Intelligent outlier detection under imperfect industrial data conditions

A Thesis Submitted for the Degree of Doctor of Philosophy

By

Jingzhong Fang

Department of Computer Science,Brunel University London

2026

I would like to dedicate this thesis to my family.

Declaration

I, Jingzhong Fang, hereby declare that this thesis and the works presented in it are entirely my own. Some of the works have been previously published in journal or conference papers, and this has been mentioned in the thesis. Where I have consulted the work of others, this is always clearly stated.

Jingzhong Fang

June 2026

Acknowledgements

At the beginning, I would like to take this opportunity to express my deepest gratitude to my principal supervisor Prof. Zidong Wang, my second supervisor Prof. Xiaohui Liu, and my research development advisor Dr. Weibo Liu, for supporting and guiding my Ph.D. study.

It is my great honor to have Prof. Wang as my Ph.D. supervisor. He is a wonderful supervisor, and I sincerely appreciate his invaluable contributions, dedicated guidance, continuous support, and the significant time he devoted to making my Ph.D. journey both productive and rewarding. His insightful comments and professional guidance, together with his constant encouragement and support, have taught me what high-quality research is and how to grow into a mature researcher. Under his guidance, I learned to be responsible in both my work and my personal life, which will benefit me throughout my career and beyond. I would also like to express my sincere thanks to Prof. Liu for his encouragement and kind support over the years. I greatly appreciate his continuous support and assistance during difficult times.

I would also like to take this opportunity to express my gratitude to the following people for helpful discussions, suggestions, comments, and support of my research during my Ph.D. stage: Dr. Weibo Liu, Prof. Lei Zou, Prof. Hang Geng, Dr. Kaiqun Zhu, Dr. Zhiru Cao, Dr. Mincan Li, Dr. Yibo Huang, Dr. Wenbin Yue, Dr. Jieyu Zhang, Dr. Yang Liu, Dr. Zhongyi Zhao, Dr. Bogang Qu, Dr. Guijun Ma, Dr. Tongjian Liu, Dr. Chuang Wang, Prof. Fei Han, Prof. Juan Li, Dr. Yezheng Wang, Dr. Yu-Ang Wang, Dr. Yuru Guo, Mr. Jiahao Song, Dr. Jiaxing Li, Dr. Jiyue Guo, Dr. Baoye Song, Dr. Shaoying Wang, Dr. Rong Gao, Prof. Fan Wang, Dr. Haijing Fu, Dr. Chaoqing Jia, Dr. Yimeng He, Dr. Zeng Fan, Dr. Yiming Wang, Dr. Hongbin Cai, Prof. Rui Yang, Prof. Jinling Liang, Ms. Linwei Chen, Mr. Juntao Han, Mr. Haobin Cao, Mr. Han Wu, Dr. Weihao Song, Prof. Wentao Mao, Dr. Limin Wang, Dr. Li

Wang, Dr. Hongmei Wang, Ms. Yunjie Chen, Mr. Yuhang Jin, Mr. Pengyu Wen, Mr. Jiale Hong, and Dr. Tao Li.

I would like to thank all my lab mates at the Intelligent Data Analysis Centre for the pleasant working environment: Dr. Ndipenoch Nchongmaje, Dr. Yu Cao, Ms. Linwei Chen, Mr. Juntao Han, Mr. Haobin Cao, Mr. Han Wu, Ms. Lina Huang, Dr. Nick Droutsas, Ms. Yuliya Chystaya, and Mr. Syed Hassan Fatimi. I would also like to thank the Department of Computer Science, Brunel University of London, for supporting my Ph.D. research.

Last but not least, I would like to express my deepest thanks and gratitude to my family. My parents have always supported and encouraged me with constant support, which has continually motivated me to move forward. They never complained and have always given me their unconditional love and unwavering support. Thank you so much for all your love and care!

The path of scientific research is filled with challenges and moments of uncertainty. I am deeply grateful to have been accompanied by my supervisors, colleagues, friends, and family throughout this journey. Along the way, I have gained not only knowledge and research skills but also a deeper appreciation of responsibility, integrity, and self-discipline. These valuable experiences will remain with me throughout my life and continue to guide my future journey.

Abstract

With the growing complexity and data intensity of modern industrial systems, intelligent data analysis (IDA) has become essential for reliable data interpretation and efficient operation. By leveraging advanced analytical and computational techniques, IDA can handle complex, large-scale data sets, thereby providing valuable insights for industrial operations and significantly improving the stability and efficiency of industrial processes. Nevertheless, data quality remains a fundamental prerequisite for IDA performance, as it directly affects the accuracy and trustworthiness of derived insights. Due to complex operating conditions, high cost, and limited availability of expert annotation, high-quality data is often difficult to acquire. The imperfections in the data can affect training and evaluation, thereby reducing the overall performance in real-world applications.

In practice, imperfections in industrial data generally fall into two main categories: 1) data scarcity, such as class imbalance or limited sample size; and 2) label quality issues, such as missing labels or noisy labels.

In this thesis, we deal with the above-mentioned data imperfections arising from the data acquisition process and the annotation process to develop innovative approaches for robust and reliable IDA in industrial applications. It should be pointed out that all approaches developed in this thesis have been evaluated and applied to outlier detection on imperfect industrial data collected from real-world wire arc additive manufacturing (WAAM) processes.

- To achieve outlier detection on unlabeled data, an improved optimization-based clustering algorithm is proposed, where the initial locations of the cluster centroids in the fuzzy C-means algorithm are optimized by an improved particle swarm optimization (PSO) algorithm. An adaptive switching randomly perturbed particle swarm optimization (ASRPPSO) algorithm is developed to enhance the convergence rate and particle's search ability of the PSO algorithm, where a distance-based weighting strategy and

switching strategy are introduced to update parameters of the optimizer. The particles can conduct a thorough search by accounting for both evolutionary states and the distances to the global and personal best positions, thereby improving the convergence rate and solution accuracy. Via the ASRPPSO-based selection of optimal clustering centroids, the proposed algorithm does not rely on centroid initialization, thereby facilitating a better cluster partition.

- With the aim to guarantee the outlier detection performance on imbalanced data under the small sample problem, an optimized deep transfer learning framework is developed, where a novel deep domain adaptation strategy is designed to minimize the cross-domain discrepancies, the weighting factors are designed to handle the data imbalance problem, and the PSO algorithm is utilized for hyper-parameters tuning. By leveraging the domain knowledge and optimal hyper-parameter selection, the developed framework effectively balances performance and efficiency when dealing with outlier detection tasks on imbalanced data under the small sample problem.

- To handle the noisy label problem under limited-data conditions in outlier detection, a novel Transformer-embedded learning with noisy labels framework with fuzzy-clustering-assisted contrastive learning (TFCCL) is developed, where a fuzzy-clustering-assisted contrastive learning approach, a dynamic two-stage training scheme and a joint learning strategy are introduced to train the outlier detector. The TFCCL framework integrates supervised learning with contrastive learning, thereby reducing reliance on potentially noisy labels and enhancing model robustness.

- For the purpose of outlier detection under the noisy label problem when sufficient data are available, a role-differentiated learning with noisy labels (RD-LNL) approach is put forward, where a leader-follower-inspired sample selection (LFSS) strategy is proposed for identifying potential clean samples for model training. A selection metric and an adaptive selection scheme are designed to adjust the number of selected samples. By combining the joint learning and adaptive sample selection, the RD-LNL framework achieves robust outlier detection in the presence of noisy labels.

- To fully evaluate the application potential of the developed frameworks. All the developed frameworks are applied to outlier detection tasks on WAAM data sets

collected from real-world manufacturing processes to examine their adaptability, robustness, and generalization capability. The experimental results demonstrate the effective and robust performance of the developed outlier detection frameworks on WAAM data sets across various data imperfection conditions.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

ACO ant colony optimization

AM additive manufacturing

ARI adjusted rand index

CDLN class-dependent label noise

CNN convolutional neural network

DDA deep domain adaptation

DED direct energy deposition

DL deep learning

DNN deep neural network

DTL deep transfer learning

EC evolutionary computation

FC fully connected

FCM fuzzy C-means

GA genetic algorithm

GELU Gaussian error linear unit

GMM Gaussian mixture model

IDA intelligent data analysis

IDLN instance-dependent label noise

IILN instance-independent label noise

KNN K-nearest neighbors

LNL learning with noisy labels

ML machine learning

MLP multi-layer perceptron

MMD maximum mean discrepancy

NAR noise at random

NCAR noise completely at random

NLP natural language processing

NNAR noise not at random

PSO particle swarm optimization

RI rand index

SA simulated annealing

TL transfer learning

WAAM wire arc additive manufacturing

WK weighted kappa

Chapter 1

Introduction

1.1 Motivation

In recent years, intelligent data analysis (IDA) has become increasingly important in a wide range of industrial manufacturing applications (e.g., aerospace engineering, energy systems, and automotive engineering) with the growing complexity and data intensity of modern industrial systems, which require reliable and accurate data interpretation to ensure stable and efficient operational performance. By leveraging advanced analytical and computational techniques, IDA can efficiently process large-scale data sets, handle diverse and heterogeneous data sources, and extract meaningful patterns. The successful adoption of IDA has enabled reliable data interpretation, provided valuable insights for industrial operations, and significantly improved the efficiency and intelligence of modern industrial systems.

As a disruptive technology in industrial manufacturing, additive manufacturing (AM) has attracted an ever-increasing research interest during the past few years. During the AM process, the metal material required by the production specification is deposited on the substrate layer by layer under the control of a computer [62]. Compared with some traditional subtractive manufacturing technologies, the AM technology exhibits better design flexibility and produces less waste. Thanks to its strong abilities in fabricating components with complex geometries, the AM technology has been successfully applied to a variety of fields such as electrical engineering, healthcare, and transportation [84]. To meet the requirement

of fabricating components with complex structures at fine resolutions, a large number of AM methods have been developed, e.g., selective laser sintering, direct energy deposition (DED), liquid binding in three-dimensional printing, contour crafting, and laminated object manufacturing [125]. Among existing AM methods, the DED method is a competitive one that uses high-power energy sources (including laser beam, electron beam, and electric arc) to deposit the metal powder or feedstock wire into the substrate layer by layer without the requirement of a strict seal structure.

Wire arc AM (WAAM) is a wire-based DED method with relatively high deposition efficiency, which is one of the most popular AM methods. Compared with other AM methods, the WAAM has demonstrated significant advantages in material loss and cost savings [145]. In WAAM, the quality of the fabricated component is highly dependent on the operating status, including feedstock feeding accuracy, shielding gas flow, and inter-layer temperature. As an essential part of the process, the electric arc serves as the heat source and is primarily governed by the total current and arc voltage, which strongly influence the operating status and ultimately the quality of the fabricated components [149]. During the WAAM process, the current and voltage may change irregularly and rapidly owing to abnormalities in the operating conditions (e.g., geometrical deviations and unstable metal transfer), which may adversely affect the working status. Given this, a common way to track the working status of the electric arc in practice is to monitor the total current and arc voltage, as they are straightforward to measure and provide essential information about the operating status.

Reaching a sufficient detection performance is of high industrial relevance because robust monitoring can greatly improve quality control. If the data is only available as a post-process batch, it can be used to guide and optimize post-process inspection. If change-point detection can be implemented in real time, it enables either a controlled stop of the process to perform rectifying actions or, ideally, an automatic corrective action through closed-loop control. Therefore, all important indications in the signal should be reliably detected and, if possible, classified since they may directly influence the quality of the fabricated component.

As one of the powerful analytical techniques in IDA, outlier detection plays a critical role in identifying abnormal instances that may compromise the reliability of data-driven analyses, which has been successfully exploited in various areas, such as medical engineering, finance, electrical engineering, and manufacturing [6, 69]. Outlier detection is able to detect

unusual data patterns, filter out corrupted samples, and enhance the overall reliability of monitoring and analysis in complex industrial environments. Furthermore, it is relatively easy to implement and has proven to be highly effective in real-world industrial settings. As a result, in the field of WAAM, employing outlier detection approaches to identify the abnormal changes in current/voltage has become an effective solution to monitor the operating status and further improve the WAAM process and ensure high-quality production [73].

With the development of advanced sensing technologies and data-driven modeling, machine learning (ML)–based outlier detection has become a powerful approach for identifying abnormal patterns in complex data sets. By exploiting statistical learning and pattern recognition capabilities, ML-based outlier detection approaches can adapt to irregular patterns, capture complex behaviors, and provide efficient and robust detection. Compared with traditional threshold- or rule-based approaches, they offer higher adaptability, stronger generalization, and an improved ability to capture subtle or nonlinear anomalies. Therefore, ML-based outlier detection methods are well-suited for WAAM, which enables more stable process operation and improves the consistency of fabricated components in real manufacturing environments. In the past few years, a great number of ML-based approaches have been developed and utilized in WAAM outlier detection [30, 121].

It is worth mentioning that the performance of ML-based outlier detection methods is highly dependent on data quality, as their accuracy and robustness heavily rely on reliable and representative input data. Nevertheless, in practical WAAM scenarios, data imperfections (e.g., unlabeled data, limited sample size, and noisy labels) are often unavoidable due to various factors such as sensor noise, fluctuating process conditions, data annotation challenges, and human errors. Such imperfections may hide the normal patterns in the data, greatly reduce the overall performance of the ML model, and degrade overall performance in real-world applications. Therefore, it is essential to develop reliable and effective ML-based approaches for outlier detection on imperfect WAAM data to further ensure accurate monitoring and reliable assessment of the WAAM process. As a class of promising approaches in ML, the weakly supervised learning aims to train models under imperfect training data using weak or unreliable annotation signals, which has attracted enormous research interest in recent years. Consequently, weakly supervised learning provides a practical and effective

pathway for designing outlier detection methods that can operate reliably despite the inherent imperfections of WAAM data.

In the past few years, optimization techniques have received considerable attention in the research community. As a powerful optimization algorithm, the particle swarm optimization (PSO) is a population-based algorithm inspired by social behaviors observed in nature, which provides notable advantages, including fast convergence, low computational cost, and efficient exploration of the solution space [42]. Recently, the PSO algorithm has been extensively employed in ML-related optimization problems and has shown consistently strong performance across diverse applications. By leveraging its strong global search capability and adaptive swarm dynamics, PSO is able to effectively tune model hyper-parameters, improve training convergence, and enhance overall prediction performance.

Motivated by the above discussions, the main challenging problems for WAAM outlier detection are dealing with data imperfections arising from unpredictable manufacturing environments and the annotation process. Specifically, data imperfections in WAAM can be categorized into two main types: 1) data scarcity, such as class imbalance or limited sample size; and 2) label quality issues, such as missing labels or noisy labels. The main purpose of this thesis is to conduct a comprehensive study on developing innovative approaches for outlier detection on imperfect data collected from the real-world WAAM process. In Chapter 3, we develop an improved optimization-based clustering algorithm for outlier detection on unlabeled WAAM data. In Chapter 4, an optimized deep transfer learning (DTL) framework is put forward for WAAM outlier detection under the small sample problem. We propose a novel Transformer-embedded learning with noisy labels (LNL) framework for WAAM outlier detection under noisy-label and limited-data conditions in Chapter 5. In Chapter 6, we develop a novel role-differentiated LNL framework for handling the noisy label problem in WAAM outlier detection when sufficient data are available.

1.2 Contribution

The main contributions of this thesis are outlined as follows.

- We propose an improved optimization-based clustering algorithm for outlier detection on unlabeled data. An adaptive switching randomly perturbed particle swarm optimization

(ASRPPSO) algorithm is proposed to improve the particle's search ability and the overall convergence rate. The ASRPPSO algorithm is then combined with the fuzzy C-means (FCM) algorithm to choose an optimal set of initial locations of the cluster centroids automatically. By globally optimizing FCM centroids and reducing sensitivity to initialization, the ASRPPSO-based FCM achieves more stable partitions, which facilitates clearer discrimination between normal and abnormal patterns, thereby improving outlier detection performance. A series of experiments is carried out to comprehensively validate the performance of the developed ASRPPSO algorithm via some well-known benchmark functions. The ASRPPSO-based FCM algorithm is also evaluated through experiments. Experimental results show the effectiveness of the ASRPPSO algorithm and the ASRPPSO-based FCM algorithm.

- We propose a PSO-assisted DTL framework for outlier detection on imbalanced data under the small sample problem. A novel deep domain adaptation (DDA) strategy is introduced to minimize the cross-domain discrepancies. The weighting factors are designed to balance the marginal distribution discrepancy loss and the conditional distribution discrepancy loss of both normal instances and outliers to tackle the data imbalance problem. And the PSO algorithm is utilized to select the optimal hyper-parameters automatically. The developed framework is deployed to detect the outliers on real-world data sets. Experimental results demonstrate the superiority of the developed PSO-embedded DTL outlier detection approach over popular benchmark approaches.

- We put forward a Transformer-embedded LNL framework with fuzzy-clustering-assisted contrastive learning (TFCCL) for outlier detection under noisy-label and limited-data conditions. A fuzzy-clustering-assisted contrastive learning (FCCL) approach is developed to train the Transformer encoder, where the UMAP-based FCM (UFCM) algorithm is introduced to determine positive and negative sample pairs for contrastive learning based on clustering results. A dynamic two-stage scheme is formulated for training the outlier detector. In the first training stage, the Transformer encoder is pre-trained to enhance its feature extraction capability on industrial time series data through data reconstruction. In the second stage, the outlier detector is trained jointly

with the Transformer encoder, while the UFCM algorithm is dynamically updated throughout the training process. Furthermore, a joint learning strategy is proposed for the second training stage, incorporating a label-consistency regularization term to minimize the discrepancy between outputs from the outlier detector and the UFCM algorithm, thereby improving the model's robustness against noisy labels. Experimental results validate the effectiveness of the proposed TFCCL framework on noisy labeled data sets under limited-data conditions.

- We propose a role-differentiated LNL (RD-LNL) approach for outlier detection under the noisy label problem when sufficient data are available. A leader-follower-inspired sample selection (LFSS) strategy is proposed for identifying potential clean samples. The chosen clean samples are then fed into the deep neural networks (DNNs) for outlier detection. In particular, three DNNs are trained collaboratively, where a leader network is pre-trained on clean auxiliary data and guides the joint training of two follower networks by fully leveraging its domain knowledge. A selection metric is introduced that integrates both the training dynamics and the prediction divergence across the DNNs. According to the selection metric, an adaptive selection scheme is proposed to adaptively adjust the clean sample selection ratio (CSSR) during the training process. Experimental results on noisy labeled data verify the practical utility of the developed RD-LNL approach for handling noisy labels.

- To fully evaluate the application potential of the developed frameworks, extensive experiments are conducted on real-world WAAM data sets under various data imperfection conditions, including missing labels, data imbalance, limited samples, and label noise. All the developed frameworks are applied to the outlier detection tasks under these diverse and imperfect data conditions to assess their adaptability and robustness. The experimental results demonstrate that the proposed frameworks consistently deliver effective and robust performance across different types of data imperfections, thereby validating their practicality and reliability for real-world WAAM applications.

1.3 Publication

The following papers report the research work resulting from this thesis.

- **J. Fang**, W. Liu, L. Chen, S. Lauria, A. Miron, and X. Liu, A survey of algorithms, applications and trends for particle swarm optimization, *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 24-50, 2023. (Resulting from Chapter 2)

- **J. Fang**, Z. Wang, W. Liu, Y. Zhang, Y. He, and X. Liu, A comprehensive review on deep learning with noisy labels: Challenges, solutions, and frontiers, *IEEE Transactions on Emerging Topics in Computational Intelligence*, accepted for publication. (Resulting from Chapter 2)

- **J. Fang**, Z. Wang, W. Liu, S. Lauria, N. Zeng, C. Prieto, F. Sikström, and X. Liu, A new particle swarm optimization algorithm for outlier detection: Industrial data clustering in wire arc additive manufacturing, *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 2, pp. 1244-1257, 2024. (Resulting from Chapter 3)

- **J. Fang**, Z. Wang, W. Liu, L. Chen, and X. Liu, A new particle-swarm-optimization-assisted deep transfer learning framework with applications to outlier detection in additive manufacturing, *Engineering Applications of Artificial Intelligence*, vol. 131, art. no. 107700, 2024. (Resulting from Chapter 4)

- **J. Fang**, Z. Wang, W. Liu, N. Zeng, Y. He, Y. Cao, L. Chen, and X. Liu, Learning with noisy labels for industrial time series outlier detection: A Transformer-embedded contrastive learning framework, *IEEE Transactions on Industrial Informatics*, vol. 22, no. 2, pp. 903-913, 2026. (Resulting from Chapter 5)

- **J. Fang**, Z. Wang, W. Liu, and X. Liu, A transfer-learning-assisted role-differentiated approach for industrial outlier detection under label noise, In: *Proceedings of the 2025 International Conference on Automation and Computing (ICAC)*, Loughborough, UK, Aug. 2025, DOI: 10.1109/ICAC65379.2025.11196672. (Resulting from Chapter 6)

1.4 Thesis Structure

To summarize, this thesis is organized into 7 chapters, including the current chapter (an introduction of this thesis). The contents of the rest chapters are summarized in the following manner.

In Chapter 2, we provide the necessary background information of the knowledge related to this thesis. First of all, we present the background of outlier detection. Then, we briefly introduce the data imperfections and weakly supervised learning. After that, we review one of the most challenging problems in the field of weakly supervised learning, namely the noisy-label problem. We review the development of the LNL approaches in two aspects: 1) LNL approaches based on improving the label quality; and 2) LNL approaches based on developing the noise-resistant model. We also summarize the applications of the LNL approaches with details. Next, we introduce hyper-parameter optimization and the basic knowledge of the PSO algorithm, and we review the development of the variant PSO algorithms in the following four aspects: 1) adjusting the control parameters; 2) developing new updating strategies; 3) designing various topological structures; and 4) combining with other evolutionary computation (EC) algorithms. The applications of the PSO algorithm are also presented. In addition, the application backgrounds and the data utilized in this thesis are described in detail.

In Chapter 3, an improved optimization-based clustering algorithm is proposed, where the ASRPPSO algorithm is developed to optimize the cluster centroids. A distance-based weighting strategy is proposed to make full use of the distance from each particle towards its personal best (pbest) and global best position (gbest) at each step to improve the convergence rate and the capability of finding the global best solution. The Gaussian white noises are utilized to randomly perturb the acceleration coefficients to expand the search space and mitigate the premature convergence problem. The switching strategy is also employed to balance the global and local searches. The search capability of the developed ASRPPSO algorithm is comprehensively evaluated via eight well-known benchmark functions, including both the unimodal and multi-modal cases. The developed ASRPPSO algorithm is employed to design an improved clustering algorithm. The novel ASRPPSO-based FCM algorithm combines the ASRPPSO algorithm and the traditional FCM algorithm, where the cluster centroids of the FCM algorithm are selected by the ASRPPSO algorithm automatically. The

experimental results indicate that the ASRPPSO algorithm and the ASRPPSO-based FCM algorithm achieve effective and reliable performance.

In Chapter 4, a PSO-embedded DTL framework is developed, where a new DDA strategy is put forward to reduce the cross-domain discrepancy by adjusting the weight of marginal distribution discrepancy loss and the conditional distribution discrepancy loss. In addition, two separate coefficients are introduced to balance the conditional domain discrepancies of normal instances and outliers with the hope to alleviate the data imbalance problem. The hyper-parameters are tuned by the PSO algorithm automatically. Experimental results demonstrate that the developed PSO-embedded DTL outlier detection approach outperforms the popular benchmark methods on WAAM data under the small sample problem.

In Chapter 5, a novel TFCCL framework is proposed. An FCCL strategy is proposed to aid in training the Transformer encoder, where a UFCM algorithm is designed to process high-dimensional features and identify positive and negative sample pairs for contrastive learning. A dynamic two-stage scheme is introduced for training the outlier detector. In the first stage, the Transformer encoder is pre-trained to enhance feature extraction capability on industrial time series data through data reconstruction. In the second stage, the outlier detector is jointly trained with the Transformer encoder, while the UFCM algorithm is dynamically updated throughout the training process. Furthermore, a joint learning strategy is proposed to improve the classifier's robustness against noisy labels, incorporating a label-consistency regularization term to minimize the discrepancy between the outputs of the outlier detector and the UFCM algorithm. Experimental results validate the effectiveness of the proposed TFCCL framework for outlier detection on noisy-labeled WAAM data sets under limited-data conditions.

In Chapter 6, an RD-LNL approach is developed, where an LFSS strategy is introduced to select clean samples to train the outlier detector. A pre-trained network is employed as the leader network and is utilized to guide the joint training process with two follower networks. A selection metric is designed for sample selection by integrating both training behavior and prediction divergence across the networks. Based on the selection metric, an adaptive selection scheme is put forward to improve the quality of selected samples by adaptively adjusting the CSSR during the training process. Experimental results provide strong evidence that the developed RD-LNL approach is effective for handling data affected by label noise.

In Chapter 7, the work of this thesis is concluded, and some relevant future research directions are presented.

Chapter 2

Backgrounds

In this chapter, we aim to introduce the knowledge of outlier detection, data imperfections and hyper-parameter optimization. We also review the existing weakly supervised learning approaches (especially the LNL approaches) and the development of the PSO algorithm, as well as their applications. In addition, the application backgrounds and details of the data used in this thesis are introduced. In Section 2.1, the knowledge of outlier detection is introduced. In Section 2.2, an introduction of data imperfections and an overview of weakly supervised learning approaches from both algorithmic and practical application perspectives are delivered. Specifically, the background and preliminaries of weakly supervised learning and the noisy label problem are presented. A range of representative approaches for addressing the noisy label problem are also summarized. Several selected applications of LNL are also discussed. In Section 2.3, the hyper-parameter optimization is introduced. A comprehensive review of the PSO algorithms as well as their applications is presented. To be specific, the details of the original PSO as well as the basic PSO are presented. Some existing variant PSO algorithms are summarized. Some popular practical applications of PSO algorithms are also pointed out. In Section 2.4, the application backgrounds and the data sets used in this thesis are described in detail.

2.1 Outlier Detection

Outlier detection refers to the process of identifying data instances that exhibit significant deviation from the patterns followed by the majority of the data. In most data sets, normal

instances tend to follow consistent statistical or structural regularities, whereas outliers break these regularities and appear as rare or irregular observations. The goal of outlier detection is to distinguish such irregular instances from normal ones by examining their deviations in terms of distance, density, distribution, or representation features [1]. Depending on the nature of the data and the modeling approach, outliers may be detected through statistical modeling, similarity measures, clustering-based analysis, or learned representations. Although the underlying principles vary across methods, the essential idea remains the same: to characterize the dominant patterns in the data and identify observations that do not conform to them.

As a class of powerful techniques, machine-learning-based outlier detection methods aim to learn the underlying patterns of normal data and automatically identify instances that deviate from these patterns. By leveraging data-driven modeling and representation learning, such methods provide adaptive mechanisms for detecting irregularities in complex data sets [85]. These methods enable models to automatically extract meaningful patterns from data, adapt to evolving data distributions, and improve detection accuracy by learning from historical observations. With the rapid development of neural network theory, DL has attracted considerable attention in outlier detection due to its strong feature extraction capability. Unlike traditional machine learning methods that often rely on handcrafted features and shallow models (e.g., support vector machines, decision trees, and random forests), DL focuses on building deep neural networks with multiple hierarchical layers, such as convolutional neural networks (CNNs), generative adversarial networks, and Transformers [110]. These architectures enable automatic feature extraction and representation learning, allowing models to capture complex patterns and high-dimensional dependencies within the data. The advancements in these neural network architectures have significantly expanded the potential of DL-based approaches in the field of outlier detection [133].

2.2 Data Imperfections and Weakly Supervised Learning

2.2.1 Preliminaries

In supervised learning, the performance of machine learning models (particularly those based on DL) has been found to rely heavily on the quality of the labeled training data. Unfortunately,

it is both time-consuming and resource-intensive to obtain sufficient high-quality labeled data owing to the considerable labor and computational costs associated with data collection and annotation in real-world scenarios. At present, data labels/annotations are often generated by human annotators or automated labeling tools. Nevertheless, some unexpected issues may be introduced by non-expert sources or through errors made by human annotators, leading to poor-quality labeled data. When the labeled data are of low quality, models may be guided to learn incorrect or ambiguous relationships between features and targets, thereby significantly degrading predictive accuracy and generalization ability. Recognized as a class of effective approaches, weakly supervised learning has been developed to construct reliable predictive models from low-quality labeled data by leveraging inherent structural and contextual information while mitigating the adverse effects of label deficiencies [103]. Based on the types of label deficiencies present in training data, weakly supervised learning approaches are generally classified into three categories: 1) *incomplete* supervision, 2) *inexact* supervision, and 3) *inaccurate* supervision [239]. Specifically, *incomplete* supervision refers to cases in which some data points lack labels entirely. *Inexact* supervision describes situations where only coarse-grained labels are provided. *Inaccurate* supervision pertains to scenarios where labels in the training data are incorrect or misleading. Illustrative examples of *incomplete*, *inexact*, and *inaccurate* supervision are shown in Fig. 2.1.

2.2.2 The Noisy Label Problem

As an emerging technique, deep learning (DL) has been successfully applied to various fields (e.g., image processing, natural language processing (NLP), and speech recognition), thereby demonstrating its significant impact and transformative potential [110]. Nevertheless, inaccurate labels may be introduced by non-expert sources or through errors made by human annotators, leading to poor-quality labeled data. Such inaccurate labels, which give rise to the noisy label problem, wherein incorrect labels provide misleading information during the training process, consequently resulting in incorrect predictions [54]. To date, a variety of approaches have been proposed to address this issue [161]. In this context, the purpose of this section is to provide a comprehensive survey of recent advances in DL-based methods designed to address the noisy label problem.

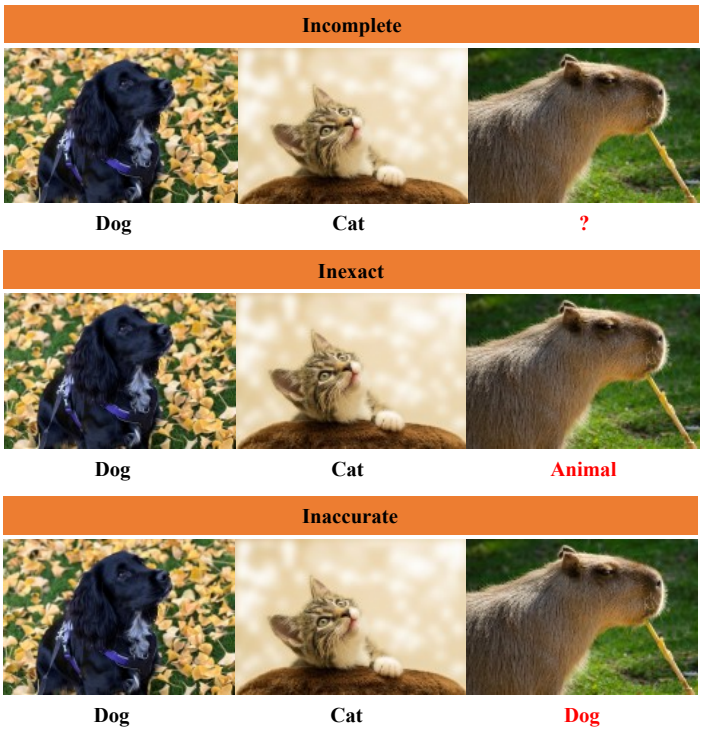


Fig. 2.1 Examples of *incomplete*, *inexact* and *inaccurate* supervised data in weakly supervised learning.

The noisy label problem is considered inevitable in most data analysis tasks. In recent years, a wide range of LNL approaches have been proposed to address the issue of noisy labels. The investigated approaches for handling the noisy label problem are categorized into two groups: i) improving the label quality of the training data, and ii) alleviating the negative influence of the noisy labels. A summary of the reviewed LNL approaches is provided in Fig. 2.2.

Noisy labels refer to labels that are inaccurate or inconsistent with the true class or category of the corresponding data points. It should be noted that the noisy label problem constitutes a subset of *inaccurate* supervision. In DL, labels are typically assumed to indicate the "correct" classes corresponding to their input features. Nevertheless, in real-world data sets, the presence of noisy labels is common and often unavoidable due to several factors: i) mistakes made by human annotators; ii) lack of expert knowledge; and iii) limitations inherent in automated annotation approaches. The inclusion of noisy labels can disrupt the training process, thereby significantly impairing the model's prediction performance and generalization capability.

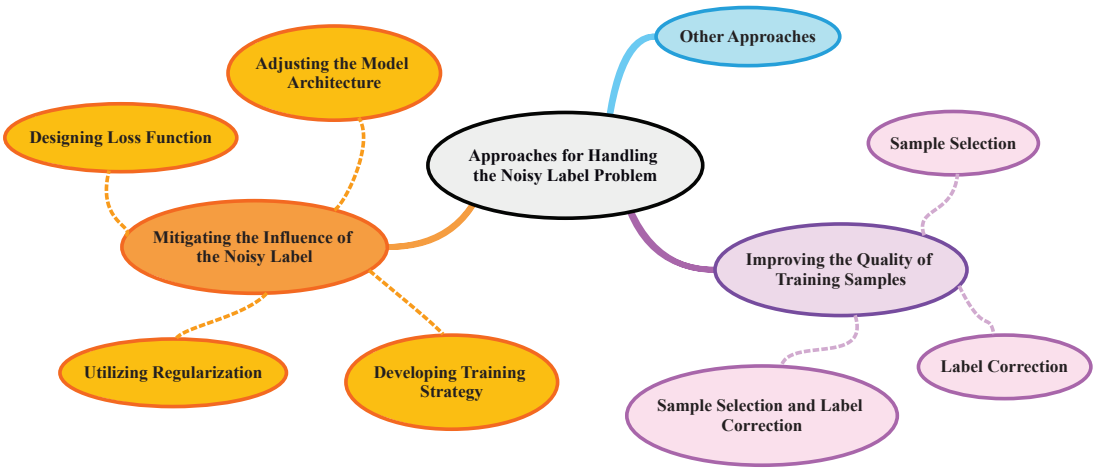


Fig. 2.2 Categorization of the approaches for tackling the noisy label problem.

Consider an M-class classification problem, where the input space is defined as $\mathcal{X} \subset \mathbb{R}^d$, and the label space is given by $\mathcal{Y} = \{y \in \{0,1\}^M \mid \|y\|_1 = 1\}$. Here, y denotes the label, and all labels are assumed to be one-hot encoded vectors corresponding to the M classes.

Let the training set be represented by $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, which consists of N independent and identically distributed samples. Each sample (x_i, y_i) is assumed to be independently drawn from a joint distribution $P(X,Y)$ defined over $\mathcal{X} \times \mathcal{Y}$. The objective of classification is to learn a mapping function $f : \mathcal{X} \to \mathcal{Y}$, implemented by a DNN parameterized by Θ, such that the empirical risk $\mathcal{R}_{\mathcal{D}}(f)$ is minimized:

$$\mathcal{R}_{\mathcal{D}}(f) = \mathbb{E}\left[\mathcal{L}\left(y_i, f(x_i)\right)\right] = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}\left(y_i, f(x_i)\right), \tag{2.1}$$

where $\mathcal{L}$ represents the loss function.

In the classification task under noisy labels, the training data set is denoted as $\tilde{\mathcal{D}} = \{(x_i, \tilde{y}_i)\}_{i=1}^N$, where $\tilde{y}_i$ refers to the (possibly noisy) label of the ith sample. Each sample in $\tilde{\mathcal{D}}$ is independently drawn from a joint distribution $P(X,\tilde{Y})$ defined over the space $\mathcal{X} \times \tilde{\mathcal{Y}}$, where $\tilde{\mathcal{Y}}$ represents the noisy label space. The corresponding empirical risk is then defined as:

$$\mathcal{R}_{\tilde{\mathcal{D}}}(f) = \mathbb{E}\left[\mathcal{L}\left(\tilde{y}_i, f(x_i)\right)\right] = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}\left(\tilde{y}_i, f(x_i)\right). \tag{2.2}$$

During model training, B samples are randomly selected (typically without replacement) from $\tilde{\mathcal{D}}$ to form a mini-batch $\mathcal{B}_t$ at iteration t, where $\mathcal{B}_t = \{(x_i, \tilde{y}_i)\}_{i=1}^{B}$ and $|\mathcal{B}_t| = B$. The model parameters Θ_t at iteration t are updated based on the gradient computed from the mini-batch $\mathcal{B}_t$, using the following update rule:

$$\Theta_{t+1} = \Theta_t - \eta \nabla \left(\frac{1}{B} \sum_{i=1}^{B} \mathcal{L} \left(\tilde{y}_i, f(x_i) \right) \right), \tag{2.3}$$

where η is the learning rate. Due to the presence of noisy labels, inaccurate gradient estimations may be introduced during the risk minimization process, thereby rendering the parameter updates potentially unreliable. Under such circumstances, the model may overfit to noisy labels or learn spurious patterns introduced by label noise. Therefore, it is essential to address noisy labels and mitigate their influence for constructing reliable and robust models.

2.2.3 Taxonomy of the Label Noise

Label noise can be categorized according to various criteria (e.g., distribution, dependency, and cause). In [54], a widely adopted taxonomy of different types of label noise has been introduced. Specifically, label noise is classified into three categories: i) noise completely at random (NCAR); ii) noise at random (NAR); and iii) noise not at random (NNAR).

