardless of the base classifier used. The reason for this performance improvement is that the proposed S-DAGMM uses a group of individual DAGMMs, that are accordingly trained based on label information of training data, to make the final decision on potential outliers. A mechanism like ensemble models helps the S-DAGMM model overcome the uncertainty of chiller data and can detect ambiguous outliers in both training data and testing data.

The statistical energy values of 1000 data samples estimated by individual DAGMMs are shown in Fig. 3.6. Out of 1000 data samples, the first 800 samples are picked randomly from the chiller dataset severity level 1, the last 200 samples are outliers generated manually from the normal and seven different faults

Table 3.1: Testing diagnostic accuracy rates of different base classifiers in case outlier detection algorithms applied.

Outlier detection algorithms	**Severity Level 1**				**Severity Level 2**			
	KNN	**SVM**	**RF**	**Ada**	**KNN**	**SVM**	**RF**	**Ada**
	Diagnostic Accuracy on Testing Set				Diagnostic Accuracy on Testing Set			
None	86.1	84.3	94.8	63.0	87.6	85.4	96.8	72.3
Supervised DAGMM	**97.7**	**98.4**	**99.8**	**67.1**	**98.4**	**99.0**	**99.8**	**74.5**
Unsupervised DAGMM	94.1	94.6	97.9	64.4	95.9	95.6	98.9	73.9
OC-SVM	93.2	93.0	97.2	63.8	94.9	94.7	98.9	73.1
IF	91.3	91.0	96.9	63.6	93.5	93.9	97.8	72.5
None: *do not apply outlier detection on the data (use the original data)*								
Outlier detection algorithms	**Severity Level 3**				**Severity Level 4**			
	KNN	**SVM**	**RF**	**Ada**	**KNN**	**SVM**	**RF**	**Ada**
	Diagnostic Accuracy on Testing Set				Diagnostic Accuracy on Testing Set			
None	88.9	86.3	98.8	74.0	91.2	86.6	98.8	85.2
Supervised DAGMM	**99.4**	**99.6**	**99.9**	**77.5**	**99.2**	**99.4**	**99.8**	**88.9**
Unsupervised DAGMM	96.1	94.8	98.9	76.2	97.8	97.0	99.6	86.5
OC-SVM	95.4	94.7	98.6	76.7	97.2	96.8	99.4	86.0
IF	95.2	94.0	98.8	75.4	96.5	94.2	98.9	85.5
None: *do not apply outlier detection on the data (use the original data)*								

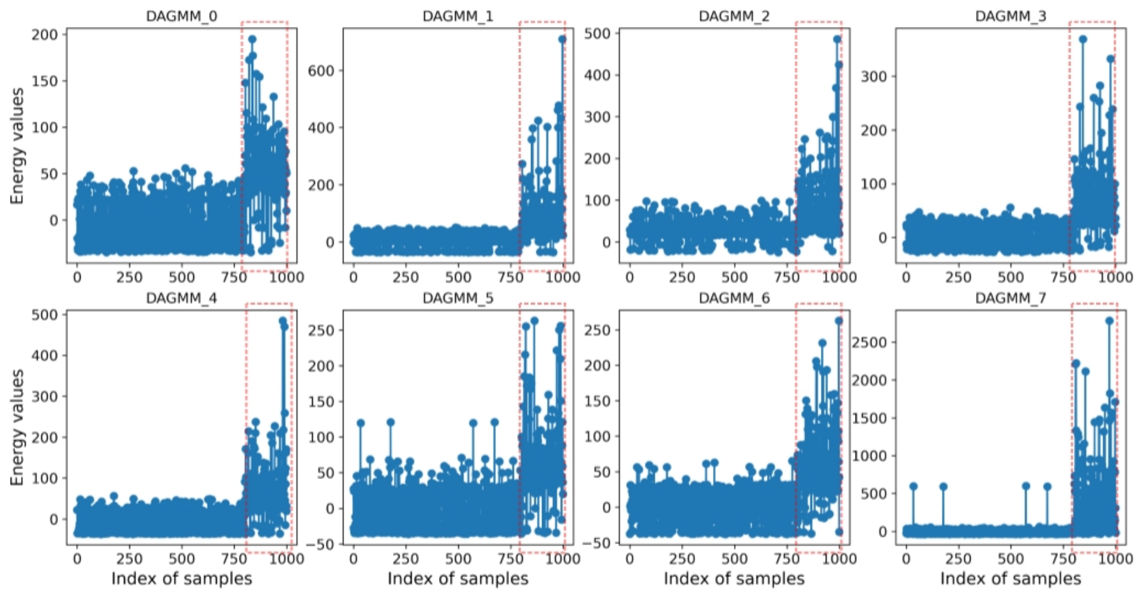

Figure 3.6: Statistical energy values of data samples estimated by individual DAGMMs.

Table 3.2: Testing F1-score of different base classifiers in case S-DAGMM algorithm applied.

Fault types	**Severity level 1**											
	KNN			**SVM**			**RF**			**AdaBoost**		
	P	R	F	P	R	F	P	R	F	P	R	F
F0-Normal	94.1	97.8	**95.9**	97.8	97.9	**97.8**	99.4	99.8	**99.6**	51.2	15.2	**23.5**
F1-ReduCF	99.3	98.2	**98.8**	100	98.1	**99.0**	100	100	**100**	100	100	**100**
2 - ReduEF	99.9	98.9	**99.4**	100	99.7	**99.8**	100	100	**100**	100	100	**100**
F3 - RefLeak	94.7	95.5	**95.1**	97.7	96.4	**97.0**	99.8	99.2	**99.5**	28.5	43.3	**34.3**
F4 - RefOver	96.8	95.9	**96.4**	93.6	98.5	**96.0**	99.9	100	**99.9**	42.7	94.7	**58.9**
F5 - ExcsOil	97.8	97.2	**97.5**	98.9	98.8	**98.9**	99.9	100	**99.9**	13.9	1.0	**1.9**
F6 - ConFoul	99.6	98.9	**99.2**	99.8	99.0	**99.4**	99.9	100	**99.9**	100	90.2	**94.8**
F7 - NonCon	99.9	99.0	**99.4**	99.8	98.8	**99.3**	100	99.9	**99.9**	99.3	97.0	**98.1**
***P:** Precision; ***R:** Recall; ***F:** F1-score;*												

chiller data of the same severity level. From Fig. 3.6, we can see that an outlier can be simply detected by statistical energy values estimated by individual DAGMMs of the S-DAGMM model. The voting scheme enforces reliability and ensures that S-DAGMM can detect efficiently outliers, even ambiguous ones.

F1-score is one of the common statistics besides accuracy. This metric, which is calculated from the precision (P) and recall (R) of the test, reflects the comprehensive diagnostic performance of classification models. Precision is the ratio of correctly predicted samples over prediction samples, while recall refers to the fraction of correctly predicted samples in the real samples. In some cases, high recall is obtained by sacrificing precision. The F-score is calculated as in Eq. (3.7). The larger the F-score is, the better the comprehensive performance of the model gets.

$$F1 - score = \frac{2P * R}{P + R} \tag{3.7}$$

Table 3.2 and Fig. 3.7 show the F1-score for all categories of the base classifiers on the testing set of severity level 1 when the S-DAGMM algorithm is applied, in terms of statistical and graphical perspectives, respectively. From Table 3, we can see that the F-score for all categories of some base classifiers mostly reached 100% (i.e., RF, SVM). This result proves that the application of the S-DAGMM algorithm comprehensively improves the diagnostic performance of the base classifiers. In addition, Fig. 3.7 shows that for all classifiers, especially AdaBoost, F1-scores for normal state and system-level faults (RefLeak, RefOver, ExcsOil) are lower than those of element-level faults (ReduCF, ReduEF, ConFoul, NonCon). While the F1-score for component-level faults is higher than 94%, regardless of the base classifier used, the F1-score for system-level faults of the AdaBoost classifier is less than 60%. This means system-level faults such as

RefLeak, RefOver, and ExcsOil are more difficult to distinguish than component-level faults.

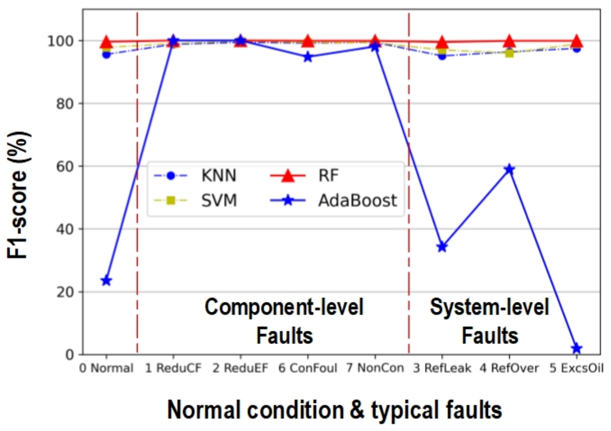

Figure 3.7: F1-score curves of different base classifiers in case S-DAGMM algorithm applied (severity level 1).

Fig. 3.8 is the graphical representation of Table 3.1. Fig. 3.8 shows that the performance of the outlier detection algorithms has the same trend when tested on different severity levels (i.e., severity levels 1, 2, 3, and 4). Another noticeable point is that the diagnostic accuracy rate of the base classifiers increases when the severity level becomes higher. However, even if the faults are in the incipient stage (i.e., severity level 1), the faults can be accurately diagnosed from outliers non-contaminated chiller data. From the statistics shown in Table 3.1, we can observe that, unlike base classifiers such as the KNN and SVM that are sensitive to outliers, the accuracy performance of RF and AdaBoost classifiers is less affected by outliers mixed in training data. For the case of SVM, the reason is that outliers cause SVM to misidentify the separating hyperplane. While KNN is the distance-based method and outliers dramatically change its class boundaries. In the contrary, for all 4 severity levels, the diagnostic accuracy rates of RF and AdaBoost classifiers on outliers-contaminated data are close to those of RF and AdaBoost classifiers when tested on non-contaminated data.

3.3.2 The efficacy of DAGMM-based pretraining technique

Our experiment aims to create a scenario as close to the real-life situation as possible, in which only a small fraction of available historical data is labeled, while the rest are not. The labeled training dataset is formed from only a small number of chiller training samples, while the unlabeled training dataset contains training samples whose labels are neglected and assumed unknown. This unlabeled training set is used

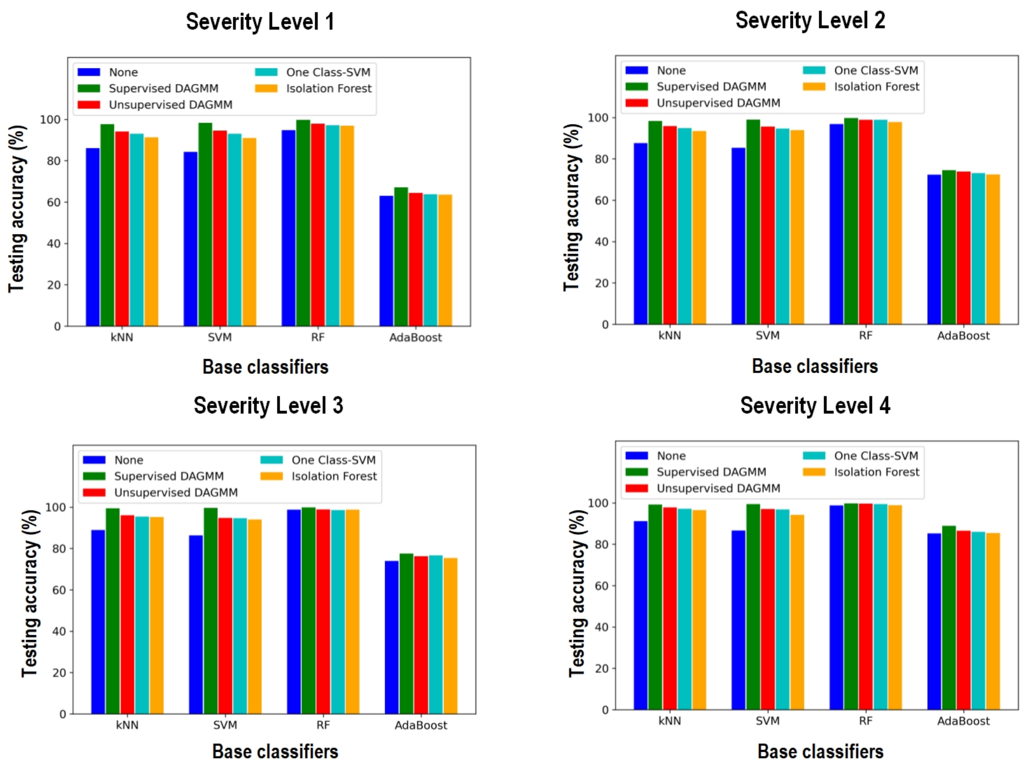

Figure 3.8: Testing accuracy rates of different base classifiers at 4 severity levels in case outlier detection algorithms applied.

to pre-train DAGMM for finetuning the DNN classifier. To show the advantage of the proposed DAGMM-based finetuned DNN classifier (DAGMM-DNN), a comparative study was conducted between the proposed DAGMM-DNN and the Stacked Autoencoder (SAE)-based finetuned DNN classifier (SAE-DNN) in terms of diagnostic accuracy metric. The DNN classifiers without using the pretraining technique and the base classifiers in the above experiments are also the objects of this comparison.

The information in the unlabeled training samples can be harnessed by our proposed DAGMM-DNN to obtain optimal weights via pretraining DAGMM. Instead of starting with randomly initialized weights, pretraining helps the DNN escape from less attractive local optima and gain relatively high accuracy even when the number of labeled training samples is limited. For each severity level, the size of the unlabeled training set and the testing set is fixed and is half the size of the original dataset (i.e., 41,528 / 2 = 20,764 samples). Meanwhile, the size of the labeled training set changes from 0.5% (103 samples), 1% (207 samples), 3% (622 samples), and 10% (2076 samples) the size of the unlabeled training set. This setting aims to validate the performance of supervised classifiers in case the labeled training data volume is small. Because the labeled training data are randomly chosen from the training set, different random seeds will result in different labeled training sets that lead to different results. Therefore, the final accuracy rate of each classifier in this experiment is calculated by taking the average of 5 repeated trials. In each trial, a random selection of the labeled training samples is performed. To make a fair comparison, the structure of DAGMM-DNN, SAE-DNN, and DNN is the same as the input layer, the output layer, and some hidden layers of diminishing sizes. Since the experimental data contain 65 features, then the input layer size is 65. The output layer has 8 neurons, according to 8 different data classes. Logically, the neural network with more hidden layers is capable of extracting more complex patterns from the input. Through many numerical tests, however, the patterns of the chiller dataset are found relatively simple B. Li et al. (2021). For the case in this study, when the depth of hidden layers increases to more than three, the DNN classifier's classification performance does not increase proportionally. Therefore, the DNN classifier's hidden structure is optimized with 3 layers of diminishing sizes (50 units, 30 units, then 10 units). The procedure to validate the performance of pretraining techniques is demonstrated in Fig. 3.9.

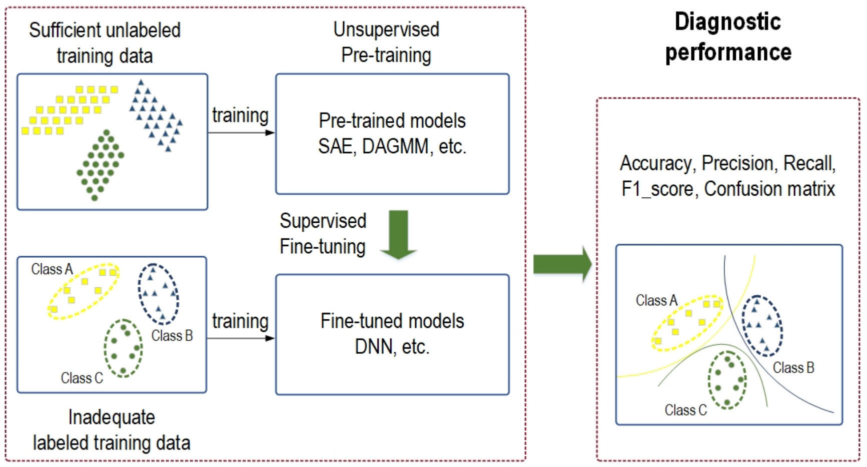
Sufficient unlabeled training data
Unsupervised Pre-training
training
Pre-trained models SAE, DAGMM, etc.
Supervised Fine-tuning
Class A
Class B
Class C
training
Fine-tuned models DNN, etc.
Inadequate labeled training data
Diagnostic performance
Accuracy, Precision, Recall, F1_score, Confusion matrix
Class A
Class B
Class C

Figure 3.9: The experimental procedure to validate the performance of pretraining techniques.

The fault diagnosis results of the classifiers are shown in Table 3.3. From Table 3.3, we can see the diagnostic improvement of the DAGMM-DNN and SAE-DNN classifiers compared with the DNN classifier without applying a pretraining technique. The difference is significant when the size of the labeled training set is small. In case the number of training samples is 103, the improvement in terms of testing accuracy rate is about 6% to 10%, regardless severity level of chiller data. The reason for this improvement is that with abundant unlabeled training data, pre-trained networks like DAGMM and SAE prevent the DNN classifier from getting stuck on local optima, especially in case the number of labeled training samples is not enough for the DNN classifier to generalize a fault diagnostic task. As the size of the labeled training set is 622

Table 3.3: Testing diagnostic accuracy rates of different classifiers according to different sizes of labeled training sets.

Diagnostic models	**Severity Level 1**				**Severity Level 2**			
	Size of the labeled training set				Size of the labeled training set			
	2,076	**622**	**207**	**103**	**2,076**	**622**	**207**	**103**
	Diagnostic Accuracy on Testing Set				Diagnostic Accuracy on Testing Set			
SAE-DNN	92.7	90.8	80.4	72.5	95.8	91.8	88.4	74.9
DAGMM-DNN	**95.4**	**92.2**	**84.2**	**73.4**	**97.0**	**96.3**	**89.2**	**76.2**
DNN	95.4	91.4	77.9	64.6	96.9	96.0	85.6	69.1
KNN	86.7	64.5	25.3	23.7	93.6	74.9	43.2	31.8
SVM	91.6	74.5	45.2	30.0	95.5	81.7	52.0	32.8
Random Forest	97.6	91.2	79.2	69.0	98.1	91.1	81.5	70.6
AdaBoost	75.3	74.2	72.3	63.8	72.9	70.1	66.2	56.4
Diagnostic models	**Severity Level 3**				**Severity Level 4**			
	Size of the labeled training set				Size of the labeled training set			
	2,076	**622**	**207**	**103**	**2,076**	**622**	**207**	**103**
	Diagnostic Accuracy on Testing Set				Diagnostic Accuracy on Testing Set			
SAE-DNN	98.5	97.4	93.9	88.0	98.2	97.4	96.3	87.8
DAGMM-DNN	**98.6**	**97.8**	**96.3**	**89.9**	**98.7**	**97.5**	**96.4**	**87.6**
DNN	98.8	97.8	92.8	79.0	98.4	97.5	93.9	81.7
KNN	94.9	78.1	48.7	38.3	96.3	86.2	64.3	53.0
SVM	95.3	81.0	50.5	33.8	96.7	90.2	59.6	39.3
Random Forest	99.2	96.2	90.2	82.5	98.9	97.1	93.8	86.6
AdaBoost	82.1	78.5	71.0	64.6	93.1	92.0	84.0	81.4
**SAE-DNN: DNN classifier uses SAE to pretrain*								
**DAGMM-DNN: DNN classifier uses DAGMM to pretrain*								

or higher, the testing accuracy rates between the DNN classifiers with pretraining and the DNN classifier without pretraining are almost similar. This is because as the labeled training data are sufficient, supervised classifiers like DNN and the base classifiers can obtain enough generalization capability to discriminate different faults via training.

Comparing between two fine-tuned models, the diagnostic accuracy of SAE-DNN is consistently inferior to DAGMM-DNN, with approximately 1% to 5% lower accuracy rates. Especially when the number of labeled training samples is 622 or higher, the accuracy rate of the SAE-DNN classifier is even lower than that of the DNN classifier without pretraining. The better performance of DAGMM is due to the regularization from the estimation network, which is outstandingly useful in terms of local optima escaping. Meanwhile, a stacked autoencoder without the regularization from the estimation network has a high chance of converging to the local minimum. Besides, the diagnostic accuracy rate of DAGMM-DNN is also superior to that of the base classifiers such as KNN, SVM, RF, and AdaBoost as the size of the labeled training set is less than 622 (i.e., 103 or 207). The improvement in terms of testing accuracy rate is about 5% to 59%, regardless severity level of chiller data. In case the size of the labeled training set is 622 or higher (i.e., 622 or 2076), the DAGMM-DNN's accuracy rate is not much better and even lower in some cases. For example, as the size of the labeled training set is 2,076, the accuracy rate of the RF classifier is highest, reaching over 97.6%, regardless severity level of chiller data.

Table 3.4 shows the F1-score for all categories of different classifiers with 207 training labeled samples. These classifiers include DNN, SAE-DNN, DAGMM-DNN, and RF, which have the highest accuracy rates between the base classifiers. Fig. 3.10 is the graphical representation of statistical results in Table 3.4. From Fig. 3.10, it is clear that the F1 scores of all the classifiers for system-level faults are lower than those of element-level faults. Compared with SAE-DNN and DNN, we can see that when the volume of the training labeled set is small, DAGMM-DNN achieves the best F1 score for all categories. Especially for system-level faults such as Refrigerant leak (RefLeak), Refrigerant overcharge (RefOver) and Excess oil (ExcsOil), the difference in F1-score of DAGMM-DNN and DNN is even more significant (at about 13% to 25%). In addition, the F1-score of DAGMM-DNN is also higher than that of RF for almost all the categories.

Table 3.4: Testing F1-score of different classifiers with 207 labeled training samples.

Fault types	**Severity level 1**											
	KNN			**SVM**			**RF**			**AdaBoost**		
	P	R	F	P	R	F	P	R	F	P	R	F
F0-Normal	66.3	68.5	**67.4**	72.5	71.5	**72.0**	70.5	64.4	**67.3**	71.3	59.7	**65.0**
F1-ReduCF	99.9	97.8	**98.8**	99.8	98.4	**99.1**	93.7	98.9	**96.2**	100	99.7	**99.8**
F2 - ReduEF	93.9	99.9	**96.8**	96.0	98.2	**97.1**	99.0	98.1	**98.5**	96.5	100	**98.2**
F3 - RefLeak	40.4	39.5	**39.9**	53.3	48.2	**50.6**	39.4	38.9	**39.1**	45.3	49.5	**47.4**
F4 - RefOver	76.2	81.2	**78.6**	77.5	83.1	**80.2**	67.2	76.9	**71.7**	66.0	78.4	**71.6**
F5 - ExcsOil	68.7	60.1	**64.1**	74.0	78.6	**76.2**	61.2	60.6	**60.9**	58.3	51.9	**54.9**
F6 - ConFoul	97.0	97.5	**97.2**	99.3	97.8	**98.6**	98.4	90.7	**94.3**	98.5	97.6	**98.1**
F7 - NonCon	98.7	98.4	**98.6**	99.3	97.1	**98.2**	94.6	94.0	**94.3**	99.8	96.2	**98.0**
***P:** Precision; ***R:** Recall; ***F:** F1-score;*												

A confusion matrix is another metric that is often used to describe the performance of classification models on a testing set for which the true values are known. For more in-depth diagnostic performance comparison, confusion matrices of the classifiers are also studied as shown in Fig. 3.11. In Fig. 3.11, the rows represent the true labels of the faults, while the columns represent the predicted labels. The reports with a great number of misclassified samples are marked with red dot circles. From Fig. 3.11, we can see that with a small number of labeled training samples (i.e., 207 samples), system-level faults (RefLeak,

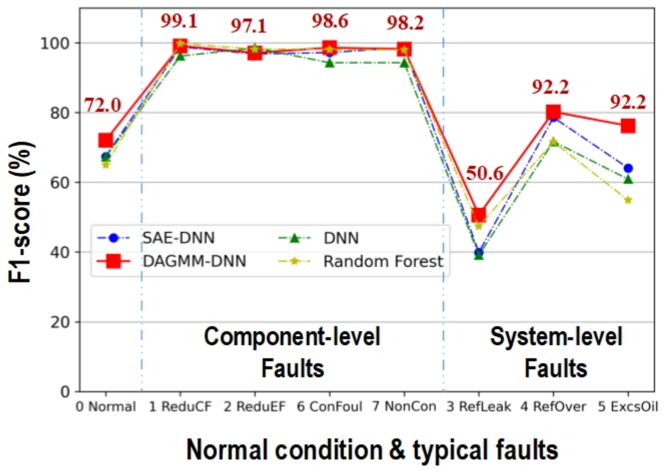

Figure 3.10: F1-score curves of DNN classifiers and RF classifier with 207 labeled training samples (severity level 1).

RefOver, ExcsOil) are misclassified at a great scale. One reason for this great misclassification is that fault samples are in the incipient stage (i.e., severity level 1) so many of them are misclassified as normal states (and vice versa). More specifically, many normal samples in row 1 are misclassified as RefLeak, RefOver, or ExcsOil. While the samples of RefLeak state are mainly misclassified as RefOver, ExcsOil, or normal state, a high proportion of ExcsOil samples are misdiagnosed as normal state, RefLeak, and RefOver.

In comparison with SAE-DNN and DAGMM-DNN, the DNN classifier does not perform well and reports the largest number of misdiagnosed samples, especially in the case of normal state and system-level faults. More specifically, for 2595 testing samples of each of the typical faults, the number of correctly diagnosed samples of the DNN classifier in case of normal state, RefLeak, RefOver, and ExcsOil are only 1662, 997, 2041, and 1572, respectively. Meanwhile, the figures for DAGMM-DNN are higher with the numbers 1844, 1236, 2207, and 2037, respectively. These results reinforce the conclusion that using DAGMM to pre-train DNN significantly increases the diagnostic performance of DNN.

Fig. 3.12 is the graphical representation of Table 3.3. From the subfigures in Fig. 3.12, we can see that the diagnostic rate curves of the base classifiers have the same trend. These curves of the classifiers are sharp in the beginning and tend to be constant with the increase of unlabeled training samples. Regardless of the size of the labeled training set, the accuracy rates of the DAGMM-DNN classifier are higher than those of base classifiers, except for very few exceptions (i.e., the accuracy rates of the Random Forest classifier are highest among compared classifiers in case the size of the labeled training set is 2076 samples). Another noticed point is that when the number of labeled training samples increases, the classification accuracy

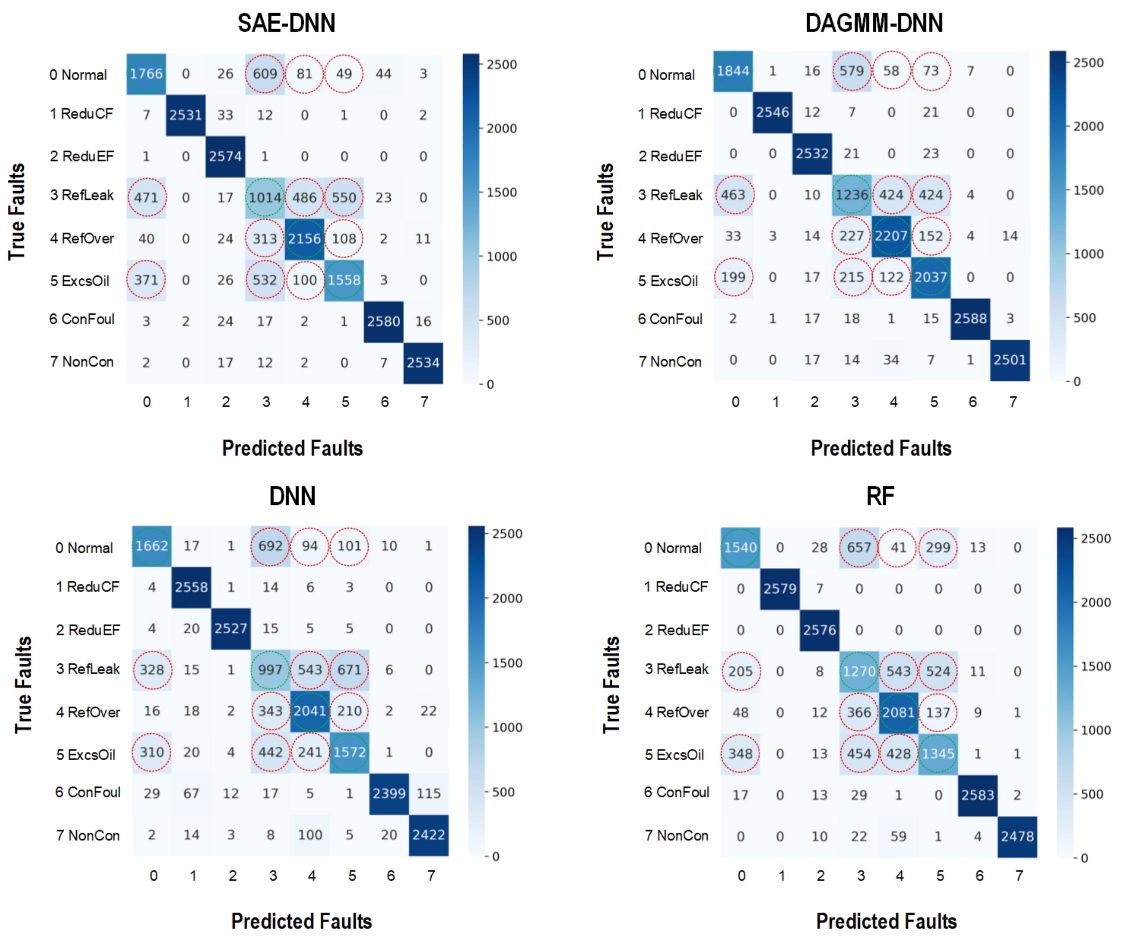

Figure 3.11: Confusion matrices of DNN classifiers and RF classifier with 207 labeled training samples.

difference between the finetuned networks and the supervised base classifiers becomes smaller. While the SVM and KNN classifiers are sensitive to the size of the labeled training sets as their accuracy rates are almost lower than 50% in case the labeled training sample is 207, the accuracy rates of the Random Forest and AdaBoost classifiers are not affected too much by the size of the training set. In both experiments, the Random Forest classifier proved to be a robust model that is resistant to outliers and unaffected by the size of the training set.

3.4 Conclusion

With a focus on the outlier detection problem, this chapter proposed a supervised DAGMM (S-DAGMM) model that can effectively detect outliers of multiclass chiller data by profiling the label information of data during the training process. A mechanism like ensemble models helps the S-DAGMM model overcome

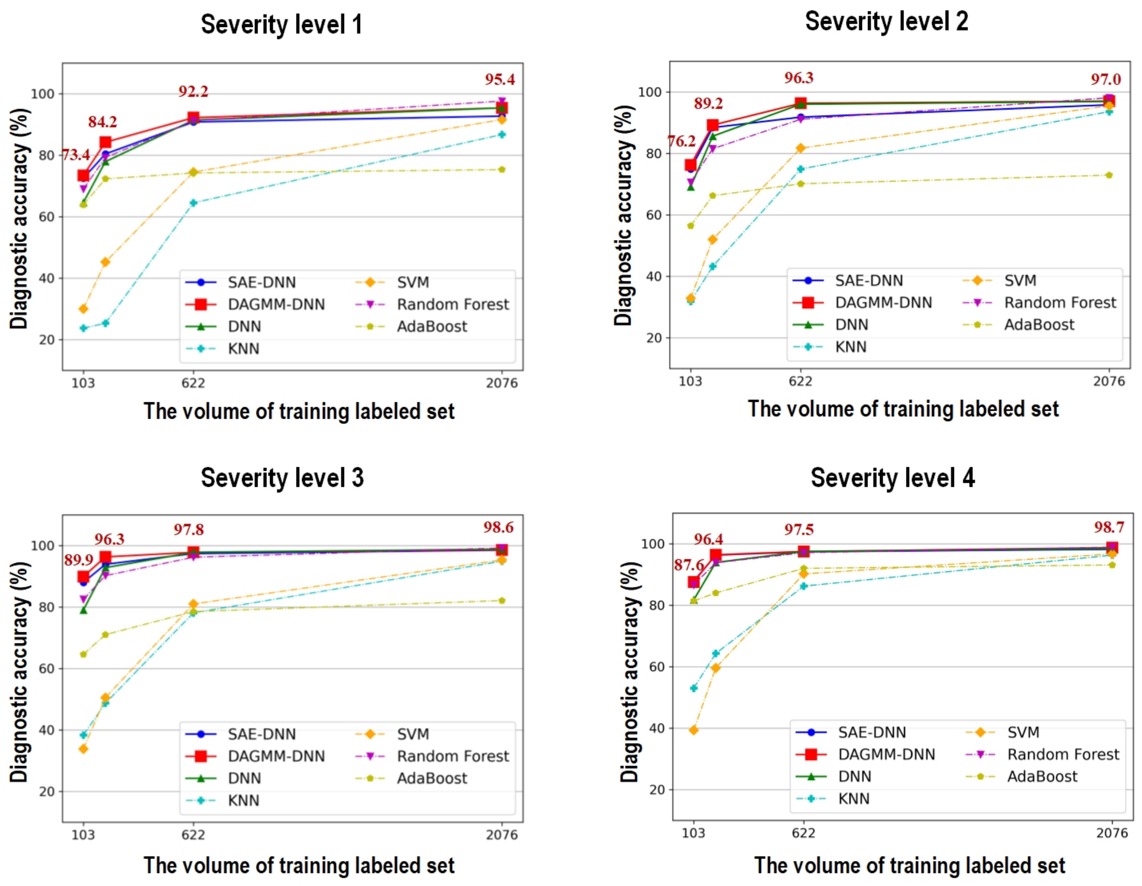

Figure 3.12: Testing accuracy rate curves of different classifiers according to different sizes of labeled training set.

the uncertainty of chiller data and can detect even ambiguous outliers in both training data and testing data as well. The experimental results have shown that the effectiveness of S-DAGMM is superior to that of DAGMM, OC-SVM, and IF. More specifically in case the data is processed by S-DAGMM, the diagnostic accuracy rate of the base classifiers such as KNN, SVM, RF, and AdaBoost is improved by 4% to 11 %, compared to the case outliers-contaminated data is directly used. Compared with the case of contaminated data processed by DAGMM, OC-SVM, and IF, the improvement is around 2% to 5%, regardless of the base classifier used. Besides the proposal of S-DAGMM, this paper also proposed to use DAGMM to pretrain the DNN classifier (DAGMM-DNN). To prove the effectiveness of the estimation network in the structure of DAGMM, we conducted a study by comparing the diagnostic performance of DAGMM-DNN with SAE-DNN in terms of diagnostic accuracy rate, F1-score, and confusion matrix. The DNN classifier without using pretraining and the base classifiers such as KNN, SVM, RF, and AdaBoost are also the objects of this comparison. The experimental results have shown that the diagnostic accuracy performance of the DAGMM-DNN classifier is always superior to SAE-DNN with about 1% to 5% higher accuracy rate. In case the size of training labeled data is small (i.e., 207 samples) DAGMM-DNN achieves the best F1-score for all categories as compared with SAE-DNN, and DNN. Especially for system-level faults such as Refrigerant leak (RefLeak), Refrigerant overcharge (RefOver), and Excess oil (ExcsOil), the difference in F1-score of DAGMM-DNN and DNN is even more significant (at about 13% to 25%). In summary, the proposed methods have more advantages in the practical use as the chiller data are easily contaminated by outliers and the cost to obtain labeled fault data is expensive. The general ability of the proposed methods is guaranteed by several factors. First, DAGMM and S-DAGMM are deep neural networks that work on the platform of Gaussian Mixture Mode (GMM) whose application is very diverse and flexible. Second, the experimental RP-1043 dataset is one of the most common chiller datasets used in chiller FDD studies, in which both fault-free and various fault data are recorded at ten-second intervals on the 90-ton chiller. This chiller is a representative of chillers used in larger installations. Third, all chiller faults were introduced at four severity levels (SL1-SL4, from slightest to severest). The data were collected not only when the system has reached steady states, but also at transient states. Such a dataset ensures that the proposed methods have been comprehensively tested under different operating conditions and system states. Future works could be devoted to the compression network of the DAGMM model. Instead of using SAE as the compression network of DAGMM, some autoencoder variants, for example, a variational autoencoder (VAE), can be used instead to better extract high-level features for the estimation network. To ensure the reproducibility of the results by the research community and a potential future improvement of our framework by other researchers the complete source code is provided in the following repository: https://github.com/viettra-xai/S-DAGMM.