Noise Completely at Random

The NCAR model, also known as symmetric label noise, refers to a noise mechanism in which the occurrence of label noise is entirely independent of both the input features and the true labels. In the NCAR setting, the probability of a label being corrupted is uniform across all data points. If a label is corrupted, an incorrect label is selected uniformly at random from the remaining classes. Due to its simplicity, NCAR is mathematically tractable and is frequently adopted for theoretical studies. An example of symmetric label noise is illustrated in Fig. 2.3, where each label has a fixed probability of remaining correct (i.e., not being flipped). Nevertheless, NCAR is rarely encountered in real-world scenarios, as it fails to capture the complexities of real-world data in which label noise often depends on the input features or the true labels.

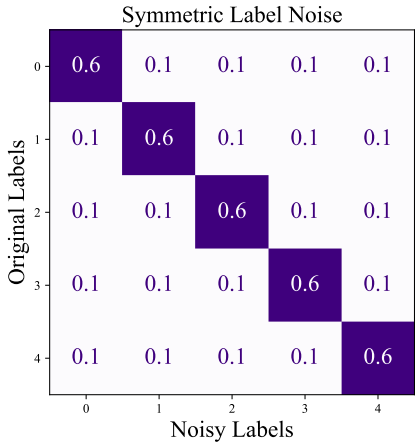


Fig. 2.3 An example of the symmetric label noise.

Noise at Random

The NAR model refers to a type of label noise in which the probability of noise depends on the true labels but remains independent of the input features. In the NAR setting, noise patterns are influenced by the characteristics of specific classes, often arising from semantic similarities or inherent ambiguities among categories. In an NAR data set, certain classes are more likely to be mislabeled than others. Compared with NCAR, the NAR model more accurately reflects real-world conditions and has been widely adopted in both theoretical analyses and practical applications.

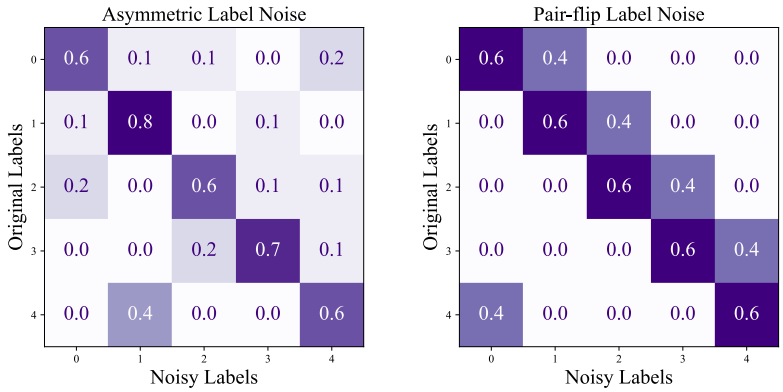
Noise Not at Random

The NNAR model refers to a type of label noise in which the probability of noise depends on both the true labels and the input features. The NNAR model captures the complexities of real-world data sets, where noise patterns are seldom uniform or purely class-dependent. In an NNAR data set, specific classes and individual data points are more likely to be mislabeled than others. Due to its ability to accurately reflect real-world noise distributions, the NNAR has been extensively studied in various domains [33].

Based on the dependency on the true label and input features, NNAR can be further categorized into two groups: i) instance-dependent label noise (IDLN) and ii) instance-independent label noise (IILN). The IDLN represents a realistic and complex form of label noise in which the noisy labels are influenced by both the true label and the specific

characteristics of the data instance. The IILN, also referred to as class-dependent label noise (CDLN), describes a form of noise where the mislabeling probability depends solely on the true label, regardless of the characteristics of individual instances. Compared to IDLN, CDLN is less complex and commonly observed in many practical applications, making it a useful model for describing systematic label noise.

Fig. 2.4 illustrates examples of two widely studied CDLN models: general asymmetric label noise and pair-flip label noise. As shown in Fig. 2.4a, under general asymmetric label noise, each label is flipped with a probability that depends on its true class. When a flip occurs, the new label is selected from a subset of alternative classes according to a non-uniform distribution, often reflecting inherent confusion or semantic similarity among classes. Pair-flip label noise constitutes a special case of asymmetric noise, wherein each flipped label is deterministically assigned to a specific alternative class, rather than being sampled from multiple candidates. An illustration of pair-flip label noise is provided in Fig. 2.4b.



(a) General asymmetric label noise. (b) Pair-flip label noise.

Fig. 2.4 Examples of the general asymmetric label noise and the pair-flip label noise.

2.2.4 LNL Approaches Based on Improving the Label Quality

Improving the label quality of training data is regarded as a direct and intuitive strategy, which focuses on correcting incorrect labels or selecting clean samples. A brief overview of recently developed LNL approaches aimed at improving label quality is provided in Table 2.1.

Table 2.1 Approaches for improving the label quality

Approach	Reference
Sample Selection	[4, 26, 37, 49, 51, 53, 59, 71, 79, 87, 92, 100–102, 119, 126, 130, 137, 154, 173, 191, 194, 195, 200, 201, 203–205, 217, 221, 223]
Label Correction	[18, 23, 24, 72, 98, 176, 181, 196, 207]
Selection and Correction	[50, 104, 107, 136, 163, 202]

It is worth noting that DNNs have been experimentally observed to memorize easy samples first and gradually adapt to harder samples as training progresses over multiple epochs [10, 229]. Remarkably, this phenomenon has been consistently observed regardless of the optimization techniques or network architectures employed. Moreover, noisy samples tend to incur higher loss values during the early stages of training, allowing for a distinction between clean and noisy samples based on their respective loss distributions. To date, numerous LNL approaches have been proposed to enhance the quality of training data. Based on their underlying strategies, the LNL methods aimed at improving label quality are mainly categorized into three groups: i) sample selection, ii) label correction, and iii) a combination of sample selection and label correction.

Sample Selection

The core idea of sample selection lies in identifying clean samples for model training, with the aim of mitigating the impact of incorrect labels. The main procedure of a basic sample selection approach is illustrated in Fig. 2.5. Due to its conceptual simplicity and ease of implementation, a wide range of sample-selection-based methods has been proposed to address the noisy label problem.

1. Multiple Network Co-training Strategies Co-training-based methods, which involve training multiple networks simultaneously, have shown strong robustness to label noise. Motivated by the notable success of the co-training strategy in semi-supervised learning and multi-view learning, multiple network structures have been effectively employed to handle the noisy label problem, demonstrating promising performance across various tasks. In [119], a novel meta-algorithm named *Decoupling* has been proposed for tackling label noise. In this method, the decision of "when" to update the parameters is decoupled from "how" to update them, which is accomplished by maintaining two networks whose parameters are updated only when their predictions differ for a given sample. This disagreement-based

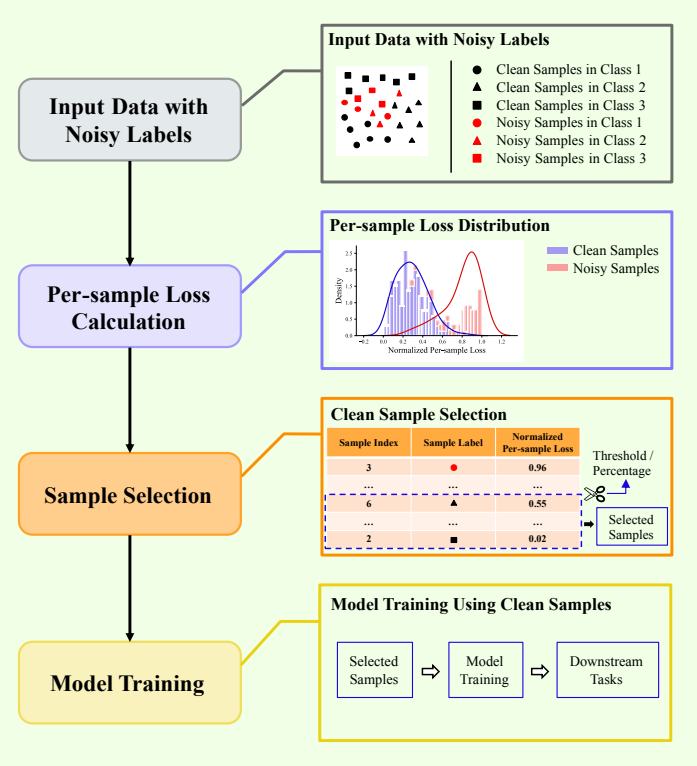


Fig. 2.5 Main procedure of the basic sample selection approach. In sample selection, the loss value of each sample is computed during the early training stage, and the per-sample loss distribution is subsequently estimated. Based on this distribution, samples with relatively low loss values are selected as clean samples and used for model training and downstream tasks.

update strategy is both simple and effective, enhancing robustness to noisy labels without introducing additional time-consuming procedures during training. However, selection errors made by the networks may accumulate over time, ultimately leading to a degradation in overall model performance.

A representative approach named *Co-teaching* has been proposed in [71] to mitigate the error accumulation issue observed in *Decoupling*. In this method, two networks (based on a Siamese architecture) are trained simultaneously, where each network selects its small-loss samples to update the parameters of its peer network. This cross-update strategy enables mutual error correction during the training process, as the two networks possess different learning capacities and tend to identify different types of errors. Unfortunately, in the later stages of training, the two networks are likely to converge to a consensus, thereby diminishing the effectiveness of the strategy. To address this limitation, *Co-teaching+* has been introduced

in [221], combining the disagreement-based updating strategy with the original *Co-teaching* framework.

Similar to its predecessor, *Co-teaching+* employs the Siamese network architecture and trains two networks concurrently. In contrast to *Co-teaching*, where small-loss samples are directly selected for cross-updating, *Co-teaching+* first identifies samples on which the two networks disagree, and then selects small-loss samples from this subset for cross-update. By incorporating the disagreement strategy, *Co-teaching+* mitigates premature convergence between the networks and enhances both robustness and generalization performance. The details of *Decoupling*, *Co-teaching*, and *Co-teaching+* are illustrated in Fig. 2.6.

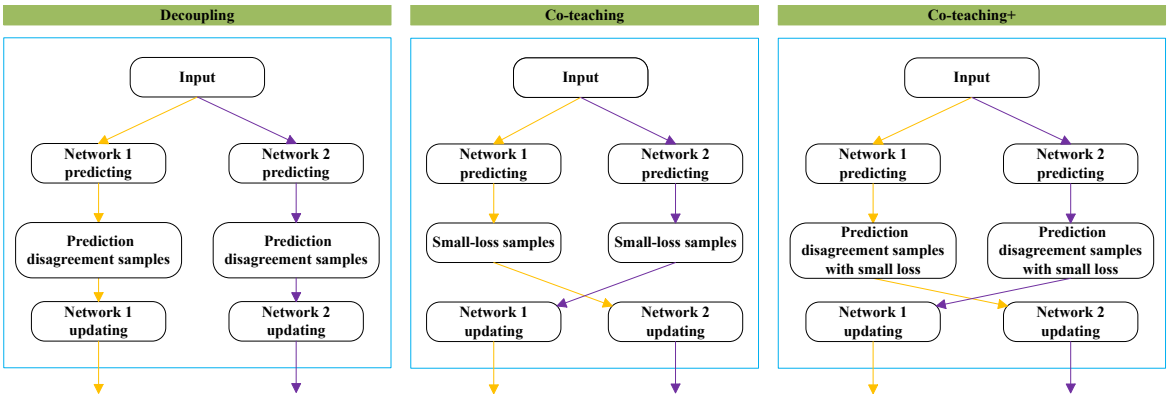


Fig. 2.6 A comparison of the *Decoupling*, *Co-teaching*, and C*o-teaching+* in a mini-batch during the training process. In *Decoupling*, each network is updated by using samples that have prediction disagreements. In *Co-teaching*, each network is updated by using small-loss samples selected by its peer network. In *Co-teaching+*, each network is updated by using small-loss samples that have prediction disagreements.

Inspired by agreement maximization, *JoCoR* [195] has jointly trained two networks by maximizing agreement between their predictions. A co-regularization term has been introduced to encourage consistent predictions, and parameter updates have been performed using small joint-loss samples. Similarly, *CoDis* [204] has employed a cross-updating mechanism between two networks, selecting samples with high prediction discrepancies as likely clean candidates. *Co-LDL* [173] has integrated co-training with label distribution learning by using small-loss and large-loss samples for peer exchange and adaptive label refinement, while also including a self-supervised module to enhance representation learning.

2. Semi-supervised Learning and DivideMix Variants Semi-supervised learning has been widely adopted for noisy label learning due to its ability to leverage both labeled and unlabeled data. *DivideMix* [101] has modeled loss distributions using Gaussian mixture models (GMMs) to separate clean and noisy samples, which are then processed using the MixMatch algorithm. *Manifold DivideMix* [53] has further enhanced this framework by learning embeddings through contrastive loss and applying K-nearest neighbors (KNN) filtering. The MixEMatch method has performed interpolation in both input and latent spaces for stronger generalization. In [200], early/late dropout regularization has been incorporated into DivideMix to prevent overfitting during initial training stages.

Other adaptations have included domain-specific extensions, such as *INL-SSL* in [26], where MixMatch and dual GMMs have been applied for time-series data with augmentations. *UNICON* [92] has addressed class imbalance in sample selection using divergence-based uniform selection and contrastive loss. *MentorNet* [87] has dynamically assigned weights to samples to guide a student network. Similarly, *SELF* [126] has integrated supervised and semi-supervised learning via ensemble predictions with a mean teacher model.

3. Contrastive Learning and Representation Robustness Contrastive learning has been increasingly incorporated into LNL methods to improve robustness against label noise. *Sel-CL* [102] has formed confident training pairs by leveraging label agreement and representation similarity, enhanced by Mixup and semi-supervised learning. *MOIT* and its refined version *MOIT+* [130] have combined supervised contrastive learning with classification using interpolation strategies and pseudo-label filtering. *CTRL* [223] has applied clustering to training loss values to differentiate clean and noisy samples for re-training.

4. Dynamic Sample Selection and Reweighting Strategies Numerous approaches have dynamically identified and reweighted training samples to mitigate noise. *O2U-Net* [79] has tracked the evolution of sample losses throughout training to identify noisy labels. *ITLM* [154] has used iterative trimmed loss minimization for sample selection. *BARE* [137] has performed mini-batch-based reweighting using loss statistics without relying on hyper-parameters. *PLS* [4] has introduced a pseudo-loss metric and confidence-guided contrastive learning to filter noisy samples.

RTME [205] has applied regularly truncated M-estimators to assign adaptive weights to both small- and large-loss samples, reducing overfitting and preserving potentially useful information. *NoiseBox* [51] has combined subset expansion and training modules for efficient label noise mitigation. *GBS* and *IGBS* [203] have applied granular computing to enhance both noisy and imbalanced classification via selective boundary sampling.

5. Instance-dependent Label Noise (IDLN) Handling Several methods have specifically addressed the challenges of instance-dependent label noise. *TCP* [201] has introduced a time-consistency prediction metric for clean/noisy sample separation, jointly optimizing a classifier and a noise-transition estimator. *InstanceGM* [59] has leveraged a hybrid generative-discriminative framework with variational autoencoders and DivideMix-inspired training to improve generalization under IDLN.

6. Open-set and Task-specific Adaptations *Sel-CL*, *MOIT*, and *RTME* have been adapted for open-set noise scenarios or tailored domains. For example, the iterative framework named *IL* in [194] has combined contrastive learning and reweighting for open-set LNL. A regression-specific method named *RNF* in [100] has introduced adaptive filtering based on local density and fitting difficulty using an ensemble mechanism. [37] has proposed a two-stage framework (*LongReMix*) combining small-loss clean set identification with oversampling. *ProMix* [191] has expanded the clean set with matched high-confidence samples, while debiasing margin-based loss and employing a semi-supervised framework for balanced classifier training.

7. Automated and Optimization-based Methods *AutoSample* [217] has applied automated machine learning for optimal sample selection. A compact search space has been designed, and optimization has been guided by stochastic relaxation and Newton's method, allowing dynamic and efficient sample selection during training. *OT-Filter* [49] has employed optimal transport and Wasserstein distance for robust sample selection, using sparsity regularization to reduce confirmation bias and improve selection accuracy.

Label Correction

Despite sample selection, in recent years, label correction has become a popular strategy for tackling the noisy label problem. Label correction attempts to correct the labels of noisy samples for the purpose of making use of all the samples in the data set. Fig. 2.7 demonstrates the main procedure of the basic label correction approach.

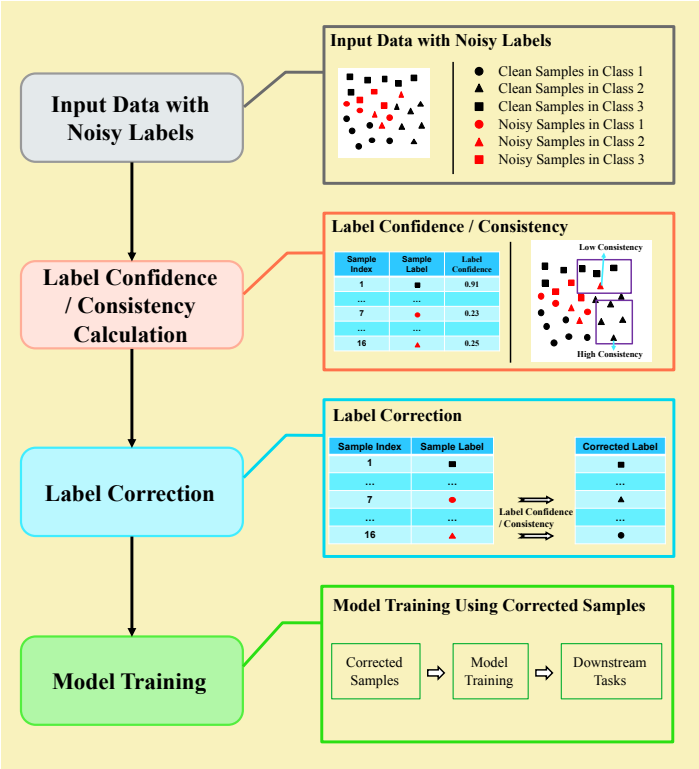


Fig. 2.7 Main procedure of the basic label correction approach. In label correction, label confidence or consistency is first calculated. Based on this measure and the corresponding model outputs, labels with low confidence or consistency are then corrected. The samples with corrected labels are subsequently used for model training and downstream tasks.

1. Joint Optimization of Labels and Network Parameters Several approaches have addressed label noise by iteratively refining both model parameters and labels within a unified optimization process. In [176], a joint optimization framework (*JOF*) has been proposed in which both labels and network parameters have been refined iteratively. A regularization term has been introduced to avoid trivial solutions in the label space, and high learning rates have been applied to prevent overfitting to noisy labels. Similarly, [207]

has presented a probabilistic graphical model (*PGM*) integrated with CNNs, where the expectation-maximization (EM) algorithm has been employed to iteratively refine true labels and update network parameters. These frameworks have effectively captured the mutual dependency between label quality and model learning dynamics.

2. Influence-based and Meta-learning Strategies for Label Correction Label noise has also been addressed by estimating the impact of individual samples on model behavior. In [72], a label detection and correction method *IFDC* has been proposed using the influence function and cross-entropy loss to detect noisy labels, which have then been corrected and reintegrated during training. *RDIA* [98] has built upon this idea by directly using influence analysis to relabel noisy samples, mitigating both label noise and distributional bias. A scalable variant has also been introduced to improve efficiency on deep models. In contrast, *DMLP* [181] has separated representation learning from label correction to prevent interference between feature extraction and label purification. A self-supervised pre-training phase and dual-process correction mechanism have been incorporated for enhanced robustness and computational efficiency.

3. Embedding-aware and Contrastive Learning Methods Embedding-based strategies have been proposed to improve label quality by exploiting feature representations. The *SERA* framework [18] has used a shared embedding space where an autoencoder, classifier, and clustering module have been jointly trained. Noisy labels have been dynamically corrected through iterative refinement without requiring prior knowledge of the noise level. Likewise, *SNSCL* [196] has introduced a supervised contrastive learning framework that is tolerant to stochastic noise. A noise-aware contrastive loss has been used along with a momentum queue update mechanism. Probabilistic feature embeddings have been generated without relying on hand-crafted augmentations, improving generalization and label correction accuracy.

4. Graph-based and Structure-aware Correction Methods Structural information such as graph topology has also been leveraged to enhance label correction. *ERASE* [23] has introduced a robust training objective by maximizing coding rate reduction and has performed noise-tolerant learning in two phases. The first phase has applied structural denoising based on graph topology to pre-correct labels, and the second has used semantic label propagation

to refine node representations during training. This two-stage approach has improved both structural consistency and semantic accuracy in graph-based tasks.

5. Fairness-aware and Confidence-based Label Correction Addressing label noise in imbalanced or biased settings has also received attention. In [24], a label correction strategy (*FLC*) has been proposed for imbalanced subpopulations using a feature-based label confidence estimation metric. Samples with low-confidence labels have been corrected, and training has been performed with distributionally robust optimization to enhance both fairness and robustness. This method has been particularly well-suited for scenarios where label noise is unevenly distributed across subgroups.

Selection and Correction

It is worth noting that the combination of sample selection and label correction has also become a widely adopted strategy in LNL. This integrated approach enables more effective sample utilization and mitigates the issue of limited clean samples, and has been increasingly applied in recent LNL methods.

1. General Frameworks Combining Sample Selection and Label Correction Several approaches have integrated sample selection with label correction in a unified framework to simultaneously identify reliable samples and correct noisy labels, thereby improving training efficiency and model robustness.

- *SSR* (Sample Selection and Relabeling) [50] has introduced a hybrid framework for handling unknown label noise. Sample selection has been carried out using a non-parametric KNN classifier, while relabeling has relied on a parametric classifier. An optional feature consistency regularization term has been included to enhance robustness under noisy conditions.

- *RUC* (Robust Learning with Unsupervised Confidence) [136] has served as an add-on method to refine predictions from unsupervised learning algorithms. Clean samples have been identified using various strategies (confidence-based, metric-based, and hybrid), and label correction has been performed using a refined classifier. Co-training and label smoothing have been further used to reduce overfitting.

- *DISC* (Dynamic Instance-Specific Selection and Correction) [104] has proposed a dynamic thresholding mechanism for adaptive sample selection based on confidence momentum. A multi-view learning strategy has been employed for label correction using weak and strong augmentations. Data have been categorized into clean, hard, and purified subsets, with tailored regularization applied to each to optimize training under noise.

2. Graph-based Sample-Label Joint Correction Approaches Structural information in graph-structured data has also been leveraged to combine sample selection with label correction. In this context, *GNN Cleaner* [202] has been designed for graph data with noisy labels. A modified label propagation algorithm has been used to generate pseudo-labels informed by graph topology. These pseudo-labels have facilitated both dynamic sample selection and label correction. The method has been made compatible with a range of GNN architectures, enhancing scalability and robustness.

3. Stage-wise Learning Pipelines with Combined Strategies Some approaches have adopted multi-stage learning pipelines that explicitly separate phases for selection, correction, and training while integrating them into a cohesive framework.

- *TSNT* (Two-Stage Noise-Tolerant Framework) [107] has proposed a two-stage LNL approach. In the first stage, clean samples have been selected based on consistency between original and predicted labels using a self-refining strategy and a dynamic mask loss. The second stage has used co-training of dual networks for mutual correction of noisy labels. Additional mechanisms such as rectified cross-entropy loss, Kullback-Leibler (KL) divergence for mutual information, and a noise-robust triplet loss have been incorporated to further improve reliability and feature discrimination.

- *TSBF* (Three-Stage Bootstrap Framework) [163] has targeted the IDLN setting without requiring clean labels. In stage one, self-supervised pre-training and early stopping have been used to identify clean samples. In stage two, clean samples have been used to bootstrap label relationships and relabel the remaining data using semi-supervised learning. In stage three, the classifier has been trained on the relabeled

data. This approach has effectively balanced noisy samples and integrated LNL with semi-supervised learning in an end-to-end pipeline.

2.2.5 LNL Approaches Based on Developing the Noise-resistant Model

Rather than improving the label quality of training data, developing noise-resistant models has emerged as an effective alternative strategy for addressing label noise. Based on their underlying methodologies, the reviewed approaches for constructing noise-resistant models in this paper are primarily categorized into four groups: i) adjusting model architecture; ii) designing robust loss functions; iii) incorporating regularization techniques; and iv) developing specialized training schemes. A summary of the reviewed approaches for building noise-resistant models is provided in Table 2.2.

Table 2.2 Approaches for developing noise-resistance models

Approach	Reference
Adjusting Model Architecture	[20, 63, 216, 220]
Designing the Loss Function	[9, 44, 75, 117, 138, 174, 215]
Introducing Regularization into the Model	[47, 48, 81, 108, 182, 222, 238]
Developing the Training Scheme	[67, 70, 116, 234, 235]

Adjusting Model Architecture

In recent years, some approaches have been presented to address the noisy label problem by adjusting the model architecture [63, 216, 220]. In [216], a quality embedding model has been proposed to alleviate the impact of noisy labels, where a quality variable is introduced to measure label trustworthiness and is embedded into different subspaces for effective supervision. A contrastive-additive noise network (*CAN*) is also developed to enhance the training quality, where a contrastive layer is utilized to estimate the quality variable and an additive layer is employed to combine noisy and predicted labels for robust learning.

A robust framework called coupled mix for graph contrast (*OMG*) for graph classification under label noise has been presented in [220], where the graph contrastive learning is combined with coupled Mixup and neighbor-aware noise removal. To further improve the representation learning performance, the Mixup augmentation is utilized to create challenging

positive and negative pairs. In addition, the curriculum-inspired noise filtering is employed to reduce noise-induced overfitting.

So far, incorporating a noise adaptation layer into the model has been demonstrated by numerous studies to be an effective and reliable approach for handling the label noise. In [63], a noise adaptation layer for DNNs (*NA-DNN*) has been introduced to model the label noise and learn the transition from true labels to noisy labels. The developed framework jointly optimizes the noise model and the classifier using a neural-network-based optimization strategy instead of traditional EM methods to further improve efficiency and scalability.

An attention-based pseudo-label propagation network (*APPN*) has been developed in [20] for few-shot learning in the presence of noisy labels, where an attention-mechanism-based feature extraction module and a multi-step noise detection module are designed. An optimized method is also proposed for improving the label propagation accuracy and overall performance.

Loss Function

The loss function is used to guide the optimization process, which plays a critical role during model training. In the past few years, a lot of approaches have been developed, which aim to modify the loss function of the model to further improve the model performance in the presence of noisy labels [75, 117, 174].

1. Robust Loss Functions for Direct Noise Suppression In [75], a bounded neural network (*BNN*) has been designed to handle the noisy label problem in fault diagnosis, where a Box-Cox loss function is introduced to improve the label noise tolerance of *BNN*, and a bounded loss function is proposed to avoid overfitting from noisy labels. Furthermore, a surrogate training strategy based on an alternative convex search is employed to ensure the stability of the model in the early stage of training.

A curriculum loss has been introduced in [117] as a tighter and more robust version of the 0-1 loss to reduce the impact of noisy labels. A noise-pruned curriculum loss (*NCL*) is also proposed to handle data sets with high levels of label noise by automatically pruning noisy samples based on their loss values.

Two robust loss correction methods named *Backward Correction* and *Forward Correction* have been put forward in [138] for training DNNs on noisy labeled data sets, where a noise transition matrix is applied to adjust the loss function to ensure the robustness of the model. Specifically, a practical algorithm is introduced for estimating the noise transition matrix in multi-class settings.

2. Self-supervised and Metric-based Loss Formulations In [174], a novel framework named *Co-learning* has been proposed, where supervised learning and self-supervised learning are combined to handle noisy labels. To be specific, the intrinsic similarity and structural similarity within the training data are employed in loss calculation to align supervised and self-supervised tasks to enhance the robustness without requiring noise rate estimation or clean validation sets.

An adaptive hierarchical similarity metric learning (*AHSML*) method is proposed for LNL in [215], where two noise-intensive information, named class-wise divergence and sample-wise consistency are introduced. To be specific, the class-wise divergence is calculated based on the average intra-class similarity and inter-class similarity to measure the similarity between different classes for capturing hierarchical relationships and structured feature embeddings. The sample-wise consistency is utilized to enhance the model's robustness by ensuring that augmented versions of the same sample produce similar feature representations. The proposed method dynamically assigns adaptive hierarchical margins to sample pairs instead of relying on fixed binary similarity, which improves its robustness against noisy labels.

3. Probabilistic and Meta-learned Loss Adaptation In [9], a robust framework *NMLC* has been developed for handling label noise, where the loss distributions are modeled by a beta mixture model to distinguish clean and noisy samples. A dynamic bootstrapping loss that adjusts prediction weights based on noise probabilities is introduced and is integrated with MixUp augmentation to improve regularization and robustness. The developed framework is able to dynamically adapt to noise characteristics without requiring a clean validation set.

A noise-aware-robust-loss-adjuster (*NARLA*) has been presented in [44] to adaptively predict hyper-parameters for each instance, where the meta-learned functions are integrated into the training process to enable better suppression of noisy samples and amplification of

clean samples. A bi-level optimization framework is also developed for training the presented loss adjuster, which improves its scalability and effectiveness in diverse noise scenarios.

Regularization

Regularization is a class of techniques utilized to prevent overfitting and improve the model's generalization ability. In the past few years, a great number of LNL approaches incorporating regularization techniques have been presented to enhance model robustness against label noise and improve generalization ability [48, 81, 182].

1. Neighborhood- and Consistency-based Regularization In [81], the *NCR* has been introduced for LNL, where the feature-space neighbor relationships are leveraged to improve label consistency and reduce noise impact. In *NCR*, the model predictions are encouraged to align with those of neighboring samples, which reduces reliance on noisy labels and improves generalization. The *NCR* can be integrated smoothly into standard training workflows, which avoids the need for complex multi-stage setups.

A consistent graph neural network (*CGNN*) has been put forward in [222] for addressing node classification in graphs in the presence of noisy labels, where the graph contrastive learning is applied as a regularization term to improve the robustness of node representations to label noise. In addition, a homophily-based noise removal strategy is put forward to identify and purify noisy nodes to ensure efficient semantic graph learning.

2. Distribution-aware and Geometry-aware Regularization The *WAR* has been proposed in [48], where the Wasserstein distance is employed to incorporate class geometry into adversarial regularization. *WAR* adjusts the regularization strength based on class similarity, thereby encouraging smoother decision boundaries while preserving robustness for overlapping classes.

In [47], a leave-noise-out regularization (*LNOR*) has been proposed to mitigate the influence of noisy samples. To be specific, a novel framework using cross-augmentation matching is developed to distinguish noisy samples in tail classes. Besides, an online prediction penalization is designed to address the class imbalance problem by dynamically rebalancing training data.

3. Temporal and Self-supervised Regularization An early-learning regularization (*ELR*) framework has been put forward in [108] to prevent DNNs from memorizing noisy labels by making use of the early-learning phenomenon. The *ELR* adjusts model training dynamically with a regularization term based on previous predictions of the model to improve the model's robustness against label noise.

In [182], a self-supervised adversarial noisy masking (*SANM*) framework has been developed, where the adversarial noisy masking and the self-supervised reconstruction are combined to handle noisy samples and provide noise-free auxiliary information, respectively. To be specific, the adversarial noisy masking strategy applies different mask ratios to noisy and clean samples based on a noise estimation model to ensure reliable learning. A decoder-based auxiliary task is introduced in self-supervised reconstruction to reconstruct original images from masked inputs to improve generalization. The noisy label regularization is applied to penalize overconfident predictions to enhance robustness against label noise.

4. Sparsity-based Regularization In [238], sparse regularization (*SR-LNL*) has been deployed to the network outputs with the aim of enhancing the model's robustness. The output sharpening and l_p-norm regularization are employed to promote sparsity while maintaining the model's learning ability. Theoretical analysis has been provided to prove that any loss function can be made robust against noisy labels by restricting the network output to a permutation set over a fixed vector and satisfying the symmetric condition for robustness.

Training Strategy

The training strategy plays a crucial part in DL as it defines the methods and approaches employed to train a model effectively, which ensures both the generalization ability and efficiency of the model.

1. Curriculum Learning Strategies In [67], a scalable framework called *CurriculumNet* has been developed for handling noisy labels by utilizing curriculum learning. Specifically, a novel curriculum learning strategy is designed, where data complexity is ranked in an unsupervised manner using density-based clustering. The model is then trained sequentially,

where the training process starts with simple and clean data and gradually incorporates noisy and complex data.

2. Meta-learning and Consistency-based End-to-End Training In [235], an LNL framework with an efficient end-to-end training process (*ETE-LNL*) has been designed, where the data reweighting, relabeling, and supervised learning are integrated during model training. Specifically, a small trusted clean data set is utilized to optimize the sample weights and pseudo-labels via meta-reweighting and meta-relabeling methods, respectively. In addition, augmentation techniques and consistency regularization are employed to improve pseudo-label initialization, thereby generating sharp and reliable label estimates even in high-noise scenarios.

3. Adversarial Training in Federated Learning A generative adversarial framework (*GA-LNL*) has been proposed in [70] for training noise-robust classifiers in federated learning by simulating label poisoning attacks. Both the CDLN and IDLN are applied to create poisoned data sets to improve the performance of adversarial training and enhance the resilience of the global model.

4. Prototype-based Co-training with Contrastive Learning In [234], a novel LNL framework named *RankMatch* has been proposed, which makes use of both confidence and consistency to train deep neural networks. To be specific, a confidence-based strategy is introduced for identifying clean samples, which first identifies high-confidence prototypes for each class and then determines whether a sample's label is correct based on its agreement with the closest prototypes. A rank contrastive loss is designed to enhance representation learning and keep consistency among similar samples by using the rank statistics of principal features. The clean and noisy samples are then trained simultaneously in a co-training manner using a joint loss function to improve the model's robustness and generalization ability.

5. Post-hoc Statistical Correction A per-class statistic estimation (*PCSE*) method has been proposed in [116] for estimating clean per-class statistics from noisy labeled data, where a mathematical relationship between noisy and clean statistics is established to provide strong theoretical guarantees and scalability to real-world noisy data sets. The *PCSE* serves as a

general post-processing method, which can be applied to various popular networks pre-trained on the noisy data set to boost their classification performance.

Other Emerging Strategies for Learning with Noisy Labels

Apart from the above-mentioned two categories of approaches, many other strategies have been recently proposed to deal with noisy label problems [80, 230].

In [80], a twin contrastive learning (*TCL*) framework has been developed for classification under noisy labels, where contrastive learning with MixUp augmentation is applied for robust representation learning. To be specific, GMMs are employed for noise detection, and cross-supervision with entropy regularization is utilized to refine noisy labels. The *TCL* detects noisy labels as out-of-distribution samples and aligns embeddings with clean targets to alleviate the noise impact effectively.

A novel and challenging type of label noise named *BadLabel* has been designed in [230], where the labels are flipped away from class boundaries and the noisy and clean labels are indistinguishable. To deal with *BadLabel*, a *Robust DivideMix* approach has also been proposed in [230], where adversarial label perturbation and Bayesian GMMs are employed for noise detection and model training.

A meta-learning framework named variational rectification inference (*VRI*) has been proposed in [170], where the adaptive rectification of loss functions is formulated as an amortized variational inference problem. The hierarchical Bayesian modeling, evidence lower bound optimization, and bi-level learning are integrated to reduce noise effects and prevent model collapse.

2.2.6 Comparative Summary

In this subsection, a systematic overview of reviewed LNL approaches is provided to facilitate a clear and structured comparison of their experimental settings and applicability, which is summarized in Table 2.3. The table compares different methods in terms of their application domains, evaluated data sets (including public and private data sets), noise types, and the highest noise ratios considered in the experiments. This summary provides a structured view of how existing LNL approaches are evaluated under different noisy-label settings.

Table 2.3 Comparative summary of selected LNL approaches in terms of application domains, data sets (public and private), noise types, and maximum noise ratios.

Category	Approach	Application Domain	Data Set		Noise Type	Highest Noise Ratio
			Public Data Set	Private Data Set		
Sample Selection	Decoupling [119]	Image Analysis	MNIST, LFW	×	NCAR	40% (NCAR)
	Co-teaching [71]	Image Analysis	MNIST, CIFAR-10/100	×	NCAR, CDLN	50% (NCAR), 45% (CDLN)
	Co-teaching+ [221]	Image Analysis, NLP	MNIST, CIFAR-10/100, Tiny-ImageNet, NEWS, ImageNet-32, SVHN	×	NCAR, CDLN	50% (NCAR), 45% (CDLN)
	JoCoR [195]	Image Analysis	MNIST, CIFAR-10/100, Clothing1M	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	CoDis [204]	Image Analysis, NLP	Food-101, Clothing1M, WebVision, MNIST, CIFAR-10, SVHN, NEWS	×	NCAR, CDLN	80% (NCAR), 45% (CDLN)
	Co-LDL [173]	Image Analysis	CIFAR-100, Web-Aircraft, Web-Car, Web-Bird	×	NCAR, CDLN	80% (NCAR), 50% (CDLN)
	DivideMix [101]	Image Analysis	CIFAR-10/100, Clothing1M, WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	Manifold DivideMix [53]	Image Analysis	CIFAR-10/100, Clothing1M, WebVision, Mini-ImageNet	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	DivideMix with Dropout [200]	Image Analysis	CIFAR-10/100, Clothing1M	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	INL-SSL [26]	Time-series Analysis	CWRU	✓	NCAR	90% (NCAR)
	UNICON [92]	Image Analysis	CIFAR-10/100, Tiny-ImageNet, Clothing1M, WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	MentorNet [87]	Image Analysis	CIFAR-10/100, ImageNet	×	NCAR	80% (NCAR)
	SELF [126]	Image Analysis	CIFAR-10/100, ImageNet	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	Sel-CL [102]	Image Analysis	CIFAR-10/100, ImageNet, WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	MOIT & MOIT+ [130]	Image Analysis	CIFAR-10/100, Mini-ImageNet, Mini-WebVision	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	CTRL [223]	Image Analysis	CIFAR-10/100, ANIMAL-10N, Food-101/101N, F-MNIST			
		Tabular Data Analysis	Cardiotocography, Credit Fraud, Human Activity, Letter, Mushroom, Satellite, Sensorless Drive	×	NCAR, CDLN	40% (NCAR), 40% (CDLN)
	O2U-Net [79]	Image Analysis	CIFAR-10/100, Mini-ImageNet, Clothing1M	×	NCAR, CDLN	80% (NCAR), 38% (CDLN)
	ITLM [154]	Image Analysis	CIFAR-10, MNIST	×	NCAR, CDLN	70% (NCAR), 40% (CDLN)
	BARE [137]	Image Analysis	CIFAR-10, MNIST	×	NCAR, CDLN	70% (NCAR), 45% (CDLN)
	PLS [4]	Image Analysis	CIFAR-100, Miny-ImageNet	×	NCAR	80% (NCAR)
	RTME [205]	Image Analysis	MNIST, CIFAR-10/100, NEWS, SVHN, Food-101, Clothing1M, CIFAR-10N/100N	×	NCAR, CDLN, IDLN	50% (NCAR), 45% (CDLN), 50% (IDLN)
	NoiseBox [51]	Image Analysis	CIFAR-10/100, Mini-ImageNet, Clothing1M, WebVision, ANIMAL-10N	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	GBS & IGBS [203]	Tabular Data Analysis	Fourclass, Svmguide1, Diabetes, Breastcancer, Sonar, Splice, Votes, Clean1, Isolet5, Codrna, Ijcnn1, Mushrooms, CreditApproval, Svmguide3,	×	NCAR	40% (NCAR)
	TCP [201]	Image Analysis	F-MNIST, SVHN, CIFAR-10/100, CIFAR-10N/100N	×	IDLN	50% (IDLN)
	InstanceGM [59]	Image Analysis	CIFAR-10/100, ANIMAL-10N, Red Mini-ImageNet, Clothing1M	×	IDLN	80% (IDLN)
	IL [194]	Image Analysis	CIFAR-10	×	CDLN	40% (CDLN)
	RNF [100]	Tabular Data Regression	Airquality, Autompg8, Bikesharing, Ele-2, Forestfires, Friedman, Gasturbine	×	NCAR	50% (NCAR)
	LongReMix [37]	Image Analysis	CNNpred, Compactiv, Concrete, Facebook, CIFAR-10/100, Red Mini-ImageNet, Clothing1M, WebVision, Food-101N	×	NCAR, CDLN	90% (NCAR), 49% (CDLN)
	ProMix [191]	Image Analysis	CIFAR-10/100, Clothing1M, ANIMAL-10N, CIFAR-10N/100N	×	NCAR, CDLN	90% (NCAR), 50% (CDLN)
	AutoSample [217]	Image Analysis	CIFAR-10/100, MNIST	×	NCAR, CDLN	50% (NCAR), 45% (CDLN)
	OT-Filter [49]	Image Analysis	CIFAR-10/100, Clothing1M, ANIMAL-10N	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
Label Correction	JOF [176]	Image Analysis	CIFAR-10, Clothing1M	×	NCAR, CDLN	90% (NCAR), 50% (CDLN)
	PGM [207]	Image Analysis	CIFAR-10	✓	CDLN	50% (CDLN)
	IFDC [72]	Image Analysis	DDSM	✓	CDLN	50% (CDLN)
	RDIA [98]	Image Analysis	MNIST, CIFAR-10,			
		Tabular Data Analysis	Covtype, Criteo (1%), Avazu, Breast-cancer, Diabetes, News20, Adult, Real-sim	×	NCAR	80% (NCAR)
	DMLP [181]	Image Analysis	CIFAR-10, CIFAR-100, Tiny-ImageNet, Clothing1M, WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	SERA [18]	Time-series Analysis	ArrowHead, CBF, Epilepsy, FaceFour, MelbourneP, NATOPS, OSULeaf, Plane, Symbols, Trace	✓	NCAR, CDLN	60% (NCAR), 40% (CDLN)
	SNSCL [196]	Image Analysis	Stanford Dogs, Stanford Cars, Aircraft, CUB-200-2011, Clothing-1M, Food-101N	×	NCAR, CDLN	40% (NCAR), 30% (CDLN)
	ERASE [23]	Image Analysis	Cora, CiteSeer, PubMed, CoraFull, OGBn-arxiv	×	NCAR, CDLN	50% (NCAR), 50% (CDLN)
	FLC [24]	Image Analysis	Waterbirds, CelebA, CIFAR-10/100, Mini-WebVision, ANIMAL-10N	✓	NCAR, CDLN	90% (NCAR), 40% (CDLN)
Selection & Correction	SSR [50]	Image Analysis	CIFAR-10/100, Clothing1M, WebVision, ANIMAL-10N	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	RUC [136]	Image Analysis	CIFAR-10/20, STL-10, ImageNet-50	×	IDLN, CDLN	N/A
	DISC [104]	Image Analysis	CIFAR-10/100, Tiny-ImageNet, ANIMAL-10N, Food-101N, WebVision, Clothing1M	×	NCAR, CDLN, IDLN	50% (NCAR), 45% (CDLN), 60% (IDLN)
	GNN Cleaner [202]	Image Analysis	Cora, CiteSeer, PubMed, OGB-Arxiv, Clothing1M, WebVision	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	TSNT [107]	Image Analysis	Market1501, DukeMTMC-ReID, CUHK03	×	CDLN	60% (CDLN)
	TSBF [163]	Image Analysis	CIFAR-10/100, ANIMAL-10N, Mini-WebVision	×	NCAR, CDLN, IDLN	90% (NCAR), 40% (CDLN), 35% (IDLN)
Model Architecture	CAN [216]	Image Analysis	WEB, AMT, VOC2007, VOC2012, SD4	×	NCAR, CDLN	N/A
	OMG [220]	Image Analysis	MUTAG, PTC, COX2, NCI1, PROTEINS, PROTEINS_F, IMDB-B, IMDB-M	×	NCAR	50% (NCAR)
	NA-DNN [63]	Image Analysis	MNIST, CIFAR-100	×	CDLN	50% (CDLN)
	APPN [20]	Image Analysis	CIFAR-FS, FC100, miniImageNet, tieredImageNet	×	CDLN	40% (CDLN)
Loss Function	BNN [75]	Image Analysis	×	✓	NCAR	40% (NCAR)
	NCL [117]	Image Analysis	MNIST, CIFAR-10/100, Tiny-ImageNet	×	NCAR, CDLN	60% (NCAR), 37% (CDLN)
	Backward/Forward Correction [138]	Image Analysis, NLP	MNIST, IMDB, CIFAR-10, CIFAR-100, Clothing1M	×	NCAR, CDLN	20% (NCAR), 60% (CDLN)
	Co-learning [174]	Image Analysis	CIFAR-10/100, ANIMAL-10N, Food-101N	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
	AHSML [215]	Image Analysis	Cars196, CUB-200-2011, Stanford Online Products	×	NCAR	70% (NCAR)
	NMLC [9]	Image Analysis	CIFAR-10/100, Tiny-ImageNet, Clothing1M	×	NCAR	90% (NCAR)
	NARLA [44]	Image Analysis	CIFAR-10/100, Tiny-ImageNet, ANIMAL-10N, Food-101N, Mini-WebVision	×	NCAR, CDLN, IDLN	60% (NCAR), 60% (CDLN), 50% (IDLN)
Regularization	NCR [81]	Image Analysis	CIFAR-10/100, Mini-ImageNet, WebVision, Clothing1M	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	CGNN [222]	Image Analysis	Amazon Photo, Amazon Computers, Coauthor CS	×	NCAR, CDLN	20% (NCAR), 20% (CDLN)
	WAR [48]	Image Analysis	F-MNIST, CIFAR-10/100, Clothing1M, ISPRS Vaihingen	×	CDLN	40% (CDLN)
	LNOR [47]	Image Analysis	CIFAR-10/100, MNIST, F-MNIST, Clothing1M	×	NCAR, CDLN	20% (NCAR), 20% (CDLN)
	ELR [108]	Image Analysis	CIFAR-10/100, Clothing1M, Mini-WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	SANM [182]	Image Analysis	CIFAR-10/100, Clothing1M, ANIMAL-10N	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	SR-LNL [238]	Image Analysis	MNIST, CIFAR-10/100, WebVision	×	NCAR, CDLN	80% (NCAR), 40% (CDLN)
Training Strategy	CurriculumNet [67]	Image Analysis	WebVision, ImageNet, Clothing1M, Food-101	×	CDLN	50% (CDLN)
	ETE-LNL [235]	Image Analysis	CIFAR-10/100, WebVision, Clothing1M, Food-101N	×	NCAR, CDLN	80% (NCAR), 80% (CDLN)
	GA-LNL [70]	Tabular Data Analysis	UNSW-NB15, NIMS	×	CDLN	40% (CDLN)
	RankMatch [234]	Image Analysis	CIFAR-10/100, Clothing1M, WebVision	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	PCSE [116]	Image Analysis	CIFAR-10/100	×	NCAR, CDLN	60% (NCAR), 40% (CDLN)
Others	TCL [80]	Image Analysis	CIFAR-10/100, WebVision, Clothing1M	×	NCAR, CDLN	90% (NCAR), 40% (CDLN)
	Robust DivideMix [230]	Image Analysis	CIFAR-10/100, MNIST, CIFAR-10N, Clothing1M	×	NCAR, CDLN, IDLN	80% (NCAR), 40% (CDLN), 80% (IDLN)
	VRI [170]	Image Analysis	CIFAR-10/100, Clothing1M, Food-101N, ANIMAL-10N, CIFAR-80N	×	NCAR, CDLN, IDLN	60% (NCAR), 60% (CDLN), 40% (IDLN)

Note: N/A indicates that the corresponding information is not reported.

2.2.7 LNL in Real-world Applications

In this subsection, several selected real-world applications of LNL are categorized into four classes and reviewed, including image analysis, NLP, time-series analysis, and other emerging applications.

Image Analysis

Image analysis has been employed in numerous real-world applications and has remained a critical research direction in computer science. In this domain, visual patterns constitute essential features for analyzing and understanding the structural characteristics of data. However, the presence of noisy labels in image data has caused models to learn inaccurate visual patterns across different classes, thereby reducing overall model performance. Fig. 2.8 presents detailed information regarding the noisy label problem in image analysis, including the underlying causes of label noise, typical forms of noisy labels, and prevalent application areas affected by label noise. In recent years, a variety of LNL approaches have been proposed to address the noisy label issue in practical image analysis tasks [128, 139, 233].

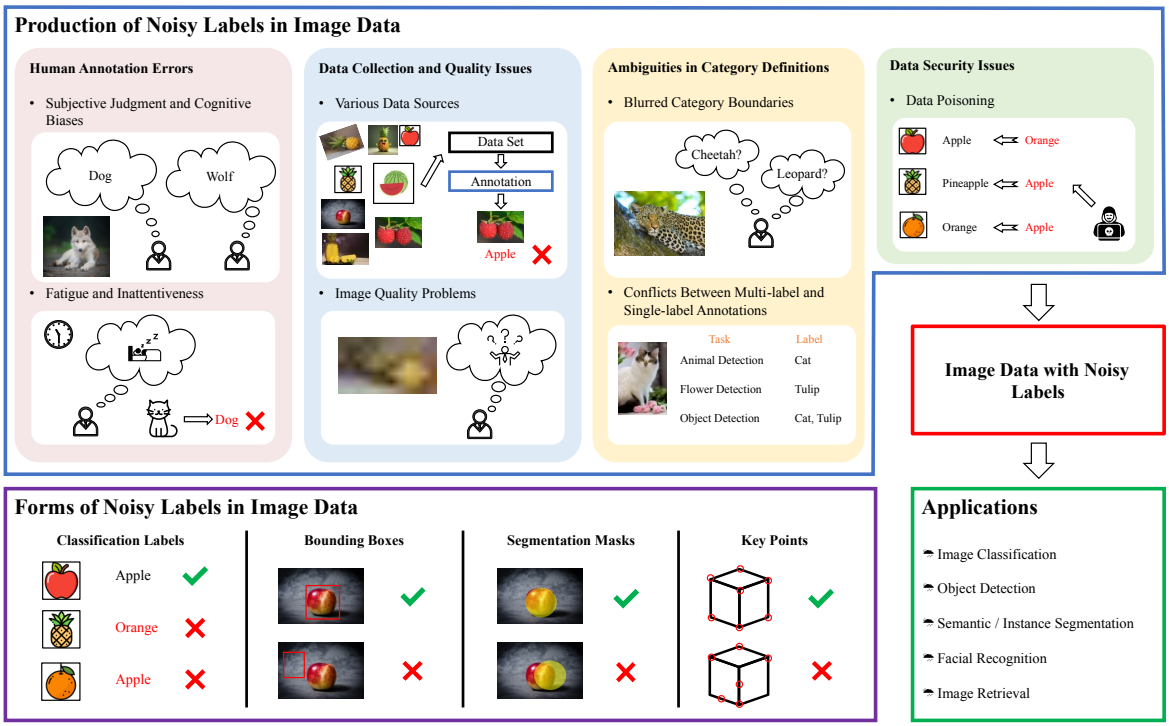


Fig. 2.8 Causes, forms, and application areas of label noise in image data.

1. Image Analysis in Healthcare Image data has been extensively used in the field of healthcare, supporting a wide range of critical applications. Healthcare has relied heavily on various image analysis technologies to diagnose, monitor, and treat diseases. With the advancement of DL techniques, image analysis has become a powerful tool in healthcare. Nevertheless, the presence of noisy labels in data sets has led to erroneous diagnoses. Therefore, it is essential to address such issues to ensure the reliability and accuracy of image analysis tasks in healthcare [55, 97, 139].

In [139], a robust multi-label classification framework for chest X-ray interpretation with uncertain labels has been developed, which has demonstrated the potential to assist radiologists in diagnosing thoracic diseases more accurately and efficiently. In [97], a novel unsupervised-learning-based framework has been proposed for medical image-to-image translation, where a registration network has been employed to adaptively manage images with noisy labels. An iterative learning framework has been proposed in [211] for the classification of skin lesion images with label noise. In [55], a weakly supervised learning model has been introduced for the classification of aortic valve malformations. A noise-robust framework has been developed in [188] for segmenting COVID-19 pneumonia lesions from CT scans, which has improved segmentation performance under noisy label conditions. In [90], a dual-uncertainty-based framework has been proposed to address the label noise problem in medical image classification, where two common types of label noise (i.e., disagreement noise and single-target noise) have been considered. In [155], a graph temporal ensembling method has been developed for robust histopathology image classification in the presence of noisy labels, where a self-ensembling deep architecture has been adopted to combine labeled and unlabeled data while enhancing robustness to label noise.

2. Image Analysis in Industry In industrial domains, image analysis has played a transformative role in automating tasks, improving efficiency, and driving innovation. It has been considered vital to detect and manage label noise in data sets to ensure the quality of image analysis tasks, thereby further reducing operational costs and enhancing overall efficiency and accuracy in industrial applications. To date, a number of approaches have been proposed to address label noise issues in industrial image analysis tasks [128, 175, 231].

In [128], a novel framework has been developed for surface defect detection under the challenge of noisy labels, where rough set theory and Bayesian neural networks have been employed to capture uncertainty in noisy labels for robust detection. A robust framework for defect classification in noisy industrial data sets has been proposed in [175], in which a memory-augmented autoencoder with sparse addressing and trust-region updates has been adopted to enhance noise resilience. In [231], a robust DL framework named adaptive loss-weighted meta-ResNet has been introduced for fault diagnosis of wind turbine gearboxes under noisy label conditions, where a meta-learning strategy has been utilized to adaptively reweight the loss function to improve model robustness and diagnostic accuracy.

3. Image Analysis in Person Identification Person identification is a popular application area that identifies or verifies a person based on their features. It is worth mentioning that the performance of person identification heavily relies on the quality of labels in the training data set. In recent years, many approaches have been developed to handle the noisy label problems in person identification [107, 218, 232, 233].

In [233], a novel method named erasing attention consistency has been put forward to address noisy label problems in facial expression recognition, where an imbalanced framework with a consistency regularization mechanism has been employed to automatically suppress noisy samples during training. A novel framework based on graph convolutional networks has been proposed in [232] for cleansing large-scale face recognition data sets with noisy labels, where a cascaded global-to-local approach has been used to identify and remove noisy samples, thereby improving data quality and enhancing face recognition performance.

In [218], a collaborative refining framework has been presented to improve person re-identification under label noise, where self-label refining and online co-refining strategies have been employed to iteratively improve labels and network predictions in a collaborative manner. A robust framework has been developed in [219] for person re-identification under noisy label conditions, where joint label refinement and hard-aware instance reweighting have been applied to improve model robustness and re-identification performance. In [31], a part-based pseudo-label refinement framework has been proposed to address unsupervised person re-identification with noisy pseudo-labels effectively. In [107], the *TSNT* framework

has been employed for person re-identification under noisy labels, and it has demonstrated satisfactory performance under various noise conditions.

4. Image Analysis in Other Fields Besides the aforementioned application scenarios, numerous approaches have recently been introduced for image analysis tasks under noisy label conditions in various other fields [86, 210].

In [86], a robust label noise cleansing framework based on multilayer spectral-spatial graphs has been developed for hyper-spectral image classification, where spectral and spatial priors have been employed to formulate the noisy label cleansing as an optimization problem regularized by graph constraints. A multiscale superpixel segmentation has also been incorporated to enhance classification accuracy in noisy environments. In [210], a robust DL framework named dual-channel residual network has been proposed for hyper-spectral image classification under noisy label conditions, in which a dual-channel architecture and a noise-robust loss function have been utilized to alleviate the influence of label noise while maximizing the utility of mislabeled samples.

Natural Language Processing (NLP)

NLP is a prominent research area in artificial intelligence, which aims to enable computers to understand and analyze human language. In recent years, the utilization of NLP has increased significantly to meet the demands of tasks such as text analysis and speech recognition. In NLP, the recognition of semantic patterns is critically important for ensuring robust language understanding. Nonetheless, the presence of noisy labels may disrupt these patterns, leading to ambiguous representations and diminishing the model's ability to accurately interpret text. In Fig. 2.9, the causes, forms, and application areas of label noise in textual data are illustrated in detail. To address the label noise problem in NLP, various LNL approaches have been proposed to date [33, 60].

In [33], pre-trained language models have been employed to identify label errors in natural language data sets, demonstrating that sorting data points by their out-of-sample loss during fine-tuning outperforms more complex error detection mechanisms. In [60], a novel approach has been introduced to handle label noise in text classification tasks, where a noise modeling method incorporating an auxiliary noise model trained alongside the

classifier has been employed. This approach addresses both random and input-conditional label noise by utilizing a denoising loss to enhance classification robustness. In [240], the robustness of BERT to label noise in text classification tasks has been investigated, and its performance in combination with noise-handling techniques has been assessed. In [88], a training framework for DNNs has been developed for text classification with noisy labels, in which a non-linear noise modeling layer has been applied to enable the network to learn robust sentence representations under extreme noise conditions. In [27], the robustness of the in-context learning capabilities of Transformers in the presence of noisy labels has been examined, focusing on the model's behavior under varying noise conditions during training and inference. A novel framework for LNL in sentence-level sentiment classification has been proposed in [190], where two CNNs have been utilized to estimate the noise transition matrix and predict clean labels, respectively.

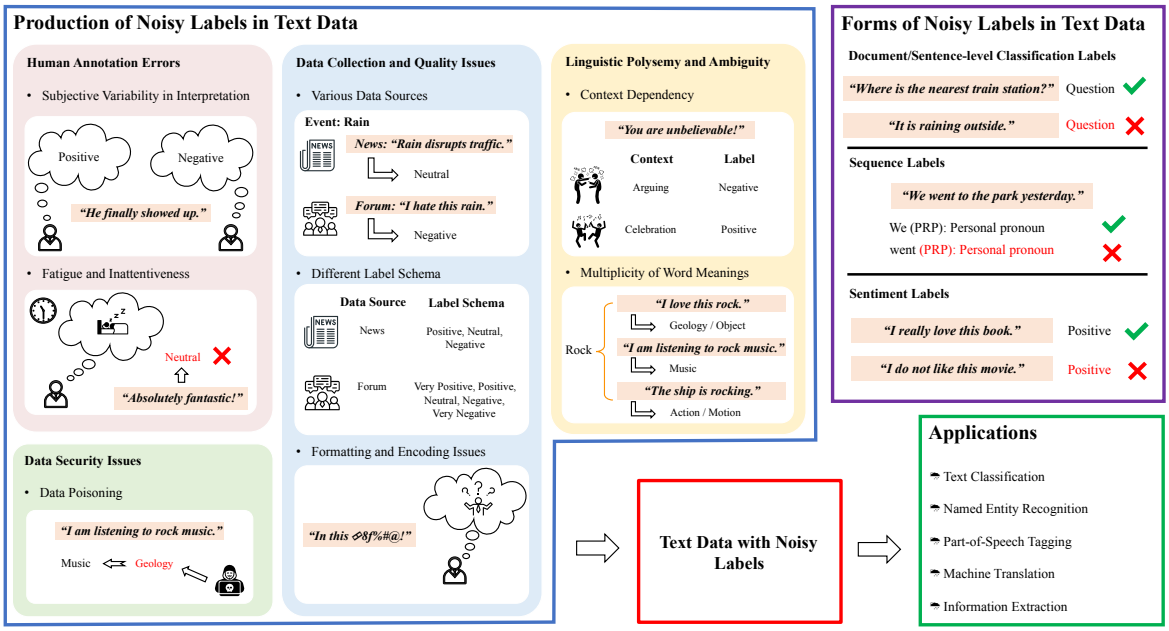


Fig. 2.9 Causes, forms, and application areas of label noise in text data.

Time-Series Analysis

Time-series data refers to a collection of observations or measurements recorded periodically, where the order of the data represents the changes over time. So far, time-series analysis has become a powerful tool in various fields to recognize patterns, trends, and other temporal

structures in data. It is worth mentioning that in time series analysis, noisy labels can severely affect the extraction of temporal dependencies, which leads to degraded performance in forecasting and trend analysis. The causes, forms, and application areas of label noise in time series data are demonstrated in Fig. 2.10. To further improve the performance of time-series analysis tasks under noisy labels, a large number of LNL approaches have been presented in the past few years [18, 26, 113].

In [18], a multi-task DL approach has been developed for time-series classification with label noise, which addresses challenges caused by noisy sensor readings, particularly in estimating the power output of combined heat and power machines. In [26], a robust method has been introduced for fault diagnosis of industrial systems using time-series rolling element-bearing data in the presence of noisy labels. In [113], a new DL paradigm has been proposed for time series classification in the presence of noisy labels, where the multi-scale properties of time series data are utilized and a cross-scale fusion mechanism and graph-based label correction are employed to enhance the robustness of the classification models. In [144], a joint learning framework has been introduced for the classification of incomplete time series with noisy label conditions, which integrates missing data interpolation with feature learning and classification and employs a robust loss function to reduce the effects of label noise to enhance classification performance.

Other Applications

Apart from the above-mentioned application areas, a great number of LNL approaches have been developed for many other real-world applications with noisy label problems [38, 75]. For example, in [75], a framework for DL-based fault diagnosis in high-speed train components has been proposed, where a bounded neural network is utilized to address the challenges of label noise and improve learning robustness in large-scale monitoring data sets. In [38], a two-stage learning framework has been presented for noisy label learning for security defect prediction. In [206], a unified noisy pseudo-label learning framework has been proposed for handling the noisy label problem in semi-supervised temporal action localization, which contains three main steps, namely noisy label ranking, noisy label filtering, and noisy label learning to improve pseudo-label quality, address class imbalance, and enhance noise tolerance. In [237], a graph convolutional label noise cleaner has been designed for video

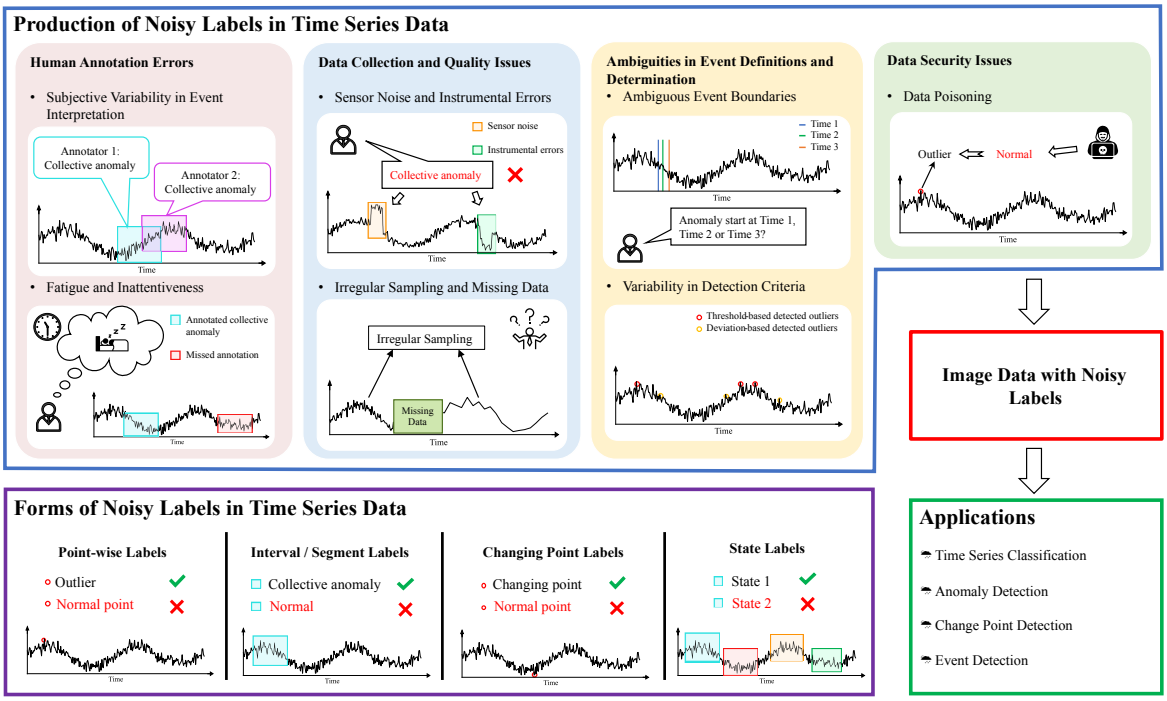


Fig. 2.10 Causes, forms, and application areas of label noise in time series data.

anomaly detection with noisy labels, where a graph convolutional network is used to propagate supervisory signals and refine noisy labels. To summarize, the study of LNL has become increasingly important due to the high costs of obtaining high-quality labels and the volume of data required to train effective AI agents is extraordinarily large.

2.3 Hyper-Parameter Optimization

Optimization plays a critical role in a variety of research fields such as mathematics, economics, engineering, and computer science. Recognizing as a popular class of optimization techniques, the EC methods have demonstrated competitive performance in effectively tackling optimization problems in an easy way. So far, the EC methods have been widely applied in numerous research fields thanks to their strong abilities in finding the optimal solutions [141]. Among the EC algorithms, some algorithms which are based on biological behaviors (e.g., the genetic algorithm (GA), the ant colony optimization (ACO) algorithm and the PSO algorithm) have been well-adopted in a number of research areas, e.g., energy systems, robotics, aerospace engineering and artificial intelligence.

As a population-based EC method, PSO is developed on the basis of the mimics of social behaviors, e.g., the birds-flocking phenomenon and the fish-schooling phenomenon. Notably, the potential optimization solution is represented by a particle (also called an individual). During the searching process, each individual learns from the "movement experience" of itself and others. It should be mentioned that the advantages of PSO can be summarized into three aspects: 1) the number of parameters required to be adjusted is relatively few; 2) the convergence rate of the PSO algorithm is relatively fast; and 3) the implementation of the PSO algorithm is simple [42]. Owing to its technical merits and easy implementation, PSO has become a widely-used technique for tackling optimization problems in recent years [11, 17]. In this section, we aim to deliver a comprehensive and timely review of the PSO algorithms and their applications.

2.3.1 The PSO Algorithms

The Original PSO Algorithm

Original PSO is a prominent EC algorithm, which aims to discover the global optimum in the problem space by tuning the velocity and position of the particles [45, 95]. Basically, each element of the swarm is an individual particle serving as a candidate solution. The movement of each particle is guided by 1) its previous "flying experience" which is the personal best location (i.e., pbest); and 2) the "group flying experience" which is the global best location (i.e., gbest) detected by the entire swarm.

All the individuals search the problem space that has D dimensions to seek the optimal solution. The velocity and position of the ith particle at the kth iteration are denoted by $v_{i,k}$ and $x_{i,k}$, respectively. At the beginning, $v_{i,k}$ and $x_{i,k}$ are randomly initialized. During the evolution process, the velocity and position updating equations of the ith particle are given as follows:

$$\begin{aligned} v_{i,k+1} &= v_{i,k} + c_1 r_1 \left(pi_{i,k} - x_{i,k}\right) \\ &\quad + c_2 r_2 \left(pg_k - x_{i,k}\right) \\ x_{i,k+1} &= x_{i,k} + v_{i,k+1} \end{aligned} \tag{2.4}$$

where k denotes the iteration number; c_1 is an acceleration factor named the cognitive parameter; c_2 is the social acceleration parameter; r_1 and r_2 are two separate random numbers

selected within $[0, 1]$; $pi_{i,k}$ represents the personal best location found by the ith particle itself, which is denoted by $pi_{i,k} = [pi_{i,k}^1, pi_{i,k}^2, \cdots, pi_{i,k}^D]$; pg_k represents the global best location of all the particles, which is denoted by $pg_k = [pg_k^1, pg_k^2, \cdots, pg_k^D]$. c_1 and c_2 indicate the degree of each particle affected by itself and other particles, respectively [93]. The acceleration coefficients play an important role in balancing the local discovery and global search performance, and also have a significant influence on the population diversity, solution quality, and convergence behavior of the algorithm.

The Basic PSO Algorithm

Original PSO has shown strong abilities in solving optimization problems. Nevertheless, the particles may be easily trapped in the local optimal solutions. To guarantee the search performance and balance the global detection and local discovery, the inertia weight w has been introduced in [157] as an important factor to improve the PSO algorithm, and such a factor indicates the ability of particles to inherit their previous velocities. The inertia weight embedded PSO algorithm has been recognized as the basic PSO algorithm. The velocity as well as the position of the ith particle at the $(k + 1)$th iteration are expressed as follows:

$$\begin{aligned} v_{i,k+1} &= w v_{i,k} + c_1 r_1 \left(pi_{i,k} - x_{i,k}\right) \\ &\quad + c_2 r_2 \left(pg_k - x_{i,k}\right) \\ x_{i,k+1} &= x_{i,k} + v_{i,k+1} \end{aligned} \tag{2.5}$$

where w indicates the inertia weight. The basic PSO process is presented in Algorithm 1.

Algorithm 1 The procedure of the basic PSO algorithm

1: **Initialize** the parameters of the PSO algorithm, including the population size P, inertia weight w, acceleration coefficients c_1 and c_2, and maximum velocity $V_{\max}$.
2: Set a swarm with P particles.
3: Initialize the position $x_{i,1}$, the velocity $v_{i,1}$, and the personal best $p_{i,1}$ of each particle $(i = 1, 2, \ldots, P)$; initialize the global best g_1 of the swarm.
4: Calculate the fitness value of each particle.
5: Update the personal best $p_{i,k}$ of each particle and the global best g_k of the swarm.
6: Update the velocity $v_{i,k}$ and the position $x_{i,k}$ of each particle according to Eq. (2.5).
7: Check whether the maximum number of iterations is reached or the fitness value meets the threshold; if not, go back to Step 4.

2.3.2 Developments of the PSO Algorithm

Similar to most population-based EC algorithms, the PSO algorithm also faces the premature convergence problem. In this case, it is of practical importance to put forward new PSO algorithms, especially for solving large-scale optimization problems and multi-modal optimization problems. In this paper, the reviewed PSO variants can be categorized into four groups: 1) adjusting the control parameters; 2) developing new updating strategies; 3) designing various topological structures; and 4) combining with other EC algorithms. In this section, the aforementioned four types of PSO variants are reviewed and summarized.

Adjusting Control Parameters

In the PSO algorithm, the control parameters refer to the inertia weight and the acceleration factors, which are of practical significance in maintaining the balance between global discovery and local detection. In the past few decades, plenty of work has been conducted to adjust these control parameters for improving PSO. Some selected PSO variants with modified control parameters are reviewed and summarized in Fig. 2.11.

1. Inertia Weight As an important parameter, the inertia weight is designed to achieve a proper balance between global discovery and local detection. A brief summarization is presented in Table 2.4 on the recently developed inertia-weight-based PSO variants.

Table 2.4 A summary of reviewed inertia weight updating strategies

Approach	Abbreviated Name	Reference	Approach	Abbreviated Name	Reference
Linear Decreasing	Li-DIW	[156, 158]	Randomizing	RIW	[159]
	MIW-LD	[209]	Optimization-based	SAIW	[5]
	CLi-DIW	[135]	Fuzzy Theory	FAPSO	[160]
Non-linear Decreasing	NLFDIW	[19]	Logistic Map-based	CDIW	[52]
Sigmoid Function-based	SIIW	[120]		CRIW	[52]
Logarithmic Decreasing	Lo-DIW	[58]			

It is known that a smaller inertia weight could lead to a better local search, while a larger inertia weight could contribute to a better global discovery [158]. The particles with satisfactory search ability would thoroughly exploit the solution space at the early step of search and avoid trapping into the local optima with high possibility. Additionally, the inertia weight would greatly affect the search ability of the particles. In [156, 158], starting from

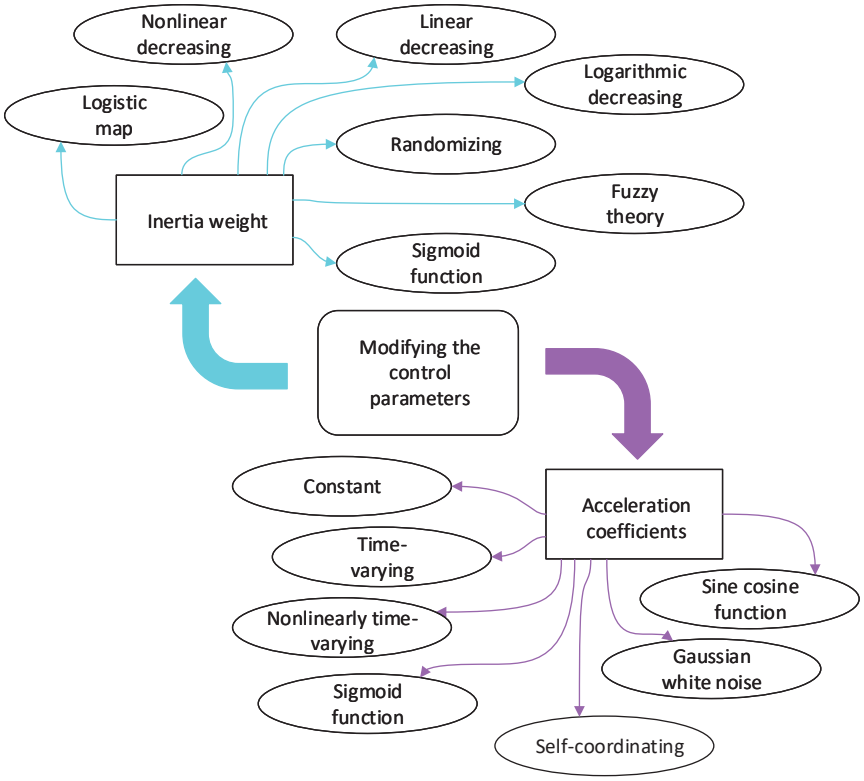


Fig. 2.11 PSO variants with modified control parameters

a relatively large value, the inertia weight is deployed to guarantee the global exploration performance. Then, the inertia weight is adjusted following the linear-decreasing strategy during the searching process with the hope to enhance the local exploration. PSO with the linear decreasing inertia weight (Li-DIW) strategy has been proposed in [156, 158] where the inertia weight (w_k) is updated as follows:

$$w_k = w_{\max} - (w_{\max} - w_{\min}) \times \frac{k}{K} \tag{2.6}$$

where $w_{\max}$ and $w_{\min}$ denote the maximal and minimal inertia weight, respectively; k is the number of the current iteration; and K denotes the number of the maximum iteration during the evolution process. In [14], $w_{\max}$ and $w_{\min}$ are set to be 0.9 and 0.4, respectively, which becomes a normal setting in later development of PSO. The Li-DIW strategy has been widely used in developing various PSO algorithms.