Chapter 4

Unsupervised Outlier Detection using Neural Network-based Mixtures of Probabilistic Principal Component Analyzers for Building Chiller Fault Diagnosis

4.1 Introduction

Heating, ventilation, and air conditioning (HVAC) systems are indispensable in commercial buildings nowadays to ensure thermal and air comfort for occupants Mirnaghi Haghighat (2020); Yan, Huang, et al. (2020); Yan, Su, et al. (2020). Contrary to the benefits that these systems bring is the concern about large energy consumption. Although an HVAC system is just one of many systems and utilities in a building, the energy consumed by such systems accounts for 33% of the total energy consumption of commercial buildings in the USA B. Li et al. (2021). In tropical countries with hot and humid weather, i.e., Singapore, this number can be up to 60% or more Yan, Chong, Mo (2020). Among the important components of an HVAC system, the chiller is the main energy consumer, which exhausts 35-40% of the total energy supplied to the building M. C. Comstock et al. (2002). Besides the essential energy consumed for chillers to regulate the temperature in building spaces, large amounts of energy may leak due to improper maintenance, underperformed components, and installation failure. If this situation persists, chiller faults will increase, causing chiller gradual degradation and eventual failure. Such corruption should obviously cause thermal discomfort as well as energy waste. Therefore, an accurate fault detection/diagnosis (FDD) model that can detect and diagnose incipient chiller faults at an early stage is very crucial.

Besides sufficient data requirement, the diagnostic reliability of data-driven FDD methods for HVAC systems also depends on data purity. This means that historical training data should not be contaminated by noise instances (i.e., outliers) which are generally generated during erroneous recording process or faulty measurements. Over time, sensors in HVAC systems, especially chiller systems, may experience issues like noise, drift, or malfunctions, leading to inaccurate readings and false alarms. Since these systems rely on numerous sensors to monitor parameters such as temperature, humidity, and pressure, the emergence of outliers in the collected data becomes inevitable. Since outlying instances have an adverse impact on FDD model's robustness, they need to be detected and removed during preprocessing steps by outlier detection (OD) algorithms before constructing FDD model. Similar to an anomaly, an outlier is referred to an instance located in low probability regions of data distribution. Therefore, outliers contaminated in regular data can be precisely detected by anomaly detection (AD) algorithms. In other words, AD algorithms can be well used for OD applications.

Along with the proliferation of probabilistic models, density estimation-based methods have been becoming mainstream in AD applications Su et al. (2019); Zong et al. (2018). Such methods predict anomalies through estimating density of data instances with respect to the normal data distribution. Most of density estimation-based methods can be divided into two groups: classic generative models and deep neural generative models. Among classic generative models, KDE Härdle (1990), histogram estimator Izenman (1991), and GMM Bishop (1994); Roberts Tarassenko (1994) are the most famous. For high-dimensional problems, deep neural generative models are more favored than classic generative models. Some methods such as VAEs An Cho (2015); Xu et al. (2018) and GANs Sabokrou, Khalooei, Fathy, Adeli (2018); Schlegl, Seeböck, Waldstein, Schmidt-Erfurth, Langs (2017) have been widely adopted to AD applications recently. The common point of these models is to map latent codes sampled from a source distribution (i.e., Gaussian distribution) to the actual input distribution via a neural network.

While a lot of studies have been done on probabilistic models such as GMM, KDE, VAE, etc., for AD applications, the potential of probabilistic PCA (PPCA) and a mixture of probabilistic principal component analyzers (MPPCA) Tipping Bishop (1999) for anomaly detection has been overlooked over years. PPCA is the probabilistic adaption of PCA that is extended for probabilistic interpretation. Standard PCA-based models typically use reconstruction loss as an anomaly indicator, while PPCAs have the flexibility to use either reconstruction loss or reconstruction probability for this purpose Ma et al. (2021). However, HVAC systems exhibit nonlinear behavior due to factors like fluctuating load conditions, changing weather, and system degradation over time, leading to nonlinear and dynamic recorded data. As a result, PCA-based or PPCA-based models may struggle to accurately capture these complexities. In such cases, MPPCA emerges as a strong candidate, offering the potential to effectively model the intricate patterns present in HVAC data. Similar to GMM with a component mixture mechanism, MPPCA combines single probabilistic PCAs in order to model complex data structures. Some studies have applied a mixture of principal component analyzers for FDD applications in the recent time. Zhang et al. J. Zhang, Chen, Hong (2021), for example, used MPPCA to exploit complex data patterns from monitoring data. Such features are then utilized for fault detection in industrial processes. In Sharifi Langari (2017), Sharifi et al. proposed a MPPCA-based sensor fault diagnosis model for HVAC systems. The model isolates the data space into locally linear regions that each region is associated with an individual probabilistic PCA. In another study J. Zhang, Chen, Chen, Hong (2017), Zhang used the technique of MPPCA to model underlying nonlinear process with local PPCA models. However, to my best knowledge, only very few studies have applied MPPCA for anomaly detection Ma, Luo, Yun, Wan, Shen (2023); Yairi et al. (2017). In these studies, MPPCA models the data distribution and estimates the anomaly score of instances based on their reconstruction probability.

However, the limitation of MPPCA-based methods is training difficulty. Similar to GMM, MPPCA is trained using the notable Expectation-Maximization (EM) algorithm. During training process, the inverse of component covariance matrices needs to be calculated after each iteration. After several iterations, however, covariance matrices become irreversible due to the singularity problem leading to training interruption. This problem occurs more often with high-dimensional observed data because the size of the covariance matrices is proportional to the observation dimension. To overcome this limitation, we propose to use a neural network to model MPPCA framework (NN-MPPCA), in which NN-MPPCA's parameters are optimized via the back-propagation algorithm. The loss function to train MPPCA-NN is the combination of the reconstruction error and the minus reconstruction probability of training instances. Besides, we also include a penalty component into the loss function to penalize small values on the diagonal entries of covariance matrices. This minimizes the singularity problem when calculating the inverse of covariance matrices. While the stopping criterion of a standard MPPCA is as its loading matrices and noise variances do not change anymore (i.e., MPPCA's parameters converge), the training process of NN-MPPCA stops when the reconstruction error and the minus reconstruction probability of training instances cannot be minimized further. This stopping criterion guarantees that NN-MPPCA fits well the training data distribution at the end of training process. After data modeling process, the trained NN-MPPCA is exploited for anomaly prediction.

In summary, the main contributions of the paper are listed as follows:

(1) To overcome the training hardship of regular MPPCA models, the paper has proposed a novel approach (i.e., NN-MPPCA) that uses a multilayer neural network (MLN) to model a mixture of probabilistic principal component analyzers (MPPCA). All the parameters of NN-MPPCA are updated via back-propagation algorithm.

(2) The loss function of NN-MPPCA is thoroughly defined so that its minimization guarantees NN-MPPCA to converge to the optimum. Especially, a penalty component is included into the loss function to minimize the singularity problem when calculating the inverse of covariance matrices during training process.

(3) The proposed method is thoroughly validated on a 90-ton centrifugal water-cooled chiller data. To prove the superiority of the proposed method over OD state-of-the-art algorithms, a comprehensive set of experimental tests has been conducted in this study.

The paper is organized as follows: In Section 4.2, we first theoretically describe MPPCA and its EM algorithm. Then, we introduce the proposed NN-MPPCA model and its training process. The way to adopt NN-MPPCA model for AD and OD applications is also explained in this section. The experimental chiller dataset RP-1043 is presented in Section 4.3, followed by a detailed methodology to generate artificial local outliers and global outliers from regular chiller data for OD validation. In Section 4.4, comprehensive experiments are presented, along with an explanation and discussion of experimental results. Finally, conclusions are summarized in Section 4.5.

4.2 Methodologies

In this section, we first theoretically describe MPPCA and how MPPCA's parameters are updated by the EM algorithm. Details about MPPCA are presented in the original paper of Tipping and Bishop Tipping Bishop (1999). Following, we introduce the proposed neural network-based MPPCA (NN-MPPCA) in which MPPCA framework is modeled by a multilayer neural network called the estimation network. Different from a standard MPPCA, NN-MPPCA's parameters are updated via the network's training process.

4.2.1 A mixture of probabilistic principal component analyzers

A mixture of probabilistic principal component analyzers (MPPCA) is the combination of single probabilistic PCAs. This combining mechanism permits MPPCA to model complex data structures. For a set of observed d-dimensional data vectors $\{\mathbf{x}_n\}$, $n \in \{1, \dots, N\}$, the completed data log-likelihood of a mixture model is:

$$\mathcal{L}_C = \sum_{n=1}^{N}\sum_{k=1}^{M} r_{nk}\log\{\pi_k p(\mathbf{x}_n, \mathbf{z}_{nk})\} \tag{4.1}$$

where π_k is the corresponding mixing proportion (i.e., $\pi_k \geq 0$ and $\sum_{k=1}^{K}\pi_k = 1$), and $\mathbf{z}_{nk}$ is a q-dimensional latent variable of $\mathbf{x}_n$ for each model component k. Variables r_{nk} define which model is responsible for generating an observed point $\mathbf{x}_n$. Note that for MPPCA framework, mean vector $\boldsymbol{\mu}_k$, factor loading matrix $\mathbf{W}_k$, and noise variance $\boldsymbol{\sigma}_k^2$ are now associated with each of the M mixture components, and can be optimized via an iterative EM algorithm with respect to the completed data log-likelihood in Eq. (4.1).

An iterative EM algorithm initiates with the E-step, which involves computing the posterior probabilities R_{nk} of r_{nk} (referred to as a responsibility value), along with the expected values $\langle\mathbf{z}_{nk}\rangle$ and $\langle\mathbf{z}_{nk}\mathbf{z}_{nk}^T\rangle$ of latent variables, based on previous values of the parameters π_k, $\boldsymbol{\mu}_k$, $\mathbf{W}_k$, and $\boldsymbol{\sigma}_k^2$:

$$R_{nk} = p(k|\mathbf{x}_n) = \frac{p(\mathbf{x}_n|k)\pi_k}{p(\mathbf{x_n})} \tag{4.2}$$

in which $p(\mathbf{x}_n|k)$ is the likelihood of a single PPCA (i.e., a mixture component) and given by:

$$p(\mathbf{x}_n|k) = (2\pi)^{-d/2}|\mathbf{C}_k|^{-1/2}\exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T\mathbf{C}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)\right) \tag{4.3}$$

where $\mathbf{C}_k = \boldsymbol{\sigma}_k^2\mathbf{I} + \mathbf{W}_k\mathbf{W}_k^T$ is the model covariance of the k-th mixture component.

$$\langle \mathbf{z}_{nk} \rangle = \mathbf{M}_k^{-1}\mathbf{W}_k^T(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{4.4}$$

$$\langle \mathbf{z}_{nk}\mathbf{z}_{nk}^T \rangle = \sigma_k^2\mathbf{M}_k^{-1} + \langle \mathbf{z}_{nk} \rangle \langle \mathbf{z}_{nk} \rangle^T \tag{4.5}$$

where $\mathbf{M}_k = \sigma_k^2\mathbf{I} + \mathbf{W}_k^T\mathbf{W}_k$. Note that $\mathbf{C}_k$ is $d \times d$ and $\mathbf{M}_k$ is $q \times q$.

To simplify the M-step equations of MPPCA, a two-stage EM procedure is adopted. In the first stage, we ignore the presence of latent variables. A new form of the complete-data log-likelihood function is:

$$\hat{\mathcal{L}}_C = \sum_{n=1}^{N}\sum_{k=1}^{M} R_{nk}\log\{\pi_k p(\mathbf{x}_n|k)\} \tag{4.6}$$

Maximization of Eq. (4.6) with respect to π_k and $\boldsymbol{\mu}_k$ gives the M-step equations as follows:

$$\tilde{\pi}_k = \frac{1}{N}\sum_{n=1}^{N} R_{nk} \tag{4.7}$$

$$\tilde{\boldsymbol{\mu}}_k = \frac{\sum_{n=1}^{N} R_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} R_{nk}} \tag{4.8}$$

In the second stage, we reintroduce the missing variables $\mathbf{z}_{nk}$, the expected complete-data log-likelihood now takes the form:

$$\begin{aligned}\langle \mathcal{L}_C \rangle = \sum_{n=1}^{N}\sum_{k=1}^{M} R_{nk} \Bigg\{ \ln\tilde{\pi}_k - \frac{d}{2}\ln\boldsymbol{\sigma}_k^2 - \frac{1}{2}\mathrm{tr}(\langle \mathbf{z}_{nk}\mathbf{z}_{nk}^T \rangle) - \frac{1}{2\boldsymbol{\sigma}_k^2}\|\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_k\|^2 \\ + \frac{1}{\boldsymbol{\sigma}_k^2}\langle \mathbf{z}_{nk} \rangle^T\mathbf{W}_k^T(\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_k) - \frac{1}{2\boldsymbol{\sigma}_k^2}\mathrm{tr}(\mathbf{W}_k\mathbf{W}_k^T\langle \mathbf{z}_{nk}\mathbf{z}_{nk}^T \rangle)\Bigg\}\end{aligned} \tag{4.9}$$

Maximizing Eq.(4.9) with respect to $\mathbf{W}_k$ and $\boldsymbol{\sigma}_k^2$ (keeping $\tilde{\boldsymbol{\mu}}_k$ fixed), we obtain the much simplified M-step equations:

$$\tilde{\mathbf{W}}_k = \mathbf{S}_k\mathbf{W}_k\left(\boldsymbol{\sigma}_k\mathbf{I} - \mathbf{M}_k^{-1}\mathbf{W}_k^T\mathbf{S}_k\mathbf{W}_k\right)^{-1} \tag{4.10}$$

$$\tilde{\boldsymbol{\sigma}}_k^2 = \frac{1}{d}\text{tr}\left(\mathbf{S}_k - \mathbf{S}_k\mathbf{W}_k\mathbf{M}_k^{-1}\tilde{\mathbf{W}}_k^T\right) \tag{4.11}$$

where $\mathbf{S}_k$ is given by:

$$\mathbf{S}_k = \frac{1}{\tilde{\pi}_k N}\sum_{n=1}^{N} R_{nk}(\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_k)(\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_k)^T \tag{4.12}$$

Iteration of equations (4.2), (4.7), and (4.8) followed by equations (4.10) and (4.11) in sequence guarantees that a local maximum of the likelihood can be found.

4.2.2 Neural network-based mixture of probabilistic principal component analyzers

In this study, we used an estimation network (i.e., a multilayer perceptron) to model the MPPCA framework, which parameters are updated via the estimation network's training process. While the estimation network's input size depends on the dimension of observation data, the output size reflects the number of MPPCA's components. For an MPPCA with M components, the outputs of the estimation network are M-dimensional responsibility vectors $\mathbf{R}_n$. Since $\sum_{k=1}^{M} R_{nk} = 1$, the responsibility variables R_{nk} are calculated by softmaxing the output vector $\mathbf{p}_n$. The process to obtain responsibility vectors are formulated as follows:

$$\mathbf{p}_n = \text{MLP}(\mathbf{x}_n, \theta) \tag{4.13}$$

$$\mathbf{R}_n = \text{softmax}(\mathbf{p}_n) \tag{4.14}$$

where $\mathbf{x}_n$, $\mathbf{p}_n$, and θ are the observed input, output, and trainable parameters of the estimation network. The overview of NN-MPPCA is shown in Fig. 4.1.

The learning of a neural network-based MPPCA (NN-MPPCA) model is executed via training process with N-samples batches. At the early training stage, the parameters of NN-MPPCA need to be initialized. Normally, the parameters of a standard MPPCA can be randomly initialized. However, this setting often causes the model to be stuck in local optima that are inferior to the global ones. Instead, k-means clustering is applied in the initialization step in which each sample is temporarily assigned to one of the clusters. This assignment determines the initial responsibility values R_{nk} of the observation data $\mathbf{x}_n$. In our proposed

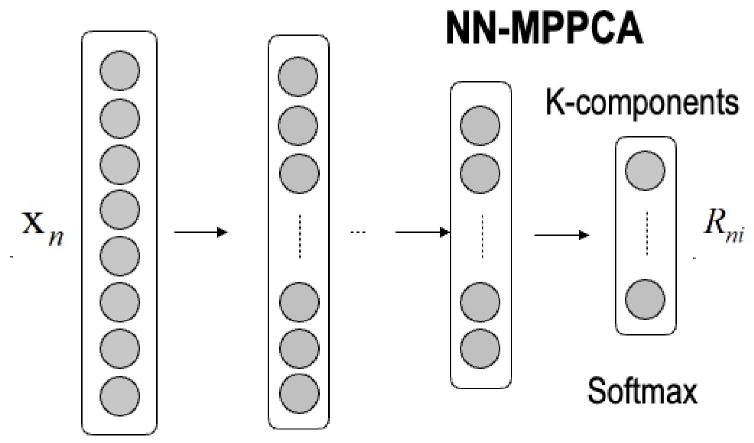

Figure 4.1: The pictorial diagram of NN-MPPCA.

framework, R_{nk} is initialized by the estimation network's output layer using (4.14). From initial R_{nk}, the k-th mixture component's weight π_k, mean $\boldsymbol{\mu}_k$ are accordingly initialized using (4.7), and (4.8).

To initialize the noise variance $\boldsymbol{\sigma}_i^2$ and loading factor matrix $\mathbf{W}_i$, we need to calculate the component covariance matrix $\mathbf{S}_i$ using (4.12). For $d \times d$ covariance matrix $\mathbf{S}_k$, its eigenvalues and eigenvectors are $\{\lambda_{k1}, \lambda_{k2}, \dots, \lambda_{kd}\}$, and $\{\mathbf{u}_{k1}, \mathbf{u}_{k2}, \dots, \mathbf{u}_{kd}\}$, respectively. If q is the number of principal components that we want to retain, the initial value of $\boldsymbol{\sigma}_k^2$ and $\mathbf{W}_k$ are estimated as:

$$\boldsymbol{\sigma}_k^2 = \frac{1}{d-q} \sum_{j=q+1}^{d} \lambda_{kj} \tag{4.15}$$

$$\mathbf{W}_k = \mathbf{U}_{k,q} \boldsymbol{\Lambda}_{k,q}^{1/2} (\mathbf{I}_q - \boldsymbol{\sigma}_k^2 \boldsymbol{\Lambda}_{k,q}^{-1}) \tag{4.16}$$

where $\mathbf{U}_{k,q}$ and $\boldsymbol{\Lambda}_{k,q}$ are defined as:

$$\mathbf{U}_{k,q} = [\mathbf{u}_{k1}, \mathbf{u}_{k2}, \dots, \mathbf{u}_{kq}] \tag{4.17}$$

$$\boldsymbol{\Lambda}_{k,q} = \mathrm{diag}(\lambda_{k1}, \lambda_{k2}, \dots, \lambda_{kq}) \tag{4.18}$$

After parameters R_{nk}, π_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\sigma}_k^2$, and $\mathbf{W}_k$ are initialized, their new values begin to be updated through minimizing the loss function of the NN-MPPCA model. For an N-samples batch, the total loss function of the model is defined as follows:

$$\mathcal{L}(\theta) = \frac{\eta_1}{N}\sum_{n=1}^{N} a(\mathbf{x}_n) + \frac{\eta_2}{N}\sum_{n=1}^{N} L(\mathbf{x}_n, \mathbf{x'}_n) + \eta_3 P(\mathbf{M}) \tag{4.19}$$

The objective function includes three components:

(1) $a(\mathbf{x}_n)$ indicates minus reconstruction probability of an observation $\mathbf{x}_n$ by NN-MPPCA which is its minus log-likelihood:

$$\begin{aligned} a(\mathbf{x}_n) = -\log p(\mathbf{x}_n) = -\log\left(\sum_{k=1}^{M} \pi_k p(\mathbf{x}_n|k)\right) \\ -\log\left(\sum_{k=1}^{M} \pi_k (2\pi)^{-d/2} |\mathbf{C}_k|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \mathbf{C}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)\right)\right) \end{aligned} \tag{4.20}$$

Note that the model covariance $\mathbf{C}_k = \boldsymbol{\sigma}_k^2\mathbf{I} + \mathbf{W}_k\mathbf{W}_k^T$ is calculated from current values of $\boldsymbol{\sigma}_k$ and $\mathbf{W}_k$.

(2) $L(\mathbf{x}_n, \mathbf{x'}_n)$ indicates the reconstruction error of an observation $\mathbf{x}_n$, which is the difference between the observation and its reconstruction via MPPCA. For an observed sample $\mathbf{x}_n$, $L(\mathbf{x}_n, \mathbf{x'}_n)$ is calculated as follows:

$$\mathbf{x}'_{nk} = \mathbf{W}_k(\mathbf{W}_k^T\mathbf{W}_k)^{-1}\mathbf{M}_k\langle \mathbf{z}_{nk}\rangle = \mathbf{W}_k(\mathbf{W}_k^T\mathbf{W}_k)^{-1}\mathbf{W}_k^T(\mathbf{x}_n - \boldsymbol{\mu}_k) + \boldsymbol{\mu}_k \tag{4.21}$$

$$\mathbf{x}'_n = R_{nk}\mathbf{x}'_{nk} \tag{4.22}$$

$$L(\mathbf{x}_n, \mathbf{x}'_n) = |\mathbf{x}_n - \mathbf{x}'_n| \tag{4.23}$$

where $\langle \mathbf{z}_{nk}\rangle$ is the expectation of the latent variable regarding the kth component.

(3) $P(\mathbf{M}) = \sum_{k=1}^{M}\sum_{j=1}^{d}\frac{1}{M_{kj}}$ is the component utilized to penalize small values on diagonal entries. This penalty helps to minimize the singularity problem when calculating the inverse of matrix $\mathbf{M}$. Actually, when calculating the loss function of the model, we need to calculate the inverse of $d \times d$ model covariance matrix $\mathbf{C}_k^{-1}$. However, we can express $\mathbf{C}_k^{-1}$ as $(\mathbf{I} - \mathbf{W}_k\mathbf{M}_k^{-1}\mathbf{W}_k^T)/\boldsymbol{\sigma}_k^2$. As a result, we can reduce computational operation from $O(d^3)$ to $O(q^3)$ by calculating $q \times q$ matrix $\mathbf{M}_k^{-1}$.

Three loss components are combined using heuristic coefficients η_1, η_2, η_3 to produce the total loss value that is used to update weights of the estimation network via back-propagation.

During minimizing the loss function, the estimation network directly updates the responsibility variables R_{ni} at the output layer. After each training epoch, new values of the other parameters such as π_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\sigma}_k^2$ and $\mathbf{W}_k$ are adjusted from the updated R_{nk} using (4.7), and (4.8), followed by (4.10), and (4.11). These updated parameters are then used to calculate the loss value for the next training epoch. Iterative training epochs guarantee NN-MPPCA to converge to the optimum.

4.2.3 Anomaly detection

The trained NN-MPPCA is now able to determine whether an observation $\mathbf{x}_n$ is anomalous or not by calculating its minus reconstruction probability $a(\mathbf{x}_n)$. Denote an anomaly score as a_n. The way to calculate an anomaly score is expressed in Eq. (4.20). If the observed sample $\mathbf{x}_n$ follows the normal patterns of data distribution, it can be reconstructed with high confidence and its anomaly score a_n is low. On the other hand, if the observed sample is less likely able to be reconstructed, it is more likely to be anomalous and its anomaly score a_n is high. Formally, if a_n is higher than an anomaly threshold, then $\mathbf{x}_n$ is marked as anomalous; otherwise $\mathbf{x}_n$ is normal. The steps of the NN-MPPCA-based AD model are illustrated in Fig. 4.2.

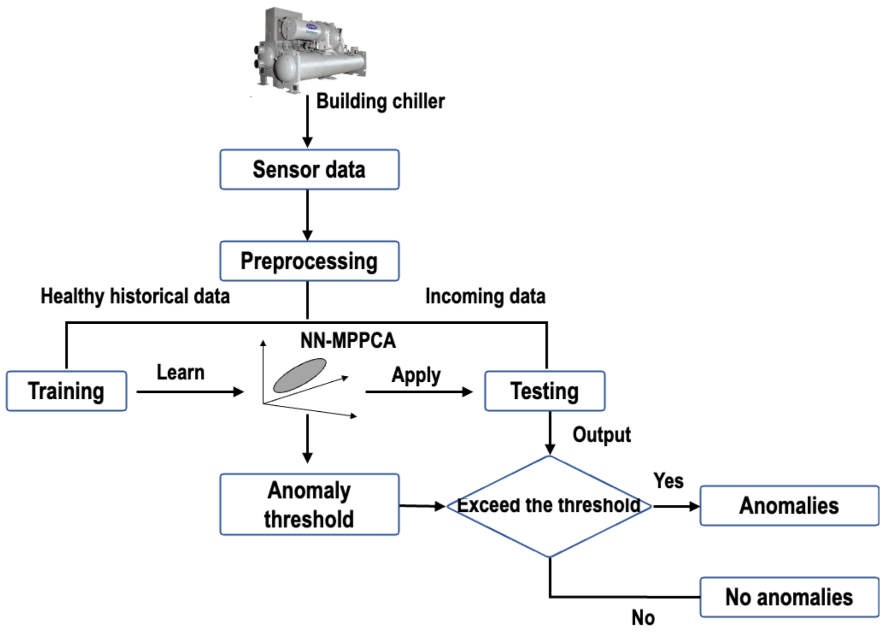

Figure 4.2: Flowchart of NN-MPPCA anomaly detection model.

4.3 Chiller Datasets and Synthetic Outlier Generation

In this study, we continue to employ the ASHRAE RP-1043 chiller dataset as a critical resource for validating the performance of our proposed models. This dataset has proven instrumental in previous research for benchmarking model efficacy under varied operational conditions. Its comprehensive coverage of typical chiller faults and operational data points provides a robust basis for assessing the effectiveness and reliability of fault detection and diagnostic models in real-world scenarios.

To benchmark outlier detection methods, both regular instances and outliers are necessary. In the RP-1043 dataset, however, outliers are not available. Hence, we need to generate artificial outliers from regular instances available in the RP-1043 dataset. For general applicability validation, both local outliers and global outliers are artificially generated and used in this study. While local outliers are noise instances that are outlying with respect to their local neighborhoods, global outliers often scatter across the whole regular instance space. The location of local outliers and global outliers with respect to regular data are illustrated in Fig. 4.3.

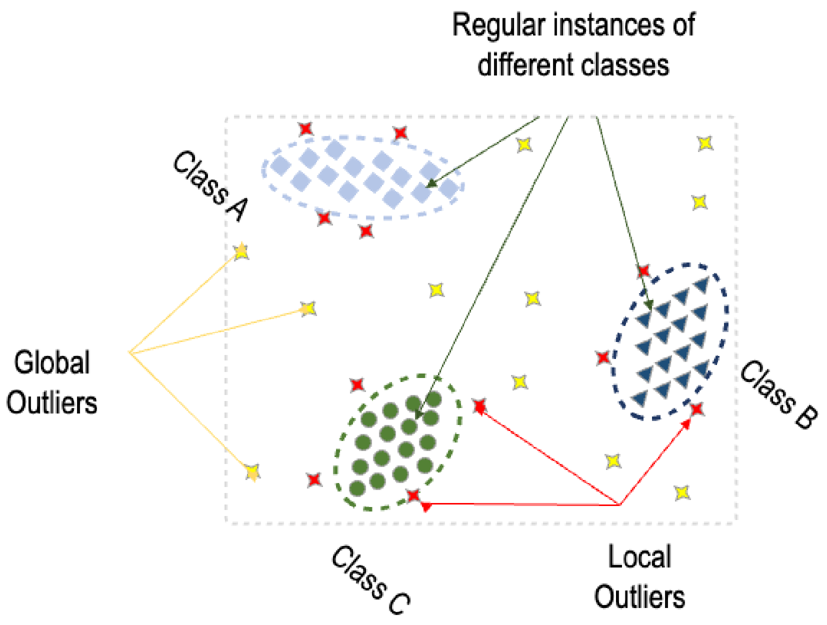

Figure 4.3: Local outliers and global outliers definition.

For local outlier generation, we apply the study in Milligan (1985), in which a GMM model is first utilized to fit regular data distribution (i.e., regular chiller data). This step returns GMM parameters such as the number of components K, mixing proportion π_k, the mean vectors $\boldsymbol{\mu}_k$, and the covariance matrices

$\mathbf{\Sigma}_k$, where $k \in \{1, \ldots, K\}$. From the characteristics of local outliers, the local outlier distribution should have the same shape as the regular data distribution but the boundary of outlier clusters is larger than that of regular data clusters. This means GMM parameters of the two distributions are the same, except the covariance matrices of local outliers are scaled with $\alpha > 1$. This guarantees that generated outliers are outlying but still close to regular instances of the same GMM component. Suggested by the study Steinbuss Böhm (2021), we set $\alpha = 5$ so that generated outliers are far enough away from regular instances but still close to regular instances of the corresponding clusters. This is an important characteristic since our aim is to generate local outliers.

In contrast to local outliers, global outliers often scatter in a wide range around the regular instance space; therefore, they should be sampled from a uniform distribution Iglesias, Zseby, Ferreira, Zimek (2019); Pei Zaïane (2006). For our dataset, the global outlier distribution is the multivariate uniform distribution, which boundary planes are determined just by computing the minimum and maximum of each attribute of the regular dataset (i.e., 64 attributes), respectively. However, these boundary values are increased by 5% to guarantee that generated instances are global outliers. In this case, the global outlier distribution looks like a high-dimensional rectangular box covering the regular data space.

Since the distributions of local outliers and global outliers all cover the regular data space, instances sampled from these distributions are not only outliers but also regular instances. Therefore, we should separate artificial outliers from artificial regular instances to create an outliers set for validating experiments. To filter artificial outliers from artificially generated instances, we used the K-nearest neighbors (KNN) classifier.

4.4 Experimental Results

4.4.1 NN-MPPCA Configuration and Baseline Methods

For the RP-1043 dataset, we use an estimation network with three layers to model the MPPCA framework. We consider MPPCA with 3 mixture PPCA components for the best performance, so the output layer of the estimation network has 3 neurons. The input layer size is 65, which is equal to the number of attributes in each regular instance. The hidden layer size is optimized with 15 neurons. All NN-MPPCA layers are implemented by TensorFlow and the model is trained by the Nadam algorithm with a learning rate of 0.001. The number of epochs and the size of mini-batches are 200 and 4096, respectively. Moreover, we choose λ_1 as 1, λ_2 as 0.5, and λ_3 as 0.005. These parameters are chosen by heuristic searches. To optimize the model performance, SELU activation function and GlorotNormal kernel initialization are jointly used. Besides the early stopping technique, to avoid overfitting, we apply the dropout technique with a rate of 0.2 and L2 regularization with a regularization factor of 0.01 for all the layers.

To validate the effectiveness of NN-MPPCA, its outlier detection performance is compared to those of both cutting-edge and classic OD methods. OD baselines used in this study include:

- Isolation forest (IF) F. T. Liu, Ting, Zhou (2008): An ensemble of random trees where anomalies are isolated based on the partition of random features and their values.

- Local outlier factor (LOF) Breunig, Kriegel, Ng, Sander (2000): An unsupervised OD method in which the outlying score of an instance is proportional to its local density deviation with respect to nearest neighbors. Instances with local density significantly lower than those of their neighbors are considered as outliers.

- Elliptical envelope (ELE) Rousseeuw Driessen (1999): The algorithm forms an imaginary elliptical area around a given Gaussian distributed data and detects new instances as outliers based on their location with respect to the envelope.

- Mixture of probabilistic principal component analyzers-based OD method (MPPCA) Tipping Bishop (1999): The method uses MPPCA and the EM algorithm to fit given regular data. For a new instance, its minus reconstruction probability is regarded as an anomaly score.

- Deep autoencoding Gaussian mixture model (DAGMM) Zong et al. (2018): A cutting-edge method that jointly trains AE and GMM in an end-to-end way. The method uses reconstruction probability with respect to GMM as the outlier detection criterion.

For the OD baselines, the meta-parameters are optimized by exhaustive search in order to find their best performance.

4.4.2 Validation of NN-MPPCA Outlier Detection Performance

To validate the outlier detection performance of NN-MPPCA, both local outliers and global outliers are used. However, outlier types are separately used so that we can accurately assess the proposed model's

Table 4.1: Local outlier detection performance of NN-MPPCA and OD baseline methods

Method	**Dataset 1 (severity level 1)**			**Dataset 2 (severity level 2)**		
	80% regular instances + 20% local outliers			80% regular instances + 20% local outliers		
	Precision (%)	**Recall (%)**	**F1-score (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
IF	87.0 ± 0.9	87.1 ± 0.8	87.0 ± 0.8	88.8 ± 1.2	89.1 ± 1.6	88.9 ± 1.4
LOF	95.2 ± 0.6	95.2 ± 0.5	95.2 ± 0.5	92.3 ± 0.5	92.4 ± 0.3	92.3 ± 0.2
ELE	63.5 ± 3.7	71.4 ± 10.5	67.1 ± 6.8	62.8 ± 2.8	68.8 ± 8.9	65.5 ± 5.4
MPPCA	92.3 ± 0.9	91.7 ± 0.6	92.0 ± 0.6	93.3 ± 1.2	93.3 ± 0.8	93.3 ± 0.9
DAGMM	83.6 ± 0.7	84.1 ± 0.8	83.8 ± 0.7	83.3 ± 0.5	83.9 ± 0.7	83.6 ± 0.5
NN-MPPCA	$\mathbf{96.1 \pm 0.6}$	$\mathbf{95.9 \pm 0.6}$	$\mathbf{96.0 \pm 0.5}$	$\mathbf{96.9 \pm 0.9}$	$\mathbf{97.0 \pm 0.4}$	$\mathbf{96.9 \pm 0.3}$
Method	**Dataset 3 (severity level 3)**			**Dataset 4 (severity level 4)**		
	80% regular instances + 20% local outliers			80% regular instances + 20% local outliers		
	Precision (%)	**Recall (%)**	**F1-score (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
IF	86.7 ± 1.3	86.7 ± 1.1	86.7 ± 1.1	90.3 ± 0.9	90.2 ± 1.0	90.3 ± 0.9
LOF	94.1 ± 0.4	93.8 ± 0.4	94.0 ± 0.1	93.7 ± 0.5	93.4 ± 0.4	93.5 ± 0.2
ELE	62.5 ± 4.4	69.6 ± 12.9	65.6 ± 8.1	53.3 ± 4.0	45.9 ± 8.3	49.2 ± 6.3
MPPCA	93.3 ± 0.9	93.3 ± 0.8	93.3 ± 0.8	92.9 ± 0.6	92.7 ± 1.2	92.8 ± 0.8
DAGMM	84.3 ± 1.0	84.9 ± 0.9	84.6 ± 0.9	80.2 ± 5.6	80.7 ± 6.1	80.4 ± 5.8
NN-MPPCA	$\mathbf{96.2 \pm 0.8}$	$\mathbf{96.4 \pm 0.5}$	$\mathbf{96.3 \pm 0.5}$	$\mathbf{95.5 \pm 0.6}$	$\mathbf{95.3 \pm 0.5}$	$\mathbf{95.4 \pm 0.4}$

performance on each outlier type. With respect to local outliers, 4 datasets according to 4 different severity levels of chiller faults are used for validation. Each dataset includes regular instances (i.e., chiller data) of the corresponding severity level and local outlier instances generated from regular instances in a dataset. The number of outliers included accounts for 20% of the total instance number in a dataset. Similarly, 4 chiller datasets with global outliers are created. In total, 8 datasets are used to validate the outlier detection performance of NN-MPPCA and the OD baseline methods.

For imbalanced classification problems like outlier detection, F1-score is the most favored metric for validation. To compute F1-score, we need to calculate two metrics: Precision (P) and Recall (R). While Precision is the ratio of correctly predicted instances over predictions, Recall refers to the fraction of correctly predicted instances in true instances. Eq. (4.24) demonstrates the F1-score calculation. The high F1-score reflects the satisfying performance of OD models.

$$F1\text{-score} = \frac{2 \times P \times R}{P + R} \tag{4.24}$$

The OD performance of NN-MPPCA and the other rival methods on local outliers and global outliers is shown in Table 4.1 and Table 4.2, respectively. Each metric value is the mean value of 10 test runs using cross-validation and their variance. In each run, 50% of data are randomly sampled for training and the rest 50% reserved for testing. Note that only regular instances of training data are used to train models.

Table 4.2: Global outlier detection performance of NN-MPPCA and OD baseline methods

Method	**Dataset 5 (severity level 1)** 80% regular instances + 20% global outliers			**Dataset 6 (severity level 2)** 80% regular instances + 20% global outliers		
	Precision (%)	**Recall (%)**	**F1-score (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
IF	98.7 ± 0.7	98.8 ± 0.5	98.7 ± 0.4	98.8 ± 0.4	98.8 ± 0.7	98.8 ± 0.3
LOF	97.0 ± 0.4	96.3 ± 0.8	96.6 ± 0.3	94.6 ± 0.5	94.9 ± 0.4	94.7 ± 0.3
ELE	71.2 ± 0.5	$\mathbf{100.0 \pm 0.0}$	83.2 ± 0.4	71.5 ± 0.8	$\mathbf{100.0 \pm 0.0}$	83.4 ± 0.6
MPPCA	$\mathbf{99.6 \pm 0.6}$	99.7 ± 0.4	$\mathbf{99.6 \pm 0.3}$	$\mathbf{99.7 \pm 0.5}$	99.5 ± 0.5	$\mathbf{99.6 \pm 0.2}$
DAGMM	94.0 ± 0.4	94.2 ± 0.6	94.1 ± 0.2	93.6 ± 0.4	94.1 ± 1.0	93.8 ± 0.5
NN-MPPCA	99.5 ± 0.5	$\mathbf{99.8 \pm 0.3}$	$\mathbf{99.6 \pm 0.2}$	$\mathbf{99.8 \pm 0.3}$	99.4 ± 0.8	$\mathbf{99.6 \pm 0.3}$
Method	**Dataset 7 (severity level 3)** 80% regular instances + 20% global outliers			**Dataset 8 (severity level 4)** 80% regular instances + 20% global outliers		
	Precision (%)	**Recall (%)**	**F1-score (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
IF	98.9 ± 0.4	99.0 ± 0.5	98.9 ± 0.3	99.4 ± 0.8	99.4 ± 0.5	99.4 ± 0.2
LOF	98.4 ± 0.4	97.4 ± 0.7	97.9 ± 0.2	99.7 ± 0.4	99.4 ± 0.7	99.6 ± 0.3
ELE	71.0 ± 1.0	$\mathbf{100.0 \pm 0.0}$	83.0 ± 0.7	71.3 ± 0.5	$\mathbf{100.0 \pm 0.0}$	83.2 ± 0.3
MPPCA	$\mathbf{99.6 \pm 0.6}$	99.3 ± 0.8	99.5 ± 0.3	$\mathbf{99.8 \pm 0.3}$	99.0 ± 0.7	99.4 ± 0.2
DAGMM	92.5 ± 0.5	93.4 ± 0.7	92.9 ± 0.6	97.7 ± 1.9	98.3 ± 1.4	98.0 ± 1.6
NN-MPPCA	$\mathbf{100.0 \pm 0.1}$	99.3 ± 0.4	$\mathbf{99.7 \pm 0.2}$	$\mathbf{99.9 \pm 0.1}$	99.2 ± 0.7	$\mathbf{99.6 \pm 0.3}$

From Table 4.1 and Table 4.2, we can see that the severity level of regular data has no effect on the detection difficulty of outliers generated from respective regular data. In other words, the performance of OD methods on datasets of different severity levels (i.e., regular instances and outliers) differ only slightly. By comparing results in Table 4.1 with those in Table 4.2, it can be seen that global outliers are easier to detect than local ones. While global outliers can be correctly detected by most of the OD algorithms, a few of these algorithms like NN-MPPCA, and LOF can handle local outliers almost completely. The reason behind this is that global outliers scatter in a wide range around regular instances, so they are easy to detect. In contrast, local outliers often located near regular instances, so they are often misdetected as regular instances. Compared to the OD baseline methods, the local outlier detection performance of the proposed NN-MPPCA is highest with F1-scores achieving from 95.4% to 96.9% for datasets 1–4. In contrast, the performance of elliptical envelope (ELE) is lowest with F1-scores only ranging from 49.2% to 67.1%. The reason for this poor performance is that Gaussian distributions used by ELE cannot fit complex data space of chiller data. As a result, the decision elliptical boundary of ELE is incorrectly estimated leading to outlier misdetection. Compared to classic OD algorithms like ELE, IF, or MPPCA, the local outlier detection performance of LOF algorithm is higher with F1-scores ranging from 92.3% to 95.2%. The reason is that the LOF was originally designed to detect local outliers. Also, both LOF and the classifier used to filter outlier instances in Section 4.3 (i.e., KNN) are distance-based methods. Therefore, LOF can efficiently detect outliers filtered by the KNN classifier. DAGMM is a state-of-the-art deep OD method for multivariate data Zong et al. (2018). However, its local outlier detection performance is not so impressive with F1-scores just ranging from 80.4% to 84.6%. This may be because DAGMM is a deep model with many hyperparameters and optimizing the hyperparameters for each dataset (i.e., chiller dataset) is not an easy task. In this study, the DAGMM's hyperparameters are set according to the recommendation in the original paper Zong et al. (2018).

Both NN-MPPCA and MPPCA-based OD methods use a mixture of PPCA to model a complex data distribution and use instance reconstruction probability as an indicator to detect outliers. However, the performance of MPPCA is lower than that of NN-MPPCA with F1-scores ranging from 92.0% to 93.3%. MPPCA uses the EM algorithm to update parameters and the convergence sign is that parameters do not change further. In the case of NN-MPPCA, the parameters are updated via back-propagation so that the minus reconstruction probability and the reconstruction error of training instances decrease over training epochs. This decrease guarantees that the NN-MPPCA model can fit well regular data distribution so that any instance with low reconstruction probability can be considered as an outlier. Fig. 4.4 shows NN-MPPCA's component losses during the first 200 training epochs. From Fig. 4.4, it can be seen that the training process helps to reduce both minus reconstruction probability and reconstruction error of validation instances. Besides, the diagonal penalty curve almost remains unchanged after 40 epochs to ensure the stability of the training process. To avoid overfitting, the training process stops if the total loss value of NN-MPPCA on validation data does not reduce after waiting 20 epochs.

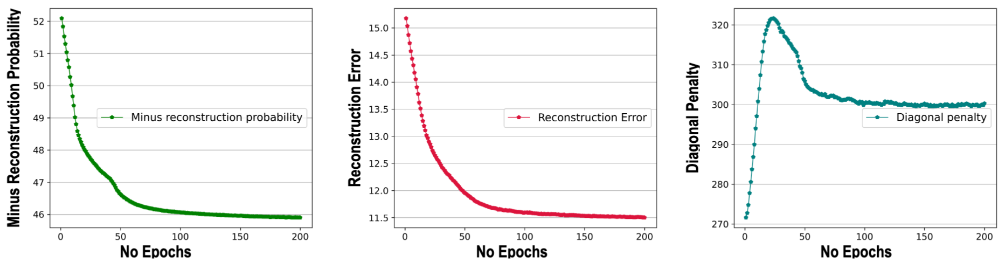

Figure 4.4: NN-MPPCA component losses on validation data during training.

Except for ELE with F1-scores of only about 83%, the global outlier detection performance of the other OD methods is relatively high and does not differ too much with F1-scores ranging from 92.29% to 99.6%. However, the performance of NN-MPPCA is still highest with F1-scores ranging from 99.6% to 99.7%. MPPCA has approximately the same performance as NN-MPPCA and higher than those of the other methods. In the case of ELE, the recall value is absolutely high, achieving almost 100%, but at the cost of low precision (i.e., around 71%). The reason behind this is that the elliptical boundary created by ELE strictly covers the regular data distribution. As a result, there are high chances that many regular instances are located outside the boundary and are misdetected as outliers (i.e., False Positive is high, and Precision is low). Meanwhile, outliers are all outside the boundary and are correctly detected (i.e., False Negative is low, and Recall is high).

4.4.3 Effectiveness of NN-MPPCA for FDD Application

To demonstrate the effectiveness of NN-MPPCA in particular and other OD methods in general for chiller FDD application, we set up a practical scenario in which the observed data are contaminated by both local outliers and global outliers. The outliers should deteriorate the FDD models' performance, and the use of OD algorithms in this case is essential. For experimental validation, 4 datasets containing both local outliers and global ones are used. Each corresponds to a respective severity level, and the ratio of outliers and total instances in each dataset is 20% (i.e., 10% local outliers and 10% global outliers). To demonstrate the necessity of a data cleaning step for data-driven FDD applications, we first directly use contaminated data (i.e., contain outliers) for FDD model training and testing. Then, FDD performance in this case is compared to that of using the data cleaning step (i.e., outliers are detected and removed by OD algorithms). The procedure to apply OD algorithms for data cleaning is illustrated in Fig. 4.5 and comparative results are shown in Table 4.3.

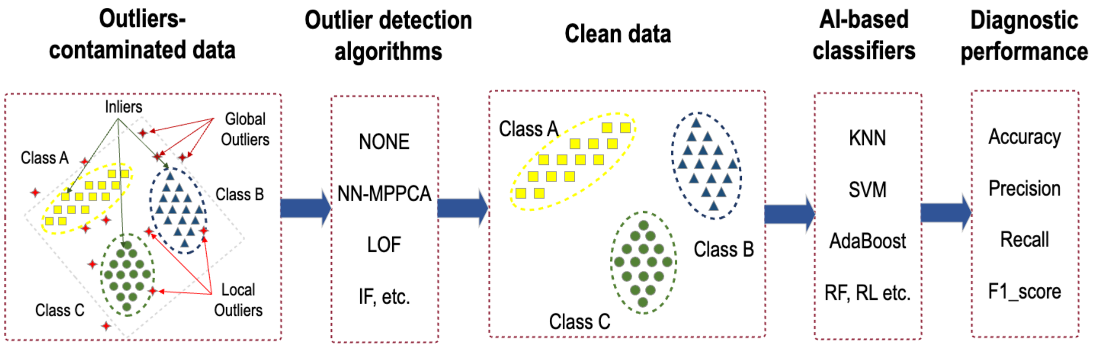

Figure 4.5: The procedure to apply data cleaning step for FDD applications.

To enforce the objectivity of the comparison, a wide range of common algorithms are chosen as base classifiers, including K-nearest neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF), Logistic Regression (LR) and Adaptive Boosting (AdaBoost). To find the best estimators and parameters for these base classifiers, the grid search technique with 3-fold cross-validation is applied. In the case of the

RF and AdaBoost classifiers, decision tree (DT) is used as a base estimator. Feature scaling is useful and applied for the KNN classifier. Experimental metric values in Table 4.3 are produced from 10 independent test runs using cross-validation to increase results reliability. Results in the first row of sub-tables in Table 4.3 correspond to the case of contaminated data (i.e., contain outliers) directly used for constructing and testing FDD models. The other rows contain testing accuracies of FDD models as these models use clean data for FDD model construction and validation. OD methods for data cleaning in this experiment are NN-MPPCA, LOF, and IF, respectively. From Table 4.3, we can see that the diagnostic accuracy of AI-based classifiers heavily depends on the ratio of outliers contaminated in regular data, regardless of severity level. The diagnostic accuracy of base classifiers only reaches up to 82.6%, precisely reflecting the proportion of outliers in the dataset (i.e., 20%). Also in Table 4.3, we can see that the diagnostic accuracy of FDD models is proportional to the performance of OD methods applied in the processing steps (i.e., the number of outliers removed). More specifically, the OD F1-scores of NN-MPPCA on 4 datasets reach 97.8%, higher than those of LOF and IF algorithms with maximum F1-scores of 96.1% and 93.9%, respectively. As a result, the diagnostic performance of the base classifiers on the data cleaned by NN-MPPCA reaches up to 99.5% (i.e., in the case of RF classifier), higher than those using LOF and IF for outlier detection with maximum diagnostic accuracy of 99.1% and 98.7%, respectively.

In addition to the accuracy metric, F1-scores for all categories are also analyzed to comprehensively evaluate the diagnostic performance of FDD models. In case the data are cleaned by NN-MPPCA, F1-score curves of the base classifiers are shown in Fig. 4.6. From Fig. 4.6, we can see that F-scores for all categories of some base classifiers mostly reached 100% (i.e., RF, SVM). This result proves that the application of NN-MPPCA significantly improves the diagnostic performance of FDD models. For all base classifiers, especially AdaBoost, Fig. 4.6 shows that F1-scores for normal and system-level fault categories (RefLeak, RefOver, ExcsOil) are lower than those of element-level fault categories (ReduCF, ReduEF, ConFoul, Non-Con). Regardless of the base classifiers, F1-scores for component-level faults are mostly higher than 99%, regardless of severity levels and the used base classifiers. In contrast, F1-scores for system-level faults and normal state are always lower than those of component-level faults. This difference is clear for the base classifiers such as KNN, SVM, Linear Regression, and especially AdaBoost. In the case of AdaBoost, for example, the F1-scores for normal state, RefLeak, RefOver, ExcsOil of severity level 1 and 2 are less than 60%, which are much lower than those for component-level faults. This means chiller system-level faults such as RefLeak, RefOver, ExcsOil are misdiagnosed with a higher probability than component-level faults.

Table 4.3: Testing diagnostic accuracy rates of different base classifiers in cases of using contaminated data and clean data.

OD Methods	Dataset 9 (severity level 1)					
	80% regular instances + 10% local outliers + 10% global outliers					
	OD F1-score	**KNN**	**SVM**	**RF**	**LR**	**AdaBoost**
None		80.9 ± 0.0	76.4 ± 0.2	82.5 ± 0.1	73.8 ± 0.1	50.9 ± 0.1
NN-MPPCA	**97.5 ± 0.2**	**97.4 ± 0.1**	**98.2 ± 0.1**	**99.5 ± 0.1**	**96.0 ± 0.1**	**57.7 ± 4.6**
LOF	96.1 ± 0.1	97.1 ± 0.1	97.8 ± 0.1	99.1 ± 0.1	95.8 ± 0.1	57.6 ± 5.3
IF	92.4 ± 0.9	96.1 ± 0.2	97.1 ± 0.2	98.3 ± 0.2	94.9 ± 0.2	55.6 ± 3.7
OD Methods	**Dataset 10 (severity level 2)**					
	80% regular instances + 10% local outliers + 10% global outliers					
	OD F1-score	**KNN**	**SVM**	**RF**	**LR**	**AdaBoost**
None		81.7 ± 0.1	79.0 ± 0.1	82.6 ± 0.1	76.9 ± 0.2	41.8 ± 0.1
NN-MPPCA	**97.8 ± 0.5**	**98.1 ± 0.2**	**99.0 ± 0.2**	**99.5 ± 0.2**	**97.9 ± 0.2**	**62.0 ± 3.3**
LOF	93.8 ± 0.2	97.5 ± 0.1	98.2 ± 0.1	98.7 ± 0.1	97.0 ± 0.1	63.4 ± 3.7
IF	92.8 ± 0.3	97.2 ± 0.1	98.0 ± 0.1	98.4 ± 0.1	96.9 ± 0.2	61.2 ± 3.5
OD Methods	**Dataset 11 (severity level 3)**					
	80% regular instances + 10% local outliers + 10% global outliers					
	OD F1-score	**KNN**	**SVM**	**RF**	**LR**	**AdaBoost**
None		82.0 ± 0.3	81.6 ± 0.3	82.6 ± 0.3	80.5 ± 0.3	41.8 ± 0.0
NN-MPPCA	**97.8 ± 0.3**	**98.7 ± 0.1**	**99.5 ± 0.0**	**99.5 ± 0.0**	**99.2 ± 0.2**	**85.2 ± 0.7**
LOF	95.4 ± 0.2	98.4 ± 0.1	99.1 ± 0.1	99.1 ± 0.1	98.7 ± 0.2	84.0 ± 0.7
IF	92.2 ± 0.4	97.5 ± 0.1	98.3 ± 0.1	98.3 ± 0.1	98.1 ± 0.2	83.9 ± 0.6
OD Methods	**Dataset 12 (severity level 4)**					
	80% regular instances + 10% local outliers + 10% global outliers					
	OD F1-score	**KNN**	**SVM**	**RF**	**LR**	**AdaBoost**
None		82.0 ± 0.1	81.5 ± 0.2	82.5 ± 0.1	81.2 ± 0.2	45.6 ± 5.0
NN-MPPCA	**97.0 ± 0.2**	**98.7 ± 0.2**	**99.3 ± 0.1**	**99.3 ± 0.1**	**99.3 ± 0.2**	**92.1 ± 0.4**
LOF	95.5 ± 0.2	98.3 ± 0.2	99.0 ± 0.1	98.9 ± 0.1	97.0 ± 0.1	**92.1 ± 0.5**
IF	93.9 ± 0.4	98.1 ± 0.1	98.7 ± 0.1	98.7 ± 0.1	98.1 ± 0.3	91.9 ± 0.3

For more detailed analysis and explanation of the above results, confusion matrices of the FDD models are also studied. Fig. 4.7 shows confusion matrices of KNN, SVM, LR, and RF models with respect to dataset 9 in which outliers are processed by NN-MPPCA. The reports with a great number of misclassified samples are marked with red dot circles. From Fig. 4.7, we can see that system-level faults (RefLeak, RefOver, ExcsOil) are misclassified at a great scale in comparison with component-level faults. In most of the cases, a system-level fault is misdiagnosed with the other system-level faults. For example, RefLeak is misclassified as RefOver or ExcsOil, and vice-versa. In addition, a high proportion of system-level fault instances is misclassified as normal-state instances. One reason is that system-level faults in this dataset are just in the incipient stage (i.e., severity level 1) causing FDD models to predict them as normal state. In contrast, component-level faults symptomize a certain location of the chiller, so these faults are highly distinct from each other and easy to be correctly classified.

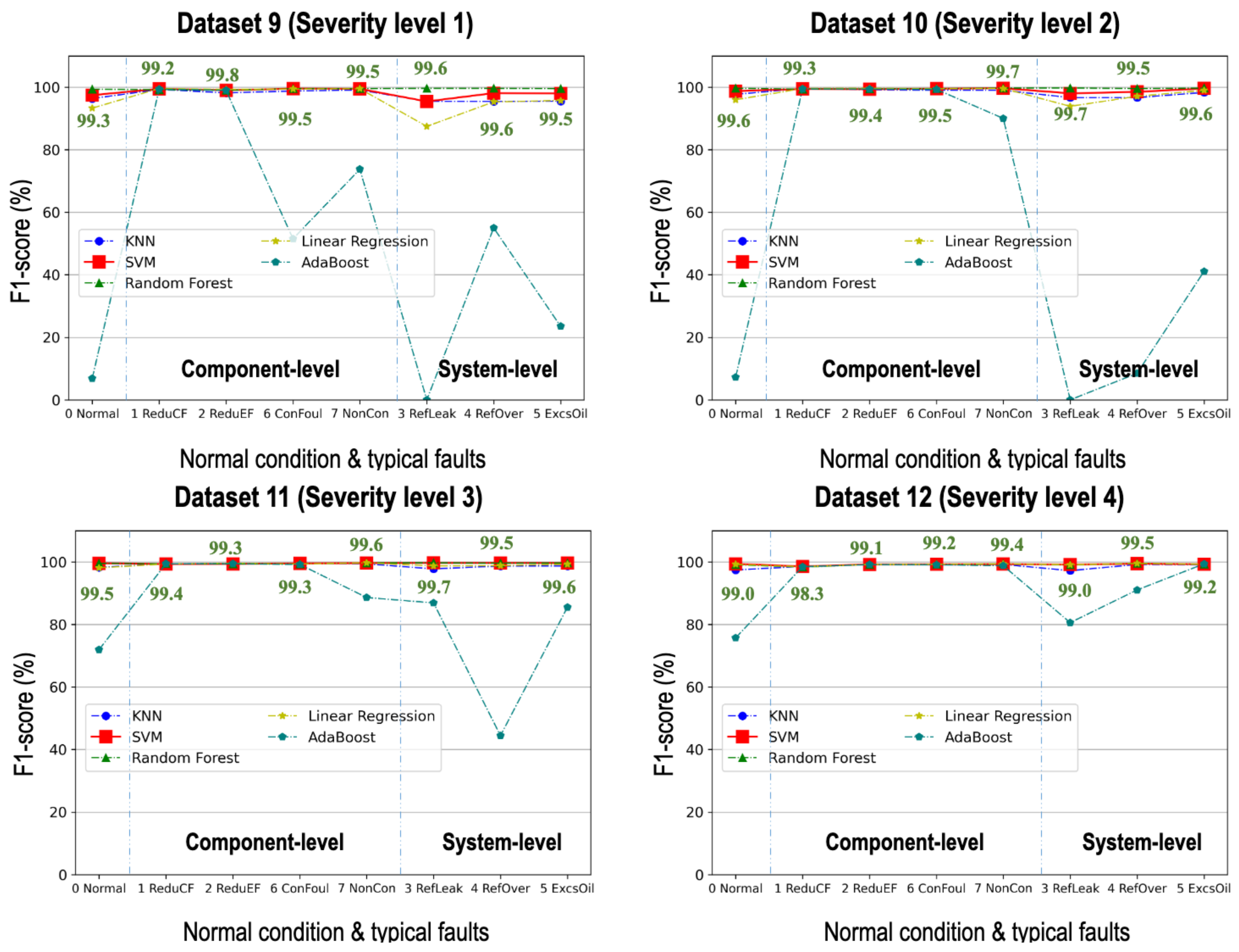

Figure 4.6: F1-score curves of the base classifiers on testing data of different datasets.

4.5 Conclusion

To purify outliers from contaminated data, the paper has proposed a novel OD method called NN-MPPCA in which MPPCA is modeled by an estimation network. Different from standard MPPCA, the parameters of NN-MPPCA are updated through back-propagation so that NN-MPPCA can well fit the complex distribution of chiller data. The definition of loss function and its minimization guarantee NN-MPPCA to converge to the optimum. Comprehensive experimental results have shown the superiority of NN-MPPCA to other OD baseline methods like IF, LOF, ELE, MPPCA, and DAGMM in terms of Recall, Precision, and F1-score metrics for both local and global outlier detection. In cases where the datasets contain 20% local outliers, the outlier detection performance of NN-MPPCA is highest with F1-scores up to 96.9%. The figures for the datasets that contain 20% global outliers are from 99.6% - 99.7%. While F1 scores of the best local outlier detection baseline (i.e., LOF) and the best global outlier detection baseline (i.e., MPPCA) are

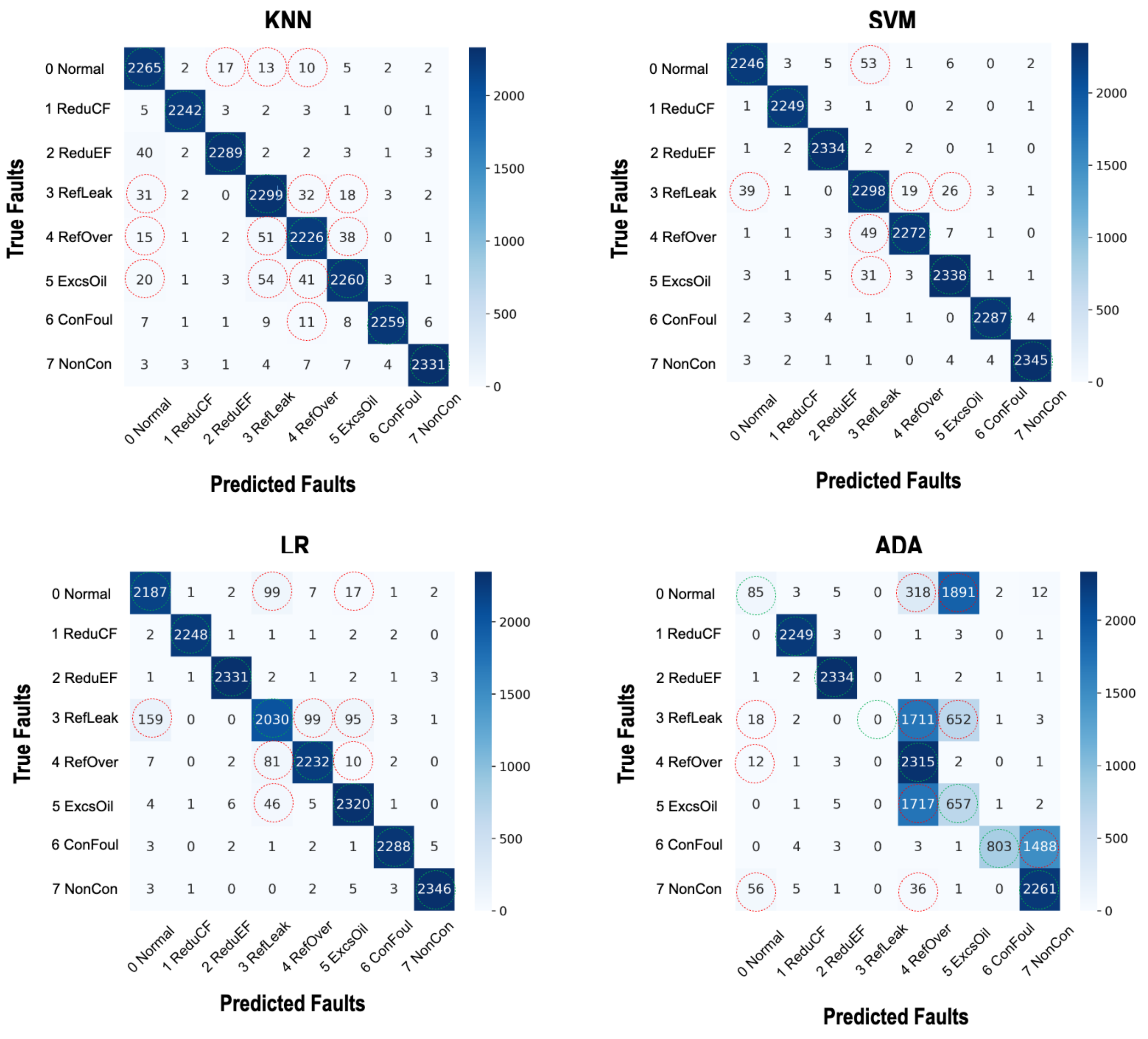

Figure 4.7: Confusion matrices of the FDD models on the dataset 9 (severity level 1).

only up to 95.2% and 99.6%, respectively.

The experimental results have also proved that NN-MPPCA is necessary for data processing before constructing FDD models. More specifically, with NN-MPPCA applied in the data purification step, the best FDD model (i.e., RF) can achieve the diagnostic accuracy up to 99.5%. While the best FDD model only reaches the diagnostic accuracy of 82.6% at most if contaminated data are directly used. The diagnostic accuracy improvement of nearly 20% reflects the proportion of outliers in the dataset. Although NN-MPPCA is more robust to high-dimensional input thanks to a penalty component included in the loss function, the singularity problem still occurs with extremely high-dimensional input during training. Therefore, future works will be devoted to the integration of the compression network into MPPCA model. The candidate for the compression network may be the deep autoencoder (DAE) or the variational autoencoder (VAE). In this framework, the compression network plays a role as a latent features extractor and provides the low-dimensional input to the estimation network NN-MPPCA.

For reproducibility and future improvement by other researchers, the complete source code of this study is provided in the following repository: `https://github.com/viettra-xai/NN-MPPCA`.

Chapter 5

Unsupervised Fault Detection for Building Air Handling Unit Systems Using Deep Variational Mixture of Principal Component Analyzers

5.1 Introduction

Heating, ventilation, and air-conditioning (HVAC) systems are ubiquitous in commercial and residential buildings nowadays. They play an important role in regulating thermal and air comfort for buildings. To secure thermal pleasure and fresh air for buildings, HVAC systems need to consume a huge amount of energy. The statistics show that HVAC systems account for 57% of the energy used in U.S. buildings and more than 40% of total global energy consumption B. Li et al. (2021); Yan, Chong, Mo (2020). Besides the essential usage, a significant proportion of this energy is leaked due to underqualified maintenance, component degradation, and control failure. Without prompt intervention, incipient faults in components develop and eventually, lead to system disruption. This disruption not only causes energy loss but also leads to thermal discomfort for commercial buildings. To a certain extent, scheduled maintenance can prevent destructive failures and prolong the operational reliability of the system. However, expenses related to tools and experts to perform periodic maintenance are costly. In addition, unforeseen failures might occur during maintenance intervals. Therefore, to prevent the catastrophic consequence of unexpected disruption, incipient faults of building HVAC systems need to be early detected by accurate fault detection and diagnosis (FDD) techniques Tra et al. (2022a, 2022b); Yan, Huang, et al. (2020); Yan, Su, et al. (2020).

Functioning as process monitoring tools, fault detection (FD) methods are designed to identify early-stage faults emerging within a system, subsequently alerting operators to their presence Jang, Hong, Kim, Na, Moon (2021); Jiang, Yan, Huang (2019); Kumar Hati (2021). This early detection enables system operators and maintenance personnel to promptly intervene and implement appropriate measures to address the identified issues. In practical scenarios, obtaining unlabeled data is relatively straightforward, while acquiring labeled fault data tends to be costly and sometimes unfeasible. This underscores the significance of fault detection techniques that can operate in an unsupervised fashion, eliminating the requirement for labeled training data. Similar to anomalies, instances of faults that signify aberrant system conditions frequently exhibit distinctive patterns, divergent from those of normal operations. Consequently, the application of unsupervised anomaly detection (AD) methods proves notably effective for the task of fault detection. By learning inherent normal patterns from data sourced directly from a well-functioning device, unsupervised AD algorithms possess the capability to identify and flag any atypical states (i.e., unforeseen faults) that may arise within the device Brito, Susto, Brito, Duarte (2022); Riazi et al. (2019). However, defining normal and abnormal behavior (i.e., faults) in HVAC systems is particularly difficult, especially in dynamic environments where operating conditions can vary significantly. Thus, designing a fault detection model that performs well for HVAC systems is both a challenging and crucial task.