It should be mentioned that a number of inertia weight updating strategies have been introduced based on the Li-DIW strategy. For example, a multi-stage inertia weight linearly decreasing strategy (MIW-LD) has been developed in [209] to better balance the global discovery and the local detection than the PSO with Li-DIW. Different from the Li-DIW strategy, another inertia weight updating strategy is the nonlinear decreasing strategy. A nonlinear function modulated inertia weight (NLFDIW) has been developed in [19]. Another nonlinear function, the sigmoid function has been adopted in [120] to control the inertia weight, where a sigmoid-based increasing inertia weight (SIIW) strategy is put forward to improve the convergence speed. A logarithm decreasing inertia weight (Lo-DIW) strategy has been introduced in [58] to improve the convergence rate.

The global search ability becomes weak when the inertia weight decreases, and the particles may fall into the local optima [42]. In the past few decades, improving the search ability of PSO has been a hot research topic by changing the inertia weight in various aspects. It is found that the PSO algorithm with Li-DIW cannot obtain satisfactory results when dealing with a nonlinear dynamic system. In this case, the random inertia weight (RIW) scheme has been proposed in [159] for tracking and optimizing a dynamic system.

Many inertia weight updating strategies have been designed by employing some mathematical methods (e.g., logistic map, chaotic-embedded methods, fuzzy theory, and optimization) [5, 52, 135, 160]. For example, in [5], the simulated annealing (SA) integrated inertia weight (SAIW) has been introduced to enhance the convergence speed. In [135], the chaotic-embedded linearly decreasing inertia weight (CLi-DIW) strategy has been proposed, which embeds the chaotic sequences into the Li-DIW strategy. Using the logistic map, the chaotic-descending-based inertia weight (CDIW) strategy and the chaotic-embedded random inertia weight (CRIW) strategy have been proposed in [52] to improve the convergence rate, the convergence accuracy and the global discovery ability. A fuzzy system-based PSO algorithm (FAPSO) has been designed in [160], where the inputs of the fuzzy-based method are the normalized current best performance evaluation as well as the current inertia weight, and the output variable of the fuzzy system is the change of the inertia weight.

2. Acceleration Coefficients In recent years, many PSO variants that focus on the modification of the acceleration coefficients have been introduced. The reviewed acceleration coefficient updating strategies are summarized in Table 2.5.

Table 2.5 A summary of reviewed acceleration coefficient updating strategies

Approach	Abbreviated Name	Reference	Approach	Abbreviated Name	Reference
Constant	Constant AC	[35, 180]	Sigmoid Function Based	SBAC	[179]
Time-varying	TVAC-1	[147]		AWAC	[111]
	TVAC-2	[89]	Sine Cosine Function Based	SCAC	[22]
	ATVAC	[15]	Adding Gaussian White Noises	GWNAC	[112]
Non-linear Time-varying	NDAC	[21]	Self-coordinating	SAC	[214]
	NTVAC	[99]			

The effect of two acceleration coefficients on each particle's movement is illustrated in Fig. 2.12. According to [95], a relatively larger cognitive component could make particles search in a wider space comparing with the social component. In the original PSO algorithm, the acceleration components are constant values that are set to be 2. A number of new PSO methods have been put forward, where different acceleration factor updating strategies have been adopted for specific optimization problems. In [35], the acceleration coefficients are set to be 2.05 to guarantee the convergence of the optimizer. In [180], c_1 and c_2 are set to be 0.5 and 2.5, respectively.

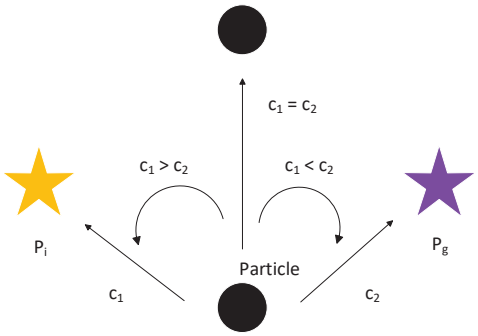


Fig. 2.12 The effect of the value of the acceleration coefficients on the movement of the particle

It is worth noting that a relatively larger social component may easily lead to the premature convergence problem. The time-varying acceleration coefficient-embedded scheme (TVAC-1)

has been proposed in [147], where the updating equations of the acceleration factors ($c_{1,k}$ and $c_{2,k}$) are expressed as follows:

$$\begin{aligned} c_{1,k} &= c_{1i} + (c_{1f} - c_{1i}) \times \frac{k}{K} \\ c_{2,k} &= c_{2i} + (c_{2f} - c_{2i}) \times \frac{k}{K} \end{aligned} \tag{2.7}$$

where c_{1i} and c_{1f} indicate the initial and final values of the cognitive component, respectively; c_{2i} and c_{2f} are the initial and final values of the social component, respectively; k is the current iteration number; and K is the maximum iteration number. The time-varying scheme for altering acceleration factors could not only improve the global discovery at the early step of the searching process, but also improve the convergence of particles towards the globally optimal solution at the latter step of the evolution process. According to the experimental results reported in [147], the parameters are set to be $c_{1i} = 2.5$, $c_{1f} = 0.5$, $c_{2i} = 0.5$, and $c_{2f} = 2.5$. Based on the TVAC-1, a new updating strategy has been introduced to adjust the time-varying acceleration coefficients with unsymmetrical transfer range (UTRAC) in [68], which improves the convergence speed of the PSO algorithm. The asymmetric time-varying-based strategy for controlling acceleration coefficients (ATVAC) has been proposed in [15], where the ATVAC demonstrates merits in balancing global and local search and improving the convergence as well as the robustness.

In [89], another time-varying acceleration coefficients (TVAC-2) updating strategy has been proposed, which could further improve the solution accuracy. The updating equations of the acceleration coefficients ($c_{1,k+1}$ and $c_{2,k+1}$) using the TVAC-2 strategy are given by:

$$\begin{aligned} c_{1,k+1} &= c_{1,k} - \frac{0.5}{K} \\ c_{2,k+1} &= c_{2,k} + \frac{0.5}{K} \end{aligned} \tag{2.8}$$

where k and K represent the current and the maximum iteration number, respectively.

Apart from two popular strategies TVAC-1 and TVAC-2, some acceleration coefficients updating strategies have been introduced by incorporating time-varying mechanisms and nonlinear dynamics [21, 22, 99, 111, 179]. In [99], a set of non-linear time-varying acceleration coefficients (NTVAC) has been put forward to alleviate premature convergence.

The nonlinear dynamic mechanism has been introduced in [21] to alter the acceleration factors. As a popular nonlinear function, the sigmoid function has been used to adjust acceleration coefficients in [179], where the sigmoid-function-based acceleration coefficients (SBAC) updating strategy has been proposed. In [111], a sigmoid-function-based adaptive weighted acceleration coefficients (AWAC) strategy has been developed, where an adaptive weighting updating (AWU) function has been proposed by exploiting the distance from the particle to its pbest and gbest in order to adjust the acceleration coefficients. The sine cosine acceleration coefficients (SCAC) updating strategy has been proposed in [22], which could not only motivate a thorough local detection but also force the candidates to move to the global optimal solution.

Some other acceleration coefficients updating strategies have also been developed [112, 214]. For example, the Gaussian white noise-embedded acceleration coefficients (GWNAC) have been introduced in [112] to maintain the population diversity and enhance the possibility of particles slipping away from the local optima by randomly perturbing the acceleration factors. In [214], a novel adaptive method has been proposed, where the acceleration factors are modified adaptively to make the target particle self-coordinates.

Developing Updating Strategies

Apart from adjusting control parameters of the PSO algorithm, many researchers have focused on designing new updating strategies of the PSO algorithm in the past few decades. The reviewed approaches on developing new algorithm updating strategies are summarized in Table 2.6.

Table 2.6 Summarization of novel algorithm updating strategies

Approach	Abbreviated Name	Reference	Approach	Abbreviated Name	Reference
Re-initialization	RPSO	[66]	Switching Strategy	APSO	[228]
	RRPSO	[29]		SPSO	[177]
	ERPSO	[29]		ISPSO	[224]
Clustering Algorithm	CAPSO	[94]		SDPSO	[226]
Constriction Factor	CFPSO	[34, 35]		MDPSO	[164]
Comprehensive Learning	CLPSO	[105]		ARFPSO	[166]
Multi-elitist Strategy	MEPSO	[41]		DNSPSO	[225]
Fractional Velocity	FVPSO	[140]		RODDPSO	[109]
FPGA Based	PMPSO	[78]		FVSPSO	[165]
Cooperative	CPSO	[184]		SASPSO	[12]

1. Re-initialization Re-initialization is a strategy that helps alleviate premature convergence. In [66], an improved PSO algorithm with re-initialization mechanisms (RPSO) has been introduced, where the re-initialization process is determined based on the estimation of the varieties and activities of the particles. A new factor named "steplength" is employed to determine whether the particle should be re-initialized or not. In [29], two re-initialization methods have been proposed (which are the random re-initialization strategy and the elitist re-initialization strategy) to promote the PSO diversity. The first method is random re-initialization (RRPSO), where particles are reserved randomly, which could improve the ability of exploration. The second method is elitist re-initialization (ERPSO), where the worse preferred particles in the searching area are re-initialized, which could make the particle obtain a better fitness value than standard PSO.

2. Switching Strategy So far, a new class of switching strategies has been designed for improving the PSO algorithm, where the evolution process is divided into several evolutionary states. In [228], an adaptive PSO (APSO) algorithm has been proposed, where an evolutionary factor (EF) is introduced to determine the exploration, exploitation, convergence, and jumping out states. The evolutionary state estimation technique and the elitist learning strategy have been proposed in the APSO. According to the evolutionary state estimation, the evolution process can be divided into four states based on the evolutionary factor.

In [177], a switching-motivated PSO algorithm (SPSO) has been developed. Based on the evolutionary factor, the velocity updating process could jump from one state to another by using a Markov chain, and the acceleration factors are thus updated based on the evolutionary state. Furthermore, a leader competitive penalized multi-learning approach is introduced in order to help the globally best particle slip away from the local optimal areas and accelerate the convergence speed. A dynamic-neighborhood-based SPSO (DNSPSO) algorithm has been developed in [225], where the evolution information of the swarm is used in the velocity updating process based on 1) a distance-based dynamic neighborhood; and 2) the switching strategy. The DNSPSO algorithm improves the solution accuracy and enhances the particle's ability to slip away from the local optima compared with the SPSO algorithm. Based on the SPSO algorithm, an improved SPSO (ISPSO) algorithm has been introduced in [224], where a non-stationary multistage assignment penalty function is introduced. The updating strategy

of velocity jumps from each mode based on the non-homogeneous Markov chain, which uses the swarm diversity as the current search information to adjust the probability matrix in order to balance the global and local search. In [226], a switching time-delay-embedded PSO (SDPSO) algorithm has been introduced, where the time delay is employed to alter the system dynamics of the SPSO algorithm. The utilized time delays contain the historical information of the evolutionary process. The multi-modal delayed PSO (MDPSO) method has been developed in [164], where the so-called multi-modal time-delay is embedded into the velocity model to enlarge the search space and reduce the probability of being trapped in the local optima. A novel randomly-occurring-distributed-time-delay PSO (RODDPSO) algorithm has been proposed in [109], where the distributed time-delays are employed to perturb the dynamic behavior of the particles. The delays occur randomly, which contributes to a better search ability than the SDPSO. A modified switching PSO algorithm with adaptive random fluctuations (ARFPSO) has been introduced in [166], where the velocity is updated based on the evolutionary states, and the adaptive random fluctuations are added to the pbest and the gbest particles. According to the simulation results reported in [166], the ARFPSO algorithm shows superior performance in terms of search the optimal solution. In [140], a PSO algorithm with fractional velocity (FVPSO) has been developed, where the fractional-order velocity terms are added to the velocity updating equation. The FVPSO algorithm enhances the particle's ability of jumping out of the local optima. Based on the FVPSO algorithm, an adaptive fractional-order velocity SPSO (FVSPSO) algorithm has been presented in [165], where the fractional velocity is updated based on the evolutionary state. In [12], an adaptive switching asynchronous-synchronous PSO (SASPSO) algorithm has been proposed, where the asynchronous updating strategy and the synchronous updating strategy are hybridized, which could switch from one to another according to the fitness value of the gbest.

3. Other Strategies In [94], the clustering algorithm has been utilized in the PSO (CAPSO) algorithm to group the particles based on their previous best positions, and the cluster centroids are replaced by particles' personal best positions and neighbors' best positions. According to the experimental results, it is found that the CAPSO algorithm (using personal best positions as the cluster centroids) performs better than algorithms using neighbors' best

positions as cluster centroids. The constriction factor and its variants have been put forward in [34, 35, 46], which improves the convergence rate of the optimizer by limiting the motion of individuals in the optimal region. In [184], two cooperative PSO (CPSO) algorithms have been proposed (i.e., CPSO-S_K and CPSO-H_K), where the cooperative behaviors are utilized to improve the solution accuracy. In [105], a comprehensive learning (CL) strategy has been proposed, where the particle's velocity is updated according to the personal best locations of all other particles. The CLPSO algorithm demonstrates better performance in solving multi-modal problems by comparing with the standard PSO algorithm. In [41], a multi-elitist PSO algorithm (MEPSO) has been introduced, where the multi-elitist strategy has been employed in the PSO algorithm to improve the global search, which improves the search ability of the PSO algorithm. In [78], a field-programmable gate array (FPGA)-based parallel meta-heuristic PSO (PMPSO) algorithm has been proposed, which employs the parallel computing strategy to run three parallel PSO algorithms in the same FPGA chip. According to the experimental results, the PMPSO algorithm shows merit in solving some global path planning problems. In [171], the quantum PSO (QPSO) algorithm has been proposed, where the quantum theory is introduced into the PSO algorithm. A trial method for adjusting parameters has also been proposed. A new parameter tuning method has been put forward in [172] to adjust the control parameters of the QPSO algorithm, where a global reference point has been introduced to evaluate the search range of the particle. The QPSO algorithm with the new parameter selection strategy has shown better performance than the standard QPSO algorithm in terms of convergence and solution accuracy.

2.3.3 Applications of the PSO Algorithm

In this part, some selected practical applications of the PSO algorithm are reviewed, which are divided into 6 categories, including robotics, the renewable energy system, the power system, data analytics, image processing, and some other applications.

Robotics

1. Path Planning for Robots Autonomous navigation of the mobile robot is a crucial task in robotics. The autonomous navigation process is illustrated in Fig. 2.13. As an important

task in autonomous navigation, path planning has been widely investigated, which is known as an optimization problem in certain indices with certain constraints [146]. So far, a number of PSO algorithms have been adopted to solve the robot's path planning problems.

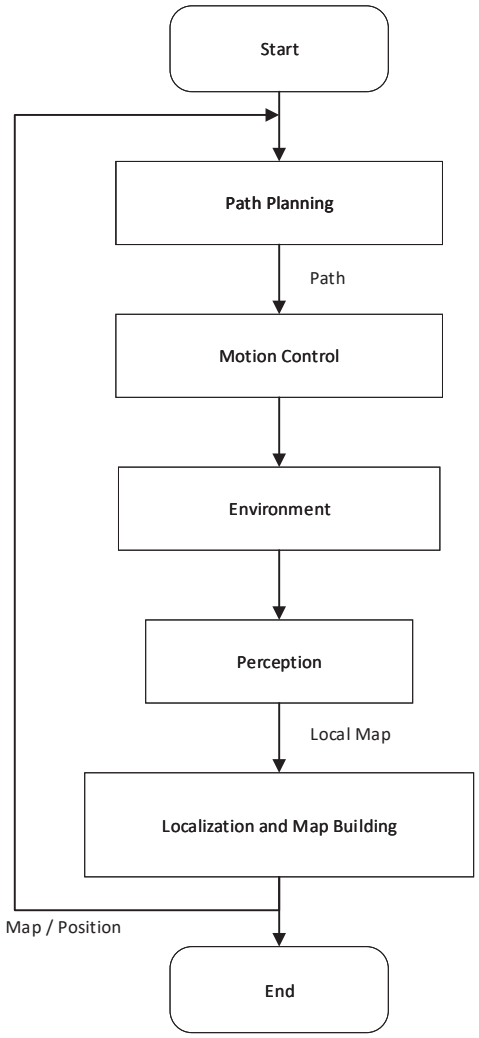


Fig. 2.13 The autonomous navigation process of the mobile robot

In [78], the PMPSO algorithm has been applied to deal with the navigation of the autonomous robot in structured environments with obstacles. In [227], the switching-local-evolutionary-based PSO (SLEPSO) algorithm has been used to solve the path planning problem of the intelligent robot. In [164], the MDPSO algorithm has been successfully applied to the path planning for mobile robots. In [166], a switching PSO algorithm with

adaptive random fluctuations has been combined with the η^3-splines to solve a double-layer smooth global path planning problem with several kinematic constraints. In [165], the FVSPSO algorithm has been applied for smooth path planning for the mobile robots based on the continuous high-degree Bezier curve.

2. Robot Learning The purpose of robot learning is to teach robots to learn skills and adapt to the environment under the guidance of learning algorithms. As an efficient class of EC algorithms, the PSO-based robot learning algorithms have been employed in the robot learning field. In [142], a PSO-based robotic learning method has been proposed, where a noise reduction technique used in the GA has been applied to the PSO algorithm to improve the performance of robot learning. In [143], an adapted PSO algorithm has been applied for unsupervised robotic learning, which shows competitiveness over some existing ones. In [43], an improved PSO-based method has been utilized in robot learning, where a statistical technique, named the optimal computing budget allocation, is utilized to improve the performance of the PSO algorithm with noise.

Renewable Energy System

In recent years, the utilization of alternative energy has increased significantly in order to fulfill the requirements of environmental protection and electricity demands. Fig. 2.14 shows the structure of the basic alternative energy system. In this case, the optimal design plans of the energy system have been made to utilize the energy resources efficiently and economically. In recent years, a number of PSO-based approaches have been put forward for solving the optimization problems in energy systems [96].

1. Sizing The sizing problem is an important optimization problem in energy systems, which aims to find the optimum number and types of devices to be used [96]. The optimum unit size can ensure the system works in the best conditions with the lowest cost, which could keep the system efficient and economic. In recent years, many PSO-based methods have been developed to solve the sizing problem in energy systems. In [183], the GA and the PSO algorithms have been used to optimize the unit size of the stand-alone hybrid energy system. A PSO-based approach has been proposed in [3] to solve the sizing problem of a hybrid solar

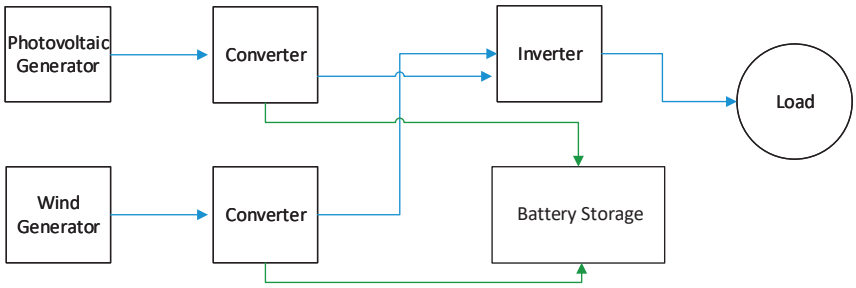


Fig. 2.14 A basic alternative energy system with battery storage

photovoltaic energy system. In [16], the PSO algorithm has been applied to a new hybrid renewable energy system. In [124], the PSO algorithm has been applied to the sizing problem of the distributed energy system in micro grid.

2. Maximum Power Point Tracking The maximum power point tracking (MPPT) on the solar photovoltaic module of the energy system is also a challenging optimization task, which aims to help the solar photovoltaic module produce maximum power output [96]. In recent years, a number of MPPT approaches have been developed. Among the proposed approaches, the PSO-based methods have demonstrated better noise immunity, which has become a popular way to track the maximum power point. In [11], a tracking method has been proposed where the PSO algorithm has been used to recognize the maximum power point in the photovoltaic system with high efficiency and fast convergence rate. In [83], a PSO-based MPPT controller of the photovoltaic system has been introduced by using the direct control method. In [82], an MPPT method with an improved PSO algorithm has been presented, where the steady-state oscillations are decreased when the maximum power point is found, which could reduce the tracking time. In [122], a new maximum power point tracking approach with the PSO algorithm has been proposed, where the multiple photovoltaic arrays are controlled by only one pair of sensors. Results show that the developed method reduces the cost and improves the efficiency of the system.

3. Others In [236], a PSO algorithm with the constriction factor and the mutation operator has been applied to solve the capacity configuration problem. The PSO algorithm has been

used in [169] to identify the parameters of the photovoltaic model. In [7], a parameter optimizing method of the current control strategy for a 3-phase photovoltaic grid-connected voltage source inverter system has been proposed, where the PSO algorithm is used to tune the current control parameters. Results showed that the optimized system improves the power quality.

Power System

1. Economic Dispatch The economic dispatch problem has become a popular research topic in the power system, which plays a crucial role in the operation and planning of the power system. The economic dispatch problem aims to adjust the output of generating units in order to meet the load requirement with the lowest cost and also satisfy the operational constraints of units as well as the system. In fact, the economic dispatch problem can be treated as an optimization problem, and a large number of optimization techniques have been adopted to solve the economic dispatch problem. In [25], the PSO-RDL algorithm has been applied to the economic dispatch problem for the 3-unit power system and has discovered the currently best solution for the 40-unit system. The PSO algorithm has been hybridized with the sequential quadratic programming method to deal with the economic dispatch problem with the valve-point effects [187]. In [36], the PSO algorithm has been combined with the Gaussian probability distribution functions to solve the economic cost dispatch problem. In [134], an improved PSO algorithm has been employed to the economic load dispatch problem, where the inertia weight of the PSO algorithm is adaptively adjusted according to each particle's rank.

2. Others The unit commitment is an efficient way to provide high-quality electric power to customers securely and economically. The unit commitment problem aims to find the optimal unit commitment, which is an optimization problem. In [152], the FAPSO-based method has been developed to compute the unit commitment of the power system.

Data Analytics

1. Clustering Clustering algorithms organize a collection of similar items into the same cluster. An illustration of clustering is in Fig. 2.15, where each point represents an instance

in the data set. It is known that clustering plays a vital role in data analysis. Unfortunately, some studies have shown that the clustering performance of the distance-based clustering algorithms is highly dependent on the initial value of the clustering centroids and may be unsatisfactory when dealing with some complex data sets [41]. In the past few years, some researchers have employed optimization algorithms to choose an optimized location of the initial cluster centroids to improve the performance of distance-based clustering algorithms.

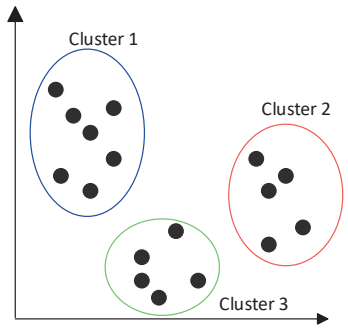


Fig. 2.15 An illustration of a cluster analysis output

In [91], a new K-means-based clustering method has been proposed, where the basic K-means clustering approach is combined with the Nelder-Mead simplex search approach and the standard PSO approach. In [127], a hybrid optimization algorithm which combines the FAPSO algorithm with the ACO algorithm (FAPSO-ACO) has been embedded to improve the basic K-means approach. The RODDPSO algorithm presented in [109] has been utilized for improving the performance of the basic K-means approach. In addition, PSO has also been utilized for optimizing the FCM algorithm. In [162], two improved FCM algorithms based on the PSO algorithm have been developed, both of which demonstrate fast speed and high solution quality. A novel clustering algorithm has been introduced in [153], where the FCM algorithm is combined with the QPSO algorithm. The performance of the new method outperforms some other clustering algorithms [171].

Apart from the K-means algorithm and the FCM algorithm, some other PSO-based clustering algorithms have also been developed. For example, in [208], PSO has been applied to the self-organizing maps, where a conscience factor is added to the self-organizing maps. The weights of the self-organizing maps are tuned by PSO, which could improve

the robustness of the clustering algorithm. The MEPSO-based automatic kernel clustering algorithm has been introduced in [41], where the MEPSO algorithm is used to find the optimal number of clusters, and a kernel function is applied to cluster data in a high-dimensional transformed space.

2. Feature Selection As an important member in data mining, classification aims to classify data into different categories based on the features of the data. Nevertheless, selecting useful features in the data set is a difficult task. Some PSO-based approaches have been introduced with the purpose of dealing with the feature selection problems. A self-adaptive PSO algorithm has been introduced and applied in feature selection [213]. In [212], two PSO-based multi-objective feature selection methods have been proposed, where the non-dominated sorting idea is employed in the first method, and the crowding, mutation, as well as dominance are applied to the second method. Both methods could automatically discover a set of non-dominated solutions.

Image Processing

Image processing is an important field in computer science, which includes image segmentation, image enhancement and so on. In recent years, a number of PSO-based algorithms have been proposed to improve the performance of image processing [2, 61, 65].

1. Image Segmentation Image segmentation focuses on analyzing and interpreting images. Many image segmentation methods have been developed using PSO algorithms to improve the performance of image segmentation. In [61], a multilevel thresholding method has been proposed for image segmentation, where a fractional-order Darwinian PSO algorithm is proposed to improve the accuracy of image segmentation. In [2], PSO has been combined with the hidden Markov random field model to improve the quality of the image segmentation.

2. Image Enhancement Image enhancement is another important task in image processing. The basic PSO approach has been widely adopted for image enhancement. A PSO-based automatic image enhancement technique has been proposed in [17], where PSO has been utilized to maximize an objective fitness criterion with the purpose of enhancing the contrast

and details in the picture. Results show that the PSO-based automatic image enhancement technique obtains satisfactory performance in maximizing the number of pixels in the edges. A PSO-based image enhancement method has been introduced in [65], where a parameterized transformation function and an objective criterion have been utilized to improve the performance of the image enhancement.

Others

A large number of PSO-based methods have been developed for various real-world problems. For example, the PSO algorithm has been used in [106] for parameter identification of the permanent magnet synchronous motors. In [56], the PSO algorithm has been applied in the control theory to find the optimal parameters of the proportional-integral-derivative controller in an automatic voltage regulator system, which provides fast parameter adjusting ability and is easy to implement. In [13], a FAPSO algorithm-based optimal bidding strategy of a thermal generator in the uniform price spot market has been developed to find the optimal bidding strategy in the power market, where the operating cost function and the minimum up/down constraints are considered.

PSO has been successfully employed to solve electromagnetic-related optimization problems. In [32], an intelligent PSO algorithm has been utilized for optimizing the electromagnetic device, where the complexity of the particle is enlarged at the group level to improve the efficiency of the optimization process.

It should be pointed out that using the PSO algorithm to design the trajectory of the spacecraft has been a popular research topic. In [28], a recovery trajectory planning method has been introduced for the recovery process of the reusable launch vehicle, which combines the PSO algorithm with the polynomial guidance law.

2.4 Application Backgrounds

2.4.1 Outlier Detection in WAAM

AM, also known as 3D printing, has been recognized as a breakthrough technology which shows great application potentials in industrial machinery, assembly processes, and supply

chains [57]. An AM system consists of three parts: a motion system, heat source and feedstock. Due to the advantages of high deposition rate, high material utilization, low cost and environmental friendliness, the WAAM has become a popular AM method, which uses the electric arc as the heat source and the wire as the feedstock. Depending on the heat source, the heat and metal transfer in WAAM can be generally divided into three categories of equipment including the Gas Tungsten Arc Welding, the Gas Metal Arc Welding, and the Plasma Arc Welding [149].

In fact, there are several key factors affecting the performance of WAAM, e.g., the programming strategies, manipulator- and feedstock feeding accuracy, shielding gas flow, metal surface contaminations, heat- and metal transfer, heat accumulation, and part distortion [198]. In recent years, a variety of advanced techniques (which focuses on the aforementioned factors) have been put forward to improve the performance of the WAAM. For instance, in [131], an improved heat transfer and fluid flow model has been introduced in WAAM, which could determine the parameters that influence the microstructure, properties, and defect formation of the component. In [123], a WAAM modelling strategy has been developed based on a novel heat source model, which has shown high accuracy in measuring distortions.

Note that the quality of AM products is highly dependent on the welding equipment used. To monitor the WAAM process, outlier detection techniques are widely adopted to analyze the sensor data (e.g., current and voltage) of the welding machine used with the purpose of detecting abnormal points. In general, the abnormal points indicate the sudden change of the process, which could bring negative influence on the manufacturing process. In this case, it is of practical importance to use outlier detection techniques for online monitoring of the WAAM process.

Over the past few years, outlier detection has attracted increasing attention in the research community due to its critical role in various real-world applications. So far, a large number of outlier detection methods have been introduced for data analysis in WAAM [30, 121, 148, 168]. For instance, a semi-supervised method has been proposed in [121] to enable real-time anomaly detection in WAAM using current and voltage data, further enhancing the reliability and stability of the manufacturing process. A modular anomaly detector has been put forward in [148] for analyzing multivariate time-series data in the WAAM process. In [30], a convolutional neural network-based method has been proposed for real-time anomaly

detection in WAAM. Furthermore, in [168], a two-stage unsupervised framework has been presented for anomaly detection in WAAM.

2.4.2 Data Description

The utilized data is collected from a WAAM pilot line deployed in Sweden, which aims at developing an end-to-end digital solution by integrating automation methodologies for metal component manufacturing. In total, there are five data sets, each representing a time series corresponding to an individual manufacturing process. Each time series consists of 98,000 instances, where each instance contains measurements of welding current and voltage recorded at a sampling interval of 0.0002 seconds during the manufacturing process. Fig. 2.16 presents a visualization of the WAAM data. In the collected data sets, there are four variables which are "X_Value", "WeldCurrent", "WeldVoltage", and "ComputerTime". "X_Value" represents the time stamp. "WeldCurrent" and "WeldVoltage" denote the processing current and voltage, respectively. "Computer time" is the total time of the process. The details of the variables are summarized in Table 2.7. Ground-truth labels are manually annotated by domain experts, classifying each sample as either "Normal" or "Outlier" based on operational characteristics.

Table 2.7 Data description

Variable	Description	Data Type	Unit
X_Value	The time stamp	Numerical	*s*
WeldCurrent	The current of the equipment	Numerical	*A*
WeldVoltage	The voltage of the equipment	Numerical	*V*
ComputerTime	The total time of the process	Numerical	*s*

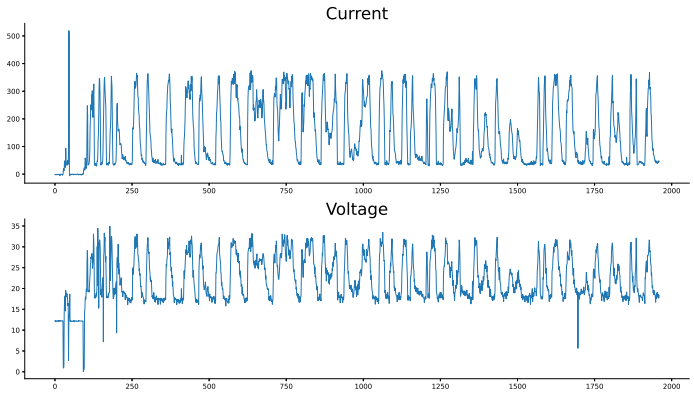


Fig. 2.16 Visualizations of the current and voltage of the WAAM data

Chapter 3

Outlier Detection under Unlabeled Data: An Improved Particle-Swarm-Optimization-Based Clustering Framework

3.1 Motivation

WAAM has become an important and rapidly developing technology in industrial manufacturing, which has attracted substantial research interest in recent years. To improve the overall performance and fabrication quality of WAAM, extensive efforts have been devoted to analyzing process behaviors and refining operating strategies [145]. In WAAM, the annotation process is usually time-consuming and labor-intensive due to the large volume of collected data and the requirement for expert knowledge. As a result, unlabeled data are widespread in practical WAAM applications. The prevalence of unlabeled data significantly limits the applicability of supervised learning methods and poses challenges for effective analysis, which reduces the reliability of data-driven monitoring and control in WAAM.

As a powerful family of outlier detection methods for unlabeled data, the clustering-based outlier detection methods are utilized to identify abnormal instances according to the corresponding clustering results. Compared with existing clustering algorithms including

the density-based spatial clustering of applications with noise (DBSCAN) algorithm and the K-means algorithm, the FCM algorithm has the advantages of easy implementation and high efficiency, which has been successfully applied to a large number of real-world applications [186]. Nevertheless, as a distance-based clustering algorithm, the clustering performance of the FCM algorithm is highly dependent on the initial locations of the cluster centroids [185]. Selecting an optimal set of initial cluster centroids seems to be an effective way to guarantee the performance of the FCM algorithm.

It is worth mentioning that EC has been widely used to solve various optimization problems. Among the EC algorithms, the PSO algorithm is a population-based one that offers notable advantages. It should be noted that the optimization of cluster centroids is non-convex and highly sensitive to the starting solution. Although gradient-based or local optimization methods could be applied to continuous optimization problems, they usually start from a single or a limited number of initial solutions and may converge to nearby local optima. In contrast, the population-based search mechanism of PSO allows multiple candidate centroid configurations to be explored simultaneously, which can reduce the dependence on a single initial solution and provide a more robust search process for identifying suitable initial centroids. Compared with some powerful black-box optimizers, such as CMA-ES, PSO can provide a clear and traceable optimization process, which is particularly valuable for WAAM outlier detection. As such, adopting the PSO algorithm to optimize the initial locations of the cluster centroids with the purpose of improving the clustering performance of the FCM algorithm becomes a seemingly reasonable idea.

It is worth mentioning that most existing population-based EC algorithms suffer from premature convergence, and this is particularly true when dealing with complex and large-scale optimization problems [105]. During the past few decades, a great many PSO variants have been proposed to improve the convergence rate and the search ability of the optimizer. For instance, the PSO algorithm with a linear decreasing inertia weight (PSO-Li-DIW) has been proposed in [156, 158]. A PSO algorithm with time-varying acceleration coefficients (PSO-TVAC) has been introduced in [147]. In [111], an AWU strategy has been proposed to update the acceleration coefficients to improve the convergence rate of the optimizer.

A switching PSO (SPSO) algorithm has been put forward in [177] by employing a switching strategy to update the velocity of the particles, where the switching strategy

divides the evolutionary process into four evolutionary states (i.e., convergence, exploitation, exploration, and jumping-out). By using the switching strategy, the SPSO algorithm has shown a relatively fast convergence rate. Nevertheless, the deployment of the switching strategy as well as the AWU could not solve the premature convergence problem.

In the literature, some popular PSO variants have been proposed by embedding random perturbations (e.g., time delays and noises) in the PSO algorithm with the hope to alleviate the premature convergence problem [112, 226]. In [226], a switching delayed PSO (SDPSO) algorithm has been developed, where time delays are embedded in the velocity updating equation of the SPSO algorithm. Compared with the SPSO algorithm, the SDPSO algorithm exhibits stronger search ability, especially for multi-modal optimization problems. It should be noticed that the employment of the perturbation has proven to be another effective way to help particles jump out of the local optima [99]. In this situation, it is reasonable to adaptively embed the random perturbation into the PSO algorithm based on the switching strategy to alter the dynamical behavior of the particles and expand the search range of the optimizer.

To summarize, the main objective of this chapter is to propose an adaptive switching randomly perturbed particle swarm optimization (ASRPPSO) algorithm so as to automatically choose an optimal set of initial locations of the cluster centroids of the FCM algorithm. Specifically, a distance-based AWU strategy is designed to adaptively control the acceleration coefficients. The switching strategy is employed to accelerate the searching process and balance the global and local searches. In addition, Gaussian white noises are embedded in the velocity updating equation to randomly alter the system dynamics of the optimizer, which could expand the search space and alleviate the premature convergence problem. The proposed ASRPPSO-based FCM algorithm is applied to detect outliers in real-world data.

The main contributions can be summarized in the following three aspects:

1. A novel ASRPPSO is proposed, where an AWU strategy is designed to adaptively adjust the acceleration coefficients, and the Gaussian white noises are adaptively embedded into the velocity updating equation based on the switching strategy.

2. An ASRPPSO-based FCM algorithm is developed, where the ASRPPSO is employed for selecting the optimal locations of the initial cluster centroids of the FCM algorithm.

3. The developed ASRPPSO-based FCM algorithm is applied to outlier detection of the real-world industrial data collected from a WAAM pilot line. Experimental results demonstrate the effectiveness of the proposed outlier detection method.

The rest of this chapter is organized below. In Section 3.2, the proposed ASRPPSO and the ASRPPSO-based FCM algorithm are introduced. Benchmark functions, test PSO algorithms, parameter setting, experiment results, and discussions are illustrated in Section 3.3. Finally, conclusions and discussions on relevant future directions are presented in Section 3.4.

3.2 Methodology

The FCM algorithm is a competitive clustering algorithm, whose initial location of the cluster centroid is a decisive factor affecting the clustering results of the FCM algorithm. Clearly, it is highly desirable to seek the best locations of the initial centroids for the best clustering performance, which gives rise to a rather challenging optimization problem that has received little research attention so far, except for the preliminary efforts made in [151]. In fact, as one of the powerful optimization algorithms, the PSO algorithm is ideally suited for optimizing the initial locations of the cluster centroids, see [151] for more details.