With the flourishing growth of probabilistic models, density estimation-based methodologies have emerged as a fundamental component within anomaly detection applications Ruff et al. (2021); Zong et al. (2018). In these approaches, the task of anomaly detection entails employing probabilistic models, such as the Gaussian mixture model (GMM), to accurately capture the characteristics of historical training data. Subsequently, the anomaly score for a test instance is derived by evaluating the negative log-likelihood of the instance with respect to the fitted probabilistic model. Much like GMM, a mixture of probabilistic principal component analyzers (MPPCA) Tipping Bishop (1999) consists of individual components, each being a single probabilistic principal component (PPCA). This mixture framework enables it to effectively model complex data distributions, making it particularly well-suited for anomaly detection applications. However, its application in anomaly detection has remained relatively limited. This motivated us to undertake a comprehensive study and enhancement of existing MPPCA-based AD methods. Like other shallow AD methods such as GMM, one-class support vector machine (OC-SVM), kernel density estimation (KDE), etc., MPPCA has satisfactory performance for low-dimensional data. However, they are susceptible to high-dimensional problems by the constraint of the curse of dimensionality. Furthermore, within high-dimensional feature spaces, instances are more likely to be perceived as rare events, contributing to an elevated false alarm rate. In instances where statistical methods, such as GMMs and MPPCAs, are trained using the expectation-maximization (EM) algorithm, training iterations involve the computation of inverse input covariance matrices. As the input dimensionality increases, the risk of encountering non-invertible covariance matrices grows. This issue is particularly pronounced in HVAC systems, where the need to monitor a large number of variables and parameters leads to high-dimensional data, which adds complexity to anomaly detection tasks. Hence, high-dimensional original input should be transformed into low-dimensional high-level features by a compression model before being fed to classic AD methods for an AD task. In this regard, autoencoders (AEs) and their variants emerge as preferred options due to their adaptable nonlinear encoding and decoding architecture. Among AE variants, a variational autoencoder (VAE) is selected as a compression network for our framework. This selection stems from the fact that, in contrast to other AE variants, VAE can be interpreted in a probabilistic manner that aligns with the principles of MPPCA. Additionally, VAE itself serves as an anomaly detection model Su et al. (2019); Xu et al. (2018). The negative reconstruction probability assigned by the VAE model to an instance acts as an indicator of its anomaly. Using both anomalous indicators of VAE and MPPCA should increase the certainty of the final decision in AD tasks. In our framework, both the compression model VAE and the estimation model MPPCA undergo joint training. Throughout this training process, tasks involving dimensionality reduction and density estimation are concurrently executed, ensuring that the framework converges toward the global optimum.

Beyond the above reasons, the employment of VAE in the proposed framework is primarily driven by its robust ability to resist incomplete data Xu et al. (2018). This is achieved by adapting the adaptive evidence lower bound (ELBO) loss function during the training phase. The statistics show that incomplete data is one of the most tricky problems that energy and building researchers frequently encounter. More specifically, to collect empirical building data, a network of wireless sensors is installed around the building to continuously record outside and inside environmental factors, internal loads, and mechanical system working conditions. Because building sensors often operate in harsh environments with interfering and disruptive conditions, the raw sensor data may be sparse, missing, or incomplete—particularly in older or poorly maintained systems—making it rarely suitable for direct use Jin, Jia, Spanos (2017); D. Li, Zhou, Hu, Spanos (2019). Hence, our objective is to create an HVAC fault detection framework that operates effectively even in scenarios involving incomplete data. By integrating the adapted evidence lower bound (ELBO) loss function into the VAE component, the resulting framework showcases exceptional effectiveness when dealing with incomplete datasets.

To undertake the integration of VAE and MPPCA, it becomes necessary to unify both models under a cohesive architecture. VAE adopts a neural network-based format, wherein parameters are optimized through the back-propagation algorithm. In contrast, MPPCA operates as an algebraic model and its training relies on the EM algorithm. Therefore, for successful integration, an initial step involves transforming MPPCA into a neural network format. This can be achieved by utilizing a neural network to model MPPCA's parameters. Through the training of the neural network using a designated loss function, the parameters of MPPCA are systematically optimized. This newly-formed model is referred to as NN-MPPCA, with its parameters being updated via the back-propagation algorithm. In essence, the VAE, functioning as the compression network, operates at the front stage to create a low-dimensional representation. Situated in the following stage, the estimation network NN-MPPCA processes this representation to perform an estimation task. Given that the framework combines VAE and NN-MPPCA, it is aptly termed the deep variational mixture of probabilistic principal analyzers (DV-MPPCA). Through the minimization of the defined total loss function, both the VAE and NN-MPPCA models within the framework seamlessly converge toward the global optimum.

The main contributions of the paper are summarized as follows:

(1) The neural network form of MPPCA is proposed. The new form of the MPPCA model (NN-MPPCA) concatenates to VAE in the preceding block to constitute the unified framework DV-MPPCA for an anomaly detection task.

(2) A modified ELBO function is adeptly employed within the VAE model to enhance the proposed framework's resilience against incomplete multivariate data.

(3) The total loss function of DV-MPPCA is precisely defined to ensure the convergence of both constituent models, VAE and NN-MPPCA, towards the global optimum.

(4) Case studies are undertaken using distinct seasonal datasets acquired from a real-world air handling unit system, designed by the ASHRAE Research Project 1312 (RP-1312). Comparative experiments have substantiated both the applicability and superiority of the proposed framework over competing

state-of-the-art models.

The rest of the paper is arranged as follows: Section 5.2 provides an overview of related works. Section 5.3 introduces the proposed framework and presents the mathematical innovations of the constituent models. Section 5.4 showcases the validation datasets from RP-1312 and presents comprehensive experiments. Lastly, Section 5.5 offers concluding remarks and summaries.

5.2 Related Works and Problem Statement

5.2.1 Data Interpolation Techniques

For industrial applications, incomplete data are widely associated with missing data points caused by sensor limitations and device constraints. To construct reliable FDD models, missing data points need to be refined in the pre-processing step. Various approaches can be employed to address missing data, with listwise deletion being a commonly used approach. This method entails eliminating entire observations from the dataset if any variable is missing. Although widely favored due to its computational efficiency, this method may result in the loss of significant information when dealing with large amounts of missing data.

Another widely utilized approach for data preparation involves reconstructing missing values before proceeding with further analysis. Within the field of pattern classification, naive data interpolation techniques have been extensively employed to handle missing data García-Laencina, Sancho-Gómez, Figueiras-Vidal (2010). These techniques can be categorized into three main groups: model-based methods, statistical analysis-based methods, and machine learning-based methods. The model-based imputation method assumes a joint distribution for all variables in the model. Among the popular approaches is the use of mixture models trained with the expectation-maximization (EM) algorithm Ghahramani Jordan (1993). These model-based methods primarily focus on estimating model parameters, and missing values are then approximated by calculating their expected values based on the estimated parameters. A drawback of these methods is their high computational cost and disregard for the temporal relationship in the data. Statistical analysis-based data imputation methods Little Rubin (2019), such as mean imputation and regression imputation, are also commonly used. Mean imputation replaces missing data points of a variable with the average of observed data points for that variable. While simple, this method often underestimates the correlation between variables. Regression imputation, on the other hand, fills missing data points by predicting values from a regression analysis. In cases with two or more incomplete features, a multivariate regression model is employed. However, using regression to impute missing values for independent variables can lead to multicollinearity problems since the imputed values are perfectly correlated with other variables in the model. Additionally, for complex systems like building HVAC systems with mass sensor variables, building numerous regression models becomes cumbersome. Machine learning techniques present effective alternatives for filling in missing data with reliable values. Well-known methods encompass K-nearest neighbors (KNN) imputation Batista, Monard, et al. (2002), self-organizing map (SOM) imputation Fessant Midenet (2002), recurrent neural network (RNN) imputation Bengio Gingras (1995), and more. However, current machine learning-based data interpolation methods typically depend on either time series adjacency or channel adjacency in sensor streams to predict the missing data. Additionally, the effectiveness of most of these techniques is contingent on the availability of abundant and reliable training data, and their performance diminishes substantially in the presence of a significant number of missing data points.

Nonetheless, incorporating an additional model for data imputation adds complexity to the suggested framework. Moreover, the VAE itself is a potent generative model. It seems unreasonable to train a generative model using data generated by another algorithm, as the primary purpose of generative models is to create data. Hence, rather than adopting missing data imputation, we integrate the VAE directly into the proposed framework. By adeptly applying the modified ELBO loss function to the VAE, we effectively eliminate the contribution of missing data points. As a result, the resulting framework becomes resilient to incomplete data.

5.2.2 Density Estimation Models

Classic density estimation methods play a pivotal role in anomaly detection applications. These methods detect anomalies among data instances by estimating their relative probability density. Instances located in low-density areas are classified as anomalies. A notable example is the Gaussian mixture model (GMM) L. Li, Hansman, Palacios, Welsch (2016); Zong et al. (2018), which employs a combination of K Gaussian components to effectively model complex data distributions. Samples falling outside the scope of this mixed Gaussian distribution are identified as anomalies. Other widely employed density estimators include histogram estimators and kernel density estimators (KDEs). KDEs, especially when combined with adaptive kernels, offer theoretical advantages over histogram estimators, contributing to their extensive application in anomaly detection (AD) contexts Hu et al. (2018); Kim Scott (2012).

PPCA and its mixture version MPPCA are among the notable probability density estimation methods that have been widely used for FDD applications. For example, Qu et al. Qu, Li, Zhang, Hu (2009) used PPCA to impute the missing values of traffic flow volume via learning historical data. Yu et al. Yu, Snapp, Ruiz, Radermacher (2010) developed a PPCA-based method to classify 3D reconstructions with missing data. Zhang et al. J. Zhang et al. (2021) incorporated clustering techniques into MPPCA for process monitoring. In their work Sharifi Langari (2017), Sharifi et al. introduced a sensor fault diagnosis model for HVAC systems based on MPPCA. This model partitions the data space into local linear regions, each associated with an individual PPCA. Although the theoretical foundation of PPCA and MPPCA have proven their potential for anomaly detection, to the best of our knowledge, very few studies have investigated the utility of PPCA and MPPCA for anomaly detection. For instance, Ma et al. Ma et al. (2021) employed PPCA for the detection of structural anomalies and their localization. In another study, Ma et al. Ma, Luo, Yun, Wan, Shen (2022) used MPPCA to formulate a model for the detection and localization of structural anomalies. Yairi et al. Yairi et al. (2017) introduced a data-driven health monitoring technique for artificial satellites using MPPCA. A shared characteristic among these studies is the utilization of PPCA and MPPCA to model the distribution of historical data. Using the resulting fitted models, they then calculate the negative log-likelihood of unseen instances, which is utilized as an anomaly score.

5.2.3 Deep generative models

Deep neural network-based generative models, including variational autoencoders (VAEs) Kingma Welling (2013), generative adversarial networks (GANs) Goodfellow et al. (2020), and normalizing flows (NFs) Rezende Mohamed (2015), have emerged as preferable alternatives to classical density estimation models for high-dimensional data. Such models map latent codes sampled from a source distribution (i.e., a Gaussian distribution) to a real data distribution through their generative networks. In the context of GANs, the learning process can be viewed as a zero-sum game in which a generative network is trained in competition with a discriminative network to generate samples whose distribution is similar to real data distribution. Many GAN variants have been proposed for AD applications. For example, Schlegl et al. Schlegl et al. (2017) proposed AnoGAN in which an intermediate layer of the discriminator is used to calculate the discrimination loss. A convex combination of the reconstruction loss and the discrimination loss is then used as an anomaly score. Zhou et al. X. Zhou, Xiong, Zhang, Liu, Wei (2021) proposed a radio anomaly detection algorithm based on a modified GAN. In this modified version, an encoder network is incorporated into the original GAN to reconstruct the spectrogram. In another study Ngo et al. (2019), Ngo et al. modified the loss function of a typical GAN to directly use the discriminator score as an anomaly threshold.

Normalizing flows are generative models used to construct complex distributions by transforming a probability density through a sequence of invertible mappings. This process involves applying the rule for changing variables iteratively, causing the initial density to 'flow' through the series of invertible transformations. More recently, there has been a growing trend in utilizing NFs for unsupervised anomaly detection and localization. For instance, Gudovskiy et al. Gudovskiy, Ishizaka, Kozuka (2022) introduced CFLOW-AD, a conditional normalizing flow framework tailored for anomaly detection with localization. CFLOW-AD comprises a discriminatively pre-trained encoder, followed by a multi-scale generative decoder that explicitly estimates the likelihood of the encoded features. Rudolph et al. Rudolph, Wehrbein, Rosenhahn, Wandt (2022) devised CSFLOW, a cross-scale normalizing flow capable of processing multiscale features derived from a pre-trained network. CSFLOW enables each scale to integrate information from other scales, allowing for the efficient processing of multiscale feature maps using a single flow model.

VAE is the probabilistic version of AE where both latent code and reconstruction can be sampled from the estimated distributions. This can be made possible through a stochastic autoencoding process. For AD applications, VAEs directly use the reconstruction probability as an anomaly score. For example, Xu et al. Xu et al. (2018) devised Donut, an unsupervised anomaly detection approach based on VAE, targeted for seasonal KPI analysis. In this method, a modified ELBO and Markov chain Monte Carlo (MCMC) imputation were introduced to enable Donut to achieve outstanding performance on KPI data. Su et al. Su et al. (2019) proposed OmniAnomaly, an innovative anomaly detection model named for multivariate time-series data, featuring a concatenation of a recurrent neural network (RNN) and VAE to capture temporal patterns from historical data. The authors also adopted planar normalizing flows to learn non-Gaussian distributions in stochastic space. Li et al. L. Li, Yan, Wang, Jin (2020) introduced smoothness-inducing prior to a VAE-based anomaly detection model as a regularizer to penalize non-smooth reconstructions.

5.2.4 Problem Statement

Both density estimation models and deep generative models have their own advantages. While classic density estimation models are widely recognized for their simple configuration yet robust performance. Deep generative models perform really well for high-dimensional data. To leverage the advantages of both approaches, we proposed a novel anomaly detection method which is a combination of the deep generative model VAE and the density estimation model MPPCA. With VAE playing a role as the compression network, the proposed model is explicitly designed to work on high-dimensional multivariate data. In addition, by modifying the ELBO loss function, our model is highly resistant to incomplete input data.

5.3 Methodologies

5.3.1 Framework Overview

The proposed DV-MPPCA framework is visually depicted in Fig. 5.1. This architecture consists of two primary constituent models: (1) the VAE model, which functions as the compression network, and (2) the NN-MPPCA model, responsible for density estimation. VAE at the front prepares the low-dimensional derivative vectors from incomplete original data and feeds them to the subsequent model NN-MPPCA. Within our framework, each derivative vector consists of two main components: the latent features $\mathbf{z}_c$ and the input's reconstruction error features $\mathbf{z}_r$ from the VAE model. These latent features capture the essential information present in the high-dimensional original data, while the reconstruction error features offer initial indications of anomalies within the original input. As a result, both types of features play a crucial role in enabling the NN-MPPCA model to perform accurate estimation tasks. Another reason for using the reconstruction error features in our study is that their involvement is essential for the training process. Without their contribution, the loss value of the framework would no longer depend on the output of VAE. Consequently, the back-propagation flow will not propagate through the decoder block of the VAE model, leading to the scenario where the decoder's parameters remain unaltered. In the context of anomaly detection, the NN-MPPCA receives derivative vectors from the VAE model. It calculates the negative log-likelihoods of these vectors, which in turn function as the anomaly scores for individual instances.

5.3.2 Adaptive Variational Autoencoder for Incomplete Data Feature Extraction

VAE is a deep Bayesian network that models the relationship between the observed variable $\mathbf{x}$ and the latent variable $\mathbf{z}$. Given a prior distribution $p_\theta(\mathbf{z})$, $\mathbf{x}$ is sampled from the posterior distribution $p_\theta(\mathbf{x}|\mathbf{z})$. While $p_\theta(\mathbf{z})$ is normally a multivariate unit Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, $p_\theta(\mathbf{x}|\mathbf{z})$ is chosen depending on the task and derived from a generative model θ_d (the decoder). More specifically, $p_\theta(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\mathbf{x}|\mu_\theta(\mathbf{z}), \sigma^2_\theta(\mathbf{z}))$, where $\mu_\theta(\mathbf{z})$ and $\sigma_\theta(\mathbf{z}))$ are the mean and standard deviation of an isotropic Gaussian distribution parameterized by the decoder network and $\mathbf{z}$ is sampled from the K-dimensional latent space. However, it is

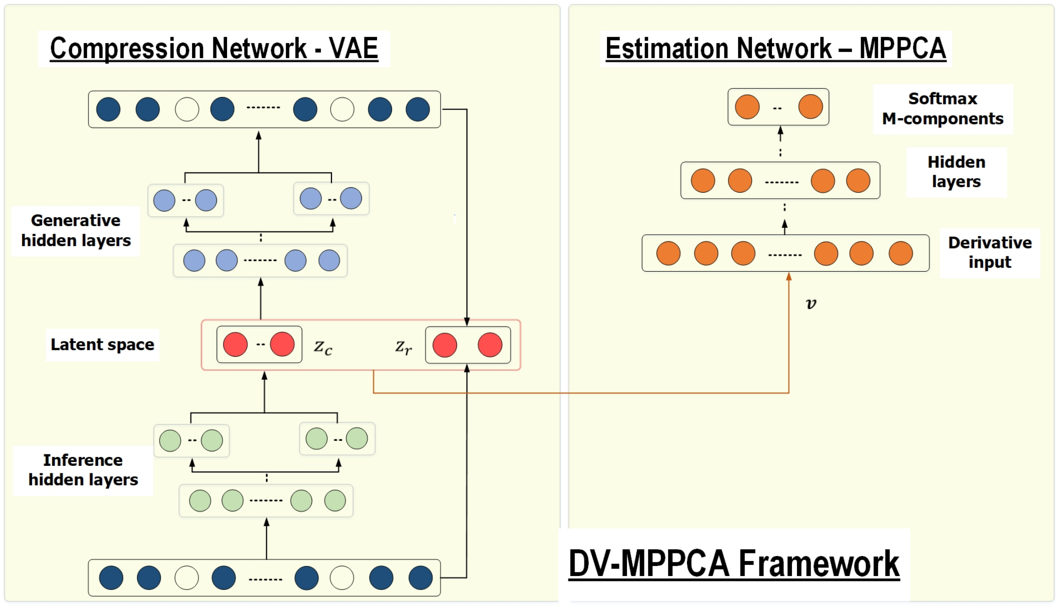

Figure 5.1: The architecture of the DV-MPPCA framework with two component models: (1) the VAE model acts as a compression network, and 2) the NN-MPPCA model performs a density estimation task.

intractable to compute $p_\theta(\mathbf{z}|\mathbf{x})$. Therefore, an inference model ϕ_e (the encoder) is introduced to parameterize the approximate distribution $q_\phi(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \sigma^2_\phi(\mathbf{x}))$ of the true posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, where $\mu_\phi(\mathbf{x})$ and $\sigma_\phi(\mathbf{x}))$ are the mean and standard deviation of an isotropic Gaussian distribution parameterized by the encoder network.

Let $f_\phi(\mathbf{x})$ and $f_\theta(\mathbf{z})$ be shared hidden layers of the encoder network and the decoder network, respectively. The mean μ_ϕ and the standard deviation σ_ϕ of the latent variable distribution are derived from the encoder: $\mu_\phi = \mathbf{W}^T_{\mu_\phi} f_\phi(\mathbf{x}) + \mathbf{b}_{\mu_\phi}$; $\sigma_\phi = \text{SoftPlus}[\mathbf{W}^T_{\sigma_\phi} f_\phi(\mathbf{x}) + \mathbf{b}_{\sigma_\phi}] + \epsilon$, where $\mathbf{W}_{\mu_\phi}$, $\mathbf{b}_{\mu_\phi}$, and $\mathbf{W}_{\mu_\sigma}$, $\mathbf{b}_{\mu_\sigma}$ are weights and biases of the linear layer and the soft-plus layer, respectively; $\text{SoftPlus}[a] = \log[\exp(a) + 1]$ and ϵ is a non-negative small number Xu et al. (2018). Similarly, the mean μ_θ and the standard deviation σ_θ of the observed variable distribution are derived from the decoder: $\mu_\theta = \mathbf{W}^T_{\mu_\theta} f_\theta(\mathbf{z}) + \mathbf{b}_{\mu_\theta}$; $\sigma_\theta = \text{SoftPlus}[\mathbf{W}^T_{\sigma_\theta} f_\theta(\mathbf{z}) + \mathbf{b}_{\sigma_\theta}] + \epsilon$, where $\mathbf{W}_{\mu_\phi}$, $\mathbf{b}_{\mu_\phi}$, and $\mathbf{W}_{\mu_\sigma}$, $\mathbf{b}_{\mu_\sigma}$ are weights and biases of the linear layer and the soft-plus layer of the decoder network, respectively.

The generative model and the inference model of VAE are jointly trained by maximizing the evidence lower bound $\mathcal{L}(\phi, \theta; \mathbf{x})$ (ELBO) with respect to their parameters.

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{x}) &= \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z}) - \mathrm{D}_{\mathrm{KL}}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})]\right] \\ &= \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z}) + \log p_\theta(\mathbf{z}) - \ \log q_\phi(\mathbf{z}|\mathbf{x})\right]\end{aligned} \tag{5.1}$$

Since the latent variable $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$ is stochastic that hinders backpropagation, the stochastic gradient variational Bayes (SGVB) Kingma Welling (2013) is applied to express the random variable $\mathbf{z}$ as the differentiable form of another random variable ε using reparameterization trick: $\mathbf{z} = g(\varepsilon, \phi, \mathbf{x})$, ε is sampled from the random noise distribution $p(\varepsilon)$. To approximate the expectation in (5.1), Monte Carlo integration Geweke (1989) is adopted as follows:

$$\mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[f(\mathbf{z})] \approx \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z}^{(l)}) \tag{5.2}$$

where $\mathbf{z}^{(l)}, l = 1...L$ are sampled from $q_\phi(\mathbf{z}|\mathbf{x})$.

To handle missing values of incomplete data, the contribution of missing data points to the loss function should be excluded. To do that, we filled missing points with zeros and applied a modified ELBO (M-ELBO) Xu et al. (2018) for training VAE with multivariate data. Let α_w be the missing indicator of data points in an observed instance, where $\alpha_w = 0$ indicates x_w being missing, and $\alpha_w = 1$ otherwise. β is defined as $(\sum_{w=1}^{W} \alpha_w)/W$. By applying (5.3), the contribution of missing values to $p_\theta(\mathbf{x}|\mathbf{z})$ is excluded by α_w, while the scaling factor β shrinks the contribution $p_\theta(\mathbf{z})$ according to the ratio of missing points in x. The modification of the ELBO function eliminates the bias caused by missing data points and trains VAE to extract accurate latent representations.

$$\mathcal{L}_{\text{M-ELBO}}(\mathbf{x}) = \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{w=1}^{W} \alpha_w \log p_\theta(\mathbf{x}|\mathbf{z}) + \beta \log p_\theta(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right] \tag{5.3}$$

After being extracted by VAE, accurate latent features of the original input combine with reconstruction error features to feed the estimation network NN-MPPCA. To be specific, the derivative vector $\mathbf{v}$ of NN-MPPCA comprises two components: 1) low-dimensional features $\mathbf{z}_c$ in the compression space (i.e., latent variable $\mathbf{z}$ of VAE); and 2) reconstruction error features $\mathbf{z}_r$. These features are calculated from an input $\mathbf{x}$ as follows:

$$\mathbf{z}_c = f(\mathbf{x}, \phi) \qquad \mathbf{x}' = f(\mathbf{z}_c, \theta) \tag{5.4}$$

$$\mathbf{z}_r = r(\mathbf{x}_o, \mathbf{x}'_o) \qquad \mathbf{v} = [\mathbf{z}_c, \mathbf{z}_r] \tag{5.5}$$

where $\mathbf{x}_o$ and $\mathbf{x}'_o$ are the observed part of an input instance and its reconstruction (i.e., normal points).

5.3.3 Neural Network-based Mixture of Probabilistic Principal Component Analyzers for Estimation Task

In this section, we revisit the neural network-based mixture of probabilistic principal component analyzers (NN-MPPCA) model, which was extensively detailed in the previous chapter. Here, NN-MPPCA is employed as an estimation network, leveraging its capabilities to further enhance the accuracy and efficiency of the fault detection process. This application builds on the foundation laid earlier, utilizing the robustness of NN-MPPCA to analyze and predict abnormal system behaviors effectively. This reapplication serves to demonstrate the versatility and adaptability of NN-MPPCA in varying operational contexts within fault detection frameworks.

A mixture of PPCAs (MPPCA) can be used to fit any complex data distribution. Let $\mathbf{v}_n$, $n \in 1, ..., N$ be the d-dimensional input data set of MPPCA, the complete data log-likelihood for such a mixture model with M components is formulated as follows:

$$\mathcal{L}_C = \sum_{n=1}^{N} \sum_{k=1}^{M} r_{nk} \log\{\pi_k p(\mathbf{v}_n, \mathbf{t}_{nk})\} \tag{5.6}$$

where $\mathbf{t}_{nk}$ is the q projections of $\mathbf{v}_n$ on principal axes of a k^{th} mixture component, and π_k is the corresponding mixing proportion (i.e., $\pi_k \geq 0$ and $\sum \pi_k = 1$). Variables r_{nk} define the responsibility of the k^{th} mixture component for generating an observed data $\mathbf{v}_n$.

For the MPPCA framework, the mean vector $\boldsymbol{\mu}_k$, factor loading matrix $\mathbf{W}_k$, and noise variance $\boldsymbol{\sigma}_k^2$ of each mixture component are separately calculated and optimized via an iterative EM algorithm Tipping Bishop (1999). More specifically, the parameters of MPPCA are updated by maximizing the complete data log-likelihood (Eq. 5.6) using a two-stage EM procedure as follows:

$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} R_{nk} \tag{5.7}$$

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} R_{nk}\mathbf{v}_n}{\sum_{n=1}^{N} R_{nk}} \tag{5.8}$$

where R_{nk} is the posterior probability of r_{nk} and is calculated by (5.9)

$$R_{nk} = p(k|\mathbf{v}_n) = \frac{p(\mathbf{v}_n|k)\pi_k}{p(\mathbf{v_n})} \tag{5.9}$$

in which $p(\mathbf{v}_n|k)$ is the likelihood of the k^{th} mixture component and given by:

$$p(\mathbf{v}_n|k) = (2\pi)^{(-d/2)}|\mathbf{C}_k|^{(-1/2)} \times \exp\{(-1/2)(\mathbf{v}_n - \boldsymbol{\mu}_k)^T\mathbf{C}_k^{-1}(\mathbf{v}_n - \boldsymbol{\mu}_k)\} \tag{5.10}$$

where $\mathbf{C}_k = \boldsymbol{\sigma}_k^2\mathbf{I} + \mathbf{W}_i\mathbf{W}_i^T$ is the model covariance of the k^{th} mixture component.

Keeping $\boldsymbol{\mu}_k$ fixed, the component noise variance and loading matrix are updated from their previous $\hat{\boldsymbol{\sigma}}_k$ and $\hat{\mathbf{W}}_k$ as follows:

$$\mathbf{W}_k = \mathbf{S}_k\hat{\mathbf{W}}_k(\hat{\boldsymbol{\sigma}}_{\boldsymbol{k}}\mathbf{I} + \hat{\mathbf{M}}_k^{-1}\hat{\mathbf{W}}_k^T\mathbf{S}_k\hat{\mathbf{W}}_k)^{-1} \tag{5.11}$$

$$\boldsymbol{\sigma}_k = \frac{1}{d}\text{tr}(\mathbf{S}_k - \mathbf{S}_k\hat{\mathbf{W}}_k\hat{\mathbf{M}}_k^{-1}\mathbf{W}_k^T) \tag{5.12}$$

where $\mathbf{M}_k = \boldsymbol{\sigma}_k^2\mathbf{I} + \mathbf{W}_k^T\mathbf{W}_k$. Please note that $\hat{\mathbf{M}}_k$ is calculated from $\hat{\boldsymbol{\sigma}}_k$ and $\hat{\mathbf{W}}_k$, and $\mathbf{S}_k$ is calculated from the current R_{nk} and $\boldsymbol{\mu}_k$ as follows:

$$\mathbf{S}_k = \frac{1}{\pi_k N}\sum_{n=1}^{N} R_{nk}(\mathbf{v}_n - \boldsymbol{\mu}_k)(\mathbf{v}_n - \boldsymbol{\mu}_k)^T \tag{5.13}$$

In this study, we used a neural network (i.e., a multilayer perceptron) to model the MPPCA framework in which its parameters are updated via the back-propagation process. This neural network form of MPPCA is called NN-MPPCA and plays a role as the estimation network of our framework. While the input layer of the estimation network fits the data dimension, the size of the output layer depends on the number of mixture components used to model the data distribution. For an MPPCA with M components, the output of the estimation network is the M-dimensional vector $\mathbf{P}_n$. To get the responsibility vector $\mathbf{R}_n$, we applied the softmax function at the output so that $\sum_{k=1}^{M} R_{nk} = 1$. In other words, instead of using Eq. (5.9) to calculate the responsibility vector as in the EM algorithm, we use the output of the estimation network to obtain this vector. This process is formulated as follows:

$$\begin{aligned} \mathbf{P}_n &= f(\mathbf{v}_n, \psi) \\ \mathbf{R}_n &= \text{SoftMax}(\mathbf{P}_n) \end{aligned} \tag{5.14}$$

where $\mathbf{v}_n$ is the input of NN-MPPCA; ψ and $\mathbf{P}_n$ are the trainable parameters and output of the estimation network NN-MPPCA.

5.3.4 Training Procedure

In our framework, the parameters of the VAE are trained by maximizing the modified evidence lower bound (ELBO), as referenced in Eq. (5.3). Simultaneously, the parameters of NN-MPPCA are updated by maximizing the log-likelihood of observing complete derivative vectors of training data and minimizing the reconstruction error. Consequently, to facilitate the joint training of both the VAE and the NN-MPPCA models, we have introduced a comprehensive total loss function that encompasses three distinct loss components: (1) the negation of the modified ELBO; (2) the negation of the complete log-likelihood of MPPCA; and (3) the reconstruction error of MPPCA. The minimization of this total loss function drives both the VAE and NN-MPPCA models to converge toward the global optimum in a coordinated manner.

For a N-samples batch, the loss function of DV-MPPCA is formulated as follows:

$$\mathcal{L}(\phi, \theta, \psi) = -\frac{\eta_1}{N}\sum_{n=1}^{N}\mathcal{L}_{\text{M-ELBO}}(\mathbf{x}_n) - \frac{\eta_2}{N}\sum_{n=1}^{N}\log\{p_\psi(\mathbf{v}_n)\} + \frac{\eta_3}{N}\sum_{n=1}^{N}\mathcal{L}(\mathbf{v}_n, \mathbf{v}'_n) \tag{5.15}$$

The VAE model operates on the original high-dimensional input, and the minimization of the negation of ELBO encourages the VAE to achieve two key objectives: firstly, to effectively reconstruct the input (achieved by maximizing the reconstruction probability $\text{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[p_\theta(\mathbf{x}|\mathbf{z})]$) and secondly, to optimize the packing of latent codes within the prior distribution (achieved by minimizing the Kullback–Leibler divergence $\text{D}_{\text{KL}}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})]$. Remarkably, the modified version of ELBO employed in this study extends this functionality by compelling the model to disregard the contribution of missing data points in the input.

The NN-MPPCA model of the framework operates on the derivative low-dimensional vectors $\mathbf{v}$ and is trained with the objective of maximizing the likelihood of observing complete derivative vectors of training data. For a derivative vector $\mathbf{v}_n$, the log-likelihood is expressed as follows:

$$\begin{aligned}\log\{p_\psi(\mathbf{v}_n)\} &= \log\left\{\sum_{k=1}^{M}\pi_k p(\mathbf{v}_n|k)\right\} \\ &= \log\left\{\sum_{k=1}^{M}\pi_k(2\pi)^{-d/2}|\mathbf{C}_k|^{-1/2}\text{exp}\left(-\frac{1}{2}(\mathbf{v}_n-\boldsymbol{\mu}_k)^T\mathbf{C}_k^{-1}(\mathbf{v}_n-\boldsymbol{\mu}_k)\right)\right\}\end{aligned} \tag{5.16}$$

Note that $\mathbf{C}_k$ is calculated from the current values of $\boldsymbol{\sigma}_k$ and $\mathbf{W}_k$ which depend on their previous values. Therefore at the early training stage, the values of $\boldsymbol{\sigma}_k$ and $\mathbf{W}_k$ can not be calculated by (5.11), and (5.12). Instead, their initial values need to be estimated. To initialize the noise variance $\boldsymbol{\sigma}_k$ and loading factor matrix $\mathbf{W}_k$, we need to calculate the component covariance matrix $\mathbf{S}_k$ using (5.13). For the $d \times d$ covariance matrix $\mathbf{S}_k$, its eigenvalues and eigenvectors are $\lambda_{k1} \geq \lambda_{k2} \geq ... \geq \lambda_{kd}$, and $u_{k1} \geq u_{k2} \geq ... \geq u_{kd}$, respectively. If q is the number of principal components that we want to retain, the initial value of $\boldsymbol{\sigma}_k$ and $\mathbf{W}_k$ is estimated as:

$$\boldsymbol{\sigma_k} = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_{kj} \tag{5.17}$$

$$\mathbf{W_k} = U_{k,q}(\lambda_{k,q} - \boldsymbol{\sigma}_k^2\mathbf{I})^{1/2} \tag{5.18}$$

where $\mathbf{U}_{k,q}$ and $\boldsymbol{\Lambda}_{k,q}$ are defined as:

$$\begin{aligned}\mathbf{U}_{k,q} &\equiv [u_{k1}, u_{k2}, ..., u_{kq}] \\ \boldsymbol{\Lambda}_{k,q} &\equiv \text{diag}(\lambda_{k1}, \lambda_{k2}, ..., \lambda_{kq})\end{aligned} \tag{5.19}$$

Except for the initial phase, for the next iterations the new values of π_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\sigma}_k$, and $\mathbf{W}_k$ are accordingly updated by (5.7) - (5.8), followed by (5.11) - (5.12). By maximizing the complete data log-likelihood (or minimizing its negative counterpart), the NN-MPPCA model is optimized to precisely fit the distribution of the derivative training vectors.