In the PSO algorithm, the control parameters (i.e., the inertia weight and acceleration coefficients) are utilized to balance the global and local search. In recent years, some variant PSO algorithms (which modify the control parameters) have been proposed to balance the global and local search [147, 156, 158]. The PSO-Li-DIW algorithm and the PSO-TVAC algorithm both demonstrate competitive performance in maintaining the balance between the global and local search compared with the original PSO algorithm.

It should be noticed that the solution accuracy of many PSO variants is improved with the sacrifice of the convergence rate. With the purpose of adequately improving the convergence rate and the capability of finding the global best solution, the PSO-AWAC algorithm has been put forward in [111], where a sigmoid-function-based AWU strategy has been proposed to adjust the acceleration coefficients. Specifically, the AWU strategy makes full use of the distance from each particle towards its pbest and gbest at each step, which significantly

improves the convergence rate of the optimizer. Unfortunately, the search ability of the PSO-AWAC algorithm is not satisfactory when dealing with complex optimization problems.

Recently, some PSO algorithms have been proposed to alleviate premature convergence based on different switching strategies [112, 177, 226]. By using the switching strategy, the evolution process is divided into four evolutionary states (including the exploration, exploitation, convergence, and jumping-out states). The velocity is updated based on different updating strategies at each state, which offers the opportunity of improving the search ability and guaranteeing the convergence rate of the optimizer at the same time. More recently, adding noises to perturb the particle's movement has been proven to be an effective way to improve the search ability of each individual particle. In [112], the intensity-adjustable Gaussian white noise has been added in the velocity updating equation to randomly alter the acceleration constants in order to explore the search space thoroughly. Motivated by the above discussions, it becomes natural to embed the random noises into the PSO algorithm according to the evolutionary states to further alleviate the premature convergence problem and improve the search ability of the optimizer.

In this chapter, a novel ASRPPSO is put forward where an AWU strategy is designed to adjust the acceleration coefficients. In the ASRPPSO, the switching strategy is utilized to improve the particle's search ability, and the Gaussian white noises are embedded in the velocity updating equation based on the switching strategy to help the particles escape from the local optima. In addition, an ASRPPSO-based FCM algorithm is proposed for outlier detection, where the ASRPPSO is adopted to choose an optimal set of initial locations of the cluster centroids in the FCM algorithm.

3.2.1 The ASRPPSO

Framework of the ASRPPSO

The updating equations of the proposed ASRPPSO in terms of velocity and position of the ith particle at the $(k+1)$th iteration are given as follows:

$$\begin{aligned} v_{i,k+1} &= w_k v_{i,k} + c_{1,k} r_1 \left(pbest_{i,k} - x_{i,k}\right) + c_{2,k} r_2 \left(gbest_k - x_{i,k}\right) \\ &\quad + \alpha_{1,\xi_k} \delta_1 \left(pbest_{i,k} - x_{i,k}\right) + \alpha_{2,\xi_k} \delta_2 \left(gbest_k - x_{i,k}\right), \\ x_{i,k+1} &= x_{i,k} + v_{i,k+1}, \end{aligned} \tag{3.1}$$

where k represents the current iteration number; w_k denotes the inertia weight at the kth iteration; $c_{1,k}$ and $c_{2,k}$ are the acceleration coefficients; r_1 and r_2 represent random numbers selected within $[0, 1]$; $pbest_{i,k}$ represents the personal best position found by the ith particle itself at the kth iteration; $gbest_k$ represents the global best position of the entire swarm at the kth iteration; α_{1,ξ_k} and α_{2,ξ_k} are parameters which are used to adjust the Gaussian white noises according to the evolutionary state; and δ_1 and δ_2 represent two independent Gaussian white noises.

The procedure of the proposed ASRPPSO is presented in Algorithm 2.

Algorithm 2 The Procedure of the ASRPPSO

1: **Initialize** the parameters of ASRPPSO, including the population size P, inertia weight w_1, acceleration coefficients $c_{1,1}$ and $c_{2,1}$, and the maximum velocity $V_{\max}$.
2: Set a swarm with P particles.
3: Initialize the position $x_{i,1}$, the velocity $v_{i,1}$, and the personal best $pbest_{i,1}$ of each particle $(i = 1, 2, \ldots, P)$; initialize the global best $gbest_1$ of the swarm.
4: Calculate each particle's fitness value.
5: Update $pbest_{i,k}$ of each particle and $gbest_k$ of the swarm.
6: Calculate E_f according to Eqs. (3.5) and (3.6), and determine the evolutionary state according to Eq. (3.7).
7: Update w_k, $c_{1,k}$, and $c_{2,k}$ for each particle according to Eqs. (3.2)-(3.4).
8: Update the velocity $v_{i,k}$ and the position $x_{i,k}$ of each particle according to Eq. (3.1).
9: Terminate if the maximum number of iterations is reached or the fitness value meets the threshold; otherwise, repeat Steps 4-8.

Inertia Weight

The inertia weight is an important parameter that is designed to adequately balance the global and local search. It shows the ability of particles to inherit their previous velocities. In order to balance the global search and the local search, the inertia weight is adaptively altered by a linear decreasing inertia weight strategy, which has been introduced in [156, 158]. In the proposed ASRPPSO, the updating equation of the inertia weight is shown as follows:

$$w = w_{\max} - (w_{\max} - w_{\min}) \times \frac{k}{K}, \tag{3.2}$$

where $w_{\max}$ and $w_{\min}$ denote the maximal and minimal value of the inertia weight, respectively; k and K represent the current iteration number and the maximum iteration number, respectively; and the maximum and minimum inertia weights are set to be 0.9 and 0.4, respectively.

Acceleration Coefficients

According to [111], using an AWU function to demonstrate the relationship between the acceleration coefficients and the distances from the particle to its pbest and gbest is a good way to adaptively adjust acceleration coefficients, which can significantly improve the convergence rate. The AWU function needs to be monotonically increasing and bounded. The tanh function is a typical activation function of neural networks, which perfectly fits the requirements mentioned above. In addition, the tanh function is differentiable, which can iteratively reflect the characteristics of the weight updating process. Hence, the tanh function seems to be an appropriate choice. In this proposed ASRPPSO, a tanh-function-based AWU strategy is introduced to accelerate the movement of particles, which aims to improve the convergence rate. The updating equations of the acceleration coefficients are demonstrated as follows:

$$c_{1,k} = \frac{-2b}{1 + \exp\left(2a\left(pbest_{i,k} - x_{i,k} - m\right)\right)} + n, \tag{3.3}$$

$$c_{2,k} = \frac{-2b}{1 + \exp\left(2a\left(gbest_{k} - x_{i,k} - m\right)\right)} + n, \tag{3.4}$$

where a and b are two parameters used to describe the curve, which represent the steepness and the peak value, respectively; m denotes the offset of the central point; n is a negative value; and $\exp(\cdot)$ is the natural exponential function.

It is worth mentioning that, in Eq. (3.3) and Eq. (3.4), a, b, m, and n are all constant values. Appropriate values of the parameters would effectively enhance the performance of the optimization algorithm. In the proposed ASRPPSO, the parameters are set by $a = -0.035$, $b = -0.275$, $m = 0$, and $n = 1.2$ based on the experimental experience.

Evolutionary States

In the ASRPPSO, the particle's velocity as well as position are adjusted based on the evolutionary states, which are identified by the evolutionary factor (calculated based on the mean distance from each particle i to other particles). The equation of the mean distance d_i is shown as follows:

$$d_i = \frac{1}{S-1} \sum_{j=1}^{S} \sqrt{\sum_{r=1}^{D} \left(x_{i,r} - x_{j,r}\right)^2}, \tag{3.5}$$

where S and D represent the swarm size and the dimension of the particle, respectively.

Denote d_g as the global best particle of d_i, the evolutionary factor E_f can thus be calculated by:

$$E_f = \frac{d_g - d_{\min}}{d_{\max} - d_{\min}}, \tag{3.6}$$

where $d_{\max}$ and $d_{\min}$ are the maximum and minimum of d_i, respectively. It is worth mentioning that E_f belongs to $[0, 1]$.

Based on the evolutionary factor, the four states can be classified as follows:

$$\xi_k = \begin{cases} 1, & 0.00 \leq E_f \leq 0.25 \\ 2, & 0.25 < E_f \leq 0.50 \\ 3, & 0.50 < E_f \leq 0.75 \\ 4, & 0.75 < E_f \leq 1.00 \end{cases} \tag{3.7}$$

where $\xi_k = 1$ denotes the convergence state, $\xi_k = 2$ represents the exploitation state, $\xi_k = 3$ denotes the exploration state, and $\xi_k = 4$ represents the jumping-out state.

An Adaptive Weighted Velocity Updating Strategy

The novel velocity updating strategy can be explained based on four evolutionary states, which are summarized in Table 3.1. The decision factors α_{1,ξ_k} and α_{2,ξ_k} are used for determining the value of the Gaussian white noise, which are dependent on evolutionary states; and k denotes the current iteration number.

Table 3.1 Parameters in the velocity updating strategy of the ASRPPSO

Evolutionary State	ξ_k	α_{1,ξ_k}	α_{2,ξ_k}
Convergence	1	0	0
Exploitation	2	1	0.3
Exploration	3	0.3	1
Jumping-out	4	1	1

Gaussian White Noise

Inspired by [112], to improve the search ability of the optimizer by altering the system dynamics of the PSO algorithm, the Gaussian white noises δ_1 and δ_2 are added into the velocity updating equation to randomly perturb the movement of the particles. Note that the mean value and variance of δ_1 and δ_2 remain the same for the four evolutionary states.

3.2.2 The ASRPPSO-based FCM Algorithm

Due to the fact that the FCM algorithm's performance is highly dependent on the initial cluster centroids, the ASRPPSO is employed for optimally selecting the initial cluster centroids. As discussed before, the Gaussian white noise is embedded in the velocity updating model so that the particle's ability to get rid of local optima is enhanced, which indicates that the probability of getting better cluster centroids would be improved. The procedure of the ASRPPSO-based clustering algorithm is described in Algorithm 3.

Algorithm 3 The Procedure of the ASRPPSO-based FCM Algorithm

1: **Initialize** the parameters of ASRPPSO and FCM clustering, including the population size P, inertia weight w_1, acceleration coefficients $c_{1,1}$ and $c_{2,1}$, and the maximum velocity $V_{\max}$.
2: Set a swarm with P particles.
3: Initialize the position $x_{i,1}$, the velocity $v_{i,1}$, and the personal best $pbest_{i,1}$ of each particle i; initialize the global best $gbest_1$ of the swarm.
4: Obtain the cluster centroids.
5: Calculate the fitness value of each particle.
6: Update $pbest_{i,k}$ of each particle and $gbest_k$ of the swarm.
7: Compute E_f according to Eqs. (3.5) and (3.6), and determine the evolutionary state according to Eq. (3.7).
8: Update w_k, $c_{1,k}$, and $c_{2,k}$ according to Eqs. (3.2)-(3.4).
9: Update the velocity $v_{i,k}$ and the position $x_{i,k}$ according to Eq. (3.1).
10: Terminate if the maximum number of iterations is reached; otherwise, repeat Steps 4-9.

3.3 Experimental Results

3.3.1 Evaluation of the ASRPPSO

The performance of the developed ASRPPSO is evaluated on a series of benchmark functions. Here, 13 selected CEC basic benchmark functions are chosen for performance evaluation. The details of selected benchmark functions are presented in Table 3.2. The experimental results of the ASRPPSO are compared with the experimental results of some existing PSO algorithms. In this experiment, both the swarm size and the dimension of the selected benchmark functions are set to 30. For all chosen algorithms, the maximum iteration is set to be 10000. To ensure statistical reliability, each experiment is repeated 30 times independently. In the experiments, the mean and variance of the Gaussian white noises are set to be 0.5 and 1, respectively. The CPU used in the experiments is Intel Core i7-10700K. The programming platform used in the experiment is MATLAB R2021a.

In the experiments, five PSO algorithms (including the basic PSO algorithm, the PSO-Li-DIW algorithm [156, 158], the SPSO algorithm [177], the SDPSO algorithm [226], and the PSO-AWAC algorithm [111] are selected for comparison. Experimental results of the adopted algorithms via thirteen selected benchmark functions are summarized. The convergence plots of the algorithms on each benchmark function are displayed in Figs. 3.1-3.13. The vertical

Table 3.2 Details of selected benchmark functions

Function number	Function name	Search Range	Minimum	Threshold
$F_1(x)$	Sphere Function	$[-100, 100]$	0	0.1
$F_2(x)$	Schwefel 1.2 Function	$[-100, 100]$	0	0.1
$F_3(x)$	Schwefel 2.21 Function	$[-100, 100]$	0	0.1
$F_4(x)$	Schwefel 2.22 Function	$[-10, 10]$	0	0.1
$F_5(x)$	Rosenbrock Function	$[-30, 30]$	0	100
$F_6(x)$	Step Function	$[-100, 100]$	0	0.1
$F_7(x)$	Bent Cigar Function	$[-100, 100]$	0	100
$F_8(x)$	Rastrigin Function	$[-5.12, 5.12]$	0	50
$F_9(x)$	Zakharov Function	$[-5, 10]$	0	0.1
$F_{10}(x)$	Levy Function	$[-10, 10]$	0	0.1
$F_{11}(x)$	Ackley Function	$[-32, 32]$	0	0.1
$F_{12}(x)$	Griewank Function	$[-600, 600]$	0	0.1
$F_{13}(x)$	Sum of Different Powers Function	$[-1, 1]$	0	0.1

coordinate and the horizontal coordinate of the convergence plots indicate the logarithm value of the mean fitness value and the generation number, respectively.

The statistical results (including minimum, mean, and standard deviation) of the fitness values of the utilized algorithms on each benchmark function are listed in Table 3.3 and Table 3.4 for performance evaluation. It is worth mentioning that the results of the PSO algorithms on selected unimodal functions are listed in Table 3.3, and the results of the PSO algorithms on selected multi-modal functions are listed in Table 3.4. The success ratio and the iteration number when the algorithm converges are listed in Table 3.3 and Table 3.4 as well.

It can be found from the figures that the proposed ASRPPSO shows better convergence performance than the selected PSO algorithms. In Fig. 3.1, Figs. 3.5-3.7, Fig. 3.10, Figs. 3.12-3.13, the ASRPPSO obtains the smallest mean fitness value. In Figs. 3.2-3.3, Figs. 3.8-3.9 and Fig. 3.11, the mean fitness value of the ASRPPSO is smaller than most of the selected PSO algorithms. In Fig. 3.4, the differences between the mean fitness values of selected PSO algorithms are not obvious. To summarize, the proposed ASRPPSO exhibits satisfactory convergence performance.

The statistical results of selected PSO algorithms are presented in Table 3.3 and Table 3.4, which contain the minimum, the mean fitness value, and the standard deviation of the

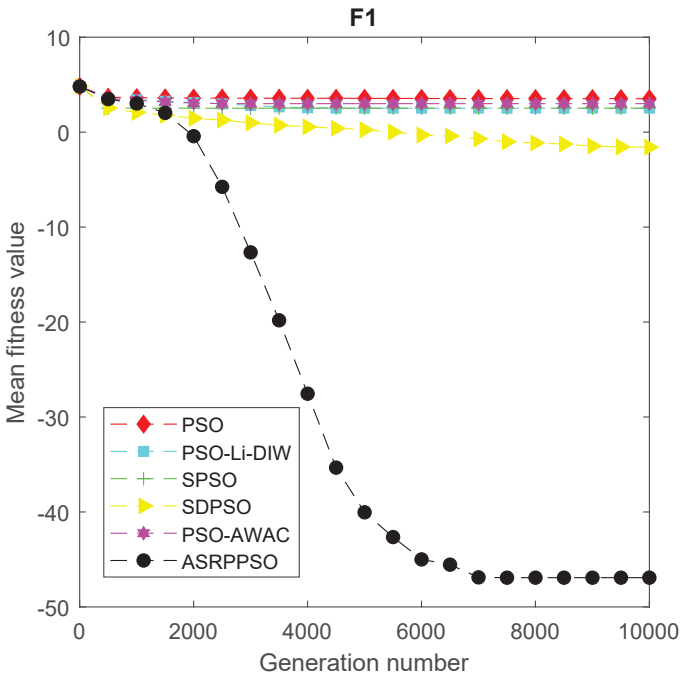


Fig. 3.1 Convergence plot for the Sphere Function $F_1(x)$

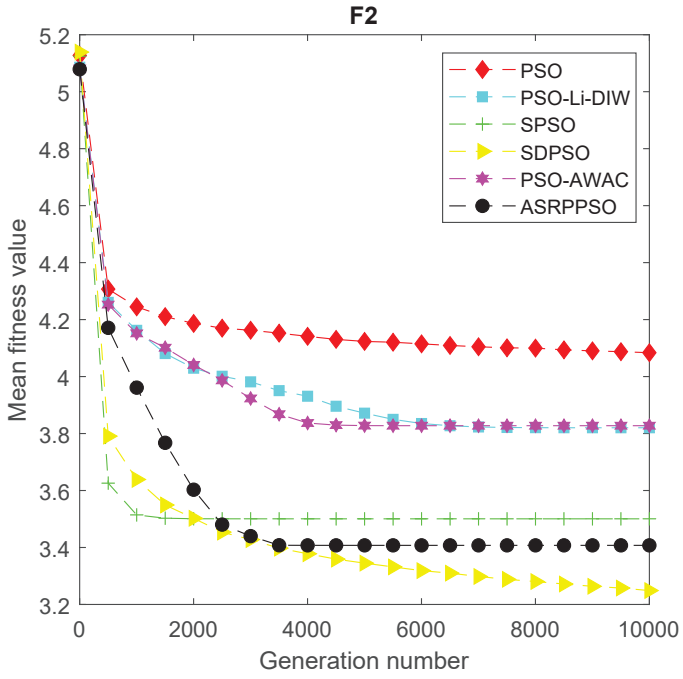


Fig. 3.2 Convergence plot for the Schwefel 1.2 Function $F_2(x)$

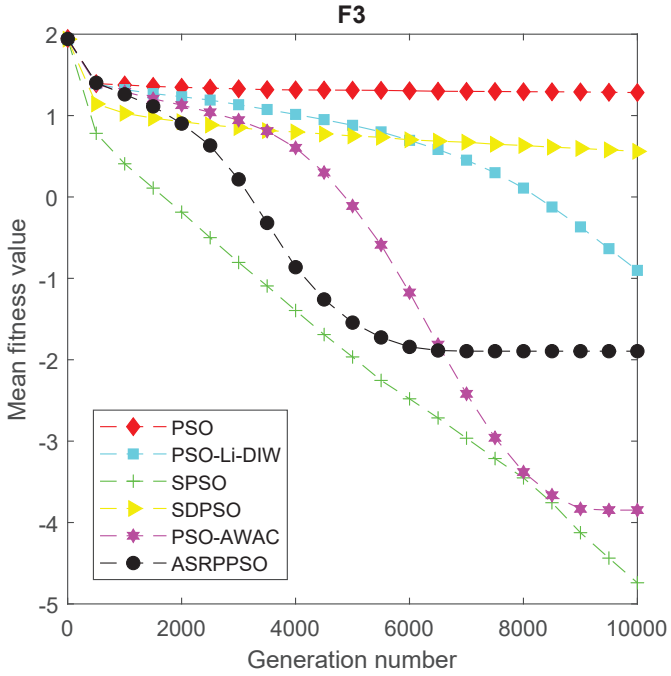


Fig. 3.3 Convergence plot for the Schwefel 2.21 Function $F_3(x)$

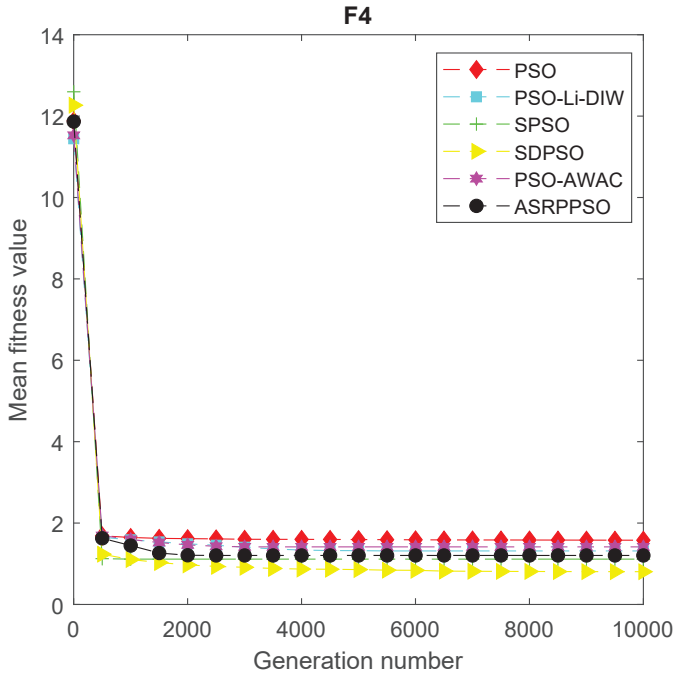


Fig. 3.4 Convergence plot for the Schwefel 2.22 Function $F_4(x)$

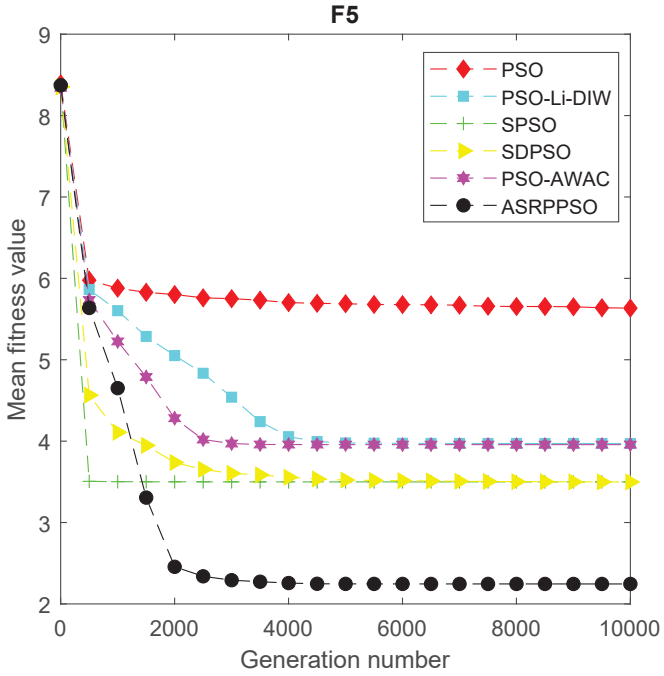


Fig. 3.5 Convergence plot for the Rosenbrock Function $F_5(x)$

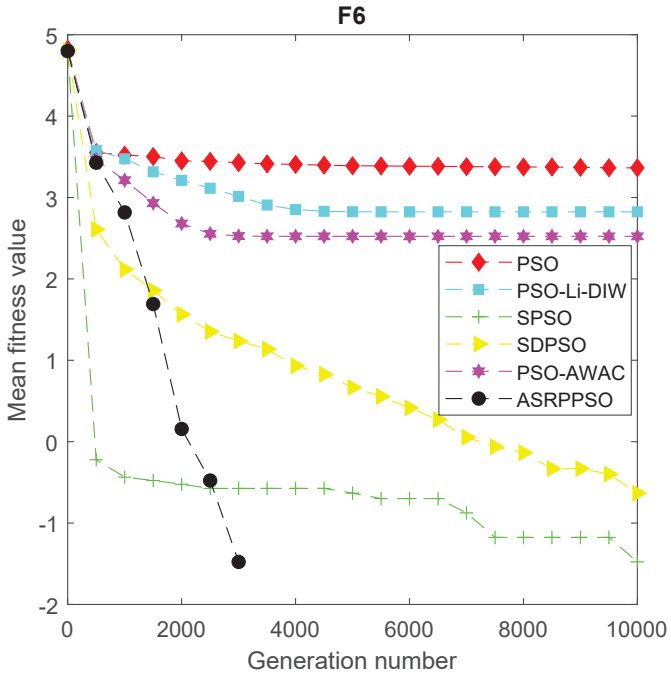


Fig. 3.6 Convergence plot for the Step Function $F_6(x)$

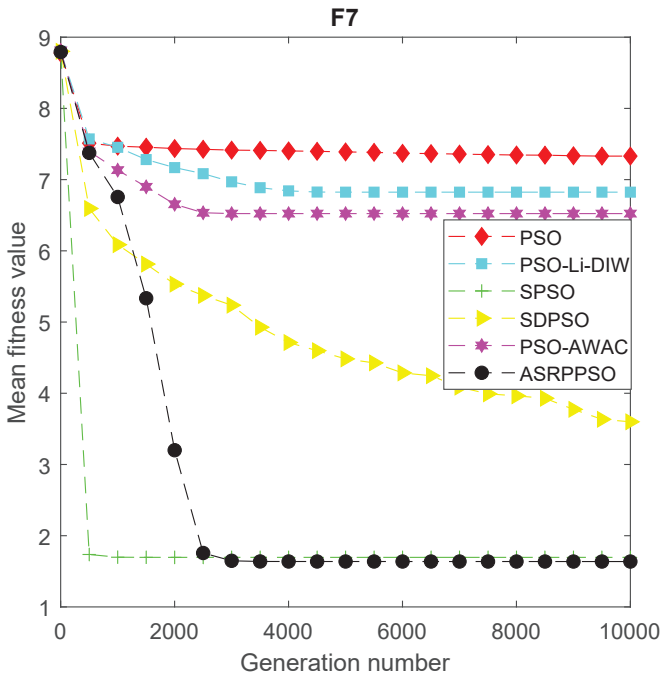


Fig. 3.7 Convergence plot for the Bent Cigar Function $F_7(x)$

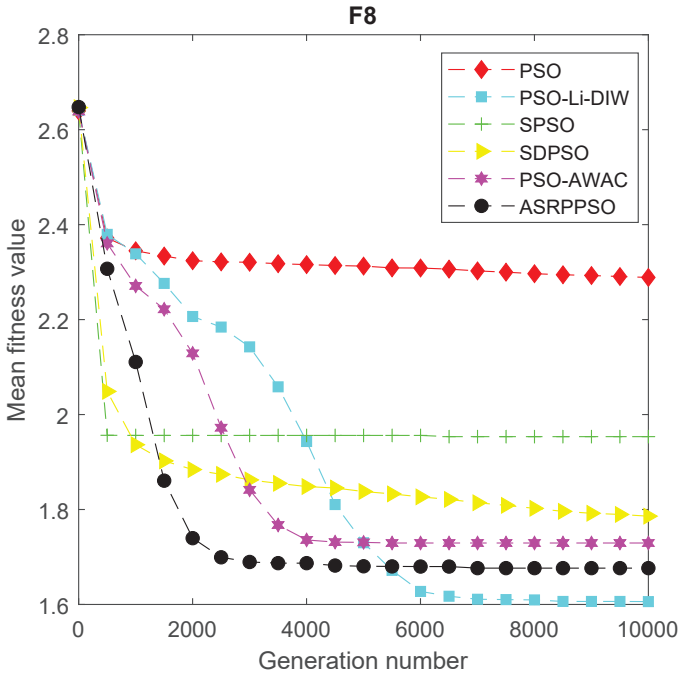


Fig. 3.8 Convergence plot for the Rastrigin Function $F_8(x)$

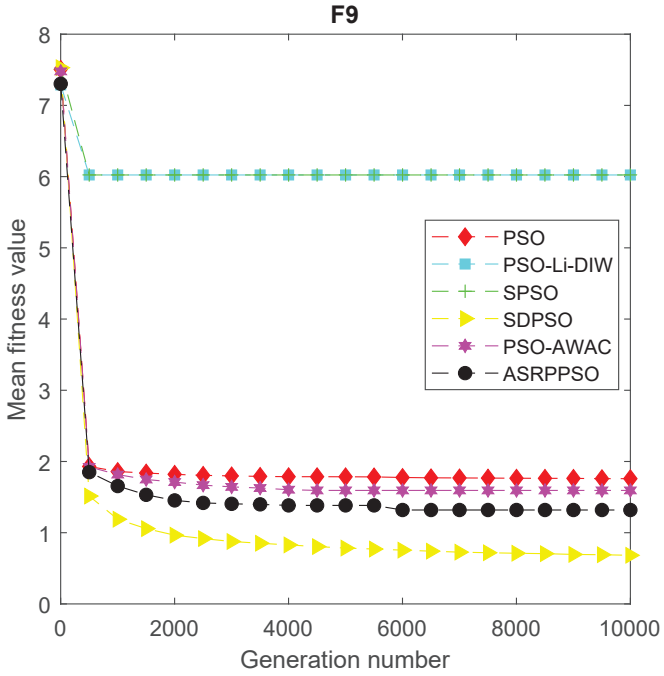


Fig. 3.9 Convergence plot for the Zakharov Function $F_9(x)$

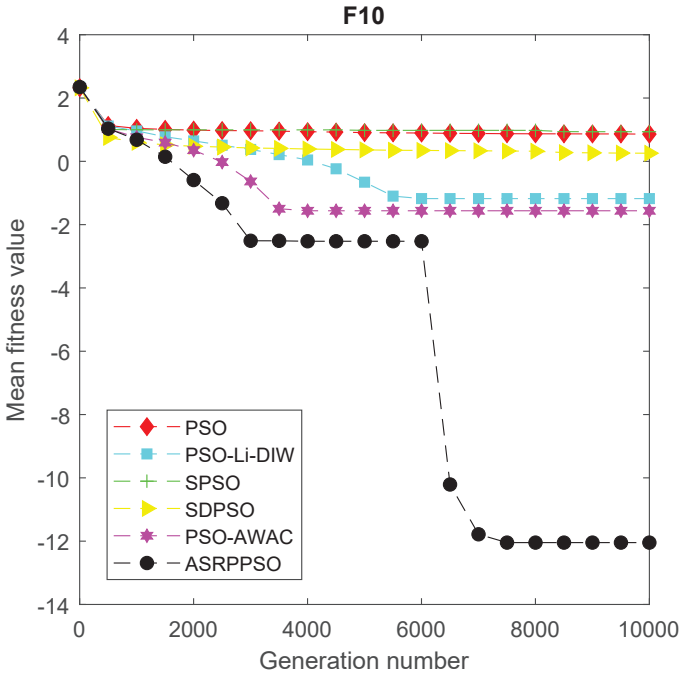


Fig. 3.10 Convergence plot for the Levy Function $F_{10}(x)$

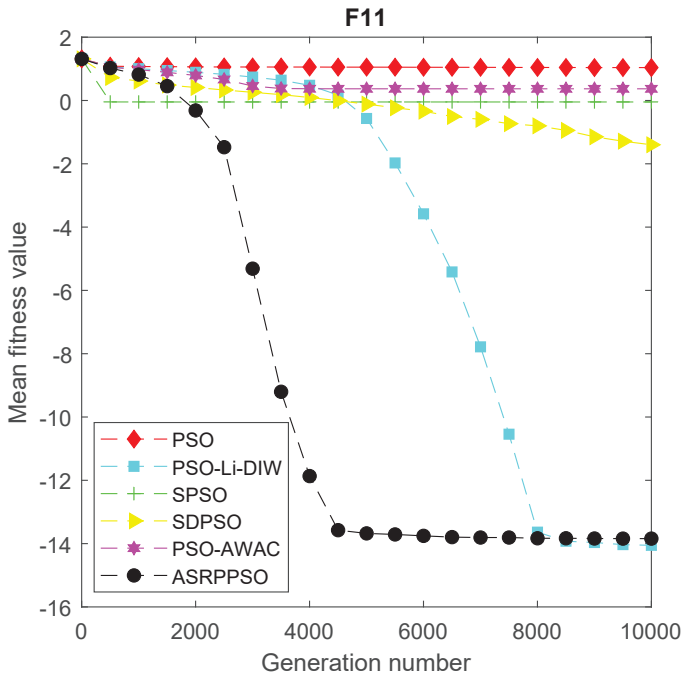


Fig. 3.11 Convergence plot for the Ackley Function $F_{11}(x)$

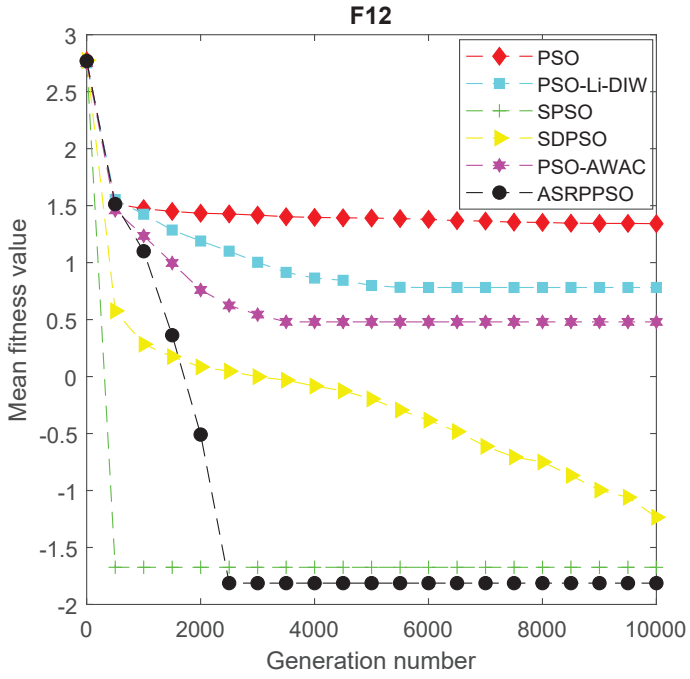


Fig. 3.12 Convergence plot for the Griewank Function $F_{12}(x)$

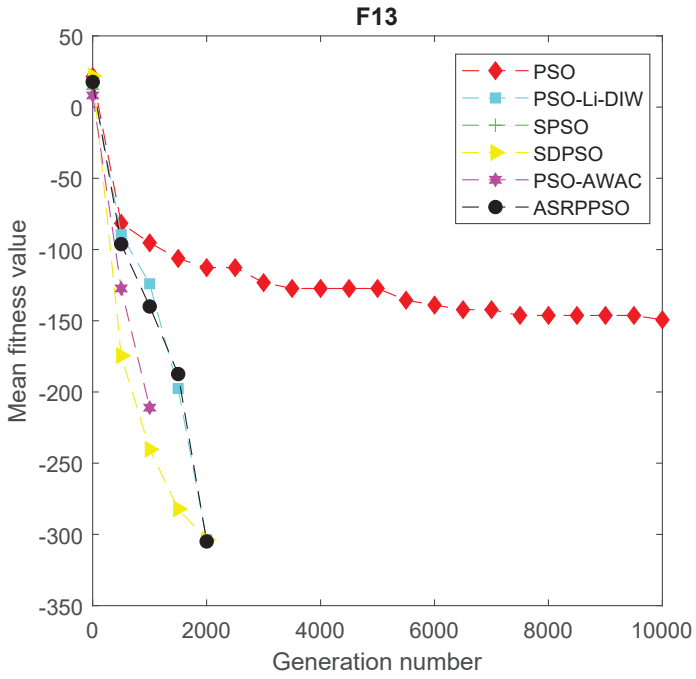


Fig. 3.13 Convergence plot for the Sum of Different Power Function $F_{13}(x)$

solutions. The minimum fitness value, the mean fitness value, and the standard deviation are the evaluation indices of the search ability of the algorithm. As the selected benchmark functions are all minimization problems, the smaller the fitness value, the better the solution found by the particles.

In Table 3.3 and Table 3.4, the minimum fitness value of the ASRPPSO is the smallest among all that of the selected PSO algorithms on $F_6(x)$, $F_8(x)$ and $F_{10}(x)$-$F_{13}(x)$. On the rest of the benchmark functions, the ASRPPSO also shows competitive performance in terms of the minimum fitness value. Compared with other selected PSO algorithms, the ASRPPSO obtains the smallest mean fitness value on $F_1(x)$, $F_5(x)$-$F_7(x)$, $F_{10}(x)$ and $F_{12}(x)$-$F_{13}(x)$. Considering the standard deviation of the fitness value, the ASRPPSO obtains the smallest one on $F_1(x)$, $F_5(x)$-$F_7(x)$, $F_{10}(x)$ and $F_{12}(x)$-$F_{13}(x)$ compared with other chosen algorithms. To conclude, the ASRPPSO has shown competitive performance on the benchmark functions in terms of the solution quality and the search capability based on the statistical analysis.