The reconstruction error of MPPCA, denoted as $\mathcal{L}(\mathbf{v}_n, \mathbf{v'}_n) = |\mathbf{v} - \mathbf{v'}|$ assesses the efficacy of MPPCA in a reconstruction context. For a derivative vector $\mathbf{v}_n$, its reconstruction $\mathbf{v'}_n$ is estimated as follows:

$$\mathbf{v'}_n = R_{nk}\mathbf{v'}_{nk} \tag{5.20}$$

where $\mathbf{v'}_{nk}$ is the reconstruction of $\mathbf{v}_{nk}$ with respect to the k^{th} mixture component and derived from:

$$\begin{aligned}\mathbf{v'}_{nk} &= \mathbf{W}_k(\mathbf{W}_k^T\mathbf{W}_k)^{-1}\mathbf{M}_k\langle\mathbf{t}_{nk}\rangle \\ &= \mathbf{W}_k(\mathbf{W}_k^T\mathbf{W}_k)^{-1}\mathbf{W}_k^T(\mathbf{v}_n - \boldsymbol{\mu}_k) + \boldsymbol{\mu}_k\end{aligned} \tag{5.21}$$

where $\langle\mathbf{t}_{nk}\rangle$ is the expectation of the latent variable with respect to the k^{th} mixture component. Minimizing this loss function drives MPPCA to achieve an accurate reconstruction of its input, hence, refining the parameters of MPPCA for enhanced performance.

The combination of three loss components is optimally adjusted by using respective heuristic coefficients η_1, η_2, η_3. For an insightful understanding of equations (5.6) - (5.13), (5.17) - (5.21) the interested reader is referred to the original paper of Tipping and Bishop Tipping Bishop (1999).

5.3.5 Online Anomaly Detection

In the proposed framework, both the trained VAE and NN-MPPCA models are able to estimate their own anomaly score. However, we mainly use the anomaly score estimated by NN-MPPCA to make the prediction on anomalous instances. For a test instance $\mathbf{x}_n$, the anomaly score in regard to NN-MPPCA is the minus log-likelihood of the derivative vector $\mathbf{v}_n$. This score reflects how the derivative vector follows the normal patterns learned by the trained NN-MPPCA model and is formulated as $a_{\mathrm{M}}(\mathbf{x}_n) = -\log\{p_\varphi(\mathbf{v}_n)\}$ (Eq. 5.16). Remind that the low-dimensional derivative vector $\mathbf{v}_n$ is constituted from the robust latent representation of the original input $\mathbf{x}_n$. Therefore, anomalous information inherent in the original input is perfectly inherited in the derivative vector. In other words, if the derivative vector is out of the distribution modeled by the trained NN-MPPCA, its corresponding original input is inferred as an anomaly.

Similarly, given a trained VAE, the likelihood of a test instance $p_\theta(\mathbf{x}_n) = \mathrm{E}_{p_\theta(\mathbf{z}_n)}[p_\theta(\mathbf{x}_n|\mathbf{z}_n)]$ is directly used for AD. This score reflects how well an instance $\mathbf{x}_n$ follows the normal pattern. However, the studies Nalisnick, Matsukawa, Teh, Gorur, Lakshminarayanan (2018); Xu et al. (2018) have shown that this score tends to perform worse than using the reconstruction probability, which conditions on $\mathbf{x}$ to estimate $\mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[p_\theta(\mathbf{x}|\mathbf{z})]$. Therefore, this study used the latter one with a modification to calculate the anomaly score. Since the VAE model in our framework is trained to exclude the contribution of missing values, therefore, the impact of missing values on the reconstruction probability should be eliminated

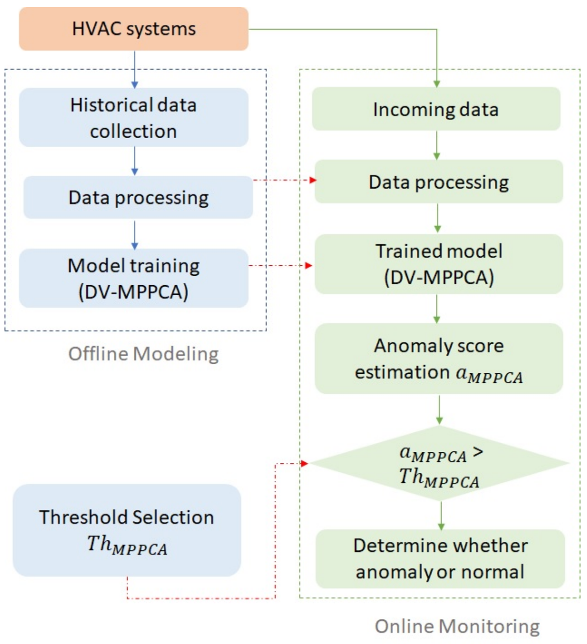
HVAC systems
Historical data collection
Data processing
Model training (DV-MPPCA)
Offline Modeling
Incoming data
Data processing
Trained model (DV-MPPCA)
Anomaly score estimation a_{MPPCA}
$a_{MPPCA} > Th_{MPPCA}$
Threshold Selection Th_{MPPCA}
Determine whether anomaly or normal
Online Monitoring

Figure 5.2: Flowchart of anomaly monitoring with two stages: offline modeling and online monitoring.

by introducing missing indicator α_w. The adaptive form of the reconstruction probability is $a_V(\mathbf{x}_n) = \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \sum_{w=1}^{W} \alpha_w \mathrm{log} p_\theta(\mathbf{x}|\mathbf{z})$, where $\alpha_w = 0$ means x_w being missing, and $\alpha_w = 1$ means x_w being normal.

With pre-defined thresholds, both models can detect anomalous instances by comparing the anomaly scores $a(\mathbf{x}_n)$ with their own thresholds. To be specific, if $a(\mathbf{x}_n)$ is higher than the threshold, $\mathbf{x}_n$ is detected as an anomalous instance; otherwise, $\mathbf{x}_n$ is normal. Using an appropriate mechanism to combine the anomaly scores estimated by the two constituted models for an AD task is valuable but we let this interesting extended topic for future work. Instead, we only use the anomaly score estimated by NN-MPPCA to predict anomalous instances. The anomaly detection process of DV-MPPCA is illustrated in Fig. 5.2.

5.4 Experiments and Results

5.4.1 Experimental Dataset and Model Configuration

In this study, we utilize the ASHRAE RP-1312 Air Handling Unit (AHU) dataset to rigorously validate the performance of our proposed model alongside baseline models. This dataset provides a comprehensive collection of operational data from AHU systems, offering a robust testbed for evaluating model accuracy and reliability in real-world scenarios. By leveraging the ASHRAE RP-1312 dataset, we aim to demonstrate the efficacy of our model in identifying and diagnosing faults, while also benchmarking its performance against established methodologies. This validation process underscores the model's potential applicability in practical HVAC system management and optimization.

With the RP-1312 project, each seasonal dataset includes faulty data $\mathbf{D}_f$ and normal data $\mathbf{D}_n$. While faulty data instances were recorded from AHU-A every minute, normal data instances were collected from AHU-B at the same time. For each typical fault, $n = 1440$ faulty instances were recorded (i.e., 24 hours * 60 minutes). The number of normal instances is equal to that of faulty instances. For a seasonal dataset that studies k different typical faults, the total number of faulty instances and normal instances is $N = 2{*}k{*}1440$. Each data instance is originally a record of sensor values and control signals. In our study, we have chosen to exclude control signals from our consideration and therefore they are not taken into account. The analysis focuses solely on sensor values, and the number of these sensor values under analysis is denoted as $m = 122$.

To adapt the datasets for the purpose of fault detection, the multiclass labels of faulty instances are disregarded, and a binary label assignment is employed. Specifically, faulty instances and normal instances are assigned binary labels based on the following criteria: $y_i = 0$ if $\mathbf{x}_i \in \mathbf{D}_n$, and $y_i = 1$ if $\mathbf{x}_i \in \mathbf{D}_f$. It is important to note that in this study, the number of faulty instances and normal instances used in each seasonal dataset is the same (i.e., $\mathbf{D}_n = \mathbf{D}_f$). During the fault detection process, each of the seasonal datasets is randomly split into two subsets: 50% of the data is allocated to the training subset, and the remaining 50% is used for the testing subset. In accordance with the unsupervised approach, only normal data from the training subset is utilized to train the models, including DV-MPPCA and other baseline models. Given that the number of faulty and normal samples in the testing subset is equal, a threshold is calculated to identify faulty instances, ensuring that around 50% of the testing instances are predicted as faulty. The performance of the models in fault detection is evaluated using the F1-score (F1), a metric derived from both Precision (P) and Recall (R). The formula for calculating the F1-score is: F1 = 2(P $\times$ R) / (P + R). Precision is determined by the ratio of correct predictions to the total number of predictions made, while Recall represents the ratio of correct predictions to the actual instances in the ground truth. To ensure the reliability of the results, the training-testing process is repeated 10 times in a randomized manner. The final result is obtained by calculating the mean value from these repetitions, providing a confident and stable assessment of the models' performance.

Although our framework is distinctly formulated for three seasonal datasets, it shares the same optimal configuration. More specifically, the sizes of the compression network (i.e., VAE) and the estimation network (i.e., NN-MPPCA) are respectively set as (60, 30, 10) and (15, 3). The framework is trained by the Nadam back-propagation algorithm with a learning rate of 0.001. The number of training epochs and the size of mini-batches is 200 and 256, respectively. To combine component losses (refer to Eq. 5.15), the coefficients η_1, η_2, η_3 need to be optimally set. Although each component loss has a different significance and role, the heuristic searches conducted within this study revealed a balanced weighting among these loss components yielded the optimal performance for DV-MPPCA. As a result, for this study, we have established the component loss coefficients as 1, 1, and 1. In addition, the SELU activation function and GlorotNormal kernel initialization are jointly used to benefit the model training process. In addition to using the early stopping technique, we incorporate dropout with a rate of 0.2 and apply l_2 regularization with a factor of 0.01 to all layers. This combination of techniques aims to prevent overfitting of the model. All the layers and the training process of the framework are implemented in the TensorFlow environment.

Because the data in the RP-1312 project has been preprocessed before publication, the seasonal AHU datasets in the current version do not include any missing data points. In order to simulate missing data points, mirroring real-world building data scenarios, we randomly masked some normal data points in the datasets with the value ”NULL,” as if these sensor data points were not collected during data collection. However, as neural network models are unable to process missing data points, these gaps need to be identified and replaced by zeros before they can be input into our framework. These zero values will be detected by our framework, and their contribution will be excluded during the training and estimation processes. However, it’s important to note that the original data may inherently contain zero values, which correspond to sensor readings during the AHU system’s ”OFF” schedule from 18:00 to 6:00 every day. To address this, before designating simulated missing points as zeros, we substitute the zero sensor features with a small value δ (e.g., $\delta = 0.0001$). This particular step aids in distinguishing between zero features and missing data points, without affecting the overall information contained within the data samples.

5.4.2 Effectiveness of DV-MPPCA for Fault Detection Application

Ablation Study

Our proposed framework DV-MPPCA is constituted of two component models (i.e., VAE, and NN-MPPCA). To verify the role of these constituent models and the way to train the framework, we conducted an ablation study comparing the fault detection performance of DV-MPPCA with that of several competitive deep models. The first competitive model is NN-MPPCA. The model uses directly the high-dimensional original input to estimate the anomaly score. The second deep baseline method is DA-MPPCA. Instead of using VAE, DA-MPPCA uses the deep autoencoder (DAE) as the compression network. The latent representation at the compression layer is fused with the reconstruction error features of DAE to provide the feed to the NN-MPPCA model for an estimation task. To prove the advantage of the modified-ELBO function during training VAE, we also compare DV-MPPCA with its competitive version DV-MPPCA†. The difference between the two models is that while DV-MPPCA applies the modified ELBO, DV-MPPCA† uses the conventional ELBO (refer to Eq. 5.1) for training. Another competitive version of DV-MPPCA, namely DV-MPPCA*, is also studied. Instead of using the anomaly score estimated by NN-MPPCA, DV-MPPCA* uses the reconstruction probability estimated by VAE to predict anomalies. The last competitive model in this study is DONUT Xu et al. (2018). The model applies the modified ELBO and the Markov Chain Monte Carlo (MCMC) technique on VAE for an anomaly detection application.

To obtain fair results, the architecture and the meta parameters of DA-MPPCA, DV-MPPCA†, and DV-MPPCA* are configured to be similar to those of DV-MPPCA (Sec. 5.4.1). The network size of DONUT, and NN-MPPCA is similar to that of VAE and NN-MPPCA in the DV-MPPCA framework, respectively. The training data and the testing data of all the models are standardized (i.e., using StandardScaler) before being fed to the models for the training and estimation process. The fault detection performance of DV-MPPCA and the competitive deep models in terms of the F1-score metric is shown in Table 5.1. Normal data and faulty data each account for half of the dataset. Therefore, the models in this study calculate their anomaly threshold accordingly so that half of the testing instances are predicted as anomalies (i.e., faulty instances) and half of them are predicted as normal. Note that each result listed in Table 5.1 is the mean value of 10 test runs using cross-validation. The highest result of the competitive baselines for each scenario is underlined. Gain rows in Table 5.1 indicate the performance difference between DV-MPPCA and the best baseline model for every case.

From Table 5.1, it can be seen that the fault detection performance of all the models in the study is outstanding for the complete seasonal datasets (i.e., the original Summer, Winter, and Spring datasets). However, their performance decreases at a great scale when the number of missing data points in the datasets increases. More specifically, when the data is complete (i.e., missing rate $\lambda = 0$), the performance of the models is above 92% for the Summer, and Winter datasets, and above 82% for the Spring dataset. When the rate of missing data points in the data increases to 10%, the performance of the models deteriorates

Table 5.1: Fault Detection Performance of DV-MPPCA and Deep Models In Terms of F1-score (%).

Missing rate	0%	10%	20%	30%	40%	50%	0%	10%	20%	30%	40%	50%
	Dataset 1: 2007 Summer AHU Dataset						**Dataset 2: 2008 Winter AHU Dataset**					
DV-MPPCA	**98.56**	**98.09**	**95.20**	**89.24**	**87.59**	**85.03**	**99.52**	**93.49**	**87.20**	**78.99**	**72.89**	**68.66**
DV-MPPCA†	96.94	82.40	74.91	75.64	75.40	73.93	96.27	56.57	57.93	55.92	54.94	53.62
DV-MPPCA*	92.16	91.75	88.81	85.72	80.71	76.99	99.27	87.16	81.83	73.79	71.51	69.01
DA-MPPCA	97.01	88.48	84.02	76.96	76.76	72.94	99.75	59.17	56.60	54.94	54.15	53.27
NN-MPPCA	99.67	89.26	87.47	77.07	69.92	65.93	99.49	83.48	75.31	64.75	54.81	52.24
DONUT	97.01	91.80	86.40	81.00	77.56	67.74	92.07	78.95	72.43	68.21	63.24	66.69
Gain (%)	-1.10	6.29	6.39	3.51	6.87	8.04	-0.23	6.34	5.38	5.21	1.38	-0.34
	Dataset 3: 2008 Spring AHU Dataset						**Dataset 4: 2007 - 2008 All Seasons AHU Dataset**					
DV-MPPCA	**95.18**	**81.57**	**76.72**	**69.20**	**65.45**	**64.71**	**87.57**	**83.63**	**72.68**	**70.66**	**63.61**	**61.43**
DV-MPPCA†	96.08	63.42	58.74	58.09	57.99	54.63	85.39	63.55	62.21	61.06	57.98	59.86
DV-MPPCA*	82.48	80.20	76.75	70.52	69.41	66.84	70.84	80.39	70.09	68.45	64.79	63.21
DA-MPPCA	97.58	60.71	58.59	56.70	54.23	54.40	96.08	57.82	57.63	57.41	53.10	52.32
NN-MPPCA	99.80	67.47	57.66	54.12	50.45	48.91	99.63	65.19	59.01	53.98	50.89	49.99
DONUT	87.69	77.89	73.34	69.35	63.99	62.98	74.81	76.77	71.75	69.56	62.79	61.14
Gain (%)	-4.62	1.38	-0.03	-1.32	-3.96	-2.13	-12.06	3.25	0.93	1.1	-1.18	-1.78

for all three datasets, but more significantly for the Winter, and Spring datasets. Among these models, DV-MPPCA†, and DA-MPPCA are more sensitive to missing points. The F1-score of DV-MPPCA† drops from 96.27% to 56.57% for the Winter dataset and from 96.08% to 63.42% for the Spring dataset. For DA-MPPCA, the F1-score sharply decreases from 99.75% to 59.17% for the Winter dataset, and from 97.58% to 60.71% for the Spring dataset. The reason behind these results is that missing data points in the data distort the latent representation of VAE in DV-MPPCA† and DAE in DA-MPPCA. Inaccurate representation from the compression network causes the estimation network unable to accomplish the correct estimation task. The T-SNE visualization depicted in Fig. 5.3 illustrates the reduction of feature representations into a 2-dimensional space for DA-MPPCA, DV-MPPCA†, and the proposed DV-MPPCA model. In scenarios where the dataset is complete (i.e., with no missing data points), the learned feature representations for normal and faulty instances by these models exhibit clear discrimination, facilitating effective detection of faulty instances. However, when confronted with an incomplete dataset (i.e., $\lambda = 10\%$), the capability to distinguish between normal and faulty feature representations becomes challenging for both DA-MPPCA and DV-MPPCA†, resulting in the failure to adequately detect faulty instances. In contrast, the feature representations of normal and faulty instances acquired by DV-MPPCA remain distinguishable even in such circumstances. The downward trend observed in NN-MPPCA's performance is similar to that of DV-MPPCA† and DA-MPPCA, albeit to a lesser extent. Notably, the F1-scores of NN-MPPCA for the Summer and Winter datasets with a 10% missing rate remain relatively high at 89.26% and 83.48%, respectively. However,

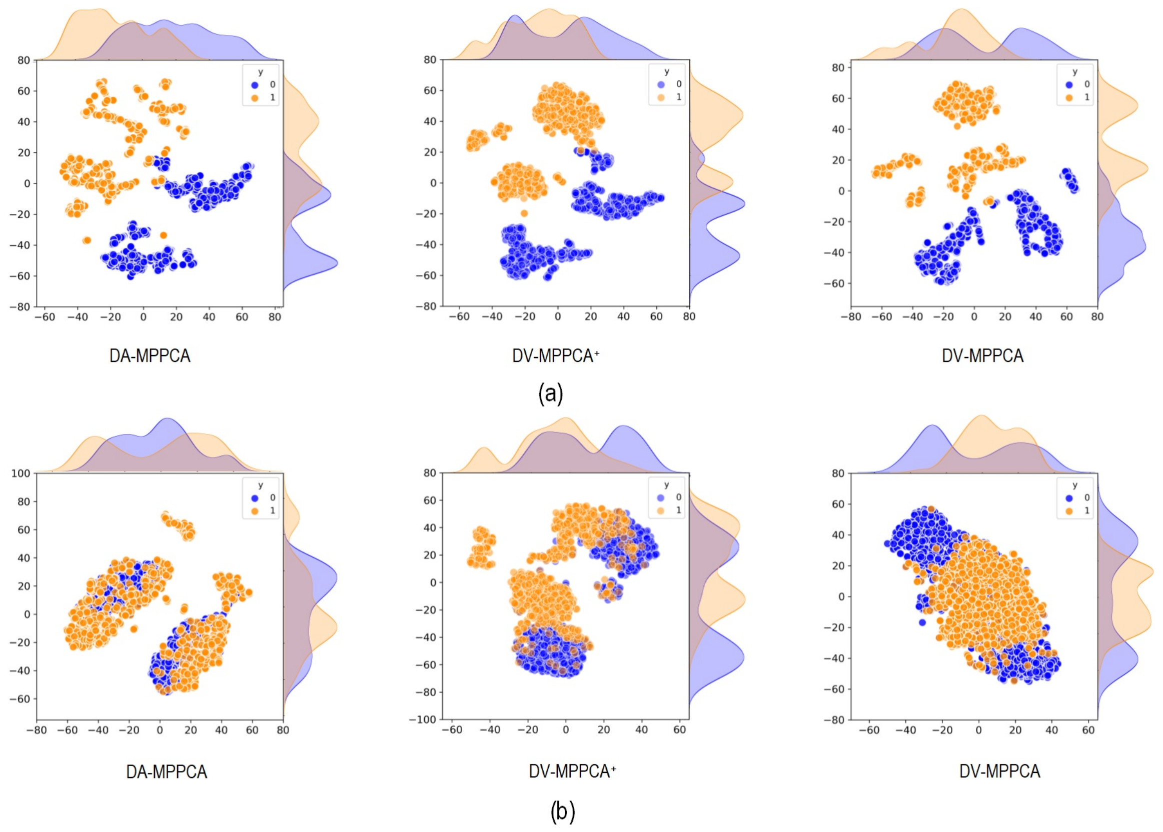

Figure 5.3: T-SNE visualization of the features learned by DA-MPPCA, DV-MPPCA†, and DV-MPPCA on the summer dataset with a missing rate of a) 0%, and b) 10%

for the Spring dataset, there is a significant drop in the F1-score from 99.80% to 67.47%. As the rate of missing data points further increases (e.g., 20%, 30%, etc.), the models' performance experiences a notable decline. For instance, at a 30% missing rate, the F1-scores of the models drop below 90% for the Summer dataset and below 80% for the Winter and Spring datasets.

Fig. 5.4 provides a visual representation of the statistical findings presented in Table 5.1. The graph underscores that while DV-MPPCA follows a comparable downward trend as the other models when the missing rate increases, it exhibits superior performance compared to the competing models across a majority of the experimental scenarios, particularly under conditions of high missing rates. The advantage of DV-MPPCA over the baseline models is highlighted by the green areas of improvement depicted in the sub-figures of Fig. 5.4. More specifically, with a missing rate of 10%, the performance of DV-MPPCA for the Summer, Winter, and Spring datasets are the highest among the models, with F1-score reaching 98.10%, 93.50%, and 81.57%, respectively. Even with a very high missing rate (i.e., 30%), the performance of DV-MPPCA is still satisfactory with F1-score for the Summer, Winter, and Spring datasets equal to 89.24%, 79.00%, and 69.20%, respectively. These results are solid proof of the DV-MPPCA's resistant capability to missing data points in the data. For a more comprehensive analysis, we delve into the training process of DV-MPPCA on the Summer dataset, with a missing rate of 20% (refer to Fig. 5.5). To enhance clarity, the component losses displayed in Fig. 5.5a have been normalized within the range of [0, 1]. From Fig. 5.5, we can see that even working on incomplete data, the component losses of DV-MPPCA still decrease properly over training epochs. However, at the epoch of 123, the component losses increase sharply. This means the training process of DV-MPPCA becomes unstable after the epoch of 123. By setting a patience parameter of 20 on the validation loss (i.e., the same parameter value is set for all the deep models in this study), the training process is automatically stopped at the epoch of 123 (Fig. 5.5b). Another noticeable point is that the loss curve of VAE-ELBO (i.e., the modified evidence lower bound loss) has minor fluctuations during the training process. The reason for this phenomenon is that VAE works directly on the incomplete input leading to unstable loss calculation. In contrast, NN-MPPCA works on the stable, accurate latent representation of VAE. That is why the loss curves of MPPCA-LOG (i.e., the complete data log-likelihood), and MPPCA-RES (i.e., the reconstruction loss) are smooth during the training process.

Among the deep baseline models, DV-MPPCA* stands out as the most effective. Just like DV-MPPCA, DV-MPPCA* showcases resilience to missing data points due to its utilization of the modified ELBO, which effectively mitigates the impact of missing data during both training and estimation stages. By using the reconstruction probability of VAE as the anomaly indicator, however, the fault detection performance of DV-MPPCA* is almost inferior to that of DV-MPPCA. The F1-scores of these two models listed in Table 5.1 proved that the use of MPPCA to estimate the anomaly score is more efficient than using VAE. Another model applying modified ELBO in this study is DONUT. However, the performance of DONUT is poorer than that of DV-MPPCA and DV-MPPCA*. The reason behind this inferiority is that while DONUT uses the modified ELBO to train VAE on the incomplete training data, it applies the Markov Chain Monte Carlo technique to recover missing points in testing samples and estimates the anomaly score on reconstructed testing data. The bias between the two processes is the reason for deteriorating the performance of DONUT.

We also extended the validation to encompass the entire year by combining AHU data from three distinct seasons. Specifically, the models were trained on normal data spanning all seasons and subsequently evaluated on the all-season testing dataset, encompassing both normal and faulty instances. The results outlined in Table 5.1 demonstrated that models utilizing VAE as a compression network, such as DV-MPPCA, DV-MPPCA*, DV-MPPCA†, and DONUT, exhibited comparatively poorer performance when the dataset

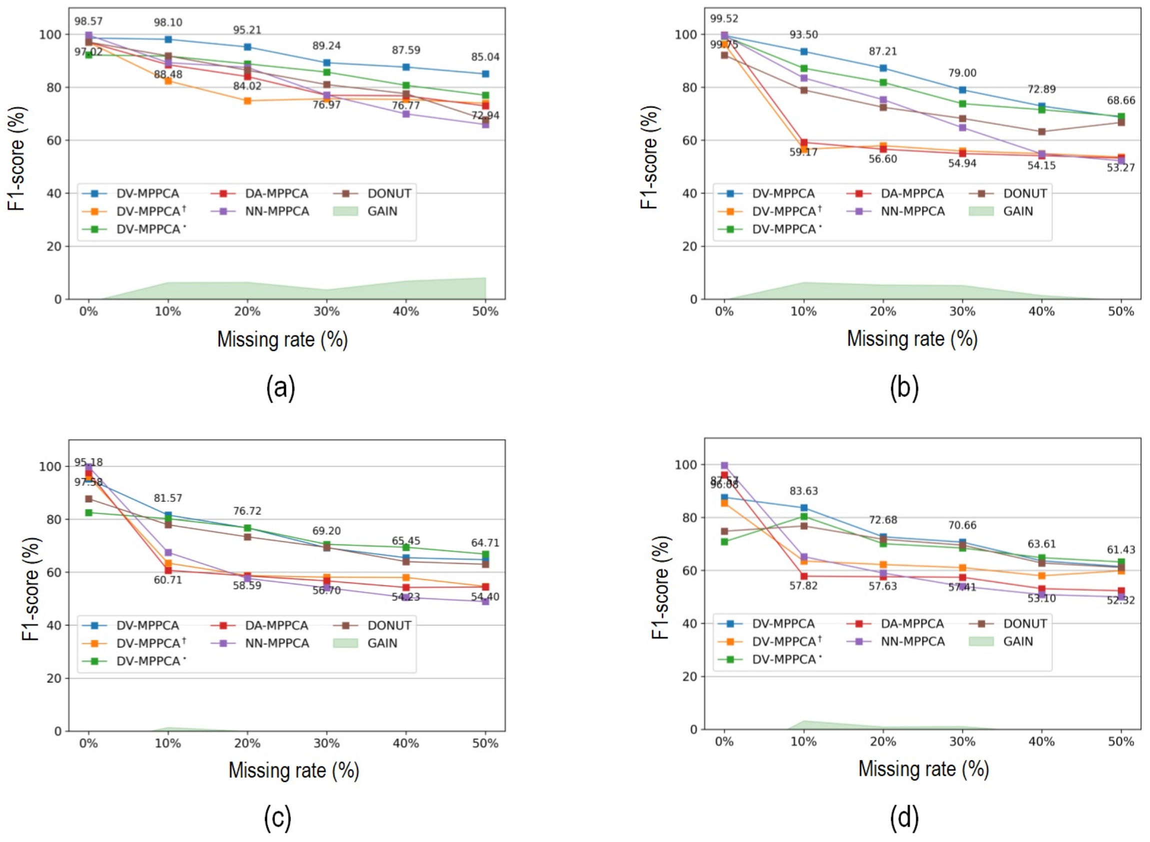

Figure 5.4: F1-scores of DV-MPPCA and deep models for three seasonal AHU datasets with different missing rates: a) summer dataset, b) winter dataset, c) spring dataset, and d) all seasons dataset

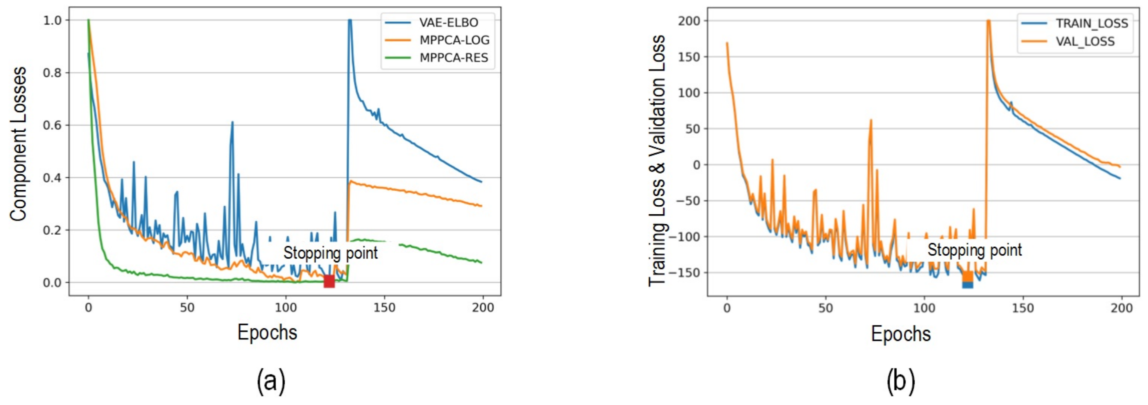

Figure 5.5: Training process monitoring: a) component losses on validation data, b) total losses on training data and validation data.

was complete (i.e., $\lambda = 0$). This outcome is attributed to the nature of these models, where the latent representations of normal training instances need to be condensed into a VAE's prior distribution (i.e., an

Table 5.2: Fault Detection Performance of DV-MPPCA and Classic Models.

Missing rate	0%	10%	20%	30%	40%	50%	0%	10%	20%	30%	40%	50%
	Dataset 1: 2007 Summer AHU Dataset						**Dataset 2: 2008 Winter AHU Dataset**					
DV-MPPCA	**98.56**	**98.09**	**95.20**	**89.24**	**87.59**	**85.03**	**99.52**	**93.49**	**87.20**	**78.99**	**72.89**	**68.66**
GMM	99.80	96.01	89.49	86.14	82.17	75.82	99.67	72.59	69.56	65.22	63.21	59.73
MPPCA	99.77	91.93	89.57	82.33	75.92	70.91	99.67	84.42	77.71	67.00	58.30	54.55
OC-SVM	99.86	89.75	87.98	84.10	78.49	73.66	99.81	82.74	78.93	73.79	69.34	61.29
IF	84.77	68.99	57.20	52.57	51.12	50.81	74.77	65.34	58.01	52.40	50.36	49.80
KDE	99.68	61.17	61.71	60.05	62.21	64.13	99.57	57.77	55.59	55.41	57.77	58.14
Gain (%)	-1.29	2.09	5.63	3.1	5.42	9.22	-0.29	9.08	8.27	5.21	3.55	7.37
	Dataset 2: 2008 Spring AHU Dataset						**Dataset 4: 2007 - 2008 All Seasons AHU Dataset**					
DV-MPPCA	**95.18**	**81.57**	**76.72**	**69.20**	**65.45**	**64.71**	**87.57**	**83.63**	**72.68**	**70.66**	**63.61**	**61.43**
GMM	99.85	68.08	59.50	54.08	54.26	52.87	99.79	64.58	60.40	56.89	53.84	52.76
MPPCA	99.59	67.99	65.28	60.41	56.49	53.98	99.85	68.61	64.01	58.38	54.86	53.41
OC-SVM	99.71	65.74	63.99	62.18	60.32	57.14	99.84	63.66	61.86	59.77	57.38	55.32
IF	78.82	63.52	55.79	52.56	48.85	48.52	65.94	57.82	53.27	51.24	50.33	49.59
KDE	99.75	55.09	52.73	54.44	53.43	52.67	99.67	55.23	54.35	52.94	51.65	51.79
Gain (%)	-4.67	13.49	11.44	7.02	5.13	7.56	-17.52	11.69	6.7	7.12	6.42	4.07

isotropic Gaussian distribution). However, due to varying weather conditions associated with different seasons, the collected normal data exhibit diverse characteristics. Consequently, the distribution of feature representations for normal data forms scattered clusters within a confined space. This increases the likelihood that latent vectors of faulty instances might lie in close proximity to these scattered normal clusters, leading to the misclassification of normal and faulty instances. For incomplete datasets, the performance of DV-MPPCA is still superior to most of the baseline models.

Comparisons with Other Methods

The superiority of DV-MPPCA is also verified by the comparison tests with classic baseline models such as GMM, MPPCA, One-class Support Vector Machine (OC-SVM), isolation forest (IF), and KDE. GMM, MPPCA, and KDE are classic density estimation models that have been introduced in Section 5.2. OC-SVM Y. Chen, Zhou, Huang (2001) is a representative of kernel-based approaches for anomaly detection. The model uses the radial basis function (RBF) as a kernel. IF F. T. Liu et al. (2012) is an ensemble model of random trees where anomalies are isolated by the partition of random features and their values. In this comparative study, the classic models are implemented using the Scikit-learn library with default parameters. The AHU data are standardized before being used by the models. Table 5.2 outlines the comparative outcomes of DV-MPPCA and the classic models in relation to the F1-score. Additionally Fig. 5.6 provides a visual representation of the statistical findings presented in Table 5.2.

From Table 5.2, it can be seen that the performance of the classic models is outstanding for the complete seasonal datasets (i.e., the Summer, Winter, and Spring datasets). Similar to the deep models in the ablation study, however, the performance of the classic models declines sharply with the increase of missing rates, especially for the Winter and Spring datasets. Without missing data points in the data, the F1-score of the classic models, except IF, is over 99%. Although all the models are affected by missing data points, KDE seems to be more fragile than the others. More specifically, the F1-score of KDE drops from 99.69% to 61.17% for the Summer dataset with a missing rate of 10%. The drop is also more significant for the Winter, and Spring datasets, from 99.57% to 57.77%, and from 99.75% to 55.09%, respectively. While GMM works well on the Summer dataset, MPPCA, and OC-SVM have better performance on the Winter, and Spring datasets. For example, the F1-score of GMM reaches 96.01% for the Summer dataset with a missing rate of 10%. MPPCA and OC-SVM have better performance than the other classic models for the Winter dataset with the F1-score of 84.42%, and 82.74%, respectively. Out of the considered models, Isolation Forest (IF) exhibits the least favorable performance. The F1-score achieved by IF is approximately 50% when the missing rate surpasses 20%. Although the performance of DV-MPPCA may appear comparable to that of the classic models when utilizing complete data, its effectiveness significantly surpasses that of the models when dealing with incomplete datasets. Especially for the Winter and Spring datasets, the improvement of DV-MPPCA in comparison to the best classic baselines reaches up to 13.49% for a missing rate of 10%, and up to 11.44% for a missing rate of 20%.

5.4.3 Potential of DV-MPPCA for Reducing False Alarm Rate

This experiment investigates the potential of DV-MPPCA to lower the rate of false alarms. As mentioned earlier in Section 5.3.5, our framework has the capability to utilize both the trained VAE and NN-MPPCA models to predict anomalies. Consequently, we can leverage the anomaly scores from both models collaboratively to effectively diminish the false alarm rate within the fault detection system. More specifically, given that the number of faulty and normal instances in the testing subset is equal, the thresholds of VAE and NN-MPPCA (i.e., Th_{VAE}, and Th_{MPPCA}) are estimated to identify faulty instances, ensuring that around 50% of the test instances are predicted as faulty. During the online monitoring process, for each individual test instance, the anomaly scores derived from the trained VAE and NN-MPPCA are compared to their respective thresholds. If both the anomaly scores are higher than their respective thresholds (i.e., $a_{\text{VAE}} > Th_{\text{VAE}}$, and $a_{\text{MPPCA}} > Th_{\text{MPPCA}}$), the anomalous alarm is triggered. By doing this, we can increase the precision

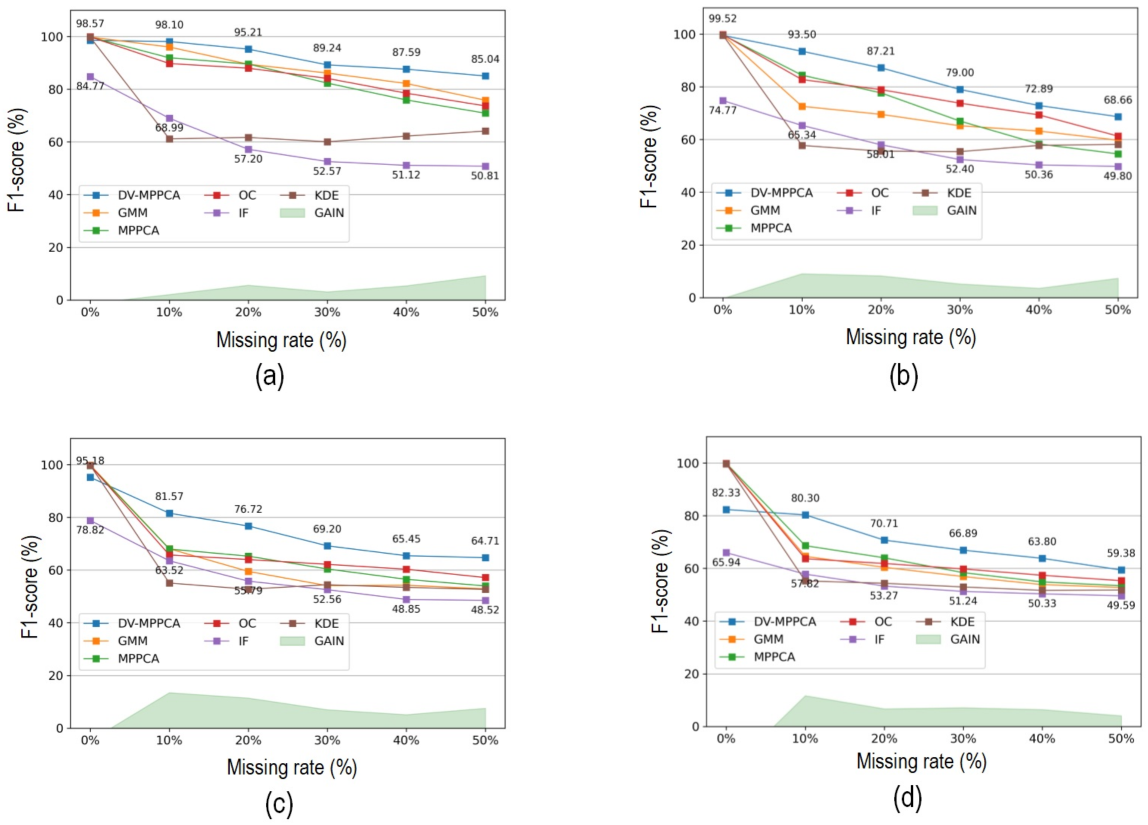

Figure 5.6: F1-scores of DV-MPPCA and classic models for three seasonal AHU datasets with different missing rates: a) summer dataset, b) winter dataset, and c) spring dataset.

of the model but at the cost of low recall. To verify this idea, we implemented the new model that uses both the anomaly scores to make a prediction and name it DV-MPPCA‡. The new model is compared with DV-MPPCA and the best baselines in the previous studies (i.e., DV-MPPCA*, MPPCA, OC-SVM) in terms of the precision metric.

Table 5.3 presents a comparison of precision results between DV-MPPCA‡ and its corresponding baselines. The table indicates that the approach of utilizing both anomaly scores for predicting anomalies demonstrates its effectiveness, particularly when dealing with datasets containing ambiguous anomalies due to a high rate of missing data points. As an instance, in the case of the Spring dataset where the missing rate exceeds 10%, the precision enhancement achieved by DV-MPPCA‡ can reach up to 6.33%. For the Winter dataset with a missing rate higher than 30%, the improvement is even higher, up to 8.25%. However, there are instances where anomalies are readily detectable. For instance, when the models operate on moderately incomplete data, the advancement achieved by DV-MPPCA‡ may not be readily apparent and, in some cases, might even experience a decline. This phenomenon can be attributed to the fact that both the anomaly

Table 5.3: Fault Detection Performance of DV-MPPCA and Best Baseline Models in Terms of Precision.

Missing rate	0%	10%	20%	30%	40%	50%	0%	10%	20%	30%	40%	50%
	Dataset 1: 2007 Summer AHU Dataset						**Dataset 2: 2008 Winter AHU Dataset**					
DV-MPPCA‡	**97.85**	**97.91**	**94.83**	**88.04**	**88.13**	**86.72**	**98.62**	**94.03**	**87.24**	**85.19**	**79.80**	**77.06**
DV-MPPCA	98.65	98.09	94.97	89.18	87.57	85.08	99.04	93.27	87.31	78.82	72.94	68.66
DV-MPPCA⋆	91.98	91.71	88.61	85.61	80.68	76.90	98.78	87.27	81.70	73.84	71.35	68.81
MPPCA	99.60	91.96	89.54	82.25	75.91	70.84	99.75	84.49	77.86	67.07	58.26	54.64
OC-SVM	99.80	89.83	87.97	84.06	78.45	73.48	99.62	82.76	79.01	73.77	69.08	61.31
Gain (%)	-1.94	-0.18	-0.13	-1.14	0.57	1.64	-1.14	0.76	-0.07	6.36	6.87	8.25
	Dataset 3: 2008 Spring AHU Dataset						**Dataset 4: 2007 - 2008 All Seasons AHU Dataset**					
DV-MPPCA‡	**96.09**	**87.69**	**83.10**	**73.31**	**72.50**	**69.90**	**89.07**	**82.07**	**73.70**	**70.01**	**68.87**	**64.82**
DV-MPPCA	95.15	81.70	76.74	69.28	65.59	64.63	86.04	83.71	72.54	70.55	63.61	61.41
DV-MPPCA⋆	82.35	80.09	76.77	70.54	69.40	66.76	70.89	80.38	70.16	68.52	64.85	63.25
MPPCA	99.89	67.87	65.12	60.52	56.54	54.08	99.78	68.68	63.97	58.36	54.81	53.34
OC-SVM	99.87	65.72	64.05	62.12	60.37	57.06	99.74	63.57	61.88	59.80	57.31	55.24
Gain (%)	-3.8	5.99	6.33	2.77	3.11	3.14	-10.72	-1.64	1.17	-0.54	4.03	1.57

score estimated by VAE and the anomaly score derived from MPPCA hold an equivalent significance in the decision-making process. If the accuracy of decisions made by VAE is considerably lower than those made by MPPCA, it could deteriorate the final decisions of the model.

5.5 Conclusion

Classic density estimation models and deep generative models are among the best anomaly detection approaches. In order to harness the strengths of both approaches, we introduced an innovative framework named DV-MPPCA. This framework combines the utilization of VAE as the compression network and NN-MPPCA as the estimation network. By applying the modified ELBO loss function on VAE, DV-MPPCA can work well on incomplete AHU datasets, even with a high missing rate. The formulation and optimization of the loss function ensure the convergence of DV-MPPCA to its optimal state. he extensive experimental outcomes clearly demonstrate that, across various scenarios, DV-MPPCA consistently outperforms both the traditional and modern baseline models. Even with a missing rate of 10%, the performance of DV-MPPCA is still guaranteed, with the F1-score for the Summer, Winter, and Spring datasets of 98.10%, 93.50%, and 81.57%, respectively.

The ability of DV-MPPCA to mitigate false alarms is also under examination. Through the combined utilization of anomaly scores derived from VAE and MPPCA, the framework showcases the potential to enhance the precision of predictions. To be more specific, when confronted with situations where data ambiguity stems from a high missing rate, the framework demonstrates a noteworthy precision enhancement.

In particular, this improvement reaches up to 6.33% for the Spring dataset and 8.25% for the Winter dataset.

For reproducibility and future improvement by other researchers, the complete source code of this study is provided in the following repository: `https://github.com/viettra-xai/DV-MPPCA`.

Chapter 6

Latent Code Description for Unsupervised AHU Fault Detection using Adaptive Adversarial Autoencoder

6.1 Introduction

Nowadays, heating, ventilation, and air-conditioning (HVAC) systems are widespread in both commercial and residential buildings. They play a crucial role in maintaining thermal and air comfort within these structures. In order to ensure optimal thermal conditions and fresh air, HVAC systems require a significant amount of energy. Statistics reveal that HVAC systems contribute to 57% of energy consumption in U.S. buildings and over 40% of the global total B. Li et al. (2021); Yan, Chong, Mo (2020). Despite their essential function, a considerable portion of this energy is wasted due to inadequate maintenance, component deterioration, and control failures. Without timely intervention, minor issues in components can escalate, ultimately leading to system breakdown. This not only results in energy wastage but also causes discomfort in commercial buildings. While scheduled maintenance can mitigate catastrophic failures and extend the system's operational reliability, the associated costs, including tools and expert services, are high. Moreover, unexpected failures may occur during maintenance periods. To prevent the adverse consequences of unexpected disruptions, it is imperative to detect early signs of faults in building HVAC systems using accurate fault detection and diagnosis (FDD) techniques Tra et al. (2022a, 2022b); Yan, Huang, et al. (2020);

Yan, Su, et al. (2020).

Operating as tools for monitoring processes, fault detection (FD) methods are formulated to spot early-stage faults emerging within a system, subsequently notifying operators of their presence. This early identification empowers system operators and maintenance personnel to instantly intervene and implement appropriate measures to address the identified issues. FD techniques are categorized into supervised, semi-supervised, or unsupervised methods Z. Chen et al. (2023). In supervised fault detection, models are trained with both normal operation and labeled fault data to determine whether incoming data is faulty or not Du, Fan, Jin, Chi (2014); Liang Du (2007). However, the challenge lies in obtaining enough labeled fault data to build robust models. Semi-supervised approaches are more suitable when labeled fault data is scarce. These methods use the limited labeled data for initial training and then predict labels for unlabeled data, iteratively enhancing the labeled dataset Yan, Zhong, Ji, Huang (2018). While effective with limited labeled data, semi-supervised methods are more computationally intensive than supervised learning. Unsupervised methods address these limitations by not requiring fault labels for model construction. Instead, they learn regular patterns from the dataset and detect faults by identifying deviations from these patterns Jang et al. (2021); Jiang et al. (2019). Many studies have applied anomaly detection (AD) techniques to unsupervised fault detection. Similar to faults, anomalies often present unique patterns distinct from normal operations. By learning normal patterns from data collected during proper functioning, unsupervised AD algorithms can detect and highlight unusual states, such as unforeseen faults Riazi et al. (2019); Tra, Amayri, Bouguila (2023).

Among anomaly detection methods, traditional shallow approaches such as one-class support vector machine (OC-SVM) Schölkopf et al. (2001), support vector data description (SVDD) Tax Duin (2004), kernel density estimation (KDE) Davis, Lii, Politis (2011), nearest neighbour algorithms Knorr, Ng, Tucakov (2000), principal component analysis (PCA)-based Hoffmann (2007), and Gaussian mixture model (GMM)-based methods Roberts Tarassenko (1994) exhibit satisfactory performance for low-dimensional data, they are prone to challenges in high-dimensional scenarios due to the limitations imposed by the curse of dimensionality. The first key issue is increased sparsity, where instances are more likely to be seen as rare events, resulting in a higher false alarm rate. The second issue is computational complexity, as algorithms handling high-dimensional data often face significantly higher computational demands. Additionally, more dimensions increase the risk of overfitting to noise rather than capturing meaningful patterns, leading to poor generalization on unseen data. These challenges have led to a growing interest in deep learning approaches for anomaly detection (AD). Deep AD aims to reduce the need for manual feature engineering by training flexible, multilayered neural networks that can learn effective representations directly from the data. Major deep AD approaches include deep autoencoder variants Xu et al. (2018), deep one-class classification Ruff et al. (2018), and methods based on deep generative models Zong et al. (2018).

In this paper, we introduce a novel deep anomaly detection (AD) approach called the adaptive adversarial autoencoder (AdaAAE), which can operate as either a one-class classification model or a density estimation model. This method is based on the adversarial autoencoder (AAE) Makhzani, Shlens, Jaitly, Goodfellow, Frey (2015). AAE is a probabilistic autoencoder that uses generative adversarial networks (GAN) Goodfellow et al. (2020) to perform variational inference Kingma Welling (2013). The AAE consists of three primary components: an encoder, a decoder, and a discriminator. It is trained to meet two objectives: a standard reconstruction error criterion (through a reconstruction phase) and an adversarial training criterion (through a regularization phase). As a result of this training process, the encoder becomes proficient at aligning the aggregated posterior distribution of the latent code vectors with a specified prior distribution. Concurrently, the decoder in the adversarial autoencoder acts as a deep generative model, mapping the imposed prior distribution onto the data distribution. Although AAE was initially designed as a generative model, its inherent properties make it highly effective for anomaly detection (AD). To adapt AAE and its variants for AD tasks, we first train the AAE using normal data, enabling it to learn the normal patterns and constrain the latent codes of normal instances within a specific distribution boundary. Specifically, we can set the prior distribution of the AAE to be an isotropic Gaussian distribution. After training, the aggregated posterior distribution (i.e., the latent distribution) of normal projections aligns with this prior distribution, taking the form of an isotropic Gaussian distribution. As a result, the projections of normal instances should be located within the spherical boundary of the posterior distribution. During the testing phase, the projections of abnormal instances in the latent space are expected to deviate from this posterior distribution, often falling outside the boundary. Detection can thus be achieved by identifying these deviations.

However, if the latent distribution of normal instances is not compact, there is a possibility that the projections of anomalous instances could fall within the decision boundary. This can result in incorrect detections. To address this issue, our proposed framework, AdaAAE, applies deep support vector data description (DSVDD) to the encoder to progressively enhance the compactness of the latent code distribution of normal instances after each training iteration. This implies that, beyond the reconstruction and regularization phases of the original AAE, each training epoch of AdaAAE incorporates a compactness phase. This phase aims to condense the latent codes (i.e., projections) within the boundary and accordingly reduce its radius. At the end of each epoch, the radius of the latent distribution boundary is measured and used to update the standard deviation of the prior distribution before advancing to the next training epoch. Repeating the three training phases—reconstruction, regularization, and compactness—enhances the compactness of the latent distribution while maintaining its spherical boundary shape in line with the prior distribution. Another innovative aspect of this method is its ability to precisely determine the anomaly threshold by using the radius value computed after training, eliminating the need for prior knowledge of the anomaly ratio in the test samples. This enhances the online anomaly detection process and makes it suitable for real-world applications. Specifically, at the end of the training phase, the radius of the latent sphere (i.e., latent boundary) is estimated, and this estimated radius can then be utilized as the anomaly threshold. Any instance whose distance from the origin in the latent sphere exceeds this threshold can be identified as anomalous. Another approach involves using the density estimation method, where the radius of the latent sphere is defined as the distance from the origin to a latent point on the boundary of the sphere. We then calculate the likelihood of this boundary point with respect to the latent distribution. This likelihood value serves as the anomaly threshold, against which the likelihood of latent codes of unseen instances is compared.

The main contributions of the paper are summarized as follows:

(1) A new anomaly detection method, called the adaptive adversarial autoencoder (AdaAAE), is introduced. This technique is based on the adversarial autoencoder (AAE) framework and incorporates deep support vector data description (DSVDD) to progressively improve the compactness of the latent code distribution.

(2) The training process and the procedure for iteratively updating the prior distribution are carefully designed to ensure the simultaneous convergence of all components of AdaAAE.

(3) The approach to accurately estimate the anomaly threshold, without requiring prior information about the anomaly ratio in the testing samples, is proposed. This threshold is calculated based on the radius of the latent distribution at the end of the training phase.

(4) Experiments involving various seasonal datasets from a real air handling unit system, developed as part of the ASHRAE research project 1312 (RP-1312), have been conducted. These comparative studies provide evidence supporting the applicability and superiority of the proposed framework compared to other state-of-the-art models.

The following sections of the paper are organized to provide a comprehensive understanding of the study. Section 6.2 delves into the fundamental background of the research, offering an overview of related works and existing literature in the field. Section 6.3 presents the proposed framework in detail, highlighting the mathematical innovations embedded within the constituent models. Section 6.4 showcases the validation datasets derived from the ASHRAE research project 1312 (RP-1312) and describes the extensive experiments conducted to evaluate the framework. Lastly, Section 6.5 offers concluding remarks and summaries, encapsulating the key findings and overall significance of the study.

6.2 Background and Related Works

6.2.1 One-Class Classification

Support vector data description (SVDD) Tax Duin (2004), is a method closely related to OC-SVM Schölkopf et al. (2001). Instead of employing a hyperplane, SVDD utilizes a hypersphere to partition the data. The goal of SVDD is to identify the smallest hypersphere with its center at $c \in F_k$, where F_k is a Hilbert space with a feature mapping associated with the kernel k, and a radius of $R > 0$ encompassing the majority of the data points within the feature space F_k. The SVDD primal problem is defined as follows:

$$\min_{R,c,\boldsymbol{\xi}} R^2 + \frac{1}{\nu n} \sum_i \xi_i \tag{6.1}$$

subject to

$$\|\phi_k(\mathbf{x}_i) - c\|^2_{F_k} \leq R^2 + \xi_i, \quad \xi_i \geq 0, \quad \forall i. \tag{6.2}$$

The inclusion of nonnegative slack variables $\xi_i \geq 0$ allows for an adaptable boundary, and the hyperparameter $\nu \in (0, 1]$ governs the balance between the penalties associated with ξ_i and the size of the hypersphere. Data points located outside the hypersphere, defined as $|\phi_k(\mathbf{x}) - c|^2_{F_k} > R^2$, are categorized as anomalies.

Selecting kernels and manually crafting relevant features can be problematic, especially with complex data. Deep one-class classification methods tackle these challenges by autonomously learning neural network feature mappings $\phi_\omega : \mathcal{X} \to \mathcal{Z}$ from the data. One-class deep SVDD Ruff et al. (2018), introduced in the work by Ruff et al. Ruff et al. (2018), presents a simplified variant with the following objective:

$$\min_{\omega,\mathbf{c}} \frac{1}{n} \sum_{i=1}^{n} \|\phi_\omega(\mathbf{x}_i) - \mathbf{c}\|^2 + R. \tag{6.3}$$

In this formulation, the neural network transformation $\phi_\omega(\cdot)$ is trained to minimize the mean squared distance between all data points and the center $\mathbf{c} \in \mathcal{Z}$. This simplified objective is known to converge faster and has proven effective in many scenarios. A common issue in deep one-class classification, particularly in deep SVDD, is preventing the collapse of the feature map, where $\phi_\omega \equiv \mathbf{c}$. Several strategies have been proposed to tackle this problem. For instance, Ruff et al. in Ruff et al. (2018) suggested measures such as fixing the center of the hypersphere, omitting bias terms in neural networks, and avoiding bounded activation functions to prevent hypersphere collapse in deep SVDD. In another study, Ruff et al. Ruff, Zemlyanskiy, Vandermeulen, Schnake, Kloft (2019) introduced the context vector data description (CVDD) method, which leverages pre-trained models for distributed vector representations and freezes the pre-trained embedding to prevent the manifold collapse phenomenon. Hojjati Hojjati Armanfard (2023) proposed the deep autoencoding support vector data descriptor (DASVDD) method, which simultaneously learns the parameters of an autoencoder while minimizing the volume of an enclosing hypersphere on its latent representation. Incorporating the reconstruction error into the loss function ensures that DASVDD avoids hypersphere collapse.

6.2.2 Adversarial Learning Models

Generative adversarial networks (GAN) Goodfellow et al. (2020) were one of the first models to use adversarial learning for generative tasks, such as image generation. GAN comprises two models: a generator G and a discriminator D. The discriminator learns to distinguish real data from generated data, while the generator aims to produce data that confuses the discriminator, making it hard to discern. This training alternates between G and D, seeking a Nash equilibrium using this objective:

$$\min_G \max_D \mathbb{E}_{\mathbf{x}\sim p_d}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{6.4}$$

Here, $\mathbf{x}$ represents a data point, and $\mathbf{z}$ is a sample from the prior distribution $p(\mathbf{z})$. The primary goal of this training process is to make the generator's sample distribution closely match the data distribution, referred to as p_d.

Numerous studies have utilized GAN for anomaly detection. AnoGAN Schlegl et al. (2017) is one of the early methods utilizing GAN for anomaly detection. In AnoGAN, the GAN is trained to map latent codes from a prior distribution to generated instances that resemble the normal instances in the dataset. When a new instance arrives, AnoGAN searches the prior distribution to identify a latent code that generates an instance closely matching the new one. The discrepancy between the generated instance and the new instance acts as an indicator of anomalies. However, AnoGAN encounters significant computational challenges due to the intensive search process required in the latent space. To mitigate this, researchers have introduced more efficient approaches. Akcay et al. Akcay, Atapour-Abarghouei, Breckon (2019) enhanced the generator in GANomaly by incorporating encoder-decoder-encoder sub-networks and using an extra encoder network to map generated images to their latent representations. Anomaly scores are calculated based on latent reconstruction errors. Schlegl et al. Schlegl, Seeböck, Waldstein, Langs, Schmidt-Erfurth (2019) introduced fast-AnoGAN (f-AnoGAN), an anomaly detection method that employs a fast mapping technique for new data into the GAN's latent space. This technique uses a trained encoder for mapping, and anomalies are identified using a combined anomaly score derived from the discriminator's feature residual error and the image reconstruction error of the trained model.

The adversarial autoencoder (AAE) Makhzani et al. (2015) is another model that utilizes adversarial learning. AAE integrates the capabilities of GANs with the feature extraction strengths of autoencoders (AE). By using the autoencoder's encoder as the generator, AAE can map a new incoming instance into the encoder's latent space. This makes AAE suitable for anomaly detection tasks without needing additional sub-networks for inverse mapping, unlike GAN-based methods. In this context, the position of the latent code vector of a new instance within the prior distribution serves as an anomaly indicator. Several studies have employed different versions of AAE for anomaly detection. For example, Li et al. D. Li, Tao, Liu, Wang (2021) developed the center-aware adversarial autoencoder (CA-AAE), which employs two encoders. Initially, an AAE is used to evaluate the anomaly levels of training instances in the pre-subspace of the first encoder. The insights gained from this step then guide the training of the second encoder, enhancing the compactness of the post-subspace. In their study, Jang et al. Jang et al. (2021) utilized the AAE technique to extract significant features from process variables. These extracted features, which conform to a predetermined distribution, are then used to create metrics that measure the degree of anomalies in the process variables. Li et al. X. Li et al. (2023) proposed the OTB-AAE model, where AAE is employed to capture the normality distribution of high-dimensional images and detect abnormalities in industrial settings. An output-turn-back structure (OTB) is incorporated into the AAE to enhance the discriminant capability of the discriminator. Wu et al. Wu, Zhao, Sun, Yan, Chen (2020) proposed the fault attention generative probabilistic adversarial autoencoder (FGPAA) to identify low-dimensional patterns in high-dimensional signal data. By utilizing autoencoder features, this approach minimizes information loss during feature extraction, enabling the creation of a fault-attention abnormal state indicator based on the probability density in the latent space and the reconstruction error.

6.3 Methodologies

In this section, we present our innovative method known as the adaptive adversarial autoencoder (AdaAAE). The architecture and training process of the AdaAAE model is comprehensively described in Section 6.3.1. This section will provide an in-depth explanation of the various components and mechanisms that constitute the model. Following, we will explore the methodology for calculating anomaly scores and estimating anomaly thresholds in Section 6.3.2. This part will cover the procedures and algorithms used to detect anomalies, offering a thorough understanding of how the model operates in real-time monitoring scenarios.

6.3.1 Offline Framework Modeling

The AdaAAE model we propose is illustrated in Fig. 6.1. Built on the foundation of the original AAE, the structure of AdaAAE closely resembles that of AAE, featuring three primary components: an encoder, a decoder, and a discriminator. The upper section represents a standard autoencoder (AE) network, while the lower section forms the discriminator D of a generative adversarial network (GAN). The key difference between AdaAAE and AAE lies in the training process. Specifically, the training of the AdaAAE model involves three distinct phases: the reconstruction phase, the regularization phase, and the compactness phase. Like AAE, the reconstruction and regularization phases aim to align the aggregated posterior distribution (i.e., the latent code distribution) with the prior distribution. The compactness phase, however, employs DSVDD to ensure that the latent code distribution becomes increasingly compact through training iterations and adaptive updates of the prior distribution.

In this study, we apply the AdaAAE to anomaly detection tasks. Thus, only normal data are used during the training stage, and normal patterns are identified without supervision. Given a training dataset $D_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, the process begins with the encoder transforming the input data $\mathbf{x}$ into a hidden code

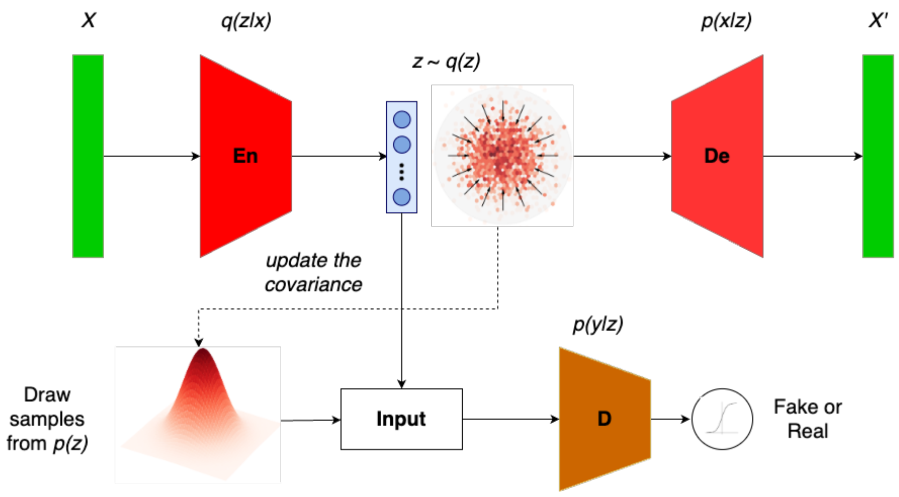

Figure 6.1: The AdaAAE framework architecture incorporating DSVDD to ensure the compactness of the latent code distribution.

vector $\mathbf{z}$. This hidden code vector is then used by the decoder to reconstruct the input as $\hat{\mathbf{x}}$ according to the following expressions:

$$\mathbf{z} = h(\mathbf{x}, \phi) \qquad\qquad \hat{\mathbf{x}} = g(\mathbf{z}, \theta) \tag{6.5}$$

Here, ϕ and θ represent the parameters of the encoder and decoder, respectively.

During the reconstruction phase, the encoder and decoder are trained with the objective of minimizing the reconstruction error, defined as:

$$\mathcal{L}_{rec}(\phi, \theta; \mathbf{x}) = -\mathbb{E}_{\mathbf{x}\sim p_d}[\mathbf{x} \log \hat{\mathbf{x}} + (1 - \mathbf{x}) \log(1 - \hat{\mathbf{x}})] \tag{6.6}$$

where p_d represents the data distribution.

In the regularization phase, the encoder section of the AE serves as the generator G within the adversarial network. The regularization phase undergoes a two-step update process to ensure that the aggregated posterior distribution $q_\phi(\mathbf{z})$ can convincingly mislead the discriminator D into perceiving that a hidden code sampled from $q_\phi(\mathbf{z})$ originates from the prior distribution $p_\psi(\mathbf{z})$. In this context, we define an aggregated posterior distribution $q_\phi(\mathbf{z})$ over the hidden layers as follows:

$$q_\phi(\mathbf{z}) = \int q_\phi(\mathbf{z}|\mathbf{x}) p_d(\mathbf{x}) d\mathbf{x} \tag{6.7}$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ refers to the encoder network.

Initially, the adversarial network updates the discriminator D to distinguish true samples (generated using the prior distribution) from generated samples (represented by the hidden codes computed by the autoencoders). This update aims to minimize the following loss function:

$$\mathcal{L}_{reg^\star}(\psi; \mathbf{x}, \mathbf{z}) = -\mathbb{E}_{\mathbf{z}\sim p_\psi}[\log D(\mathbf{z})] - \mathbb{E}_{\mathbf{x}\sim p_d}[\log(1 - D(G(\mathbf{x})))] \tag{6.8}$$

Here, ψ represents the parameters of the discriminator D. Subsequently, the adversarial network updates the generator G, which simultaneously serves as the encoder of the autoencoder (AE). This update process is designed to perplex the discriminative network and is guided by the minimization of the following loss function:

$$\mathcal{L}_{reg^\dagger}(\phi; \mathbf{x}) = -\mathbb{E}_{\mathbf{x}\sim p_\mathrm{d}}[\log D(G(\mathbf{x}))] \tag{6.9}$$

After each training epoch of the reconstruction and regularization phases, the aggregated posterior distribution of hidden code vectors, $q_\phi(\mathbf{z})$, is progressively adjusted to match the predefined prior distribution, $p_\psi(\mathbf{z})$. In this study, we aim to confine the latent representations of normal data instances within the boundaries of a defined spherical distribution. Therefore, the initial form of the prior distribution is configured as an isotropic Gaussian distribution, $\mathcal{N}(0, \sigma^2\mathrm{I})$. This choice ensures that the boundary for capturing normal latent points is a simple sphere centered at the origin. Any instance that deviates from normal instances will exhibit latent representations positioned outside this spherical boundary, making it easy to detect as an anomaly.

However, if the latent distribution of normal instances is not compact, there is a risk that the latent codes of anomalous instances may fall within the decision boundary, resulting in misdetection. To address this issue, this research devises an innovative approach to progressively enhance the compactness of the latent code distribution of normal instances after each training iteration. Specifically, immediately after completing each epoch of the reconstruction phase and the regularization phase, we take steps to condense the encoder's latent space. This consolidation process, referred to as the compactness phase, involves minimizing the objective function of the soft-boundary Deep SVDD Ruff et al. (2018) to update the parameters ϕ of the encoder, as described below:

$$\mathcal{L}_{svd}(\phi; \mathbf{x}) = R^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \max\{0, \|h(\mathbf{x}_i, \phi) - \mathbf{c}\|^2 - R^2\} \tag{6.10}$$

where R represents the radius of the latent distribution and $\mathbf{c}$ denotes its center. Minimizing R^2 reduces hypersphere volume, while the second term penalizes points outside it ($\|h(\mathbf{x}, \phi) - \mathbf{c}\| > R$). Hyperparameter $\nu \in (0, 1]$ balances sphere volume and boundary violations, allowing some points to be mapped outside the sphere. Since network parameters ϕ of the encoder and radius R typically operate on different scales, they need to be updated in an alternating fashion. In each epoch, we follow a specific process: we train the network parameters of the encoder while keeping the radius R fixed. Subsequently, once this phase is completed, we determine the radius R based on the data representations obtained from the network using the latest updated network parameters ϕ. Let $D = \{d_1, d_2, \ldots, d_n\}$ represent the distances of latent representations from the sphere's center, where $d_i = |h(\mathbf{x}_i, \phi) - \mathbf{c}|$. To find R, we employ a straightforward line search approach as follows:

$$R = \text{PERCENTILE}(D, 100 \times (1 - \nu)) \tag{6.11}$$

Here, ν denotes the proportion of points permitted to lie outside the sphere.

Through the reconstruction and regularization phases, the latent distribution can be approximated as a normal distribution, $q_\phi(\mathbf{z}) \sim \mathcal{N}(0, \sigma^{\star 2}\mathrm{I})$. Additionally, after each epoch of the compactness phase, the radius R of the latent distribution undergoes a reduction. Before proceeding to the next training epoch, we propose updating the standard deviation $\sigma^\star$ based on the current radius R using the empirical rule of a normal distribution. More specifically, to encompass more than 95% of instances of the latent distribution, the radius R of the n-sphere boundary must exceed $2 \times \sigma^\star$. In this study, we opted for $R = 3 \times \sigma^\star$, ensuring the hypersphere covers at least 97.7% of the distribution's population. Thus, the standard deviation can be computed as $\sigma^\star = R/3$. To prepare for the reconstruction and regularization phases in the next epoch, the standard deviation of the prior distribution σ ought to be adjusted to align with the standard deviation of the latent distribution, denoted as $\sigma \equiv \sigma^\star$. This means that the prior distribution is updated by decreasing its standard deviation. Repeating three training phases enhances the compactness of the latent code distribution while maintaining its adherence to the spherical shape of the prior distribution. The entire training procedure of AdaAAE is outlined in Algorithm 1.