The success ratio is also listed in Table 3.3 and Table 3.4. According to the table, the ASRPPSO has the highest success ratio on $F_1(x)$-$F_3(x)$, $F_5(x)$-$F_7(x)$ and $F_{10}(x)$-$F_{13}(x)$, which means that the ASRPPSO could find the optimum with high probability. The success

Table 3.3 Statistical results of the selected PSO algorithms on selected unimodal functions

		PSO	PSO-Li-DIW	SPSO	SDPSO	PSO-AWAC	ASRPPSO
$F_1(x)$	Minimum	1.62×10^{3}	8.20×10^{-65}	3.42×10^{-197}	3.22×10^{-5}	1.69×10^{-79}	7.02×10^{-62}
	Mean	3.30×10^{3}	3.33×10^{2}	3.33×10^{2}	2.57×10^{-2}	1.00×10^{3}	1.20×10^{-47}
	Std. Dev.	3.05×10^{3}	1.83×10^{3}	1.83×10^{3}	7.58×10^{-2}	3.05×10^{3}	6.37×10^{-47}
	Ratio	0%	96.67%	96.67%	93.33%	90%	100%
	Con. Num.	10000	5142	580	6440	3770	1927
$F_2(x)$	Minimum	5.24×10^{3}	2.55×10^{-3}	1.30×10^{-13}	2.21×10^{2}	5.63×10^{-12}	2.93×10^{-9}
	Mean	1.21×10^{4}	6.61×10^{3}	3.17×10^{3}	1.77×10^{3}	6.72×10^{3}	2.56×10^{3}
	Std. Dev.	5.40×10^{3}	6.95×10^{3}	5.33×10^{3}	3.17×10^{3}	8.17×10^{3}	4.28×10^{3}
	Ratio	0%	33.33%	66.67%	0%	40%	66.67%
	Con. Num.	10000	9770	5277	10000	8299	5987
$F_3(x)$	Minimum	1.59×10^{1}	4.03×10^{-2}	1.58×10^{-6}	2.27	1.02×10^{-5}	5.09×10^{-4}
	Mean	1.92×10^{1}	1.25×10^{-1}	1.82×10^{-5}	3.63	1.42×10^{-4}	1.27×10^{-2}
	Std. Dev.	1.40	8.04×10^{-2}	2.53×10^{-5}	8.09×10^{-1}	2.34×10^{-4}	1.94×10^{-2}
	Ratio	0%	43.33%	100%	0%	100%	100%
	Con. Num.	10000	9853	3233	10000	5820	4153
$F_4(x)$	Minimum	1.73×10^{1}	2.34×10^{-40}	3.17×10^{-102}	9.41×10^{-4}	1.03×10^{-29}	6.70×10^{-18}
	Mean	3.80×10^{1}	2.07×10^{1}	1.30×10^{1}	6.39	2.60×10^{1}	1.60×10^{1}
	Std. Dev.	1.67×10^{1}	1.47×10^{1}	1.26×10^{1}	8.88	1.54×10^{1}	1.19×10^{1}
	Ratio	0%	13.33%	33.33%	50%	6.67%	20%
	Con. Num.	10000	9284	6755	8351	9527	8350
$F_5(x)$	Minimum	2.34×10^{5}	3.02×10^{-2}	3.89	2.31×10^{1}	2.48×10^{-2}	5.00×10^{-2}
	Mean	4.31×10^{5}	9.36×10^{3}	3.16×10^{3}	3.15×10^{3}	9.06×10^{3}	1.76×10^{2}
	Std. Dev.	1.10×10^{5}	2.74×10^{4}	1.64×10^{4}	1.64×10^{4}	2.74×10^{4}	5.57×10^{2}
	Ratio	0%	70%	86.67%	56.67%	80%	86.67%
	Con. Num.	15000	11145	2388	10800	7547	4132
$F_6(x)$	Minimum	1.81×10^{3}	0.00	0.00	0.00	0.00	0.00
	Mean	2.31×10^{3}	6.67×10^{2}	3.33×10^{-2}	2.33×10^{-1}	3.33×10^{2}	0.00
	Std. Dev.	2.48×10^{2}	2.54×10^{3}	1.83×10^{-1}	5.68×10^{-1}	1.83×10^{3}	0.00
	Ratio	0%	93.33%	96.67%	83.33%	96.67%	100%
	Con. Num.	10000	5537	1748	6906	3447	2166
$F_7(x)$	Minimum	1.66×10^{7}	2.36×10^{-4}	1.44×10^{-5}	4.42	6.06×10^{-5}	1.02×10^{-4}
	Mean	2.14×10^{7}	6.67×10^{6}	4.96×10^{1}	3.98×10^{3}	3.33×10^{6}	4.34×10^{1}
	Std. Dev.	2.06×10^{6}	2.54×10^{7}	5.05×10^{1}	1.34×10^{4}	1.83×10^{7}	5.04×10^{1}
	Ratio	0%	50%	53.33%	10%	53.33%	56.67%
	Con. Num.	10000	7617	4836	9749	6359	5531
$F_9(x)$	Minimum	1.61×10^{1}	9.31×10^{-15}	1.75×10^{-46}	1.64×10^{-1}	5.91×10^{-22}	2.80×10^{-18}
	Mean	5.74×10^{1}	1.05×10^{6}	1.05×10^{6}	4.82	3.92×10^{1}	2.08×10^{1}
	Std. Dev.	4.14×10^{1}	5.78×10^{6}	5.78×10^{6}	9.39	4.99×10^{1}	3.09×10^{1}
	Ratio	0%	33.33%	76.67%	0%	33.33%	60%
	Con. Num.	10000	8888	2988	10000	8010	5537
$F_{13}(x)$	Minimum	4.62×10^{-232}	0.00	0.00	0.00	0.00	0.00
	Mean	4.92×10^{-150}	0.00	0.00	0.00	0.00	0.00
	Std. Dev.	2.69×10^{-149}	0.00	0.00	0.00	0.00	0.00
	Ratio	100%	100%	100%	100%	100%	100%
	Con. Num.	3	2	2	2	2	2

ratio of the ASRPPSO on $F_9(x)$ also shows competitive performance compared with other PSO algorithms. The ratio of all selected PSO algorithms on $F_4(x)$ and $F_8(x)$ is not satisfactory because the number of local optima of these functions is very large, which indicates that it is difficult to find the globally optimal solution.

Table 3.4 Statistical results of selected PSO algorithms on selected multi-modal functions

		PSO	PSO-Li-DIW	SPSO	SDPSO	PSO-AWAC	ASRPPSO
$F_8(x)$	Minimum	1.78×10^2	1.49×10^1	4.97×10^1	3.00×10^1	1.59×10^1	8.95
	Mean	1.94×10^2	4.04×10^1	8.98×10^1	6.11×10^1	5.36×10^1	4.75×10^1
	Std. Dev.	1.08×10^1	2.24×10^1	2.55×10^1	2.20×10^1	2.58×10^1	2.11×10^1
	Ratio	0%	70%	3.33%	36.67%	46.67%	56.67%
	Con. Num.	10000	6446	9674	8107	6805	5372
$F_{10}(x)$	Minimum	5.29	1.50×10^{-32}	8.95×10^{-2}	1.00×10^{-4}	1.50×10^{-32}	1.50×10^{-32}
	Mean	7.22	6.65×10^{-2}	8.63	1.80	2.74×10^{-2}	8.90×10^{-13}
	Std. Dev.	8.01×10^{-1}	2.56×10^{-1}	7.89	2.95	1.50×10^{-1}	4.93×10^{-12}
	Ratio	0%	93.33%	3.33%	53.33%	96.67%	100%
	Con. Num.	10000	5175	9691	7667	3210	1958
$F_{11}(x)$	Minimum	9.42	6.22×10^{-15}	6.22×10^{-15}	1.78×10^{-3}	6.22×10^{-15}	6.22×10^{-15}
	Mean	1.09×10^1	8.82×10^{-15}	8.97×10^{-1}	3.99×10^{-2}	2.34	1.44×10^{-14}
	Std. Dev.	1.78	3.48×10^{-15}	9.60×10^{-1}	4.61×10^{-2}	5.32	4.68×10^{-15}
	Ratio	0%	100%	46.67%	90%	83.33%	100%
	Con. Num.	10000	5037	5487	7082	4281	2009
$F_{12}(x)$	Minimum	1.73×10^1	0.00	0.00	7.25×10^{-5}	0.00	0.00
	Mean	2.20×10^1	6.05	2.12×10^{-2}	5.83×10^{-2}	3.02	1.54×10^{-2}
	Std. Dev.	2.56	2.29×10^1	2.46×10^{-2}	7.20×10^{-2}	1.64×10^1	2.18×10^{-2}
	Ratio	0%	93.33%	96.67%	80%	96.67%	100%
	Con. Num.	10000	5435	756	7877	3191	1960

The number of iterations when the algorithm converges is illustrated in Table 3.3 and Table 3.4 as well. It can be seen from the table that the ASRPPSO performed well in all 13 functions compared with other selected algorithms.

To summarize, by comparing with other selected PSO algorithms, the proposed ASRPPSO demonstrates superior performance over the compared ones in terms of convergence and search ability.

3.3.2 Evaluation of the ASRPPSO-based FCM Algorithm

To verify the effectiveness of the proposed ASRPPSO-based FCM clustering algorithm, the Iris data from the UCI machine learning repository is employed for performance evaluation. Experimental results are assessed by using the Rand Index (RI), the Adjusted Rand Index (ARI), and the Weighted Kappa (WK) coefficient.

The Rand Index (RI) is a similarity measure that quantifies the agreement between two clustering results, as well as the agreement between a clustering result and the ground-truth labels. Note that for random clustering, the RI does not take a fixed value but fluctuates around 0. As such, the ARI is introduced to overcome this shortcoming. The value of the

ARI is between -1 and 1. When the clustering results are compared with labeled data, the larger the ARI, the higher the accuracy of the clustering results. In addition, the larger the ARI, the higher the similarity between the two clustering results.

The WK coefficient is a statistical measure used to quantify the consistency between two data partitions. Its value reflects the level of agreement between the two results. If the value is negative, the agreement between two comparators is worse than random. If the value is 0, it means the agreement between the comparators is equal to random. If the value is closer to 1, the agreement between the two comparators is higher.

The comparison evaluation of the clustering results by using the ASRPPSO-based FCM algorithm and the original FCM algorithm is listed in Table 3.5.

Table 3.5 Evaluation of two clustering algorithms

Algorithm	WK	ARI	RI
ASRPPSO-based FCM	0.7287	0.7287	0.8797
FCM	0.7149	0.7149	0.8737

According to Table 3.5, the WK coefficient, ARI, and RI of the ASRPPSO-based FCM algorithm are 0.7287, 0.7287, and 0.8797, respectively. The WK coefficient, ARI, and RI of the FCM algorithm are0.7149, 0.7149, and 0.8737, respectively. It can be found from the results that the ASRPPSO-based FCM algorithm is an effective clustering algorithm, which exhibits better performance than the FCM algorithm in aspects of WK, ARI, and RI. Therefore, the effectiveness and reliability of the developed ASRPPSO-based FCM algorithm are proven based on experimental results.

3.3.3 Outlier Detection on WAAM Data Sets

Data Pre-Processing

As mentioned previously in Section 2.3, there are four variables (e.g., X_Value, Current, Voltage, and ComputerTime) in each data set. In the experiments, only the voltage and current are utilized for outlier detection. The missing values and null values in the data sets are removed for data cleaning. In order to avoid the influence of different attributes, it is necessary to make every instance in each data set have the same scale so that each feature is

equally measured. Thus, the min-max normalization process is carried out to pre-process the data. The min-max normalization is given by:

$$X_{N_i} = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}} \tag{3.8}$$

where X_{N_i} denotes the ith normalized value of the variable X; $X_{\max}$ and $X_{\min}$ represent the maximum and the minimum values of X, respectively.

The Results of the Outlier Detection on WAAM Data Sets

During the AM process, some values of current and voltage may occasionally change abruptly, which indicates that the process is unstable. In this case, these instances are regarded as outliers. In the experiment, the number of clusters is 2, and each data sample contains 2 measured variables. As such, the number of variables to be optimized is 4. These variables are continuous centroid coordinates, and each coordinate is constrained within the range of the corresponding measured variable. The fitness value is calculated using the FCM objective function. The population size is set to 20 and the maximum number of iterations is set to 500. Therefore, the computational budget is 10000 fitness evaluations. The results of the outlier detection are shown in Table 3.6.

Table 3.6 Outlier detection results

Data Set	Cluster1	Cluster2 (Outlier)
Data Set 1	81326	16674
Data Set 2	78979	19021
Data Set 3	79954	18046
Data Set 4	83023	14977
Data Set 5	83096	14904

The clustering performance of the developed outlier detection method on the WAAM data sets is evaluated by using the silhouette coefficients, which is an effective approach to assess the unlabeled data clustering results [150]. Generally, the silhouette coefficient value is between -1 and 1. The performance of clustering is better when the value is closer to 1. In the experiments, the clustering performance of the proposed method is evaluated by using the average silhouette coefficients of all data points in each data set. Experimental results are shown in Table 3.7 and Figs. 3.14-3.18.

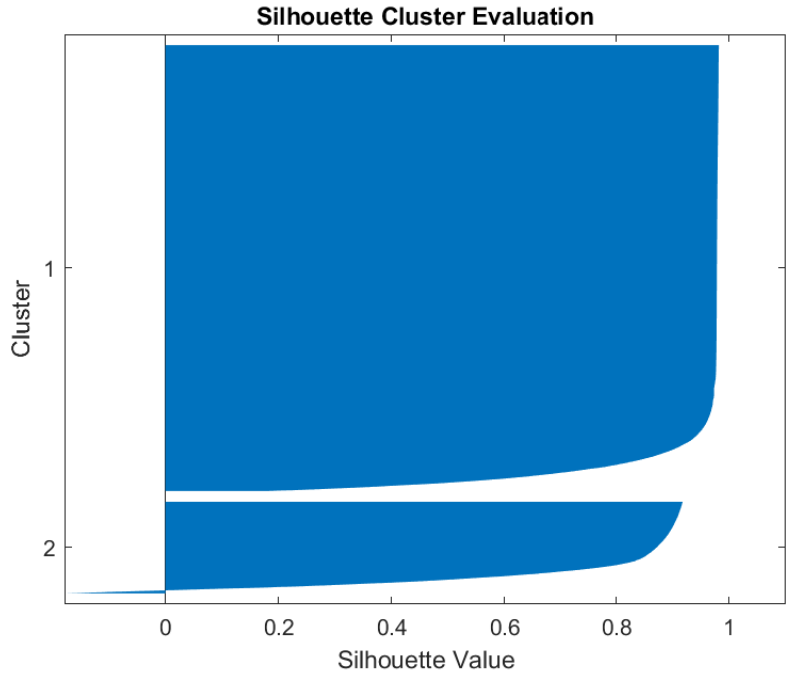


Fig. 3.14 The average silhouette coefficient of the clustering result on data set 1

Fig. 3.15 The average silhouette coefficient of the clustering result on data set 2

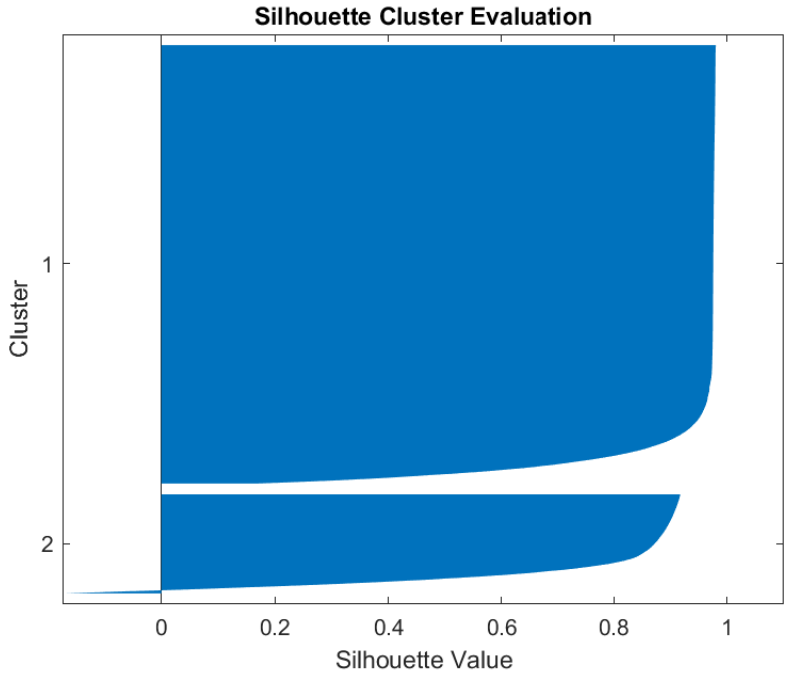


Fig. 3.16 The average silhouette coefficient of the clustering result on data set 3

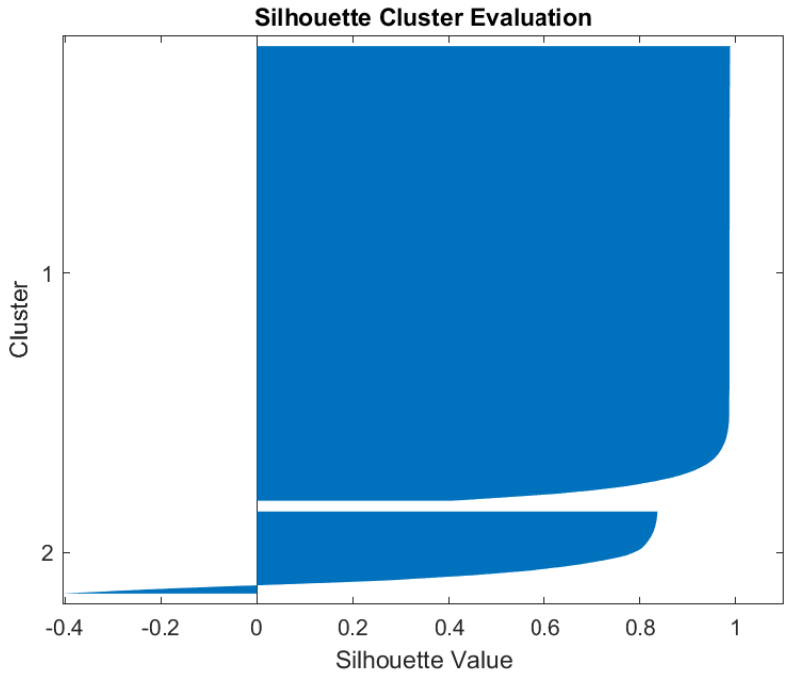


Fig. 3.17 The average silhouette coefficient of the clustering result on data set 4

Table 3.7 The average silhouette coefficients of the clustering results

Data Set	Average Silhouette Coefficients
Data Set 1	0.9135
Data Set 2	0.9034
Data Set 3	0.9073
Data Set 4	0.9123
Data Set 5	0.9315

Fig. 3.18 The average silhouette coefficient of the clustering result on data set 5

From Figs. 3.14-3.18, most of the silhouette values are positive, which means most data points are assigned to the right clusters. At the same time, according to Table 3.7, the average silhouette coefficients are all close to 1. Overall, the results of the ASRPPSO-based FCM clustering algorithm on the AM data are reasonable, which demonstrates the effectiveness of the developed method.

3.4 Conclusion

To conclude, an optimized FCM-based outlier detection method has been proposed for outlier detection on unlabeled WAAM data. Specifically, the ASRPPSO has been developed to optimize the initial locations of the cluster centroids in the FCM algorithm. An AWU strategy has been designed in the ASRPPSO to adaptively adjust the acceleration coefficients, and the Gaussian white noise has been added to the velocity updating equation based on the evolutionary states. Experimental results have shown the superiority of the ASRPPSO over some existing PSO algorithms in terms of convergence rate and solution quality. The outlier detection results have also shown the effectiveness of the proposed optimized FCM-based method. Future work can be summarized into the following five aspects: 1) designing a parameter selection strategy to automatically choose the parameters of the adaptive weighting function; 2) adjusting the mean and variance of the random noises adaptively for different evolutionary states; 3) applying the ASRPPSO to multi-objective optimization problems; and 4) deploying the proposed outlier detection algorithm to some other data analysis applications.

Chapter 4

Outlier Detection under Small-Sample Conditions: A Particle-Swarm-Optimization-Assisted Deep Transfer Learning Framework

4.1 Motivation

By identifying abnormal instances, outlier detection has been widely used in industrial data analysis. Thanks to its powerful feature extraction ability, DL has been extensively utilized in outlier detection, which contributes to the rapid development of DL-based outlier detection methods. Recently, numerous DL algorithms have been presented for tackling various outlier detection tasks [133].

Owing to the data-hungry nature of DL, a vast amount of training data with high-quality labels is required to build a reliable DL-based outlier detection model. By using DL methods, training data and testing data should be independent and identically distributed (i.i.d.). Nevertheless, it is difficult to guarantee that the collected data obey the i.i.d. rule in real-world problems. Meanwhile, the small-sample problem poses an additional challenge, since the limited amount of available data restricts the model's ability to learn robust representations. Fortunately, such constraints could be relaxed with the assistance of transfer learning (TL),

which attempts to fully leverage the knowledge discovered from the source domain for analyzing the target domain [197]. Among existing TL techniques, deep transfer learning (DTL) techniques have been widely adopted due mainly to their strong feature extraction and knowledge transfer abilities.

Serving as a popular class of DTL techniques, deep domain adaptation (DDA) has been developed to reduce the cross-domain distribution discrepancy [115]. Recently, various DDA methods have been introduced, which mainly include metric-learning-based DDA approaches, reconstruction-based DDA approaches, and adversarial-learning-based DDA approaches. Among them, the metric-learning-based approaches could quantify the cross-domain distribution discrepancy with the assistance of statistical approaches. Recognized as a well-known distance metric, in recent years, the maximum mean discrepancy (MMD) has been extensively exploited in metric-learning-based DDA approaches due to its computational efficiency and flexibility [132]. In many existing metric-learning-based DDA approaches that employ MMD, the marginal and conditional distributions are treated as having equal importance, which overlooks the distributional differences between the source and target domains. In this case, the trained DL model may be biased. To balance the influences of various distribution discrepancies and inspired by [192, 199], we will introduce weighting factors into the loss function of metric-learning-based DDA methods.

Data imbalance is a widespread problem in outlier detection, which makes it difficult to build a reliable detection model [178]. In most real-world scenarios, the quantity of normal instances is much larger than that of abnormal instances (i.e., outliers). Here, we separately consider the distribution discrepancies of outliers and normal instances in the proposed DDA strategy by introducing two independent coefficients (i.e., hyper-parameters). The coefficients are employed to balance the influences of the normal data and outliers to alleviate the data imbalance problem.

Hyper-parameters are generally chosen based on experimental experience, which is time-consuming [8]. A reasonable solution is to apply optimization techniques to select optimal hyper-parameters. It is worth mentioning that the relationship between these hyper-parameters and the training performance is indirect, since the hyper-parameters affect the balance among different loss terms and influence the model training process. Therefore, it is not straightforward to optimize these hyper-parameters directly using gradient-based mathematical

programming methods. Thanks to its fast convergence rate and low implementation difficulty, PSO has been successfully adopted to choose optimal hyper-parameters [118]. Compared with grid-based methods, PSO is more suitable for searching continuous hyper-parameters within predefined bounds without exhaustively evaluating a large number of parameter combinations. Compared with black-box optimizers, PSO has a clear update mechanism and can be conveniently embedded into the DTL training process. It is therefore reasonable to employ the PSO algorithm to automatically tune the hyper-parameters.

To sum up, this chapter develops a PSO-embedded DTL framework for outlier detection on imbalanced industrial datasets in a limited-sample scenario. The utilized data set is collected by multiple sensors deployed in a WAAM pilot line. The contributions are listed in the following threefold:

1. A novel DDA strategy is proposed, where the weighting factors are designed to balance i) the marginal distribution discrepancy loss and ii) the conditional distribution discrepancy loss of both normal instances and outliers to tackle the data imbalance problem.

2. The PSO algorithm is employed to select the optimal weighting factors automatically.

3. The developed approach is deployed to detect the outliers on real-world industrial data obtained through the WAAM process. Experimental results demonstrate the superiority of the developed PSO-embedded DTL outlier detection approach over two benchmark approaches.

The remaining sections of this chapter are organized as follows. The preliminary is provided in Section 4.2. In Section 4.3, the developed DTL framework is introduced. Experiment settings, network configurations, and results are presented in Section 4.4. Finally, conclusions and future directions are presented in Section 4.5.

4.2 Preliminary

The DDA strategy aims to align the source and target domains by minimizing the cross-domain distribution discrepancy based on the conditional distribution $P(Y|X)$. In general,

calculating $P(Y|X)$ directly is a difficult mission. Many existing DDA strategies employ the Bayes' theorem to obtain $P(Y|X)$ [114, 199]. MMD is a powerful approach to quantify the distribution discrepancy between the source domain and target domain [132]. The MMD is calculated by mapping the original data into a reproducing kernel Hilbert space (RKHS). The MMD of $P(X)$ between the source domain and target domain can be calculated by:

$$D\left(X^s, X^t\right) = \left\| \frac{1}{n} \sum_{i=1}^{n} \phi\left(x_i^s\right) - \frac{1}{m} \sum_{i=1}^{m} \phi\left(x_i^t\right) \right\|_{\mathcal{H}}^2, \tag{4.1}$$

where X^s and X^t represent the source and target data, respectively; n and m are the total number of instances in the source and target domains, respectively; x_i^s and x_i^t indicate the ith samples in the source and target domains, respectively; $\phi(\cdot)$ denotes a mapping from the original space into the RKHS; and $\|\cdot\|_{\mathcal{H}}^2$ is the squared norm in the RKHS.

It is worth mentioning that in the DDA strategy for the binary classification task, the marginal distributions $P(Y)$ of the source and target domains are the same, and the MMD of $P(X|Y)$ between the source and target domains (i.e., $D\left(X^s|Y^s, X^t|Y^t\right)$) can be calculated by accumulating the MMDs of the instances with same classes between the source and target domains. $D\left(X^s|Y^s, X^t|Y^t\right)$ can be calculated by:

$$\begin{aligned} & D\left(X^s|Y^s, X^t|Y^t\right) \\ = & \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi\left(x_{i,1}^s\right) - \frac{1}{m_1} \sum_{i=1}^{m_1} \phi\left(x_{i,1}^t\right) \right\|_{\mathcal{H}}^2 + \left\| \frac{1}{n_2} \sum_{j=1}^{n_2} \phi\left(x_{j,2}^s\right) - \frac{1}{m_1} \sum_{j=1}^{m_2} \phi\left(x_{j,2}^t\right) \right\|_{\mathcal{H}}^2, \end{aligned} \tag{4.2}$$

where n_1 and n_2 represent the total numbers of source domain instances in class 1 and class 2, respectively; m_1 and m_2 represent the total number of target domain instances in class 1 and class 2, respectively; $x_{i,1}^s$ and $x_{i,2}^t$ indicate the ith samples of the first class in the source and target domains, respectively; and $x_{j,1}^s$ and $x_{j,2}^t$ are the jth samples of the second class in the source and target domains, respectively.

4.3 Methodology

In some existing DDA-based classification methods, the loss function is designed by considering the classification loss and the domain loss. Recently, many researchers have focused on balancing the weights between the classification loss and the domain loss [114]. Nevertheless, the scale of distribution discrepancies between the source data and target data should also be considered. Using hyper-parameters to adjust the weight of each term in the loss function has been proven to be a fast and easy approach. In [114], the hyper-parameters have been introduced to adjust the weight among the domain loss, the regularization term, and the training loss.

Note that data imbalance is a frequent problem in data analysis, which could lead to a biased model with unsatisfactory results. Recently, a number of strategies have been proposed in order to tackle the data imbalance challenge [192]. In [192], the data imbalance problem has been tackled by adaptively changing the weights of samples from different classes. It is worth mentioning that the marginal distributions are assumed to be unchanged when calculating the MMD of $P(Y|X)$, which may affect the classification accuracy when dealing with other data sets. Motivated by the above discussions, the weights of different domain losses and the weights to tackle the imbalance problem are considered at the same time so as to design a proper loss function.

The selection of hyper-parameters is usually manual and based on experimental experience. In fact, selecting the optimal hyper-parameters is an optimization problem. As a popular optimization algorithm, the PSO algorithm (which is a famous evolutionary computation technique) has been widely exploited in various optimization tasks because of its high efficiency and easy implementation, which seems to be an appropriate solution to select the optimum hyper-parameters automatically.

In this chapter, a novel PSO-embedded DTL framework is developed, where a new DDA strategy is put forward to reduce the cross-domain discrepancy by adjusting the weight of marginal distribution discrepancy loss and the conditional distribution discrepancy loss. In addition, two separate coefficients are introduced to balance the conditional domain discrepancies of normal instances and outliers with the hope to alleviate the data imbalance problem. In addition, the hyper-parameters are tuned by the PSO algorithm automatically.

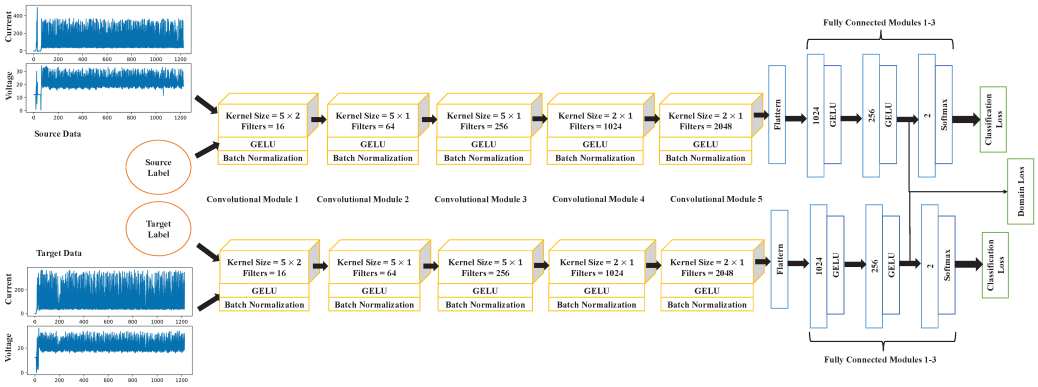


Fig. 4.1 The proposed PSO-embedded DTL outlier detection approach

4.3.1 The PSO-Embedded DTL Framework

The developed framework is displayed in Fig. 4.1. The convolutional neural network (CNN) is utilized for feature extraction, which comprises five convolutional modules and three fully connected (FC) layers. In each convolutional module, there is one convolutional layer followed by the Gaussian error linear unit (GELU) mapping and batch normalization. In the first and second FC layers, the activation function is GELU. As a popular activation function, GELU shows better robustness and generalization ability than some existing activation functions (e.g., ReLU and Leakly-ReLU) in some sense [76]. As such, GELU is selected as the activation function. In the third FC layer, the softmax activation function is adopted.

4.3.2 Loss Function

Classification Loss

The binary cross-entropy is applied to calculate the classification loss. Specifically, the classification loss includes the classification loss of the source data L_c^s and the target data L_c^t. L_c^s and L_c^t are given by:

$$L_c^s = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i^s \log\left(p_i^s\right) + \left(1-y_i^s\right)\log\left(1-p_i^s\right)\right], \tag{4.3}$$

$$L_c^t = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i^t \log\left(p_i^t\right) + \left(1-y_i^t\right)\log\left(1-p_i^t\right)\right], \tag{4.4}$$

where n and m are the source and target sample numbers, respectively; p_i^s and p_i^t represent the probabilities when the predicted label is the same as the real label of the ith sample in the source and target domains, respectively; and y_i^s and y_i^t are the real labels of the ith sample in the source and target domains, respectively.

Domain Loss

The domain loss between source and target domains contains two parts (i.e., marginal distribution discrepancy $L_{\mathrm{MMD}}^{P(X)}$ and conditional distribution discrepancy $L_{\mathrm{MMD}}^{P(X|Y)}$), which can be calculated by:

$$L_{\mathrm{MMD}}^{P(X)} = \left\| \frac{1}{n}\sum_{k=1}^{n}\phi\left(\psi\left(x_k^s\right)\right) - \frac{1}{m}\sum_{k=1}^{m}\phi\left(\psi\left(x_k^t\right)\right)\right\|_{\mathcal{H}}^2, \tag{4.5}$$

$$\begin{aligned} L_{\mathrm{MMD}}^{P(X|Y)} &= \mu_1 L_{\mathrm{MMD}}^{P(X|Y=Normal)} + \mu_2 L_{\mathrm{MMD}}^{P(X|Y=Outlier)} \\ &= \mu_1 \left\| \frac{1}{n_N}\sum_{i=1}^{n_N}\phi\left(\psi\left(x_i^s\right)\right) - \frac{1}{m_N}\sum_{i=1}^{m_N}\phi\left(\psi\left(x_i^t\right)\right)\right\|_{\mathcal{H}}^2 \\ &\quad + \mu_2 \left\| \frac{1}{n_O}\sum_{j=1}^{n_O}\phi\left(\psi\left(x_j^s\right)\right) - \frac{1}{m_O}\sum_{j=1}^{m_O}\phi\left(\psi\left(x_j^t\right)\right)\right\|_{\mathcal{H}}^2, \end{aligned} \tag{4.6}$$

where $L_{\mathrm{MMD}}^{P(X|Y=Normal)}$ ($L_{\mathrm{MMD}}^{P(X|Y=Outlier)}$) denotes the domain discrepancy loss of normal data (outliers) between source and target domains; μ_1 and μ_2 are two coefficients; n_N and n_O represent the normal data and outliers in the source domain, respectively; m_N and m_O are the normal data and outliers in the target domain, respectively; x_k^s, x_k^t represent the kth sample of the data in source and target domains, respectively; x_i^s and x_i^t are the ith sample of the normal data in source and target domains, respectively; x_j^s and x_j^t are the jth sample of outliers in source and target domains, respectively; and $\psi(\cdot)$ represents the distributions of deep features of $P(\cdot)$. $\psi(\cdot)$ is learned by five convolutional modules and the first two fully connected modules in the designed CNN model. In our framework, the Gaussian kernel is employed,

which is given as follows:

$$k(a,b) = \exp\left(\frac{-\|a-b\|^2}{2\sigma^2}\right), \tag{4.7}$$

where a and b are two random samples; $\|\cdot\|$ is the Euclidean distance; and σ represents the kernel's width.

Overall Loss Function

The entire loss function of the proposed outlier detection approach is given as follows:

$$\begin{aligned} L = L_c^s + \delta L_c^t + \varepsilon L_{\text{MMD}}^{P(X)} + \mu_1 L_{\text{MMD}}^{P(X|Y=Normal)} \\ + \mu_2 L_{\text{MMD}}^{P(X|Y=Outlier)} + \lambda \|\omega\|_2 , \end{aligned} \tag{4.8}$$

where $\|\cdot\|_2$ is the L_2 norm to avoid the overfitting problem; ω denotes the weights of the CNN; δ and ε are weighting factors to balance the domain discrepancy loss of normal instances and outliers; and λ is the penalty factor.

Remark 4.1. *During the WAAM process, the working status is normal most of the time. The current and voltage will be abnormal only when some unexpected changes occur (e.g., arc instability and irregular metal transfer). In this case, the WAAM data set seriously suffers from the data imbalance problem, where the number of normal instances is much greater than that of outliers. To alleviate the impact of class imbalance, two coefficients (i.e., μ_1 and μ_2) are introduced. The weight between $L_{\text{MMD}}^{P(X|Y=Normal)}$ and $L_{\text{MMD}}^{P(X|Y=Outlier)}$ is balanced by adjusting μ_1 and μ_2, which improves the performance of the model on WAAM outlier detection.*

4.3.3 Parameter Optimization

Basic PSO

In the basic PSO algorithm, each particle in the swarm is a candidate solution, which is directed by its personal best location and the global best location (i.e., pbest and gbest) found by the entire swarm in a D-dimensional problem space. The updating equations of the velocity and position of the αth particle at the $\beta+1$th iteration (i.e., $v_{\alpha,\beta+1}$ and $x_{\alpha,\beta+1}$) are

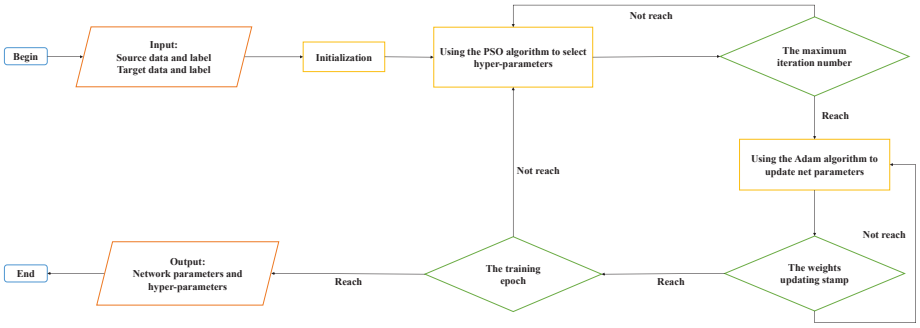


Fig. 4.2 The training procedure of the developed framework

given as follows:

$$\begin{aligned} v_{\alpha,\beta+1} &= w v_{\alpha,\beta} + c_1 r_1 \left(pi_{\alpha,\beta} - x_{\alpha,\beta}\right) + c_2 r_2 \left(pg_{\beta} - x_{\alpha,\beta}\right), \\ x_{\alpha,\beta+1} &= x_{\alpha,\beta} + v_{\alpha,\beta+1}, \end{aligned} \tag{4.9}$$

where w denotes the inertia weight; c_1 and c_2 represent two acceleration coefficients; r_1 and r_2 are two random numbers selected from $[0, 1]$; $pi_{\alpha,\beta}$ denotes the personal best location found by the αth particle itself; pg_{β} is the global best location of all particles.

Optimization Strategy

The basic PSO algorithm is used for selecting proper hyper-parameters (σ, δ, ε, μ_1, and μ_2). The CNN weights ω are optimized by the Adam algorithm. During the training process, the hyper-parameters and net parameters are updated automatically. The training procedure of the method is depicted in Fig. 4.2. The hyper-parameter selection and model training procedure can be interpreted as a bi-level optimization problem. In this setting, the upper-level problem aims to search for suitable hyper-parameters using PSO, while the lower-level problem trains the deep transfer learning model under each candidate hyper-parameter setting. After the corresponding lower-level model training process, the fitness value of each particle is

calculated based on the overall loss function of the trained model, and this value is then used to guide the upper-level PSO search.

4.4 WAAM Outlier Detection

Data Pre-Processing

In the experiments, the abnormal data points (e.g., missing or null) are removed to obtain clean data. To enhance the difference between the instances, the values of current and voltage are transformed to their cubes. Each data set is divided into several consecutive segments with the purpose of extracting and analyzing the features of time, where each segment (containing 56 instances) corresponds to a label (i.e., "Normal" and "Outlier"). The label of each segment depends on the label of each instance in the segment. Specifically, the label of the segment will be "Normal" if the labels of all the instances in this segment are "Normal", otherwise, the label of this segment will be "Outlier". In the experiments, the labels are transformed to one-hot formats for ease of classification.

4.4.1 Model Training

Parameter Setting and Model Configuration

The system of the experiments is Ubuntu 20.04.5. The GPU is NVIDIA RTX A6000 with 49 GB of memory. The experiments are conducted on Python 3.10.9 and CUDA 11.7. The configurations of 5 convolutional modules and 3 FC layers are displayed in Fig. 4.1. In the experiment, the developed approach is compared with the CNN-based approach and the basic DTL-based approach with the purpose of verifying the performance of the developed approach. The ablation study is also conducted to verify the effectiveness of the regularization term in the developed approach. Note that the same network architecture and configurations of the CNN are employed for comparison.

In the experiments, the hyper-parameters (σ, δ, ε, μ_1, and μ_2) are selected automatically by the basic PSO algorithm. The constraint intervals of hyper-parameters are set to be $[0, 100]$, $[1, 5]$, $[0, 3]$, $[0, 3]$, and $[1, 10]$, respectively. In order to put more consideration into the target data, the minimum of δ is set to be 1. The maximum of μ_2 is set to be 12,

which aims to tackle the data imbalance problem. Based on the experimental experience, the parameter λ of the L_2 regularization term is set as 0.00001. In the basic PSO algorithm, the particle number is 25, and the number of generations is set to 20, resulting in 500 fitness evaluations for each PSO search. The control parameters (e.g., w, c_1, and c_2) are set to be 0.6, 1.5, and 1.5, respectively.

Model Training and Testing

In the experiments, two different data sets are selected as the source and target data sets which both contain all segments and labels of the pre-processed data sets. The training data consists of two parts, which are the source and target training data. To be specific, the source training data is the same as the source data. The target training data is the first 450 segments of the target data. Through the training process, the source and target training data are fed to the CNN at the same time. The classification loss is calculated based on the source and target training data. The domain loss is calculated based on the deep features of the source and target training data. The total number of training epochs is 1000, and the PSO search is performed every 50 training epochs. Therefore, the PSO search is conducted 20 times during the whole training process, and the cumulative computational budget is 10000 fitness evaluations. The testing data is the rest of the target data excluding the target training data. The testing data is input to the trained model after the training process. The output of the testing process is the predicted labels of the testing data.

In realistic application scenarios, labeled WAAM data usually become available progressively from the beginning of the WAAM process, rather than being randomly sampled from the entire sequence. To better reflect practical deployment conditions, in the experiments, the initial segments of the target data are used as limited labeled training data, while the remaining segments are reserved as unseen testing data. It should be noted that such a sequential split may introduce temporal or ordering bias, since the initial segments may not fully represent the distribution of the later segments, especially when the WAAM process is time-varying or when outliers are unevenly distributed. To mitigate this issue, in the experiments, the target training and testing sets are kept strictly non-overlapping, and the same chronological splitting strategy is applied to all compared methods. The proposed

framework is also evaluated across multiple source-target data set combinations to examine its robustness under different transfer scenarios.

4.4.2 Results and Discussion

Experimental Results

In the experiments, one data set is selected as the source data set, and the other data sets are selected as the target data sets. We select three data sets (Data1, Data3, and Data5) as the source data sets in turn. The optimization process of the developed approach is shown in Figs. 4.3-4.5, where columns 1-5 represent δ, ε, μ_1, μ_2, and σ, respectively, and rows 1-4 represent four target data sets.

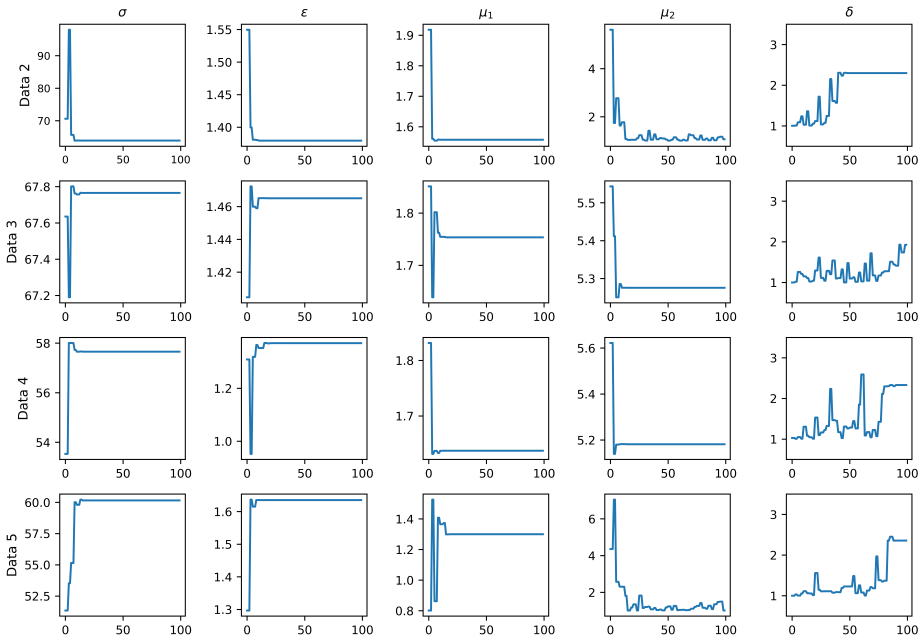


Fig. 4.3 The optimization process of the developed framework when Data 1 is the source data

Five outlier detection tasks (i.e., T1, T2, T3, T4, and T5) are explored by utilizing the designed approach. In each task, two different data sets are selected randomly. Three evaluation metrics (i.e., accuracy, precision, and sensitivity) are employed for performance evaluation. The accuracy, precision, and sensitivity of three approaches on five outlier detection tasks are summarized in Table 4.1. As mentioned previously, in each outlier detection task, the source and target data sets are randomly selected from 5 data sets. In

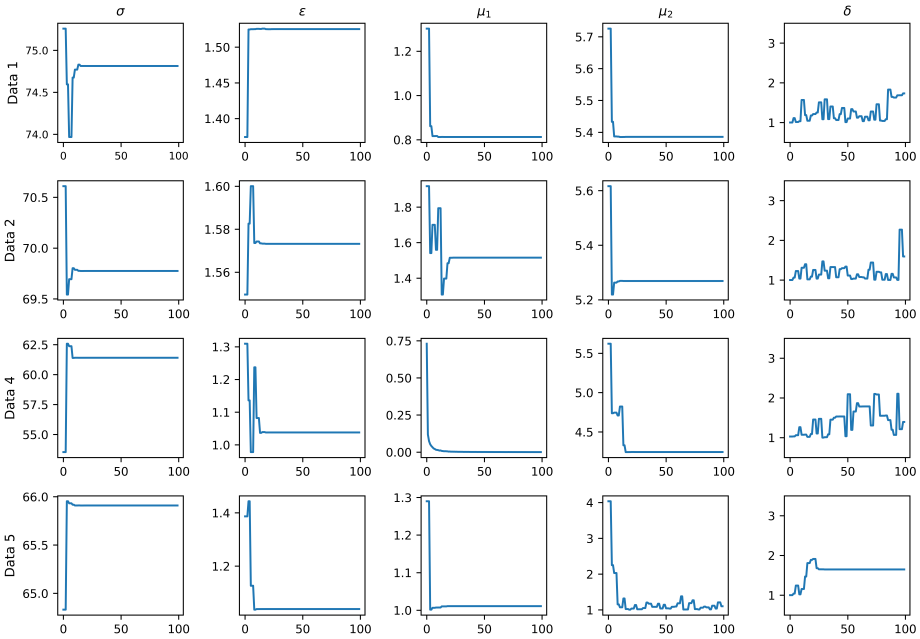


Fig. 4.4 The optimization process of the developed framework when Data 3 is the source data

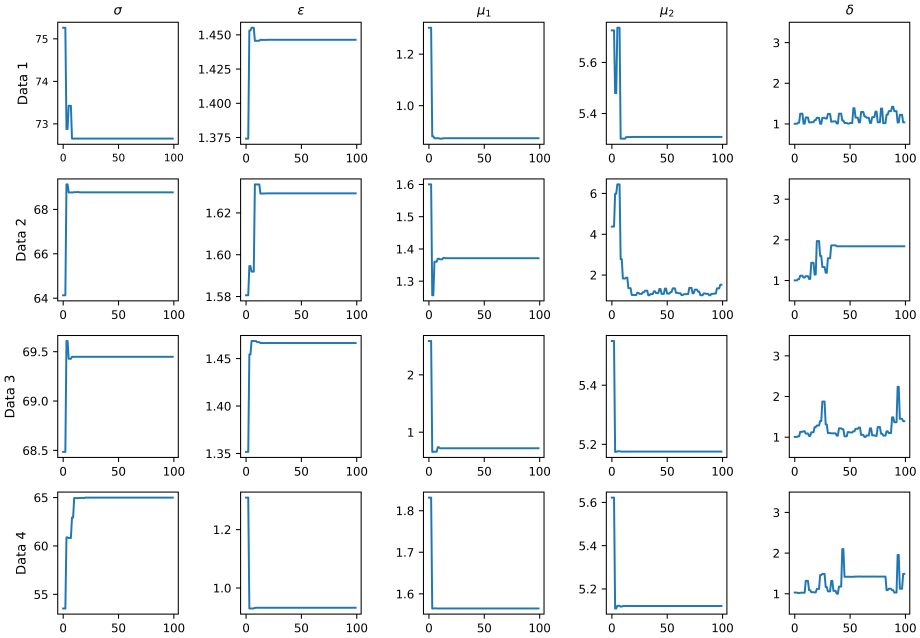


Fig. 4.5 The optimization process of the developed framework when Data 5 is the source data

the table, the selected source and target data in each task are described as "(source data, target data)". According to Table 4.1, the developed framework obtains the highest values

in accuracy, precision, and sensitivity in all five tasks. The results also demonstrate the effectiveness of the regularization term of the developed approach.

Table 4.1 The outlier detection results using selected approaches

Evaluation Metrics	Approach	T1 (Data1, Data2)	T2 (Data2, Data4)	T3 (Data3, Data1)	T4 (Data4, Data5)	T5 (Data5, Data3)
Accuracy (%)	CNN	78.85	92.46	99.00	96.38	95.23
	CNN+DDA	97.62	99.08	99.11	98.15	98.38
	Developed Approach without Regularization	97.04	98.18	97.69	98.17	97.25
	Developed Approach	**97.69**	**99.15**	**99.46**	**98.62**	**99.08**
Precision (%)	CNN	45.12	70.18	96.14	95.46	93.34
	CNN+DDA	94.65	98.52	96.15	97.55	96.15
	Developed Approach without Regularization	96.34	95.23	95.94	97.48	95.33
	Developed Approach	**96.67**	**98.86**	**96.87**	**98.33**	**97.20**
Sensitivity (%)	CNN	38.19	92.34	98.25	95.34	75.11
	CNN+DDA	90.94	95.69	98.68	96.82	94.94
	Developed Approach without Regularization	89.18	94.35	96.72	96.14	94.05
	Developed Approach	**91.34**	**96.55**	**98.71**	**97.23**	**95.57**

The optimal values of 5 hyper-parameters selected by the PSO algorithm in each outlier task are listed in Table 4.2. We can see in Tables 4.1-4.2 that the discovered hyper-parameters are different in each outlier detection task, which indicates that the solution space of each task is quite different. Thereby, there is a need to apply such kind of hyper-parameter optimization strategy for analyzing the sensor data collected in various working status. Compared with two benchmark methods, the proposed PSO-embedded DTL outlier detection approach has better detection accuracy and exhibits satisfactory generalization ability.

Table 4.2 The optimal values of hyper-parameters in each outlier detection task

Task	δ	ε	μ_1	μ_2	σ
T1	2.30	1.38	1.56	1.06	63.90
T2	1.07	1.14	1.50	4.84	57.51
T3	1.73	1.53	0.81	5.39	74.81
T4	1.54	1.01	1.17	5.08	72.06
T5	1.28	1.47	0.72	5.18	69.45

4.5 Conclusion

In this chapter, a novel PSO-embedded DTL framework has been developed for outlier detection on imbalanced WAAM data under limited-sample conditions. A new DDA strategy has been put forward to minimize the cross-domain discrepancies of both the marginal distribution and the conditional distribution. To be specific, in order to alleviate the data imbalance problem, the conditional domain discrepancies of normal instances and outliers

have been adjusted by two separate coefficients. The PSO algorithm has been adopted to adjust the hyper-parameters of the proposed method. The developed DTL framework has been applied in the outlier detection tasks on WAAM data sets. Experimental results have shown satisfactory performance in the detection accuracy of the proposed approach. The developed framework has shown promising performance of outlier detection on the WAAM data sets collected from real-world manufacturing processes. Future work can be summarized into the following five aspects: 1) applying the developed framework to the real-world outlier detection tasks in WAAM; 2) developing advanced optimization algorithms for hyper-parameters selection; 3) adjusting the model structure; 4) designing appropriate regularization terms for the model; and 5) deploying the proposed framework to some other outlier detection tasks.

Chapter 5

Outlier Detection under Limited Noisy Labeled Data: A Transformer-Embedded Contrastive Learning Framework

5.1 Motivation

DL has been extensively studied in recent years due to its remarkable ability to process large-scale data and its superior performance in feature extraction [110]. In DL, the quality of training data is crucial in determining the overall effectiveness of DL models. The accuracy of labels within the training data set is of paramount importance, as mislabeled data (commonly referred to as noisy labels) may introduce misleading information during training, thereby impairing the model's generalization ability and degrading overall performance in real-world applications.

To mitigate the adverse effects of noisy labels on DL models, LNL has recently been investigated extensively, leading to the proposal of numerous approaches [167]. Among existing LNL methods, sample selection, label correction, and robust training have been recognized as three prominent categories due to their satisfactory performance and ease of implementation. It should be noted that sample selection approaches may lead to the exclusion of potentially valuable data, thereby reducing the diversity and completeness of the training set. Meanwhile, label correction approaches can introduce further errors if the

corrected labels are inaccurate, ultimately misleading the training process and potentially reinforcing incorrect patterns in the model. Furthermore, each data point in an industrial time series contains essential intrinsic and structural information, which is critical for accurately identifying outliers, preserving temporal dependencies, and maintaining feature integrity in the presence of noisy labels [70].

To establish a reliable outlier detection model based on robust training approaches, the extraction of inherent patterns and features from industrial data is of critical importance. As a powerful family of feature extraction techniques, encoder-based methods have been widely employed in robust training-based LNL approaches due to their capability of capturing complex data representations [18, 174]. Among encoder-based methods, the Transformer has been particularly effective in handling time series data, primarily due to its self-attention mechanism. The Transformer encoder allows multivariate time series to be processed simultaneously while capturing dependencies across different dimensions, thereby eliminating the need for data pre-processing [189].

It is worth mentioning that the standard training scheme of the Transformer encoder typically follows a supervised approach, which is susceptible to the adverse effects of noisy labels. In contrast, self-supervised learning has emerged as a viable alternative, as it does not rely on label information during training. Among self-supervised learning techniques, contrastive learning has been particularly effective, as it aims to bring similar (i.e., positive) samples closer together while pushing dissimilar (i.e., negative) samples further apart. To date, contrastive learning has been widely adopted in robust training-based LNL approaches for constructing reliable encoders [80, 174].

Existing LNL approaches based on contrastive learning identify positive and negative sample pairs through data augmentation [74, 129]. Nevertheless, conventional data augmentation methods often fail to capture the intricate characteristics of time series data and tend to increase computational costs, particularly when handling large-scale data sets. To overcome these limitations, clustering algorithms, which group similar data points into distinct clusters based on inherent features, have been successfully employed for selecting positive and negative sample pairs in contrastive learning [40]. Recognized as a well-established clustering algorithm, the FCM algorithm enables data points to belong to multiple clusters with varying degrees of membership, leveraging fuzzy logic principles. Compared with hard

clustering algorithms, FCM is not only easier to implement but also offers greater flexibility in handling complex data distributions. However, its performance may degrade when applied to high-dimensional data, primarily due to the presence of sparse and redundant features extracted by the Transformer encoder [64].

A seemingly natural solution for clustering high-dimensional data is to adopt the uniform manifold approximation and projection (UMAP) method, which serves as a competitive manifold-learning-based dimensionality reduction technique, in order to obtain a low-dimensional embedding from the high-dimensional input [193]. The extracted low-dimensional representation can then serve as the input for the FCM algorithm, improving clustering effectiveness while mitigating the impact of irrelevant features. It is noticeable that clustering algorithms have been widely utilized in outlier detection as they are capable of uncovering the intrinsic structures and relationships within data, thereby providing valuable insights. As such, beyond determining positive and negative sample pairs, the partitions generated by the FCM algorithm can also be leveraged for model regularization, further enhancing the model's generalization ability and overall performance.

Motivated by the above discussions, this chapter proposes a novel Transformer-embedded LNL framework with fuzzy-clustering-assisted contrastive learning (TFCCL) for outlier detection in industrial time series data with noisy labels. Specifically, a fuzzy-clustering-assisted contrastive learning (FCCL) approach is developed to train the Transformer encoder, where the UMAP-based FCM (UFCM) algorithm is introduced to determine positive and negative sample pairs for contrastive learning based on clustering results. A dynamic two-stage scheme is formulated for training the outlier detector. In the first training stage, the Transformer encoder is pre-trained to enhance its feature extraction capability on industrial time series data through data reconstruction. In the second stage, the outlier detector is trained jointly with the Transformer encoder, while the UFCM algorithm is dynamically updated throughout the training process. Furthermore, a joint learning strategy is proposed for the second training stage, incorporating a label-consistency regularization term to minimize the discrepancy between outputs from the outlier detector and the UFCM algorithm, thereby improving the model's robustness against noisy labels.

The main contributions are summarized as follows.

1. A novel FCCL strategy is proposed for training the Transformer encoder, where the UFCM algorithm is designed to process high-dimensional features and identify positive and negative sample pairs for contrastive learning.

2. A dynamic two-stage training scheme is introduced for the outlier detector, where the Transformer encoder is pre-trained in the first stage via data reconstruction, and in the second stage, the outlier detector is jointly trained with the Transformer encoder.

3. A joint learning strategy is developed, incorporating a label-consistency regularization term to improve the model's generalization ability and robustness against noisy labels by minimizing the discrepancy between outputs from the outlier detector and the UFCM algorithm.

4. The proposed TFCCL framework is applied to outlier detection on limited WAAM data with noisy labels. Experimental results validate the effectiveness of the proposed TFCCL framework.

The remainder of this chapter is organized as follows. In Section 5.2, the proposed TFCCL framework and the training scheme are introduced. Experimental results for WAAM outlier detection under noisy labels are presented in Section 5.3. Finally, Section 5.4 concludes the chapter and provides discussions on potential future research directions.

5.2 Methodology

In the presence of noisy labels, the supervised training process of the Transformer encoder may be adversely affected, leading to degraded feature extraction performance. To mitigate this issue, an effective approach is to reduce the model's reliance on ground-truth labels. As a self-supervised learning method, contrastive learning leverages the intrinsic relationships among samples to learn meaningful representations without requiring label supervision, making it a valuable technique for improving model robustness. In recent years, contrastive learning has been widely employed to assist model training, particularly in LNL. For instance, in [174], a self-supervised contrastive learning approach has been introduced for LNL to demonstrate its effectiveness in training both the encoder and classifier despite label noise.

It is worth noting that, in conventional contrastive learning, positive and negative sample pairs for each data point are typically identified using data augmentation methods. However, this approach is not well-suited for large-scale time series data sets, as it may fail to capture complex temporal dependencies and could significantly increase computational costs. To overcome this limitation, a clustering-based approach for selecting positive and negative sample pairs presents a more efficient alternative. By leveraging the underlying structural information within the data, clustering enhances the correlation and diversity of positive and negative sample pairs while eliminating the need for additional label information, thereby improving the reliability of contrastive learning in noisy-label scenarios.

The FCM algorithm is a promising clustering technique for handling noisy and large-scale data, as it employs fuzzy logic to capture soft boundaries between clusters and assigns membership degrees to each data point across multiple clusters. Furthermore, the membership degrees generated by FCM can be used to regularize the classifier's output, thereby reducing the impact of label noise. To fully utilize both the inherent features and structural information in time series data for clustering, the Transformer encoder has recently been widely employed to extract effective feature representations [40]. However, directly using the high-dimensional features extracted by the Transformer encoder from large-scale time series data sets may degrade clustering performance. To overcome this challenge, applying dimensionality reduction techniques presents a natural and effective solution, as they can refine the extracted features by preserving essential structures while reducing computational complexity.

In this section, a novel TFCCL framework is developed for outlier detection on time series data with noisy labels. Specifically, an FCCL strategy is proposed to aid in training the Transformer encoder, where a UFCM algorithm is designed to process high-dimensional features and identify positive and negative sample pairs for contrastive learning. A dynamic two-stage scheme is introduced for training the outlier detector. In the first stage, the Transformer encoder is pre-trained to enhance feature extraction capability on industrial time series data through data reconstruction. In the second stage, the outlier detector is jointly trained with the Transformer encoder, while the UFCM algorithm is dynamically updated throughout the training process. Furthermore, a joint learning strategy is proposed to improve the classifier's robustness against noisy labels, incorporating a label-consistency

regularization term to minimize the discrepancy between the outputs of the outlier detector and the UFCM algorithm.

5.2.1 Overview of the TFCCL Framework

The developed TFCCL framework is illustrated in Fig. 5.1, which is designed to train a neural network-based classifier for time series outlier detection in the presence of noisy labels. The TFCCL framework consists of three key components: the Transformer encoder, the FCM clustering module, and the neural network-based classifier. As shown in Fig. 5.2, the Transformer encoder architecture comprises multiple identical encoder layers, each consisting of a multi-head attention mechanism and a convolutional module. Residual connections and layer normalization are applied to both the attention layer and the convolutional module to mitigate gradient vanishing. Furthermore, the neural network-based classifier and the decoder are implemented as multi-layer perceptrons. To enhance clustering performance, UMAP is employed for dimensionality reduction.

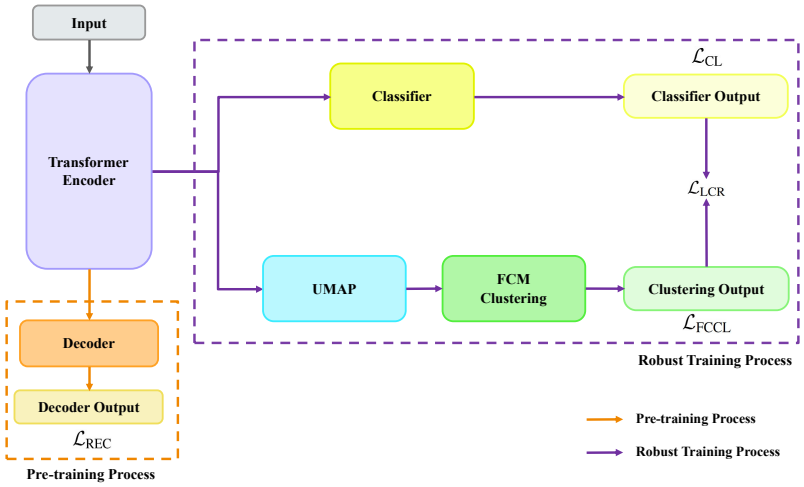


Fig. 5.1 The developed TFCCL framework, which consists of three key components: the Transformer encoder, the FCM clustering module, and the neural network-based classifier.

5.2.2 Loss Function

Classification Loss

The proposed TFCCL framework is designed to train a classifier for time series classification tasks. Although noisy labels may negatively impact supervised learning performance, incorporating the classification loss during training remains essential. The cross-entropy is used to compute the classification loss. The classification loss $\mathcal{L}_{\mathrm{CL}}$ is defined as:

$$\mathcal{L}_{\mathrm{CL}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{M}\tilde{y}_{ij}\log(\hat{y}_{ij}), \tag{5.1}$$

where B denotes the number of samples in the current mini-batch; M is the number of classes; $\tilde{y}_{ij}$ represents the jth component of the label for the ith sample; and $\hat{y}_{ij}$ is the jth component of the predicted output for the ith sample.

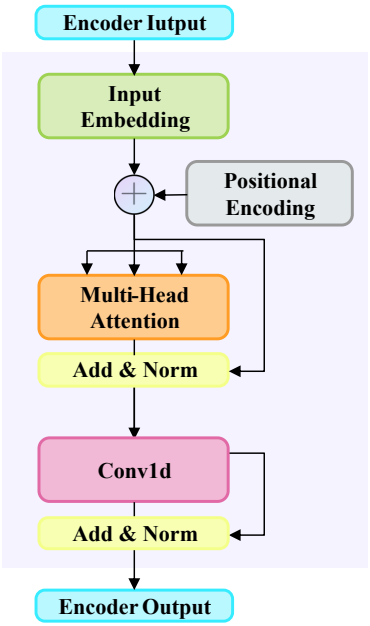


Fig. 5.2 The architecture of the Transformer encoder, which comprises multiple identical encoder layers. Each layer consists of a multi-head attention mechanism and a convolutional module.

Reconstruction Loss

The reconstruction loss is employed for pre-training the Transformer encoder. Specifically, the mean squared error (MSE) is used to calculate the reconstruction loss $\mathcal{L}_{\text{REC}}$, which is given by:

$$\mathcal{L}_{\text{REC}} = \frac{1}{B}\sum_{i=1}^{B} \|x_i - \hat{x}_i\|^2, \tag{5.2}$$

where B represents the number of samples in the current mini-batch; and x_i and $\hat{x}_i$ are the true value and the predicted value of the ith sample, respectively.

Contrastive Loss

The contrastive loss is utilized to train the Transformer encoder. A novel fuzzy-clustering-assisted contrastive loss $\mathcal{L}_{\text{FCCL}}$ is designed, which is shown as follows:

$$\mathcal{L}_{\text{FCCL}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\sum_{z_j \in \mathcal{P}(z_i)} \exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{z_k \in \mathcal{B}(z_i)} \exp(\text{sim}(z_i, z_k)/\tau)}, \tag{5.3}$$

where B is the number of samples in the current mini-batch; $z_i \in \mathbb{R}^d$ denotes the feature representation of the ith sample; $\mathcal{P}(z_i)$ is the set of positive feature representations corresponding to the ith sample; $\mathcal{B}(z_i)$ is the set of all feature representations in the current mini-batch except the feature representation of the ith sample; $\text{sim}(\cdot, \cdot)$ is the cosine similarity function between two samples; and τ represents the temperature parameter, which is a positive constant and controls the sharpness of the similarity scores.

The contrastive loss is formulated as a probability ratio that quantifies the likelihood of the model selecting the correct positive pair over all other sample pairs within the batch. The structure of the numerator and denominator reflects a probabilistic approach to pairwise discrimination, which emphasizes the relative similarity between positive and negative pairs. The exponential and logarithmic functions in (5.3) are combined in the contrastive loss to formulate a normalized, probabilistic objective based on the softmax and cross-entropy functions.

Remark 5.1. *Traditional contrastive learning methods typically identify positive and negative sample pairs for the ith sample using augmented data or ground-truth labels [129]. However,*

the large volume of data and the presence of noisy labels can interfere with the sample pairing process. To reduce computational costs and mitigate the effects of label noise, the FCCL strategy is introduced, where positive and negative sample pairs for the ith sample are determined based on clustering results. Specifically, samples belonging to the same cluster as the ith sample are considered positive, while those in different clusters are treated as negative. The FCCL strategy enables the Transformer encoder to be trained in an unsupervised manner, thereby avoiding the adverse impact of noisy labels and improving its ability to extract meaningful feature representations.

In the proposed TFCCL framework, the UFCM algorithm is introduced to assign positive and negative sample pairs based on the feature-space structure rather than relying directly on the given ground-truth labels. The contrastive learning process can therefore be guided by the intrinsic structure of the learned representations to mitigate the impact of the noisy labels. The potential transferability of this strategy lies in the fact that it is not restricted to the specific WAAM data used in this thesis. In principle, the same idea could be applied to other LNL scenarios, where reliable labels are difficult to obtain but meaningful sample relationships may still exist in the learned feature space. The strategy may also be combined with other feature extractors, as long as the extracted representations provide a suitable basis for clustering. Nevertheless, the performance of this strategy depends on the quality of the learned representations and the reliability of the clustering results. If the feature space has weak cluster structure, or if normal and outlier samples are highly overlapping, inaccurate positive and negative pairs may be generated and may affect the contrastive learning process.

Joint Learning Strategy

In the developed framework, a label-consistency regularization term is introduced to quantify the discrepancy between the output membership of the UFCM algorithm and the output probability of the classifier. Specifically, the KL divergence is employed to compute the label-consistency regularization term $\mathcal{L}_{\mathrm{LCR}}$, which is defined as:

$$\mathcal{L}_{\mathrm{LCR}} = D_{\mathrm{KL}}(P \parallel U) = \sum_{i=1}^{M} P(i) \log\left(\frac{P(i)}{U(i)}\right), \tag{5.4}$$

where M is the number of classes; P and U represent the distribution of the classifier's output probability and the distribution of the UFCM algorithm's output membership, respectively; and $P(i)$ and $U(i)$ represent the classifier's output probability for the ith class and the UFCM algorithm's output membership for the ith class, respectively.

Remark 5.2. *The output membership of the UFCM algorithm is independent of the noisy labels in the training data, making it a valuable reference for mitigating the impact of label noise on the classifier. It is important to note that KL divergence is asymmetric, meaning that it quantifies the information loss incurred when one probability distribution is used to approximate another, with the divergence value varying based on the direction of comparison. Therefore, the KL divergence $D_{KL}(P \parallel U)$ is computed instead of $D_{KL}(U \parallel P)$ as the label-consistency regularization term, aiming to better align the classifier's predictions with the UFCM algorithm's output membership.*

Overall Loss Function

The overall loss function of the developed model consists of two parts: pre-training loss $\mathcal{L}_{\text{PRE}}$ and joint training loss $\mathcal{L}$. The pre-training loss $\mathcal{L}_{\text{PRE}}$ is given as follows:

$$\mathcal{L}_{\text{PRE}} = \mathcal{L}_{\text{REC}}. \tag{5.5}$$

The joint training loss $\mathcal{L}$ is given by:

$$\mathcal{L} = \mu_1 \mathcal{L}_{\text{CL}} + \mu_2 \mathcal{L}_{\text{FCCL}} + \mu_3 \mathcal{L}_{\text{LCR}}, \tag{5.6}$$

where μ_1, μ_2 and μ_3 are three hyper-parameters to balance the weights among the classification loss, contrastive loss and label-consistency regularization term.

5.2.3 Dynamic Two-stage Training Scheme

Pre-training of the Transformer Encoder in the First Stage

The feature representations extracted from the training data by the Transformer encoder play a crucial role in the training process, as they are subsequently utilized by the classifier

for classification and by the UFCM algorithm for contrastive learning and regularization. Although the FCCL strategy is employed to train the Transformer encoder, its performance is highly dependent on the quality of initialization. To ensure the effectiveness of FCCL and further enhance the Transformer encoder's feature extraction capabilities, a decoder is employed to pre-train the encoder through input reconstruction. As illustrated in Fig. 5.1, during pre-training, only the encoder and decoder parameters are trainable, while all other parameters remain fixed. The reconstruction loss $\mathcal{L}_{\text{PRE}}$ is used as the loss function to guide the encoder and decoder in learning stable and robust feature representations. Pre-training allows the encoder to develop a deeper understanding of the underlying data structure, thereby improving training quality and overall model performance.

Dynamic Updating of the UFCM Algorithm in the Second Stage

The UMAP is employed before applying the FCM algorithm to reduce the dimensionality of the Transformer encoder output. UMAP is a non-linear dimensionality reduction method that can preserve local neighborhood structures while also maintaining useful global data structures. Compared with other dimensionality reduction techniques such as PCA and MDS, UMAP is considered more suitable for the proposed UFCM module because it is generally more scalable and is better able to preserve neighborhood relationships in high-dimensional feature spaces. Furthermore, cosine similarity is used to measure the similarity between data points and cluster centroids, facilitating effective clustering of feature representations.

It should be noted that the updating frequency of the UFCM algorithm during training is a critical hyper-parameter. A low updating frequency may cause the algorithm to become trapped in local optima, while a high updating frequency can lead to increased computational costs and unstable clustering performance. To address this issue, a dynamic updating strategy is introduced to adjust the UFCM algorithm's update frequency throughout the training process. Specifically, the update frequency is relatively high in the early training stages and gradually decreases as training progresses. The exact epoch for performing the $(k+1)$th UFCM update, denoted by e^{k+1}_{UFCM}, is calculated by:

$$e^{k+1}_{\text{UFCM}} = \frac{k * (k+1)}{2} * e^{k}_{\text{UFCM}}. \tag{5.7}$$

Training Procedure

The training procedure of the developed TFCCL framework is presented in Algorithm 4.

Algorithm 4 Training procedure of the developed TFCCL framework

Require: Noisy training data set $\tilde{\mathcal{D}} = \{(x_i, \tilde{y}_i)\}_{i=1}^{N}$, pre-training epochs E_P, training epochs E, first UFCM update epoch e^1_{UFCM}, batch size B, hyper-parameters μ_1, μ_2, μ_3
Ensure: Model parameters θ

```
Randomly initialize θ
Freeze all parameters in θ except those of the encoder and decoder
for e = 1 to E_P do
    Update θ (encoder and decoder) on D̃
end for
Unfreeze all parameters in θ
Initialize UFCM using encoder outputs on D̃
k ← 1
for e = 1 to E do
    for all mini-batches {(x_i, ỹ_i)}_{i=1}^{B} ⊂ D̃ do
        Compute the joint loss L according to (5.6)
        Update θ by minimizing L
    end for
    if e = e^k_UFCM then
        Update parameters of the UFCM algorithm
        Update e^{k+1}_UFCM according to (5.7)
        k ← k + 1
    end if
end for
```

5.3 Experiment Results

5.3.1 Experimental Setup

Data Pre-processing

In the experiments, each data set is segmented using a sliding window approach, where the window length is set to 10 and the stride to 5. Each segment is assigned a label ("Normal" or "Outlier"), determined based on the labels of individual instances within the segment. Specifically, a segment is labeled as "Normal" if all instances within it are labeled as "Normal"; otherwise, it is classified as an "Outlier". All labels are converted into a one-hot format for

ease of classification. Each segmented data set is then split into training and testing sets in a 7 : 3 ratio. Min-max normalization is subsequently applied to both sets to scale the data to a uniform range, further improving the performance of outlier detection.

Evaluation Metrics

To comprehensively assess the performance of the developed TFCCL framework in WAAM outlier detection, essential evaluation metrics such as accuracy, precision, recall, and F1-score are utilized. Each experiment is repeated five times, and the average values of these metrics are reported to ensure the consistency and reliability of the results.

Baselines

Several novel and representative approaches are selected for comparison. In addition, a vanilla outlier detection method is employed as a standard baseline for comparison. The details of each selected approach are listed as follows:

- **Co-teaching** [71], where two deep neural networks (DNNs) are trained simultaneously and select low-loss samples to update each other.
- **Co-teaching+** [221], where the disagreement-based updating strategy is combined with the original Co-teaching framework.
- **JoCoR** [195], where a co-regularization term is employed to help select clean samples for model training.
- **Co-learning** [174], where supervised learning and self-supervised learning are combined to handle noisy labels.
- **Standard Baseline**, where a classifier with the Transformer backbone is trained directly on the training data with the presence of noisy labels.

Implementation Details

For fair comparisons, all experiments are carried out on a server with an NVIDIA RTX A6000 GPU with 48 GB of memory and an Intel(R) Xeon(R) Silver 4214R CPU. All approaches are implemented using Ubuntu 20.04.6, Pytorch 2.5.1, Python 3.9.21, and CUDA 12.3.

It is worth mentioning that CNNs are employed as the original backbone networks in selected LNL approaches. As such, in the experiments, both the CNN and the Transformer architectures are employed as the backbone networks for selected approaches for comparisons to fully investigate the competitive performance of the TFCCL framework.

To verify the effectiveness of the FCM algorithm in the TFCCL framework, a comparison experiment is conducted. Specifically, the K-means algorithm and the hierarchical clustering algorithm are chosen as baseline clustering algorithms to compare against the FCM used in the TFCCL framework. Besides, the dynamic updating strategy of the proposed UFCM algorithm is evaluated through comparative experiments. To be specific, linear and logarithmic strategies are selected as baselines for comparison.