Algorithm 1 The Training Procedure of AdaAAE.

```
Require: $p_d$: the data distribution, $p_\psi(\mathbf{z})$: the prior distribution, $q_\phi(\mathbf{z})$: the latent code distribution, $m$: the batch size, n_batch: the number of data batches, max_epoch: the maximum training epochs, $\alpha_1$: the learning rate for the reconstruction phase and the regularization phase, $\alpha_2$: the learning rate for the compactness phase.
 1: while current epoch in smaller than max_epoch do
 2:     for current batch is smaller than n_batch do
 3:         Sample $\{x^{(i)}\}_{i=1}^{m} \sim p_d$                    ▷ a batch from the real data
 4:         $g_\phi, g_\theta \leftarrow \nabla_{\phi,\theta}\mathcal{L}_{rec}(\phi, \theta; \mathbf{x})$          ▷ train autoencoder
 5:         $\phi \leftarrow \phi - \alpha_1 \cdot \text{RMSProp}(g_\phi)$
 6:         $\theta \leftarrow \theta - \alpha_1 \cdot \text{RMSProp}(g_\theta)$
 7:         Sample $\{z^{(i)}\}_{i=1}^{m} \sim p_\psi(z)$              ▷ a batch of prior distribution
 8:         $g_\psi \leftarrow \nabla_\psi\mathcal{L}_{reg^\star}(\psi; \mathbf{x}, \mathbf{z})$            ▷ train discriminator
 9:         $\psi \leftarrow \psi - \alpha_1 \cdot \text{RMSProp}(g_\psi)$
10:         $g_\phi \leftarrow \nabla_\phi\mathcal{L}_{reg^\dagger}(\phi; \mathbf{x})$                ▷ train generator
11:         $\phi \leftarrow \phi - \alpha_1 \cdot \text{RMSProp}(g_\phi)$
12:     end for
13:     for current batch is smaller than n_batch do
14:         $g_\phi \leftarrow \nabla_\phi\mathcal{L}_{svd}(\phi; \mathbf{x})$                  ▷ train encoder
15:         $\phi \leftarrow \phi - \alpha_2 \cdot \text{RMSProp}(g_\phi)$
16:     end for
17:     Update $R \rightarrow$ update $\sigma^\star$ of $q_\phi(\mathbf{z}) \rightarrow$ update $\sigma$ of $p_\psi(\mathbf{z})$
18: end while
```

6.3.2 Online Anomaly Detection

Our method provides a unique advantage in that it enables the precise determination of the online anomaly threshold, even in scenarios where prior information about the anomaly ratio in the testing samples is unavailable, which is often the case in practical situations. To elaborate, towards the end of the modelling phase, we estimate the radius R of the sphere representing the latent distribution $q_\phi(\mathbf{z})$. This estimation is subsequently employed to derive the approximate standard deviation $\sigma^\star$ of the latent distribution. In this context, we offer two distinct approaches for determining the anomaly threshold and assessing anomaly scores for test instances: a distance-based approach and a density-based approach.

In the case of the distance-based approach, we can employ the estimated radius R of the latent distribution as the anomaly threshold, denoted as $Th_a = R$. Under this approach, any test instance $\mathbf{x}$ is classified as anomalous if its computed anomaly score, represented by $\text{a}(\mathbf{x}) = \|h(\mathbf{x}, \phi) - \mathbf{c}\ |$, exceeds this threshold, i.e., $\text{a}(\mathbf{x}) > R$. In the density-based approach, an instance's anomaly score is determined by the negative log-likelihood of its latent code, considering the latent distribution $q_\phi(\mathbf{z}) \sim \mathcal{N}(0, \sigma^{\star 2}\text{I})$. This score is represented as $\text{a}(\mathbf{x}) = -\text{log}q_\phi(G(\mathbf{x}))$. In this context, the anomaly threshold is determined as the negative log-likelihood of a latent point located on the boundary of the latent sphere, where the distance from the origin equals the radius R. Therefore, the threshold can be approximately computed as:

$$Th_a = -log\left(\frac{1}{\sqrt{(2\pi\sigma^{\star 2})^n}} \times \exp\left(-R^2/2\sigma^{\star 2}\right)\right) \tag{6.12}$$

where n is the dimension of the latent points, $R = 3 \times \sigma^\star$ as previously defined. Instances with anomaly scores exceeding the threshold are classified as anomalies, while those with scores below the threshold are considered normal. This study employs the density-based approach, as supported by experimental results. The anomaly detection process of AdaAAE is visualized in Fig. 6.2.

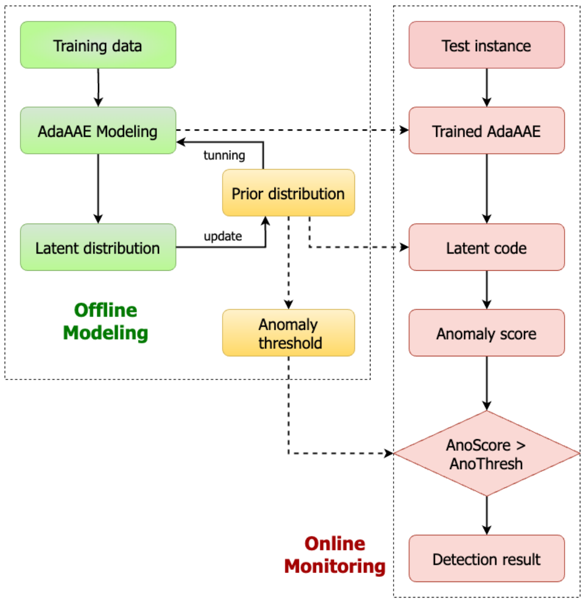

Figure 6.2: Flowchart outlining the two stages of anomaly monitoring: offline modelling and online monitoring.

6.4 Experiments and Results

6.4.1 Experimental Settings

Validation Datasets

The ASHRAE RP-1312 Air Handling Unit (AHU) dataset is reused in this study to evaluate and validate the proposed models alongside baseline models. This dataset, renowned for its comprehensive collection of data from real-world HVAC systems, offers a robust foundation for testing the effectiveness and reliability of our fault detection approaches. By leveraging this dataset, we can ensure that the comparative models are tested under realistic conditions, providing valuable insights into their performance in practical applications.

To assess the resilience of our framework to variations in faults arising from seasonal conditions of the RP-1312 AHU system, we meticulously curated three distinct datasets. Each dataset corresponds to the typical faults observed during one of the seasons: Spring, Summer, or Winter. In this context, every seasonal dataset contains a combination of faulty data acquired from AHU-A and normal data obtained from AHU-B. The number of normal instances matches the number of fault instances. For each common fault, there are $n = 1440$ instances of faults (equivalent to 24 hours * 60 minutes). Consequently, for a seasonal dataset investigating k different typical faults, the total combined instances, encompassing both fault and normal ones, amount to N_d = 2 * k * 1440. To adapt the datasets for fault detection, we ignore the multiclass labels assigned to fault instances and instead utilize a binary label assignment approach. Specifically, we assign binary labels to both fault and normal instances as follows: $y_i = 0$ for instances $\mathbf{x}_i$ in the negative class $\mathcal{X}_{\text{neg}}$ (comprising normal instances), and $y_i = 1$ for instances $\mathbf{x}_i$ in the positive class $\mathcal{X}_{\text{pos}}$ (comprising fault instances). Originally, each data instance included sensor values and control signals. Since control signals are irrelevant to our detection task, we exclude them from analysis, focusing solely on sensor values. We consider 106 sensor features in our analysis.

Nevertheless, fault instances within each original dataset possess distinct underlying characteristics compared to normal instances. Consequently, they are readily identified by the majority of standard baseline anomaly detection models. As a result, the effectiveness of the proposed model cannot be confirmed using the original datasets because the detection performance across the comparative models does not exhibit significant differences. Therefore, the research devised a technique to produce hard transient fault instances and substituted them for the original fault instances to evaluate the detection performance of the comparative models. For each fault class, let $N_{\text{pos}}^{\text{cls}}$ represent the number of fault instances $\mathcal{X}_{\text{pos}}^{\text{cls}}$, and $N_{\text{neg}}^{\text{cls}}$ represent the number of normal instances $\mathcal{X}_{\text{neg}}^{\text{cls}}$ recorded at the same time ($\mathcal{N}_{\text{pos}}^{\text{cls}}$ = $\mathcal{N}_{\text{neg}}^{\text{cls}}$). For each instance $\mathbf{x}_i$ in $\mathcal{X}_{\text{neg}}^{\text{cls}}$, locate the k nearest instances $\mathbf{x}_k$ where $\mathbf{x}_k$ belongs to $\mathcal{X}_{\text{pos}}^{\text{cls}}$. For each of these nearest instances $\mathbf{x}_k$, derive the transient fault instance $\mathbf{x}_j$, which lies along the vector from the normal instance $\mathbf{x}_i$ to the fault instance $\mathbf{x}_k$. This process can be expressed as follows:

$$\mathbf{x}_j = \frac{(d * \mathbf{x}_i + \mathbf{x}_k)}{(d + 1)} \tag{6.13}$$

Here, d denotes the distance scaling factor. When d is high, the transient fault instance closely resembles the normal instance, posing difficulty for the classifier to distinguish it from normal instances. Perform this process for every instance $\mathbf{x}_i$ in $\mathcal{X}_{\text{neg}}^{\text{cls}}$, resulting in a total of $k \times N_{\text{neg}}^{\text{cls}}$ transient fault instances being produced. In the subsequent step, the technique is utilized to identify hard instances from $\mathcal{X}_{\text{trans}}^{\text{cls}} = \{\mathbf{x}_j\}_{j=1}^{k\times N_{\text{neg}}^{\text{cls}}}$. To elaborate, let's designate a normal class as class c. We denote "hard negatives" as those instances $\mathbf{x}_j$ from a transient fault class with high confidence $p(y_j^{\text{pred}} = c|\mathbf{x}_j)$, indicating obvious mistakes. These are formally defined as follows:

$$\mathcal{N}_c = \Big\{\mathbf{x}_j; y_j \neq c, \text{high confidence } p(y_j^{\text{pred}} = c|\mathbf{x}_j)\Big\} \tag{6.14}$$

In this research, hard negative instances are employed to replace the original fault instances. The aim is to construct datasets where each dataset consists of 80% normal instances (a majority class) and 20% fault instances (a minority class). Consequently, the number of hard negative instances for each class employed equals 25% of the according number of normal instances. To accomplish this, $0.25{*}N_{\text{neg}}^{\text{cls}}$ hard negatives with the highest scores of confidence, denoted as $\mathcal{X}_{\text{hard}}^{\text{cls}} = \{\mathbf{x}_j\}_{j=1}^{0.25*N_{\text{neg}}^{\text{cls}}}$ for class c are chosen, considering the present feature space and classification model. Repeat this procedure for all the fault classes, the revised dataset comprises $\mathcal{X}_{\text{neg}} = \{\mathbf{x}_{i=1}^{N_{\text{neg}}}\}$ and $\mathcal{X}_{\text{hard}} = \cup_{cls=1}^{N_{\text{cls}}} \mathcal{X}_{\text{hard}}^{\text{cls}}$, which is then utilized as a validation dataset.

Model Configuration

Although our model is specifically tailored for three seasonal datasets, it maintains a consistent configuration throughout. Specifically, the network structures of the autoencoder (AE) and discriminator in our model are [106, 1000, 2, 1000, 106] and [2, 1000, 1], respectively. Typically, the middle layer of the AE should contain sufficient nodes to effectively represent the latent distribution of normal instances. However, in this study, we intentionally restrict the number of nodes in the middle layer to 2. This deliberate choice was made to demonstrate the AdaAAE's capability to capture the latent distribution of normal instances even within a 2D circle and to facilitate the visualization of the latent distribution in a 2D plane. Activation functions are applied throughout the model, with the rectified linear unit (ReLU) function utilized between layers, a linear function employed for the final layer of the encoder, and a sigmoid function utilized for the final layers of both the decoder and the discriminator. Consequently, input instances necessitate normalization via a MinMax scaler to ensure feature values fall within the range of [0, 1]. Training utilizes the root mean square propagation (RMSprop) algorithm, with a learning rate of 0.01 during both the reconstruction and regularization phases, and 0.005 during the compactness phase of the encoder to condense the latent space utilizing soft-boundary deep SVDD. The selection of these learning rates aims to prevent rapid reduction of the hypersphere radius of the latent distribution, which could cause divergence in the shape of the latent distribution from the prior distribution.

The training process comprises 50 epochs, with mini-batch sizes of 128. During training, we monitor the ”Gaussian level” of the latent distribution of normal instances after each epoch, using it as a stopping criterion. This metric assesses how well the latent distribution aligns with the prior distribution. To compute this metric, we first fit a Gaussian distribution to the data points in the latent space. Subsequently, with the obtained parameters of the Gaussian distribution (mean and covariance matrix), we compute the likelihood or probability density of each data point within the cluster under this distribution. A higher average likelihood or density indicates greater consistency of the data points within the latent cluster with a Gaussian distribution. Ideally, this average likelihood value should increase throughout training. If there’s a decrease in this value, the stopping criterion is triggered. We set the patience parameter to 5. Initially, a radius R is set as 3, and the prior distribution is configured as an isotropic normal distribution $\mathcal{N}(0, \sigma^2 \mathrm{I})$ with $\sigma = R/3$. The hyper-parameter ν is set to 0.05 to permit 5% of the points to be mapped outside the sphere. Following each compactness phase, the radius R is estimated and utilized to update σ of the prior distribution. The framework’s layers and training procedures are all executed within the Tensorflow environment.

6.4.2 Effectiveness of AdaAAE for Fault Detection Application

Performance Metrics and Baseline Models

In our experiments, we assess the anomaly detectors using two key performance metrics: the area under the receiver operating characteristic curve (AUC-ROC) and the area under the precision-recall curve (AUC-PR). AUC-ROC quantifies the performance by measuring the area under the ROC curve, which illustrates the relationship between true-positive rates and false-positive rates. Conversely, AUC-PR calculates the area under the PR curve, depicting the relationship between precision and recall. While AUC-ROC evaluates prediction performance across both normal and abnormal classes, AUC-PR focuses more on anomalies. Both AUC-ROC and AUC-PR values range between 0 and 1, with higher values indicating superior performance. We compute all AUC-ROC and AUC-PR results, along with their standard deviations, across ten independent runs.

To assess the proposed AdaAAE method, we utilized a range of advanced anomaly detection techniques as baseline models. These baseline models encompass various underlying approaches, including both shallow architectures and deep learning-based approaches. For the shallow AD approaches, we have selected one-class support vector machine (OC-SVM), kernel density estimation (KDE), and mixture of probabilistic principal component analyzers (MPPCA), given their exceptional performance in anomaly detection tasks. OC-SVM Schölkopf et al. (2001) is a widely used kernel-based anomaly detection algorithm. Unlike SVM, which separates classes with a hypersphere, OCSVM aims to find a hypersphere that includes all training points. KDE Hu et al. (2018) is a non-parametric method that estimates the probability density function of a random variable, making it effective for anomaly detection in complex scenarios. MPPCA Tipping Bishop (1999) models data as a mixture of probabilistic PCA models, capturing complex distributions and identifying anomalies as deviations from the learned normal subspaces. The OC-SVM and KDE implementations were carried out using Scikit-Learn libraries with optimized parameters. MPPCA was implemented following the optimized parameters suggested in the original paper Yairi et al. (2017).

Additionally, seven state-of-the-art deep anomaly detection algorithms were utilized for comparison. Some of these algorithms were previously introduced in Section 6.2, including the variational autoencoder (VAE), deep autoencoding Gaussian mixture model (DAGMM), AnoGAN, deep SVDD (DSVDD), deep autoencoding support vector data description (DASVDD), AAE, and center-aware adversarial autoencoder (CA-AAE). The VAE-based anomaly detection method utilizes a variational autoencoder rather than a traditional autoencoder for data reconstruction, leveraging the reconstruction error to detect anomalies. DAGMM Zong et al. (2018) is a density-estimation model that combines an autoencoder with a Gaussian mixture model applied to the latent representation from the autoencoder. It has demonstrated competitive performance across various anomaly detection datasets. AnoGAN Schlegl et al. (2017) is a method for anomaly detection using generative adversarial networks (GANs). AnoGAN generates realistic instances from candidate latent vectors and identifies anomalies based on the differences between the generated and input instances. DASVDD Hojjati Armanfard (2023) integrates autoencoders with support vector data description (SVDD) for anomaly detection. It utilizes reconstruction error to improve anomaly detection and prevent the collapse of the hypersphere. CA-AAE D. Li et al. (2021) enhances anomaly detection by adding center awareness to the adversarial autoencoder (AAE). It uses two encoders: one evaluates anomaly levels while the other improves latent space compactness, focusing on the central distribution of normal data to boost detection accuracy and robustness. DSVDD and AAE are component models of AdaAAE and are used as ablation models for comparison in this study. To ensure a fair comparison, the neural network architectures for VAE, DAGMM, AnoGAN, DSVDD, DASVDD, AAE and CA-AAE are aligned with that of AdaAAE. The hyperparameters of these deep models are fine-tuned based on both the recommendations from the original papers and experimental data to achieve optimal performance. All layers and training procedures of these deep baseline models are implemented within the TensorFlow environment.

Findings from Comparative Tests

In this research, three distinct datasets are formed for each season, each with a varying distance scaling factor (denoted as d = 30, 50, 70). This variation allows for assessing the method's effectiveness across different levels of data complexity. Consequently, a total of nine datasets were utilized, corresponding to the three seasons analyzed in this study. Tables 6.1 and 6.2 show the performance of various outlier detection algorithms in terms of AUC-ROC (Area Under the Receiver Operating Characteristic Curve) and AUC-PR (Area Under the Precision-Recall Curve). From these tables, it is evident that the performance of AdaAAE and the baseline methods tends to decrease as the complexity of the datasets increases, irrespective of the seasonal aspect. This trend is particularly noticeable in the lower performance metrics for the highly complex datasets such as Summer^3, Winter^3, and Spring^3. However, despite the increased complexity, AdaAAE maintains relatively high performance, underscoring its robustness and reliability compared to other methods.

More specifically, the experimental results in Table 6.1 demonstrated that AdaAAE consistently outperforms other methods, achieving AUC-ROC scores above 90%. For example, it achieved impressive scores of 98.7 $\pm$ 1.3% in the Summer^1 dataset and 99.8 $\pm$ 0.4% in the Winter^1 dataset. Even for highly complex datasets like Summer^3, Winter^3, and Spring^3, AdaAAE's scores were significantly better than those of other baseline methods. On average, AdaAAE achieved an average AUC-ROC of 97.5 $\pm$ 1.9%, showing not only high performance but also consistent results across different dataset complexities. While deep baseline models performed well in certain cases, they exhibited greater variability. For instance, AnoGAN scored 97.9 $\pm$ 2.1% in Summer^1 but dropped to 69.8 $\pm$ 8.1% in Summer^3. Similarly, DAGMM scored 94.4 $\pm$ 1.4% in Spring^1 but fell to 83.6 $\pm$ 3.2% in Spring^3. CA-AAE achieved 97.4 $\pm$ 2.3% in Summer^1, but its performance fell to 77.6 $\pm$ 4.0% in Summer^3. The high variance in deep neural models is due to their increased complexity and higher number of parameters, which makes them more sensitive to variations in

Table 6.1: Comparison of AUC-ROC (%) between Baseline Methods and AdaAAE

Datasets	AUC-ROC Performance										
	OC-SVM	**KDE**	**MPPCA**	**VAE**	**DAGMM**	**AnoGAN**	**DSVDD**	**DASVDD**	**AAE**	**CA-AAE**	**AdaAAE**
Summer[1]	96.8 ± 0.1	**99.5** ± **0.1**	98.8 ± 0.3	98.6 ± 0.1	97.7 ± 1.3	97.9 ± 2.1	96.6 ± 3.0	98.8 ± 1.0	92.1 ± 8.4	97.4 ± 2.3	98.7 ± 1.3
Summer[2]	83.4 ± 0.2	93.9 ± 0.4	95.8 ± 1.1	94.9 ± 0.3	94.5 ± 1.8	85.6 ± 4.7	89.8 ± 5.0	94.2 ± 2.1	89.9 ± 7.6	87.2 ± 4.7	**96.4** ± **2.1**
Summer[3]	69.4 ± 0.2	85.0 ± 0.7	87.9 ± 1.3	87.3 ± 0.8	90.6 ± 1.5	69.8 ± 8.1	74.6 ± 6.9	77.3 ± 7.1	79.7 ± 5.8	77.6 ± 4.0	**93.9** ± **5.5**
Winter[1]	**100** ± **0.0**	**100** ± **0.0**	**100** ± **0.0**	**100** ± **0.0**	98.8 ± 0.4	96.5 ± 2.1	**100** ± **0.0**	99.9 ± 0.1	96.6 ± 5.7	100 ± 0.0	99.8 ± 0.4
Winter[2]	**100** ± **0.0**	99.9 ± 0.0	99.5 ± 0.2	99.8 ± 0.1	95.7 ± 1.1	94.9 ± 2.7	97.8 ± 2.3	98.4 ± 1.0	92.6 ± 5.1	95.3 ± 3.1	99.1 ± 1.6
Winter[3]	96.9 ± 1.1	**99.8** ± **0.1**	97.2 ± 0.8	99.1 ± 0.2	92.7 ± 1.6	89.8 ± 3.1	91.3 ± 8.0	93.0 ± 2.7	88.1 ± 7.1	88.1 ± 5.0	99.3 ± 0.5
Spring[1]	96.1 ± 0.2	**99.7** ± **0.0**	99.3 ± 0.1	99.1 ± 0.1	94.4 ± 1.4	98.0 ± 1.2	92.2 ± 5.7	97.6 ± 1.5	93.7 ± 5.1	93.8 ± 3.0	98.9 ± 1.0
Spring[2]	83.6 ± 0.4	94.2 ± 0.6	94.6 ± 0.8	95.5 ± 0.5	89.7 ± 3.8	85.4 ± 3.8	89.4 ± 4.4	93.5 ± 2.8	91.7 ± 1.3	88.1 ± 4.2	**98.5** ± **1.3**
Spring[3]	65.5 ± 0.7	91.6 ± 0.7	91.6 ± 0.6	92.3 ± 1.2	83.6 ± 3.2	79.6 ± 4.2	85.1 ± 4.4	87.8 ± 4.3	86.0 ± 5.7	81.2 ± 3.1	**92.7** ± **3.8**
Average	88.0 ± 0.3	96.0 ± 0.3	96.1 ± 0.6	96.3 ± 0.4	93.1 ± 1.8	88.6 ± 3.5	90.8 ± 4.4	93.4 ± 2.5	90.0 ± 5.8	89.9 ± 3.8	**97.5** ± **1.9**

Table 6.2: Comparison of AUC-PR (%) between Baseline Methods and AdaAAE

Datasets	AUC-PR Performance										
	OC-SVM	**KDE**	**MPPCA**	**VAE**	**DAGMM**	**AnoGAN**	**DSVDD**	**DASVDD**	**AAE**	**CA-AAE**	**AdaAAE**
Summer[1]	86.9 ± 0.5	**97.7** ± **0.3**	93.9 ± 1.7	91.7 ± 0.6	88.3 ± 6.7	86.2 ± 5.8	91.7 ± 6.0	95.3 ± 3.1	82.0 ± 9.3	89.3 ± 2.2	97.3 ± 2.7
Summer[2]	58.7 ± 0.6	80.5 ± 1.0	79.3 ± 4.8	74.3 ± 0.6	77.4 ± 6.4	76.4 ± 4.1	75.7 ± 9.3	76.8 ± 7.6	74.9 ± 6.4	50.2 ± 7.6	**90.9** ± **5.6**
Summer[3]	34.1 ± 0.7	52.3 ± 2.0	53.0 ± 3.2	51.6 ± 2.2	63.9 ± 5.9	49.7 ± 4.8	52.4 ± 8.5	44.8 ± 9.4	53.6 ± 8.3	34.9 ± 8.7	**88.2** ± **6.8**
Winter[1]	99.3 ± 0.5	99.2 ± 0.7	**100** ± **0.0**	**100** ± **0.1**	93.5 ± 1.2	93.8 ± 3.1	99.9 ± 0.1	99.8 ± 0.1	94.1 ± 5.3	99.9 ± 0.1	99.1 ± 0.8
Winter[2]	**99.8** ± **0.3**	99.3 ± 0.7	97.0 ± 1.4	97.1 ± 1.1	78.2 ± 4.1	91.7 ± 3.8	94.2 ± 4.5	91.9 ± 2.4	88.3 ± 4.7	87.2 ± 5.1	98.3 ± 2.7
Winter[3]	81.9 ± 5.9	**98.2** ± **1.2**	76.0 ± 5.5	92.2 ± 1.4	69.3 ± 6.0	71.2 ± 8.3	67.8 ± 7.7	79.2 ± 4.0	65.7 ± 3.2	49.6 ± 6.2	97.5 ± 1.8
Spring[1]	88.9 ± 1.2	**98.7** ± **0.2**	96.9 ± 0.4	95.8 ± 0.6	78.9 ± 4.3	96.6 ± 3.1	86.4 ± 9.1	96.0 ± 2.0	83.4 ± 5.6	84.8 ± 4.1	97.6 ± 2.6
Spring[2]	66.2 ± 1.2	84.5 ± 1.5	79.9 ± 1.9	80.0 ± 2.2	68.8 ± 4.3	78.9 ± 4.5	78.8 ± 5.8	85.2 ± 3.1	81.4 ± 1.9	68.8 ± 5.1	**96.0** ± **3.6**
Spring[3]	34.9 ± 0.7	74.0 ± 2.3	68.0 ± 2.2	70.2 ± 3.5	60.8 ± 3.6	69.1 ± 6.2	71.4 ± 4.6	74.3 ± 5.2	68.5 ± 4.0	49.7 ± 3.2	**84.5** ± **7.1**
Average	72.3 ± 1.3	87.2 ± 1.1	82.6 ± 2.4	83.7 ± 1.4	75.5 ± 4.7	79.3 ± 5.6	79.8 ± 6.2	82.6 ± 4.2	76.9 ± 5.4	68.3 ± 4.2	**94.4** ± **3.7**

training data. This complexity often leads to overfitting, where the model performs well on the training data but exhibits greater variability on new data. Additionally, the optimization process in deep learning, such as backpropagation, which includes gradient descent, can lead to convergence issues, contributing to performance variability. Despite this, the advanced capabilities of deep models allow them to capture complex data patterns, leading to superior overall performance, as shown by AdaAAE's results.

In terms of AUC-PR, shown in Table 6.2, AdaAAE continues to demonstrate its superiority over the baseline models. It achieved remarkable scores, such as 98.3 $\pm$ 2.7% in the Winter^2 dataset and 97.6 $\pm$ 2.6% in the Spring^1 dataset. While the KDE method peaked in Winter^3 and Spring^1, it struggled to maintain consistent results across all datasets, ultimately falling behind the more reliable and high-performing AdaAAE. AdaAAE's average AUC-PR score of 94.4 $\pm$ 3.7% further confirms its superiority, significantly outperforming KDE and MPPCA, which achieved lower average scores of 87.2 $\pm$ 1.1% and 82.6 $\pm$ 2.4%, respectively. Deep models such as DASVDD and CA-AAE also exhibited high variance in AUC-PR performance. For instance, DASVDD achieved 95.3 $\pm$ 3.1% in Summer^1 but dropped to 44.8 $\pm$ 9.4% in Summer^3. Likewise, CA-AAE's performance varied significantly, with a score of 84.8 $\pm$ 4.1% in Spring^1, dropping to 49.7 $\pm$ 3.2% in Spring^3. The experimental results also highlight that model performance is higher and shows less variance in the winter datasets compared to the summer and spring datasets. This is because winter faults, such as heating coil fouling and damper leaks, are simpler to detect and classify than the more varied and complex faults in summer and spring, like duct leaks and air filter blockages. Additionally, the winter datasets contain 6 types of faults, while the summer datasets include 8 types. These characteristics contribute to better and more consistent anomaly detection performance in the winter datasets compared to the summer and spring datasets. Although both AUC-ROC and AUC-PR metrics are important for validating the performance of comparative methods, AUC-PR is more suitable for this study's scenarios, where anomalies (i.e., fault instances) are the minority, accounting for only 20% of test instances. Therefore, AdaAAE's significant superiority over the baseline models in terms of AUC-PR clearly demonstrates its effectiveness in handling AHU data of varying complexities. It highlights AdaAAE's superior fault detection capabilities while also emphasizing its consistency and reliability.

Exploring the Effectiveness of AdaAAE

AdaAAE is designed to condense the latent distribution of training data, which includes normal instances, while also ensuring that the normal latent distribution conforms to the adaptive spherical prior distribution during training. Increased compactness in the training latent distribution enhances the probability of normal latent codes forming a distinct cluster separate from the one of fault latent codes. In addition, the spherical shape of the latent distribution simplifies the threshold estimation, thereby facilitating effective fault detection within the latent space. Fig. 6.3 illustrates the adaptation of the latent distribution of normal instances during training with the Summer^2 dataset. The normal latent distribution adjusts to converge into a spherical shape, with the cluster's radius decreasing throughout the training phase. This process makes the latent distribution of normal instances progressively more compact. Fig. 6.4 further illustrates this, showing the reduction in the radius of the latent distribution from epoch 1 to epoch 20 across various seasonal training datasets.

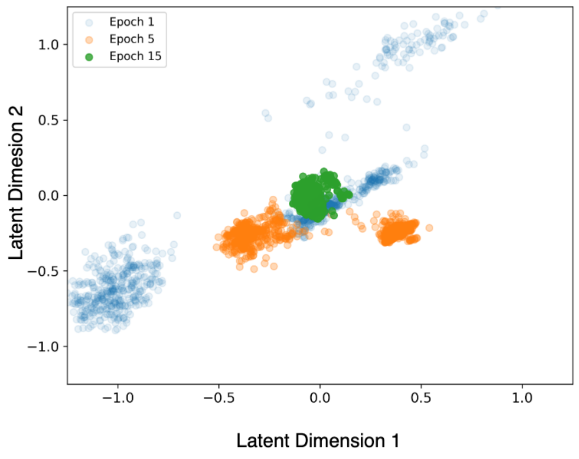

Figure 6.3: The adaptation of normal latent codes throughout the training process.

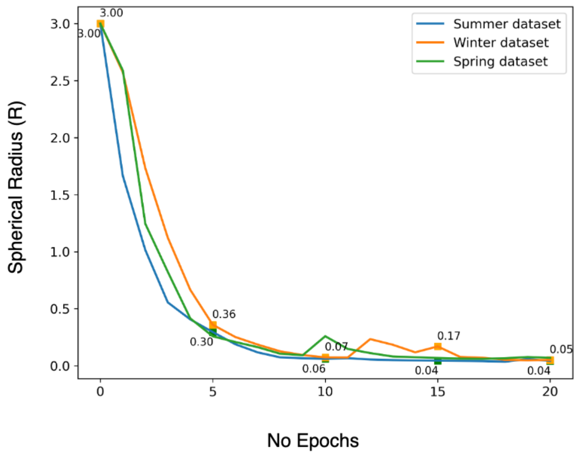

Figure 6.4: The reduction in radius of the normal latent code cluster during the training process.

To explore the performance disparities between AdaAAE and baseline methods operating in the latent space, such as VAE, DAGMM, DSVDD, DASVDD, and CA-AAE, we illustrate the latent representations of AdaAAE and these baseline methods using testing data from the Summer^2 dataset ($d = 50$). This visualization aims to evaluate the compactness of the normal latent cluster and the separation between the normal and fault latent clusters. Fig. 6.5 illustrates the features learned by these comparative models. Observations

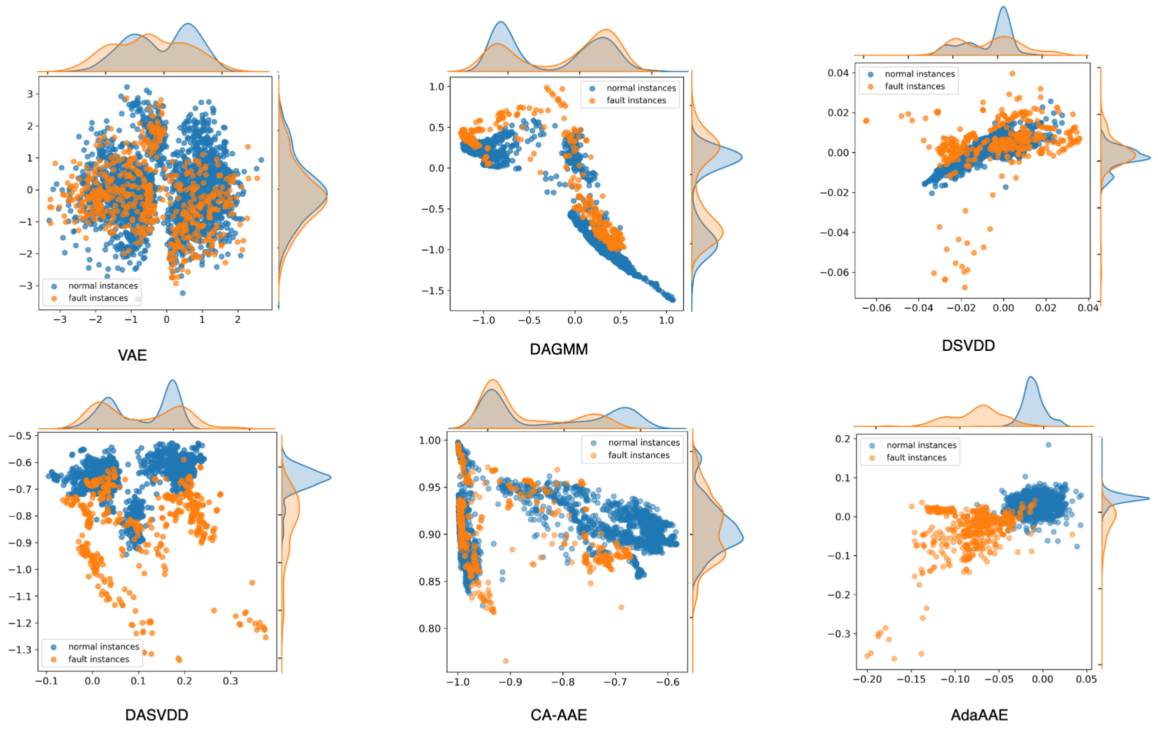

Figure 6.5: Visualization of features learned by AdaAAE and baseline methods on testing summer data.

from Fig. 6.5 reveal that AdaAAE effectively confines normal latent codes within a more condensed area, while fault latent codes are more dispersed. This clear separation between normal and fault latent codes highlights AdaAAE's superior ability to differentiate between them compared to other methods. In contrast, DSVDD's normal latent cluster tends to form an elliptical shape due to the lack of regularization, making fault separation less effective. DASVDD, though better than some, still shows a less compact normal latent cluster, complicating the clear separation from fault latent codes. VAE, DAGMM, and CA-AAE exhibit more overlap between normal and fault latent codes, reducing their effectiveness. In contrast, AdaAAE maintains a more compact and distinct separation of normal and fault data in the latent space, highlighting its enhanced performance in fault detection.

In addition, we employed the Davies-Bouldin Index (DBI) Davies Bouldin (1979) as a statistical metric to assess the compactness of the normal and fault latent clusters, as well as the separation between them.

This metric is calculated based on the normal and fault latent features learned by AdaAAE and the baseline methods using the test data from the datasets. The DBI evaluates the clustering quality by comparing the similarity of points within each cluster to the difference between clusters. Specifically, it calculates the average ratio of within-cluster dispersion to between-cluster separation. A lower DBI indicates better clustering, with more compact and well-separated clusters.

Table 6.3: Comparing Davies-Bouldin Index (DBI) between Baseline Methods and AdaAAE

Datasets	**Davies-Bouldin Index (DBI)**					
	VAE	**DAGMM**	**DSVDD**	**DASVDD**	**CA-AAE**	**AdaAAE**
Summer^1	10.6 ± 1.9	6.0 ± 2.1	2.1 ± 0.6	1.6 ± 0.9	3.1 ± 1.0	$\mathbf{1.4 \pm 0.9}$
Summer^2	11.9 ± 2.4	6.8 ± 1.6	2.5 ± 0.5	2.7 ± 0.3	7.6 ± 5.1	$\mathbf{1.9 \pm 0.6}$
Summer^3	11.6 ± 1.1	10.9 ± 3.6	6.2 ± 7.3	4.2 ± 3.9	11.2 ± 6.3	$\mathbf{1.4 \pm 0.6}$
Winter^1	10.2 ± 2.5	3.5 ± 1.9	1.7 ± 1.1	2.1 ± 0.8	3.8 ± 1.9	$\mathbf{1.0 \pm 0.4}$
Winter^2	7.6 ± 0.6	4.2 ± 2.3	2.0 ± 1.1	2.7 ± 1.9	5.9 ± 1.8	$\mathbf{1.5 \pm 1.1}$
Winter^3	10.7 ± 1.4	6.5 ± 2.2	2.4 ± 0.6	3.9 ± 1.2	8.4 ± 3.4	$\mathbf{1.6 \pm 1.4}$
Spring^1	5.7 ± 0.5	4.4 ± 1.3	2.2 ± 0.8	2.1 ± 0.4	3.6 ± 1.1	$\mathbf{2.0 \pm 1.5}$
Spring^2	4.7 ± 0.2	3.7 ± 1.2	$\mathbf{2.4 \pm 0.4}$	3.8 ± 1.2	5.1 ± 0.2	2.7 ± 1.0
Spring^3	6.2 ± 0.3	9.7 ± 3.5	3.3 ± 1.4	4.0 ± 2.4	6.4 ± 2.7	$\mathbf{2.2 \pm 0.5}$
Average	8.8 ± 1.2	6.2 ± 2.2	2.8 ± 1.5	3.0 ± 1.6	6.1 ± 2.7	$\mathbf{1.7 \pm 0.9}$

Results from Table 6.3 indicate that the proposed method, AdaAAE, exhibits superior performance in producing well-separated and compact clusters in the latent space across the seasonal datasets. For instance, in the summer datasets, AdaAAE achieved DBI scores of 1.4 ± 0.9, 1.9 ± 0.6, and 1.4 ± 0.6 for Summer^1, Summer^2, and Summer^3, respectively. During the winter season, the method attained DBI values of 1.0 ± 0.4 for Winter^1, 1.5 ± 1.1 for Winter^2, and 1.6 ± 1.4 for Winter^3. In Spring datasets, AdaAAE recorded DBI values of 2.0 ± 1.5, 2.7 ± 1.0, and 2.2 ± 0.5 for Spring^1, Spring^2, and Spring^3, respectively. The average DBI across all datasets was 1.7 ± 0.9, significantly lower than those of the baseline models. It should be noted that the results provided in Table 6.3 are based on ten independent runs. This low average DBI value across various dataset complexities demonstrates AdaAAE's superior capability in latent space manipulation.

6.4.3 Dynamic Fault Threshold Estimation

Apart from AdaAAE's established superiority in unsupervised fault detection, a noteworthy advantage of the method lies in its ability to precisely estimate an online anomaly threshold, even when there is no prior information available regarding the anomaly ratio in the testing instances—a common scenario in practical contexts. This aspect is detailed in Section 6.3.2. To demonstrate this benefit, we conduct experimental tests where we evaluate the F1-score of AdaAAE and various baseline methods under two distinct scenarios. In

Table 6.4: F1-Score (%) Comparison of Fault Detection Performance in Two Scenarios of AdaAAE and Baseline Methods

Datasets	**F1-Score**			
	IForest	**OC-SVM**	**DSVDD**	**AdaAAE**
Summer[1]	28.4 ± 1.8	96.3 ± 1.2	95.4 ± 5.6	97.1 ± 1.0
	24.0 ± 0.4	49.1 ± 1.5	79.6 ± 8.2	94.6 ± 1.3
Decrement	**4.4 ± 0.0**	**47.2 ± 0.0**	**15.8 ± 0.0**	**2.5 ± 0.0**
Winter[1]	41.3 ± 1.0	97.9 ± 0.1	96.4 ± 2.7	97.7 ± 1.2
	42.1 ± 1.8	49.3 ± 1.1	85.2 ± 8.4	95.1 ± 1.4
Decrement	**-0.8 ± 0.0**	**48.6 ± 0.0**	**11.2 ± 0.0**	**2.6 ± 0.0**
Spring[1]	28.4 ± 0.8	83.6 ± 1.7	77.5 ± 12.7	94.8 ± 1.5
	27.0 ± 1.0	50.1 ± 0.9	57.7 ± 15.8	92.2 ± 4.2
Decrement	**1.4 ± 0.0**	**33.5 ± 0.0**	**19.8 ± 0.0**	**2.6 ± 0.0**

the first scenario, we possess prior knowledge of the anomaly ratio (fault ratio) (20%), which was crucial for determining the anomaly threshold. In the second scenario, we assume no prior knowledge of the anomaly ratio, and comparative methods are required to estimate the anomaly threshold based solely on the training data. Please note that the F1 score is the primary metric employed to evaluate the effectiveness of anomaly detection techniques. It is derived from both precision and recall values. The variation in performance of the methods across the two scenarios reflects their ability to accurately estimate anomaly thresholds. Minor discrepancies indicate the method's precision in estimation. These evaluations are conducted using three datasets: Summer[1], Winter[1], and Spring[1].

The baseline methods selected for the experiments include isolation forest (IForest) F. T. Liu et al. (2008), one-class support vector machine (OC-SVM) Schölkopf et al. (2001), and deep support vector data description (DSVDD) Ruff et al. (2018) because these methods can detect anomalies without needing the anomaly ratio of new instances. AdaAAE uses Eq. (6.12) to calculate the anomaly threshold, utilizing updated values of R and $\sigma^\star$ at the end of the training phase. DSVDD, on the other hand, directly employs the value of R obtained from the latent distribution as the anomaly threshold. Instances whose latent distance from the origin in the latent space exceeds R are classified as anomalies. In the case of IForest and OC-SVM, anomaly detection is performed using the predict function from Scikit-learn. Instances predicted as -1 are labeled as anomalies, while those predicted as 1 are considered normal. The results presented in Table 6.4 are derived from ten independent runs. Each dataset includes two sets of results: the upper row corresponds to the first scenario, while the lower row displays the results of the second scenario.

Table 6.4 shows that AdaAAE experiences consistently smaller performance drops across all datasets compared to the baseline methods, demonstrating its superior capability in estimating the anomaly threshold without prior knowledge. In terms of F1-score reductions, AdaAAE shows a slight decrease of only 2.5% on the Summer^1 dataset, which is significantly smaller than the drops seen in other methods: 4.4% for IForest, 47.2% for OC-SVM, and 15.8% for SVDD. For the Winter^1 dataset, AdaAAE shows a performance reduction of just 2.6%, significantly lower than the 41.3% drop for IForest, 49.3% for OC-SVM, and 11.2% for SVDD. In the Spring^1 dataset, AdaAAE experiences a slight decrease of 2.6%, which is slightly higher than IForest's 1.4% but remains substantially lower than the 33.5% for OC-SVM and 19.8% for SVDD. These figures illustrate that AAdaAAE maintains high performance even when required to estimate the anomaly threshold without prior knowledge, a common challenge in practical scenarios. The consistently small performance drops across all datasets highlight its superior generalization and reliability compared to the other methods evaluated.

6.5 Conclusion

In this research, we proposed a new anomaly detection approach called AdaAAE, specifically designed for unsupervised fault detection in air handling units (AHUs) of HVAC systems. Our method combines key techniques from adversarial autoencoders (AAE) and deep support vector data description (DSVDD). By innovatively training the reconstruction, regularization, and compactness phases, AdaAAE generates a spherical and compact latent distribution of training data, enabling efficient fault detection within the latent space. The experimental results for AUC-ROC and AUC-PR highlight AdaAAE's superior capability to detect faults across datasets of varying complexities. Notably, AdaAAE achieved an impressive average AUC-ROC of 97.5 $\pm$ 1.9% and an AUC-PR of 94.4 $\pm$ 3.7%, outperforming the baseline methods evaluated in this study.

A significant strength of AdaAAE lies in its ability to accurately determine the anomaly threshold at the end of the training phase, which is especially useful in cases where no prior knowledge of the anomaly ratio is available for test instances—a common scenario in practical applications. Experimental results highlight AdaAAE's outstanding capability to determine the anomaly threshold even without prior information. The data show only a minimal decline in performance, with a drop of only 2.5% for the Summer^1 dataset, and a similar decrease of 2.6% for both the Winter^1 and Spring^1 datasets, even when transitioning from a scenario with a known anomaly ratio to one where the anomaly ratio must be estimated.

Furthermore, due to AdaAAE's architecture, the approach holds promise for application in other domains featuring high-dimensional data such as images, texts, genetic data, and more. However, a limitation encountered in this study pertains to determining the optimal stopping criteria for the method, which occasionally results in the method stopping at suboptimal points. This limitation is left for future exploration and improvement. To ensure reproducibility and enable future enhancements by other researchers, the entire source code for this study is available in the following repository: https://github.com/viettra-xai/Ada-AAE.

Chapter 7

Conclusions and Future Work

This chapter provides a comprehensive summary of the research conducted in this thesis, highlighting the key contributions and advancements in the development of fault detection and diagnosis (FDD) models for HVAC systems. Additionally, we explore potential directions for future work, emphasizing the importance of developing adversarially robust anomaly detection models to enhance the reliability of these systems in real-world applications.

7.1 Concluding Remarks

In this thesis, we have tackled several critical challenges in developing fault detection and diagnosis (FDD) models for HVAC systems, addressing the complexities inherent in multiclass datasets, outlier detection, fault detection, and high-dimensional data processing. Our work presents significant advancements in the field through innovative methodologies and comprehensive validations. The key conclusions drawn from our research are summarized as follows:

(1) **Supervised Multiclass DAGMM**: We developed a supervised multiclass version of the Deep Autoencoding Gaussian Mixture Model (DAGMM) that leverages label information from training data to effectively detect outliers in multiclass datasets. By utilizing the encoding layers of DAGMM's compression network as a pre-trained model for deep neural networks (DNNs), we enhanced the DNN's performance in handling classification tasks. This approach demonstrated superior performance compared to state-of-the-art algorithms, validating its effectiveness through rigorous comparative studies. This work has been published in the *Journal of Energy and Buildings* in 2022 Tra et al. (2022a).

(2) **Neural Network-Based MPPCA (NN-MPPCA)**: To enhance the training and efficacy of mixtures of probabilistic principal component analyzers (MPPCA) models, we introduced a neural network-based approach, NN-MPPCA. This method employs a multilayer neural network for parameter optimization via back-propagation, addressing singularity issues with a robust loss function that includes a penalty component. Our experimental validation using a 90-ton centrifugal water-cooled chiller dataset confirmed the method's superiority over existing outlier detection algorithms. This study has been published in the *Journal of Building and Environment* in 2022 Tra et al. (2022b).

(3) **Deep Variational Mixture of Probabilistic Principal Component Analyzers (DV-MPPCA)**: We proposed a new framework that integrates NN-MPPCA with a Variational Autoencoder (VAE) to manage high-dimensional data effectively. The framework employs a modified evidence lower bound (ELBO) loss function to improve handling of incomplete datasets, ensuring convergence to the global optimum by jointly training the VAE and NN-MPPCA components. Case studies using data from the ASHRAE Research Project 1312 validated the framework's superior performance in anomaly detection compared to existing models. This research has been published in *IEEE Transactions on Automation Science and Engineering* in 2023 Tra et al. (2023).

(4) **Adaptive Adversarial Autoencoder (AdaAAE)**: We introduced the adaptive adversarial autoencoder (AdaAAE), which enhances the adversarial autoencoder (AAE) framework with deep support vector data description (DSVDD) to improve the compactness of the latent code distribution. The AdaAAE's training process is meticulously designed to ensure simultaneous convergence of all components, with a novel method for accurately estimating the anomaly threshold. This approach was tested on various seasonal datasets from an air handling unit system under the ASHRAE Research Project 1312, demonstrating its effectiveness and superiority over current models. This study has been accepted in *IEEE Transactions on Automation Science and Engineering* in October, 2023.

In conclusion, this thesis contributes significantly to the field of fault detection and diagnosis in HVAC systems by providing innovative methodologies that address existing limitations and enhance the accuracy and reliability of anomaly detection. The proposed models and frameworks not only advance theoretical understanding but also offer practical solutions applicable in real-world settings. Through my research publications, I aim to disseminate novel ideas to fellow academics in the fields of building and energy. In addition to sharing research findings, I have uploaded the codes of my implementation online to facilitate the reproducibility of my work by other researchers. Implementing the research findings from this project could offer effective energy solutions for commercial buildings, contributing to the conservation of valuable energy resources worldwide.

7.2 Future work

The study of adversarially robust anomaly detection is a promising area for future research, focusing on creating models that can resist adversarial examples. Current anomaly detection models are easily affected by adversarial attacks, like outliers in HVAC data, which can change the model's predictions. This highlights the need for strong defense strategies specifically designed for detecting new or unusual events.

One potential future direction involves leveraging task-specific knowledge to improve the robustness of anomaly detectors. This can be achieved by manipulating the latent space of models to retain only the necessary information about normal data while discarding adversarial perturbations. Developing more effective adversarial training methods for anomaly detection is another promising area. The use of adversarial training has shown potential in enhancing robustness. However, adapting these methods to anomaly detection requires careful consideration of how adversarial examples are generated and utilized during training.

Additionally, there are challenges that future research needs to address. The first challenge involves the interactions and dependencies within HVAC systems. Specifically, components in HVAC systems are often interconnected, and a fault in one component can spread and cause anomalies in other parts of the system. Understanding these interactions is essential for accurate diagnosis. The second challenge is the impact of seasonal variations. HVAC systems function under different conditions depending on the season (e.g., heating in winter, cooling in summer). Anomalies might be seasonal, necessitating the use of adaptive detection algorithms.

In conclusion, addressing these challenges is crucial for enhancing the reliability of FDD and AD models in real-world scenarios. By creating targeted solutions and implementing rigorous evaluation methods, future research can significantly strengthen the resilience of FDD and AD models against adversarial threats and fluctuating environmental conditions.

References

Aggelis, D., Mpalaskas, A., & Matikas, T. (2013). Investigation of different fracture modes in cement-based materials by acoustic emission. *Cement and Concrete Research*, *48*, 1–8.

Akcay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2019). Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer vision–accv 2018: 14th asian conference on computer vision, perth, australia, december 2–6, 2018, revised selected papers, part iii 14* (pp. 622–637).

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, *2*(1), 1–18.

Batista, G. E., Monard, M. C., et al. (2002). A study of k-nearest neighbour as an imputation method. *His*, *87*(251-260), 48.

Bengio, Y., & Gingras, F. (1995). Recurrent neural networks for missing or asynchronous data. *Advances in neural information processing systems*, *8*.

Bishop, C. M. (1994). Novelty detection and neural network validation. *IEE Proceedings-Vision, Image and Signal processing*, *141*(4), 217–222.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). Lof: identifying density-based local outliers. In *Proceedings of the 2000 acm sigmod international conference on management of data* (pp. 93–104).

Brito, L. C., Susto, G. A., Brito, J. N., & Duarte, M. A. (2022). An explainable artificial intelligence approach for unsupervised fault detection and diagnosis in rotating machinery. *Mechanical Systems and Signal Processing*, *163*, 108105.

Candès, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis? *Journal of the ACM (JACM)*, *58*(3), 1–37.

Chen, Y., Zhou, X. S., & Huang, T. S. (2001). One-class svm for learning in image retrieval. In *Proceedings*

2001 international conference on image processing (cat. no. 01ch37205) (Vol. 1, pp. 34–37).

Chen, Z., O'Neill, Z., Wen, J., Pradhan, O., Yang, T., Lu, X., … others (2023). A review of data-driven fault detection and diagnostics for building hvac systems. *Applied Energy*, *339*, 121030.

Comstock, M., & Braun, J. E. (2002). Fault detection and diagnostic (fdd) requirements and evaluation tools for chillers. *West Lafayette, IN: ASHRAE*.

Comstock, M. C., Braun, J. E., & Groll, E. A. (2002). A survey of common faults for chillers/discussion. *Ashrae Transactions*, *108*, 819.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*(2), 224–227.

Davis, R. A., Lii, K.-S., & Politis, D. N. (2011). Remarks on some nonparametric estimates of a density function. *Selected Works of Murray Rosenblatt*, 95–100.

Du, Z., Fan, B., Jin, X., & Chi, J. (2014). Fault detection and diagnosis for buildings and hvac systems using combined neural networks and subtractive clustering analysis. *Building and Environment*, *73*, 1–11.

El-Yaniv, R., & Nisenson, M. (2006). Optimal single-class classification strategies. *Advances in Neural Information Processing Systems*, *19*.

Fan, Y., Cui, X., Han, H., & Lu, H. (2019). Chiller fault diagnosis with field sensors using the technology of imbalanced data. *Applied Thermal Engineering*, *159*, 113933.

Fessant, F., & Midenet, S. (2002). Self-organising map for data imputation and correction in surveys. *Neural Computing & Applications*, *10*, 300–310.

García-Laencina, P. J., Sancho-Gómez, J.-L., & Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: a review. *Neural Computing and Applications*, *19*, 263–282.

Geweke, J. (1989). Bayesian inference in econometric models using monte carlo integration. *Econometrica: Journal of the Econometric Society*, 1317–1339.

Ghahramani, Z., & Jordan, M. (1993). Supervised learning from incomplete data via an em approach. *Advances in neural information processing systems*, *6*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144.

Gudovskiy, D., Ishizaka, S., & Kozuka, K. (2022). Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *Proceedings of the ieee/cvf winter conference* *on applications of computer vision* (pp. 98–107).

Härdle, W. (1990). *Applied nonparametric regression* (No. 19). Cambridge university press.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527–1554.

Hoffmann, H. (2007). Kernel pca for novelty detection. *Pattern recognition*, *40*(3), 863–874.

Hojjati, H., & Armanfard, N. (2023). Dasvdd: Deep autoencoding support vector data descriptor for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*.

Hu, W., Gao, J., Li, B., Wu, O., Du, J., & Maybank, S. (2018). Anomaly detection using local kernel density estimation and context-based regression. *IEEE Transactions on Knowledge and Data Engineering*, *32*(2), 218–233.

Huang, L., Nguyen, X., Garofalakis, M., Jordan, M., Joseph, A., & Taft, N. (2006). In-network pca and anomaly detection. *Advances in neural information processing systems*, *19*.

Iglesias, F., Zseby, T., Ferreira, D., & Zimek, A. (2019). Mdcgen: Multidimensional dataset generator for clustering. *Journal of Classification*, *36*, 599–618.

Izenman, A. J. (1991). Review papers: Recent developments in nonparametric density estimation. *Journal of the american statistical association*, *86*(413), 205–224.

Jang, K., Hong, S., Kim, M., Na, J., & Moon, I. (2021). Adversarial autoencoder based feature learning for fault detection in industrial processes. *IEEE Transactions on Industrial Informatics*, *18*(2), 827–834.

Jiang, Q., Yan, X., & Huang, B. (2019). Deep discriminative representation learning for nonlinear process fault detection. *IEEE Transactions on Automation Science and Engineering*, *17*(3), 1410–1419.

Jin, M., Jia, R., & Spanos, C. J. (2017). Virtual occupancy sensing: Using smart meters to indicate your presence. *IEEE Transactions on Mobile Computing*, *16*(11), 3264–3277.

Kim, J., & Scott, C. D. (2012). Robust kernel density estimation. *The Journal of Machine Learning Research*, *13*(1), 2529–2565.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Knorr, E. M., Ng, R. T., & Tucakov, V. (2000). Distance-based outliers: algorithms and applications. *The VLDB Journal*, *8*(3), 237–253.

Kumar, P., & Hati, A. S. (2021). Review on machine learning algorithm based fault detection in induction motors. *Archives of Computational Methods in Engineering*, *28*(3), 1929–1940.

Li, B., Cheng, F., Zhang, X., Cui, C., & Cai, W. (2021). A novel semi-supervised data-driven method for chiller fault diagnosis with unlabeled data. *Applied Energy*, *285*, 116459.

Li, D., Tao, Q., Liu, J., & Wang, H. (2021). Center-aware adversarial autoencoder for anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(6), 2480–2493.

Li, D., Zhou, Y., Hu, G., & Spanos, C. J. (2019). Handling incomplete sensor measurements in fault detection and diagnosis for building hvac systems. *IEEE Transactions on Automation Science and Engineering*, *17*(2), 833–846.

Li, L., Hansman, R. J., Palacios, R., & Welsch, R. (2016). Anomaly detection via a gaussian mixture model for flight operation and safety monitoring. *Transportation Research Part C: Emerging Technologies*, *64*, 45–57.

Li, L., Yan, J., Wang, H., & Jin, Y. (2020). Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder. *IEEE transactions on neural networks and learning systems*, *32*(3), 1177–1191.

Li, S., & Wen, J. (2014). A model-based fault detection and diagnostic methodology based on pca method and wavelet transform. *Energy and Buildings*, *68*, 63–71.

Li, X., Jing, J., Bao, J., Lu, P., Xie, Y., & An, Y. (2023). Otb-aae: Semi-supervised anomaly detection on industrial images based on adversarial autoencoder with output-turn-back structure. *IEEE Transactions on Instrumentation and Measurement*, *72*, 1–14.

Liang, J., & Du, R. (2007). Model-based fault detection and diagnosis of hvac systems using support vector machine method. *International Journal of refrigeration*, *30*(6), 1104–1114.

Little, R. J., & Rubin, D. B. (2019). *Statistical analysis with missing data* (Vol. 793). John Wiley & Sons.

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 eighth ieee international conference on data mining* (pp. 413–422).

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *6*(1), 1–39.

Liu, J., Shi, D., Li, G., Xie, Y., Li, K., Liu, B., & Ru, Z. (2020). Data-driven and association rule mining-based fault diagnosis and action mechanism analysis for building chillers. *Energy and Buildings*, *216*, 109957.

Ma, Z., Luo, Y., Yun, C.-B., Wan, H.-P., & Shen, Y. (2022). An mppca-based approach for anomaly detection of structures under multiple operational conditions and missing data. *Structural Health Monitoring*, 14759217221100708.

Ma, Z., Luo, Y., Yun, C.-B., Wan, H.-P., & Shen, Y. (2023). An mppca-based approach for anomaly detection of structures under multiple operational conditions and missing data. *Structural Health Monitoring*, *22*(2), 1069–1089.

Ma, Z., Yun, C.-B., Wan, H.-P., Shen, Y., Yu, F., & Luo, Y. (2021). Probabilistic principal component analysis-based anomaly detection for structures with missing data. *Structural Control and Health Monitoring*, *28*(5), e2698.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., & Frey, B. (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Milligan, G. W. (1985). An algorithm for generating artificial test clusters. *Psychometrika*, *50*, 123–127.

Mirnaghi, M. S., & Haghighat, F. (2020). Fault detection and diagnosis of large-scale hvac systems in buildings using data-driven methods: A comprehensive review. *Energy and Buildings*, *229*, 110492.

Moya, M. M., Koch, M. W., & Hostetler, L. D. (1993). One-class classifier networks for target recognition applications. *NASA Sti/Recon technical report N*, *93*, 24043.

Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., & Lakshminarayanan, B. (2018). Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*.

Ngo, P. C., Winarto, A. A., Kou, C. K. L., Park, S., Akram, F., & Lee, H. K. (2019). Fence gan: Towards better anomaly detection. In *2019 ieee 31st international conference on tools with artificial intelligence (ictai)* (pp. 141–148).

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021). Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)*, *54*(2), 1–38.

Pei, Y., & Zaïane, O. (2006). A synthetic data generator for clustering and outlier analysis.

Principi, E., Vesperini, F., Squartini, S., & Piazza, F. (2017). Acoustic novelty detection with adversarial autoencoders. In *2017 international joint conference on neural networks (ijcnn)* (pp. 3324–3330).

Qu, L., Li, L., Zhang, Y., & Hu, J. (2009). Ppca-based missing data imputation for traffic flow volume: A systematical approach. *IEEE Transactions on intelligent transportation systems*, *10*(3), 512–522.

Rezende, D., & Mohamed, S. (2015). Variational inference with normalizing flows. In *International conference on machine learning* (pp. 1530–1538).

Riazi, M., Zaiane, O., Takeuchi, T., Maltais, A., Günther, J., & Lipsett, M. (2019). Detecting the onset of machine failure using anomaly detection methods. In *International conference on big data analytics and knowledge discovery* (pp. 3–12).

Roberts, S., & Tarassenko, L. (1994). A probabilistic resource allocating network for novelty detection. *Neural Computation*, *6*(2), 270–284.

Rousseeuw, P. J., & Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, *41*(3), 212–223.

Rudolph, M., Wehrbein, T., Rosenhahn, B., & Wandt, B. (2022). Fully convolutional cross-scale-flows for image-based defect detection. In *Proceedings of the ieee/cvf winter conference on applications of computer vision* (pp. 1088–1097).

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., … Müller, K.-R. (2021). A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, *109*(5), 756–795.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., … Kloft, M. (2018). Deep one-class classification. In *International conference on machine learning* (pp. 4393–4402).

Ruff, L., Zemlyanskiy, Y., Vandermeulen, R., Schnake, T., & Kloft, M. (2019). Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4061–4071).

Sabokrou, M., Khalooei, M., Fathy, M., & Adeli, E. (2018). Adversarially learned one-class classifier for novelty detection. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3379–3388).

Sadooghi, M. S., & Khadem, S. E. (2018). Improving one class support vector machine novelty detection scheme using nonlinear features. *Pattern Recognition*, *83*, 14–33.

Schlegl, T., Seeböck, P., Waldstein, S. M., Langs, G., & Schmidt-Erfurth, U. (2019). f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, *54*, 30–44.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., & Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging* (pp. 146–157).

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural computation*, *13*(7), 1443–1471.

Sharifi, R., & Langari, R. (2017). Nonlinear sensor fault diagnosis using mixture of probabilistic pca models. *Mechanical Systems and Signal Processing*, *85*, 638–650.

Shyu, M.-L., Chen, S.-C., Sarinnapakorn, K., & Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the ieee foundations and new directions of data mining workshop* (pp. 172–179).

Steinbuss, G., & Böhm, K. (2021). Benchmarking unsupervised outlier detection with realistic synthetic data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *15*(4), 1–20.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., & Pei, D. (2019). Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining* (pp. 2828–2837).

Tax, D. M., & Duin, R. P. (2004). Support vector data description. *Machine learning*, *54*, 45–66.

Tipping, M. E., & Bishop, C. M. (1999). Mixtures of probabilistic principal component analyzers. *Neural computation*, *11*(2), 443–482.

Tra, V., Amayri, M., & Bouguila, N. (2022a). Outlier detection via multiclass deep autoencoding gaussian mixture model for building chiller diagnosis. *Energy and Buildings*, *259*, 111893.

Tra, V., Amayri, M., & Bouguila, N. (2022b). Unsupervised outlier detection using neural network-based mixtures of probabilistic principal component analyzers for building chiller fault diagnosis. *Building and Environment*, *225*, 109620.

Tra, V., Amayri, M., & Bouguila, N. (2023). Unsupervised fault detection for building air handling unit systems using deep variational mixture of principal component analyzers. *IEEE Transactions on Automation Science and Engineering*.

Tra, V., Duong, B.-P., & Kim, J.-M. (2019). Improving diagnostic performance of a power transformer using an adaptive over-sampling method for imbalanced data. *IEEE Transactions on Dielectrics and Electrical Insulation*, *26*(4), 1325–1333.

Tra, V., Kim, J., Khan, S. A., & Kim, J.-M. (2017). Bearing fault diagnosis under variable speed using convolutional neural networks and the stochastic diagonal levenberg-marquardt algorithm. *Sensors*, *17*(12), 2834.

Tra, V., Nguyen, T.-K., Kim, C.-H., & Kim, J.-M. (2021). Health indicators construction and remaining useful life estimation for concrete structures using deep neural networks. *Applied Sciences*, *11*(9), 4113.

Wen, J., & Li, S. (2011). Tools for evaluating fault detection and diagnostic methods for air-handling units. *ASHRAE 1312-RP*.

Wu, J., Zhao, Z., Sun, C., Yan, R., & Chen, X. (2020). Fault-attention generative probabilistic adversarial autoencoder for machine anomaly detection. *IEEE Transactions on Industrial Informatics*, *16*(12), 7479–7488.

Xiao, F., Zheng, C., & Wang, S. (2011). A fault detection and diagnosis strategy with enhanced sensitivity for centrifugal chillers. *Applied thermal engineering*, *31*(17-18), 3963–3970.

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., ... others (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference* (pp. 187–196).

Yairi, T., Takeishi, N., Oda, T., Nakajima, Y., Nishimura, N., & Takata, N. (2017). A data-driven health monitoring method for satellite housekeeping data based on probabilistic clustering and dimensionality reduction. *IEEE Transactions on Aerospace and Electronic Systems*, *53*(3), 1384–1401.

Yan, K., Chong, A., & Mo, Y. (2020). Generative adversarial network for fault detection diagnosis of chillers. *Building and Environment*, *172*, 106698.

Yan, K., Huang, J., Shen, W., & Ji, Z. (2020). Unsupervised learning for fault detection and diagnosis of air handling units. *Energy and Buildings*, *210*, 109689.

Yan, K., Su, J., Huang, J., & Mo, Y. (2020). Chiller fault diagnosis based on vae-enabled generative adversarial networks. *IEEE Transactions on Automation Science and Engineering*, *19*(1), 387–395.

Yan, K., Zhong, C., Ji, Z., & Huang, J. (2018). Semi-supervised learning for early detection and diagnosis of various air handling unit faults. *Energy and Buildings*, *181*, 75–83.

Yu, L., Snapp, R. R., Ruiz, T., & Radermacher, M. (2010). Probabilistic principal component analysis with expectation maximization (ppca-em) facilitates volume classification and estimates the missing data. *Journal of structural biology*, *171*(1), 18–30.

Zhang, H., Li, C., Li, D., Zhang, Y., & Peng, W. (2021). Fault detection and diagnosis of the air handling unit via an enhanced kernel slow feature analysis approach considering the time-wise and batch-wise dynamics. *Energy and Buildings*, *253*, 111467.

Zhang, H., Li, C., Wei, Q., & Zhang, Y. (2022). Fault detection and diagnosis of the air handling unit via combining the feature sparse representation based dynamic sfa and the lstm network. *Energy and buildings*, *269*, 112241.

Zhang, J., Chen, H., Chen, S., & Hong, X. (2017). An improved mixture of probabilistic pca for nonlinear data-driven process monitoring. *IEEE transactions on cybernetics*, *49*(1), 198–210.

Zhang, J., Chen, M., & Hong, X. (2021). Nonlinear process monitoring using a mixture of probabilistic pca with clusterings. *Neurocomputing*, *458*, 319–326.

Zhao, X., Yang, M., & Li, H. (2012). A virtual condenser fouling sensor for chillers. *Energy and Buildings*, *52*, 68–76.

Zhao, Y., Li, T., Zhang, X., & Zhang, C. (2019). Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future. *Renewable and Sustainable Energy Reviews*, *109*, 85–101.

Zhou, X., Xiong, J., Zhang, X., Liu, X., & Wei, J. (2021). A radio anomaly detection algorithm based on modified generative adversarial network. *IEEE Wireless Communications Letters*, *10*(7), 1552–1556.

Zhou, Z., Chen, H., Li, G., Zhong, H., Zhang, M., & Wu, J. (2021). Data-driven fault diagnosis for residential variable refrigerant flow system on imbalanced data environments. *International journal of refrigeration*, *125*, 34–43.

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., & Chen, H. (2018). Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations.*