The Transformer encoder consists of 3 encoder layers. The CNN backbone consists of 5 convolutional layers and a fully connected layer, where each convolutional layer is followed by batch normalization. The decoder and the classifier are a 4-layer MLP and a 3-layer MLP, respectively. The hyper-parameters are selected via grid search on a held-out validation set. Each hyper-parameter is explored from a range of values and the combination that achieves the best validation performance is selected. To be specific, the temperature parameter τ in contrastive loss $\mathcal{L}_{\text{FCCL}}$ is set to 0.08. The hyper-parameters μ_1, μ_2 and μ_3 in the training loss $\mathcal{L}$ are set as 0.3, 0.8 and 0.8, respectively. In the experiments, the Adam optimizer is utilized for model training, where the learning rate is 0.0001. The pre-training and training epochs are 15 and 50, respectively. During the training process, the epoch number for the first UFCM update e^1_{UFCM} is set to 3.

The noise ratio λ is an important parameter in LNL, which indicates the ratio of the noisy labels in a data set. It should be noted that the outlier detection task is formulated as a binary classification problem with only two classes. As such, symmetrical label noise is employed in the experiment, where the label of each data point is corrupted with an equal probability defined by λ. For a comprehensive evaluation, multiple experiments are conducted on each data set with various values of λ (e.g., 30%, 40%, 50%, and 60%). It should be noted that in real-world industrial scenarios, the noise ratio usually remains below 50%. To verify the outlier detection performance of the TFCCL framework under severe label noise, the WAAM data sets with a noise ratio of 60% are utilized in the experiments.

5.3.2 Evaluation of WAAM Outlier Detection

The results of WAAM outlier detection on five data sets with noisy labels using selected approaches and the TFCCL framework are illustrated in Tables 5.1-5.5. According to the results, the TFCCL framework shows the leading performance on five data sets with five different noise ratios. The selected approaches using the Transformer backbone demonstrate better performance than those using the CNN backbone. A detailed evaluation of different noise ratios is presented below.

Table 5.1 Outlier detection performance of selected approaches on WAAM data set 1 with different noise ratios

Approaches	Accuracy				Precision				Recall				F1 Score			
	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
Co-teaching (CNN)	99.03	96.21	75.35	71.76	98.54	93.35	49.45	49.24	95.95	93.74	77.76	70.21	97.22	91.94	56.75	53.05
Co-teaching+ (CNN)	99.00	96.54	78.79	75.78	98.28	93.79	52.85	53.58	95.53	93.09	80.50	75.06	97.30	93.17	60.49	57.60
JoCoR (CNN)	98.82	98.54	83.70	14.44	99.12	98.01	53.40	14.59	94.11	93.66	70.32	81.19	96.53	95.73	60.35	24.73
Co-learning (CNN)	98.36	96.78	80.0	78.29	98.3	94.08	54.24	56.47	95.42	93.34	82.37	77.20	97.48	94.02	67.16	63.48
Co-teaching (Transformer)	99.05	97.58	77.46	76.28	98.87	94.31	46.47	63.39	95.68	92.54	81.13	72.74	97.25	93.05	57.53	55.92
Co-teaching+ (Transformer)	99.43	97.8	83.06	82.55	99.05	94.55	53.16	60.42	95.57	92.97	85.27	80.50	97.33	93.35	66.82	64.63
JoCoR (Transformer)	97.96	97.67	80.54	62.76	99.37	99.49	72.04	82.99	95.03	94.11	87.17	70.82	97.15	96.73	76.49	74.64
Co-learning (Transformer)	99.33	97.95	86.24	83.14	98.87	94.45	54.78	60.51	95.75	93.08	86.83	82.67	97.39	93.60	68.33	65.96
Standard (Transformer)	90.22	66.73	44.68	10.37	86.64	67.98	60.48	18.72	91.02	89.89	36.82	12.36	92.84	77.41	45.77	14.89
TFCCL	99.10	98.95	93.88	93.45	99.30	99.49	90.98	89.93	99.28	98.85	92.82	92.99	99.29	99.17	91.67	91.16

Table 5.2 Outlier detection performance of selected approaches on WAAM data set 2 with different noise ratios

Approaches	Accuracy				Precision				Recall				F1 Score			
	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
Co-teaching (CNN)	99.01	96.11	73.51	71.68	98.26	91.00	54.19	53.50	97.18	90.70	79.80	78.92	97.71	90.38	60.20	59.27
Co-teaching+ (CNN)	99.16	96.33	76.73	72.08	98.39	90.93	58.64	55.34	97.13	90.27	83.72	86.87	97.68	90.6	68.21	66.28
JoCoR (CNN)	98.95	98.94	80.21	15.89	97.56	97.42	50.43	16.18	97.17	97.27	94.04	79.69	97.36	97.34	65.41	26.88
Co-learning (CNN)	98.94	96.12	79.88	74.51	98.52	91.10	61.34	57.93	97.05	90.09	85.47	88.43	97.79	90.84	70.12	68.14
Co-teaching (Transformer)	99.17	96.92	76.48	68.53	98.34	94.31	60.60	61.36	97.46	90.11	84.65	83.08	97.89	92.05	65.02	64.73
Co-teaching+ (Transformer)	99.02	97.15	81.60	71.86	98.22	94.71	65.83	62.50	96.93	90.36	89.22	85.55	97.20	91.25	67.50	69.20
JoCoR (Transformer)	98.42	98.29	75.42	70.37	99.32	98.41	70.64	69.06	96.88	97.26	82.96	60.85	98.08	97.83	72.55	62.09
Co-learning (Transformer)	99.20	97.52	86.68	74.93	98.10	93.71	69.40	67.06	97.53	90.76	94.55	87.38	97.60	92.22	80.38	77.21
Standard (Transformer)	88.27	64.20	30.26	8.81	83.25	80.83	42.07	11.45	96.17	50.64	51.87	8.37	90.86	62.27	46.46	9.67
TFCCL	98.84	98.66	91.19	89.02	98.98	98.83	91.60	92.18	99.28	98.85	92.82	92.99	99.01	98.38	89.23	86.09

Table 5.3 Outlier detection performance of selected approaches on WAAM data set 3 with different noise ratios

Approaches	Accuracy				Precision				Recall				F1 Score			
	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
Co-teaching (CNN)	99.05	96.27	76.62	71.31	98.05	89.72	54.44	48.97	96.94	91.26	71.08	67.39	97.47	90.36	56.16	51.64
Co-teaching+ (CNN)	98.90	96.57	80.51	78.21	98.17	89.47	58.13	53.41	96.84	91.66	77.69	75.88	97.50	90.55	66.48	63.58
JoCoR (CNN)	99.14	98.99	81.32	15.16	98.40	98.65	50.54	15.09	97.02	96.00	94.09	77.91	97.70	97.30	65.57	25.27
Co-learning (CNN)	98.78	96.87	84.23	81.06	98.02	89.92	62.44	56.53	96.96	91.36	81.23	78.66	97.43	90.63	70.65	65.75
Co-teaching (Transformer)	99.13	96.56	85.88	82.39	98.64	91.51	61.73	53.58	96.74	90.31	72.93	62.38	97.67	90.86	65.55	57.55
Co-teaching+ (Transformer)	98.98	96.86	86.08	85.69	98.54	91.91	57.23	56.68	96.12	90.01	77.53	65.88	96.84	90.78	67.03	59.58
JoCoR (Transformer)	98.49	98.25	79.27	63.32	99.25	98.90	70.19	74.33	96.61	95.73	86.75	78.55	97.91	97.29	75.52	66.41
Co-learning (Transformer)	98.83	96.56	87.28	85.39	98.36	92.16	71.73	59.88	96.82	89.46	82.33	70.28	97.73	90.60	73.53	61.58
Standard (Transformer)	77.04	58.59	35.88	9.01	72.47	61.71	47.15	8.88	93.98	82.92	50.58	5.46	84.04	70.76	48.80	6.76
TFCCL	98.91	98.78	93.71	91.95	99.55	98.23	94.35	92.24	98.65	98.40	90.16	87.50	99.10	98.31	91.53	89.45

Table 5.4 Outlier detection performance of selected approaches on WAAM data set 4 with different noise ratios

Approaches	Accuracy				Precision				Recall				F1 Score			
	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
Co-teaching (CNN)	96.17	90.82	61.97	60.01	81.41	66.76	35.72	31.74	98.07	83.91	75.61	72.90	88.86	73.58	44.60	39.87
Co-teaching+ (CNN)	96.32	90.52	67.02	62.33	81.29	67.16	38.47	37.80	97.92	84.21	80.11	75.93	88.89	74.71	53.12	50.17
JoCoR (CNN)	98.27	96.82	73.63	12.18	91.18	83.70	37.76	11.79	98.40	99.08	99.89	74.72	94.64	90.66	54.55	20.37
Co-learning (CNN)	96.20	90.82	70.97	65.01	81.17	67.41	43.22	39.54	98.03	83.91	84.31	78.40	88.71	75.01	56.92	53.27
Co-teaching (Transformer)	96.54	90.77	66.33	63.53	83.78	65.71	37.89	30.87	96.68	86.47	83.96	79.48	89.68	74.35	49.32	43.00
Co-teaching+ (Transformer)	96.39	91.07	71.83	67.03	83.96	65.31	42.09	33.87	96.56	86.97	88.76	81.98	89.82	74.60	57.10	47.94
JoCoR (Transformer)	96.41	96.55	71.65	62.72	96.42	94.09	61.44	53.52	94.91	96.85	84.22	96.12	95.64	95.40	68.03	63.52
Co-learning (Transformer)	96.20	91.37	77.03	71.13	83.80	64.91	47.39	37.57	96.45	87.37	93.56	85.38	89.65	74.90	62.70	52.14
Standard (Transformer)	74.32	61.58	15.75	6.09	71.92	64.56	31.52	15.11	75.78	41.60	23.94	9.26	83.67	58.75	27.21	11.48
TFCCL	99.03	98.45	95.48	93.98	99.02	98.57	93.14	89.62	99.51	98.00	94.15	94.37	99.26	98.27	93.47	91.48

Table 5.5 Outlier detection performance of selected approaches on WAAM data set 5 with different noise ratios

Approaches	Accuracy				Precision				Recall				F1 Score			
	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
Co-teaching (CNN)	98.80	96.56	73.93	68.36	96.91	89.33	44.45	44.42	95.36	89.94	69.53	69.04	96.10	89.15	49.43	47.22
Co-teaching+ (CNN)	98.92	96.80	77.62	75.12	96.74	89.80	49.00	46.67	95.51	89.59	74.63	72.59	95.70	89.66	55.73	54.62
JoCoR (CNN)	99.04	98.78	82.40	12.54	96.06	96.75	47.45	12.21	97.86	95.37	95.59	77.20	96.93	96.04	63.04	21.08
Co-learning (CNN)	96.17	90.87	71.09	65.11	81.41	66.86	43.17	39.29	97.74	84.61	80.89	79.20	88.90	74.68	57.01	52.55
Co-teaching (Transformer)	98.83	97.21	79.10	76.99	98.76	92.46	44.61	39.89	97.65	89.97	69.90	70.27	98.20	90.94	52.18	51.10
Co-teaching+ (Transformer)	98.68	97.56	84.33	80.13	98.64	92.91	50.28	42.27	97.81	89.72	74.79	73.60	98.12	90.64	56.34	53.25
JoCoR (Transformer)	98.14	97.85	78.64	64.41	99.39	98.36	69.62	64.99	94.92	95.72	87.96	83.00	97.10	97.02	72.97	65.66
Co-learning (Transformer)	98.53	97.86	86.53	84.23	98.52	92.61	55.03	45.72	97.26	90.12	79.29	77.80	97.63	90.39	66.48	62.19
Standard (Transformer)	79.42	73.21	47.45	7.07	84.60	83.31	69.09	14.23	84.81	75.20	39.43	7.62	84.71	79.05	50.21	9.92
TFCCL	98.95	98.42	95.20	92.22	99.42	98.06	90.10	87.58	99.01	99.62	96.51	89.53	99.21	98.83	93.07	88.37

30% **and** 40% **Noisy Labels**

It can be found that all selected approaches with different backbones as well as the TFCCL framework all achieve satisfactory performance on five data sets. The standard approach also shows acceptable performance under a 30% noise ratio. As indicated by the results, the TFCCL framework outperforms the other approaches on most data sets in terms of accuracy, precision, recall, and F1 score. Notably, on data set 2 with a 30% noise ratio, Co-teaching, Co-teaching+ and Co-learning (with both backbones) all achieve higher accuracy than TFCCL. data sets 1 and 2 with noise ratios of 30%, Co-learning-Transformer has higher accuracy than TFCCL. On data sets 2, 3, and 5 with noise ratios of 30% and 40%, JoCoR-CNN has higher accuracy than TFCCL as well. Nevertheless, TFCCL attains the highest performance on the remaining metrics in these scenarios, which demonstrates the overall effectiveness of TFCCL in outlier detection with a relatively small number of noisy labels.

50% **Noisy Labels**

In the scenario with a 50% noise ratio, the TFCCL remains superior in performance, whereas the performance of all selected approaches decreases significantly on all five data sets. Specifically, the accuracy and the F1 score of the selected approaches are no more than 88% and 78%, respectively. For TFCCL, the accuracy and the F1 score across five data sets exceed 91% and 88%, respectively, which shows the robustness of TFCCL against a moderate number of noisy labels.

60% **Noisy Labels**

In situations with the noise ratio of 60%, the baseline approach and JoCoR-CNN are affected by severe label noise and are no longer capable of handling outlier detection tasks. The performance of the other selected approaches is also not effective enough. It should be noted that TFCCL still achieves satisfactory performance across five data sets, with each data set's accuracy above 89% and F1 score above 86%. The results indicate the competitive performance of the TFCCL in outlier detection with a large number of noisy labels.

5.3.3 Comparison Experiments

Two comparison experiments are conducted. Specifically, the first experiment aims to verify the effectiveness of the selection of the FCM algorithm. The results are illustrated in Table 5.6. According to the results, the FCM algorithm shows competitive performance across all five data sets under different noise ratios compared with other hard clustering algorithms, demonstrating its robustness and adaptability in the presence of label noise. The second experiment aims to verify the effectiveness of the proposed dynamic updating strategy for the UFCM algorithm. As shown in Table 5.7, the developed dynamic updating strategy achieves the best results compared with other updating strategies.

Table 5.6 Comparison of different clustering algorithms within the TFCCL framework on WAAM data sets under varying noise ratios

Approaches	Metrics (%)	Data Set 1				Data Set 2				Data Set 3				Data Set 4				Data Set 5			
		30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
TFCCL-Kmeans	Accuracy	98.12	96.22	89.02	84.23	97.84	95.68	86.43	80.48	97.97	96.09	88.13	82.47	98.07	95.28	89.52	83.08	97.91	95.42	89.03	82.91
	Precision	98.29	96.42	85.08	80.11	97.72	95.01	85.59	78.52	98.28	95.01	88.31	80.88	97.81	95.11	86.48	78.59	98.12	94.48	83.28	74.01
	Recall	98.34	96.09	86.12	77.68	97.81	94.01	79.02	70.01	96.68	94.32	82.52	74.48	98.42	94.02	86.12	79.79	97.62	96.58	87.92	71.01
	F1 Score	98.30	96.26	85.59	78.84	97.76	94.48	82.12	73.83	97.48	94.62	85.31	77.41	98.11	94.53	86.29	79.26	97.84	95.52	85.48	72.31
TFCCL-Hierarchical	Accuracy	95.58	90.97	82.33	75.47	94.83	90.28	80.24	72.36	95.13	90.26	81.16	73.84	95.18	89.62	80.76	72.28	94.68	89.78	80.08	71.96
	Precision	95.76	91.04	78.86	72.84	94.42	89.27	77.47	69.87	95.33	89.32	80.07	71.94	94.57	88.84	77.94	68.57	94.64	87.83	75.08	65.96
	Recall	95.27	90.23	78.16	70.48	94.34	87.77	70.77	62.27	94.02	87.93	74.16	65.47	95.04	86.28	75.36	66.07	93.78	88.03	79.06	63.44
	F1 Score	95.51	90.58	78.48	71.57	94.37	88.53	73.97	65.58	94.61	88.55	76.97	68.63	94.76	87.57	76.58	67.18	94.18	87.46	76.87	64.52
TFCCL-FCM	Accuracy	99.10	98.95	93.88	93.45	98.84	98.66	91.19	89.02	98.91	98.78	93.71	91.95	99.03	98.45	95.48	93.98	98.95	98.42	95.20	92.22
	Precision	99.30	99.49	90.98	89.93	98.98	98.83	91.60	92.18	99.55	98.23	94.35	92.24	99.02	98.57	93.14	89.62	99.42	98.06	90.10	87.58
	Recall	99.28	98.85	92.82	92.99	99.04	97.94	87.65	82.10	98.65	98.40	90.16	87.50	99.51	98.00	94.15	94.37	99.01	99.62	96.51	89.53
	F1 Score	99.29	99.17	91.67	91.16	99.01	98.38	89.23	86.09	99.10	98.31	91.53	89.45	99.26	98.27	93.47	91.48	99.21	98.83	93.07	88.37

Table 5.7 Comparison of different UFCM updating strategies within the TFCCL framework on WAAM data sets under varying noise ratios

Approaches	Metrics (%)	Data Set 1				Data Set 2				Data Set 3				Data Set 4				Data Set 5			
		30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
TFCCL (Linear)	Accuracy	98.26	94.87	88.65	73.00	95.70	91.78	85.43	76.62	95.57	89.94	85.27	84.36	92.80	90.37	83.99	77.69	95.80	92.76	86.34	80.14
	Precision	97.78	92.50	90.02	71.98	98.23	91.67	82.73	78.79	94.07	93.50	84.98	88.31	92.71	94.08	89.13	90.84	99.34	98.81	77.78	87.80
	Recall	99.54	97.76	89.18	97.62	92.63	94.53	96.52	84.42	98.90	89.57	93.82	85.09	96.66	91.10	86.47	75.60	93.79	90.32	96.23	86.91
	F1 Score	98.67	96.10	89.97	83.27	96.17	93.07	89.58	82.02	96.44	91.50	88.81	86.40	94.66	92.57	88.22	87.25	96.77	94.36	88.09	87.64
TFCCL (Logarithmic)	Accuracy	96.93	92.56	86.20	70.64	94.47	89.13	83.14	74.54	94.06	87.58	81.70	82.28	90.94	88.17	81.00	75.41	94.15	90.48	78.07	77.53
	Precision	96.65	90.00	87.83	70.42	96.78	88.81	80.03	76.95	92.54	91.31	82.13	86.46	91.40	91.72	86.79	88.57	97.55	96.57	75.33	85.58
	Recall	98.03	95.67	86.28	95.30	90.78	92.13	93.64	82.83	97.55	87.41	91.35	83.37	95.62	88.37	84.15	73.93	92.57	87.93	94.20	85.29
	F1 Score	97.10	93.21	87.81	81.36	94.92	90.50	86.69	80.03	94.82	89.16	86.19	84.83	93.50	89.77	85.82	85.32	94.95	91.70	86.62	85.53
TFCCL (Dynamic)	Accuracy	99.10	98.95	93.88	93.45	98.84	98.66	91.19	89.02	98.91	98.78	93.71	91.95	99.03	98.45	95.48	93.98	98.95	98.42	95.20	92.22
	Precision	99.30	99.49	90.98	89.93	98.98	98.83	91.60	92.18	99.55	98.23	94.35	92.24	99.02	98.57	93.14	89.62	99.42	98.06	90.10	87.58
	Recall	99.28	98.85	92.82	92.99	99.04	97.94	87.65	82.10	98.65	98.40	90.16	87.50	99.51	98.00	94.15	94.37	99.01	99.62	96.51	89.53
	F1 Score	99.29	99.17	91.67	91.16	99.01	98.38	89.23	86.09	99.10	98.31	91.53	89.45	99.26	98.27	93.47	91.48	99.21	98.83	93.07	88.37

5.3.4 Ablation Study

In the experiments, a detailed ablation study is carried out to verify the effectiveness of the main components in TFCCL under various noise ratios. To be specific, the effects of the pre-training, the FCCL strategy, the UMAP and the dynamic UFCM updating strategy in TFCCL are verified. The results of the ablation study are illustrated in Table 5.8. Based on the results, detailed analyses are given: 1) The pre-training helps the Transformer encoder exploit the internal structure and capture useful feature representations. When trained directly for classification without pre-training, the average performance of the TFCCL framework declines with increasing noise ratios. 2) The FCCL strategy significantly improves the performance of outlier detection with noisy labels by utilizing the intrinsic features of data. Employing the UMAP is able to enhance the performance of the FCCL strategy when dealing with severe label noise. 3) The dynamic UFCM updating strategy effectively stabilizes the training process and the final performance. 4) Each of the main components plays a critical role in the TFCCL framework. The best performance of TFCCL can be achieved only if all components work together.

Table 5.8 Results of ablation study on WAAM data sets with different noise ratios

	Metrics (%)	Data Set 1				Data Set 2				Data Set 3				Data Set 4				Data Set 5			
		30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%	30%	40%	50%	60%
w/o Pre-training	Accuracy	95.17	93.96	80.26	71.22	95.75	94.02	77.57	63.44	96.75	94.52	86.50	76.94	95.48	94.93	86.85	80.80	95.70	93.98	88.49	82.74
	Precision	99.84	96.42	82.23	70.66	95.77	92.07	75.03	61.49	94.95	91.71	90.85	75.34	93.65	92.85	89.35	83.48	95.84	92.63	92.85	96.35
	Recall	92.38	93.97	87.85	93.43	97.00	98.19	92.24	99.85	99.94	99.97	86.35	91.92	99.90	99.87	90.85	88.29	97.85	98.91	89.77	74.31
	F1 Score	96.04	95.18	84.95	80.46	96.38	95.03	82.75	76.11	97.38	95.66	88.54	82.81	96.67	96.29	90.09	85.81	96.83	95.67	91.29	85.26
w/o FCCL	Accuracy	92.74	92.18	46.50	11.99	94.97	94.01	72.39	27.62	95.29	93.06	66.07	34.54	88.13	85.02	52.50	46.65	94.00	93.16	41.72	20.68
	Precision	95.46	97.20	55.76	24.68	93.48	93.43	61.42	34.59	97.34	96.21	80.14	41.76	84.71	81.45	65.41	39.04	94.29	95.44	51.29	38.66
	Recall	91.95	90.27	50.66	18.91	98.22	96.53	59.10	27.03	94.79	92.15	56.25	21.11	98.04	93.54	61.84	18.90	96.94	94.33	56.15	30.78
	F1 Score	93.68	93.60	52.98	21.41	95.79	94.95	64.10	30.34	96.05	94.13	65.42	28.04	91.72	89.78	63.53	31.79	95.59	94.88	52.80	34.27
w/o UMAP	Accuracy	98.85	95.61	64.85	57.82	96.79	93.59	69.91	56.75	96.29	88.66	76.97	64.64	93.47	90.97	67.91	65.31	96.36	93.40	65.49	59.80
	Precision	99.04	93.90	66.89	69.83	96.53	94.21	70.33	62.12	97.60	92.88	76.06	65.28	90.97	98.63	70.38	59.61	95.94	95.95	71.53	71.87
	Recall	97.80	99.54	88.26	58.97	98.02	94.84	83.76	66.27	96.23	84.19	84.01	88.63	96.45	87.49	88.42	69.73	98.76	94.15	80.81	66.01
	F1 Score	98.41	96.64	76.08	63.94	97.27	94.52	76.46	64.13	96.91	91.42	82.54	75.18	95.27	92.73	78.38	64.79	97.33	95.04	75.89	68.81
w/o Dynamic UFCM Updating Strategy	Accuracy	98.28	94.83	84.76	70.83	95.70	91.75	84.32	75.20	95.58	89.90	82.62	81.92	92.81	90.34	82.13	76.29	95.78	92.72	79.05	78.57
	Precision	97.79	92.48	88.03	69.64	98.22	91.63	81.46	76.59	94.08	93.48	83.68	86.47	92.71	94.05	87.39	88.82	99.95	98.78	76.23	83.53
	Recall	99.54	99.97	87.93	95.76	92.62	94.49	94.66	82.80	98.90	89.53	92.19	83.08	96.66	91.08	85.11	73.51	93.77	90.28	94.31	84.84
	F1 Score	98.66	96.08	87.98	80.64	96.17	93.04	87.57	79.57	96.43	91.46	87.28	84.74	94.65	92.54	86.23	84.73	96.76	94.34	86.51	84.18
TFCCL	Accuracy	99.10	98.95	93.88	93.45	98.84	98.66	91.19	89.02	98.91	98.78	93.71	91.95	99.03	98.45	95.48	93.98	98.95	98.42	95.20	92.22
	Precision	99.30	99.49	90.98	89.93	98.98	98.83	91.60	92.18	99.55	98.23	94.35	92.24	99.02	98.57	93.14	89.62	99.42	98.06	90.10	87.58
	Recall	99.28	98.85	92.82	92.99	99.04	97.94	87.65	82.10	98.65	98.40	90.16	87.50	99.51	98.00	94.15	94.37	99.01	99.62	96.51	89.53
	F1 Score	99.29	99.17	91.67	91.16	99.01	98.38	89.23	86.09	99.10	98.31	91.53	89.45	99.26	98.27	93.47	91.48	99.21	98.83	93.07	88.37

5.4 Conclusion

In this chapter, a new TFCCL framework has been developed for LNL in outlier detection on limited WAAM data. Specifically, a novel FCCL strategy has been introduced to update the Transformer encoder, where a UFCM algorithm has been proposed for identifying positive and negative sample pairs. A dynamic two-stage training scheme has been designed to train the outlier detector, incorporating a joint learning strategy to enhance its robustness against noisy labels. Moreover, a dynamic updating strategy has been employed to update the UFCM algorithm throughout the training process. To further improve model reliability, a label-consistency regularization term has been introduced to minimize the discrepancy between the outputs of the outlier detector and the UFCM algorithm. Experimental results have demonstrated the effectiveness of the developed TFCCL framework. Future research directions can be summarized as follows: 1) utilizing optimization techniques for automatic hyper-parameter selection; 2) extending the developed TFCCL framework to multi-class outlier detection tasks; 3) modifying the Transformer encoder architecture to enhance feature extraction performance; and 4) refining the FCCL strategy by adjusting the contrastive loss and improving the clustering algorithm.

Chapter 6

Outlier Detection under Sufficient Noisy Labeled Data: A Transfer Learning Assisted Role-Differentiated Approach

6.1 Motivation

In many real-world applications, due to the unexpected errors in automated labeling tools and improper operation by human annotators, the obtained training data may contain noisy labels, which would corrupt the training process and mislead the mapping between the feature space and the target space, thereby affecting the performance and reliability of the trained DL model. Serving as a prominent class of LNL approaches, sample selection focuses on identifying clean samples for model training by using specific metrics (e.g., the training loss and similarity metrics). Compared with other classes of LNL approaches such as robust training and label correction, sample selection is capable of decreasing the negative influences from noisy samples and avoiding the misleading effects arising from label correction errors. A major advantage of sample selection is its simplicity and ease of implementation, which leads to the widespread adoption of sample selection in various applications for handling label noise in recent years [26, 195].

Sample selection may suffer from incorrect selection, where clean samples are misclassified as noisy samples, especially when applied to large or complex data sets with label

noise. During the model training process, noisy samples would cause error accumulation and introduce model biases, which result in degraded model performance. Recently, a number of sample selection variants have been proposed to improve the selection performance by introducing advanced training strategies or developing new network architectures [71, 101]. For instance, in [71], a well-known LNL approach named Co-teaching has been proposed, where the Siamese network structure is employed. Particularly, the potential clean samples identified by each network are utilized to update its peer network. The Siamese network structure has been widely adopted in sample selection by leveraging the unique learning capability of each network, offering complementary views on samples and improving selection accuracy. Unlike single-network methods, the Siamese-network-based sample selection (SNSS) approach leverages model diversity to reduce confirmation bias and improve robustness against label noise. Apart from the Co-teaching approach, various SNSS approaches, including Co-teaching+ [221], JoCoR [195], and DivideMix [101] have also achieved notable success in tackling the noisy label problem.

To reduce the computational cost, many SNSS methods employ identical and simple subnetworks, which would unfortunately constrain the model's learning and representation capabilities, especially when handling complex data sets. Moreover, identical architectures sometimes fail to provide diverse multi-view representations of data patterns, thereby limiting the network complementarity and impairing the model's ability to discriminate clean and noisy samples. Inspired by coordination strategies in multi-agent systems, the leader-follower framework proposed in [39] is a seemingly promising solution to break symmetry among subnetworks and promote diverse yet coordinated model learning behaviors. In the leader-follower framework, role differentiation among subnetworks is achieved, where the leader network incorporates additional domain-specific knowledge (which is imparted to the follower networks to guide their training), thereby improving the overall capacity to identify and select clean samples.

The clean sample selection ratio (CSSR) plays a significant role in sample selection, which brings a trade-off between the inclusion of clean samples and the exclusion of noisy samples. Many existing sample selection approaches use a constant CSSR based on experimental experience, which demands domain knowledge and cannot guarantee consistent performance on various data sets with multi-modal inputs or partially labeled data. In Co-teaching, a

dynamic selection strategy has been designed, where the CSSR decreases according to the stage of training to gradually exclude noisy data and retain clean samples. Unfortunately, the dynamic selection strategy in Co-teaching overlooks the variability in the training process across different data sets, potentially leading to premature exclusion of valuable samples or delayed removal of noisy ones. To enhance selection efficiency and fully utilize potentially clean samples, a reasonable idea is to adaptively adjust the CSSR based on both the training state information and the loss characteristics.

Motivated by the above discussions, in this chapter, a role-differentiated LNL (RD-LNL) approach is put forward for WAAM outlier detection under label noise. A leader-follower-inspired sample selection (LFSS) strategy is proposed for identifying potential clean samples. Then, the chosen clean samples are fed into DNNs for outlier detection. In particular, three DNNs are trained collaboratively. Here, a leader network is pre-trained on clean auxiliary data and guides the joint training of two follower networks by fully leveraging its domain knowledge. A selection metric is introduced that integrates both the training dynamics and the prediction divergence across the DNNs. According to the selection metric, an adaptive selection scheme is proposed to adaptively adjust the CSSR during the training process.

The main contributions lie in the following threefold:

1. An LFSS strategy is proposed for training the outlier detection model with selected potential clean samples, where a leader network and two follower networks are trained jointly for sample selection by leveraging the domain knowledge of the leader network.

2. An adaptive selection scheme is introduced to adjust the CSSR adaptively through the model training process, which makes use of the loss characteristics of the training samples.

3. The developed RD-LNL approach is successfully applied to a real-world industrial outlier detection application using data collected from a WAAM pilot line. Experimental results verify the practical utility of the developed RD-LNL approach for handling data with label noise.

The remaining sections of this chapter are organized as follows. In Section 6.2, the developed RD-LNL approach is introduced. Experimental results for WAAM outlier detection

under label noise are presented in Section 6.3. Finally, conclusions are presented in Section 6.4.

6.2 Methodology

Many SNSS approaches use identical network architectures with random initialization to encourage diverse learning behaviors and reduce confirmation bias [71, 195]. During the model training, samples are selected based on the prediction consensus of the networks. Nevertheless, identical architectures with random initialization may still lack sufficient representational diversity, thereby decreasing the selection quality. As a role-differentiated framework, the leader-follower framework is capable of integrating domain-specific knowledge and model diversity to support effective sample selection [39]. Besides, the variation between the leader and follower networks brings multiple perspectives of the learning process, which helps reduce confirmation bias and improve the reliability of selected samples.

As a critical factor, the CSSR directly determines how many samples are used for model training at each training epoch, and thus governs the balance between preserving clean samples and excluding noisy samples. It is worth mentioning that most existing approaches adopt the fixed selection ratio or the dynamic selection ratio based on training epochs. Such selection ratios may overlook the fact that training dynamics vary significantly across data sets, and thus decrease the generalization ability of the approaches. To address the above mentioned limitation, a seemingly reasonable solution is to consider both the training epochs and the sample loss characteristics to adaptively select clean samples during training.

In this chapter, an RD-LNL approach is developed for industrial outlier detection with noisy labels. An LFSS strategy is introduced to select clean samples to train the outlier detector. To be specific, a pre-trained network is employed as the leader network and is utilized to guide the joint training process with two follower networks. A selection metric is designed for sample selection by integrating both training behavior and prediction divergence across the networks. Based on the selection metric, an adaptive selection scheme is put forward to improve the quality of selected samples by adaptively adjusting the CSSR during the training process.

6.2.1 Overview of the RD-LNL Approach

The developed RD-LNL approach is shown in Fig. 6.1. The RD-LNL approach consists of three networks: Network 3 serves as the leader, while Network 1 and Network 2 act as followers.

The Transformer is utilized as the backbone of the leader network in order to capture long-range dependencies within the time series and improve the generalization capability. The Transformer is composed of several identical encoder layers, where each layer includes a multi-head attention mechanism and a lightweight 1D convolutional module to enhance local temporal feature extraction.

The CNN serves as the backbone of the follower networks, enabling efficient training and effective extraction of local features. Each CNN consists of multiple standard convolutional modules, each containing a convolutional layer followed by batch normalization. Residual connections are employed in both the encoder layers and the convolutional modules to facilitate gradient flow and stabilize training. Three identical multi-layer perceptrons are employed as the classifiers, each attached to the output of one network.

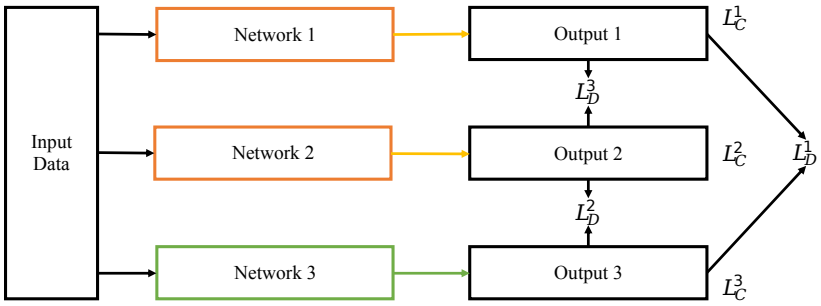


Fig. 6.1 The proposed RD-LNL approach

6.2.2 Loss Function

Classification Loss

The cross-entropy loss is applied to compute the classification losses for Network 1, Network 2, and Network 3 (i.e., L_C^1, L_C^2, and L_C^3, respectively), which are defined as follows:

$$L_C^1 = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(\hat{y}_{ij}^1), \tag{6.1}$$

$$L_C^2 = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(\hat{y}_{ij}^2), \tag{6.2}$$

$$L_C^3 = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(\hat{y}_{ij}^3), \tag{6.3}$$

where N indicates the sample size of current mini-batch; M denotes the total number of classes; y_{ij} is the jth element of the label vector corresponding to the ith sample; and $\hat{y}_{ij}^1$, $\hat{y}_{ij}^2$, and $\hat{y}_{ij}^3$ are the jth element of the predicted output for the ith sample of Network 1, Network 2, and Network 3, respectively.

Prediction Divergence

To mitigate the confirmation bias of each network and enhance collaborative sample selection performance, the prediction divergences between all pairs of networks are calculated, where the KL divergence and Jensen–Shannon (JS) divergence are employed. Specifically, the KL divergence is utilized to measure the output discrepancies between the leader network and each of the two follower networks (i.e., L_D^{13} and L_D^{23}), and the JS divergence is applied to calculate the output discrepancy between two follower networks (i.e., L_D^{12}). The prediction divergences between all pairs of networks are calculated by:

$$L_D^{13} = \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(P_i^1 \parallel P_i^3) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} P_{ij}^1 \log\left(\frac{P_{ij}^1}{P_{ij}^3}\right), \tag{6.4}$$

$$L_D^{23} = \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(P_i^2 \parallel P_i^3) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} P_{ij}^2 \log\left(\frac{P_{ij}^2}{P_{ij}^3}\right), \tag{6.5}$$

$$\begin{aligned} L_D^{12} &= \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{JS}}(P_i^1 \parallel P_i^2) \\ &= \frac{1}{2}\left[\sum_{i=1}^{N} D_{\mathrm{KL}}(P_i^1 \parallel Q_i) + \sum_{i=1}^{N} D_{\mathrm{KL}}(P_i^2 \parallel Q_i)\right], \end{aligned} \tag{6.6}$$

where $Q_i = \frac{1}{2}(P_i^1 + P_i^2)$ is the average distribution of the two follower networks for the ith sample; L_D^{13} and L_D^{23} denote the prediction discrepancy between Network 1 and Network 3, and between Network 2 and Network 3, respectively; L_D^{12} represents the prediction discrepancy between Network 1 and Network 2; P_i^1, P_i^2 and P_i^3 are the distribution of the output probability

of the ith sample from Network 1, Network 2, and Network 3, respectively; and P_{ij}^1, P_{ij}^2 and P_{ij}^3 are the output probability for the ith sample with the jth class of Network 1, Network 2, and Network 3, respectively.

Remark 6.1. *Network 3, denoted as the leader network, is obtained via pre-training and fine-tuning. Compared with the follower networks that are trained directly on the noisy labeled data set, the leader network has better feature representation and discriminative capabilities. It is worth mentioning that KL divergence measures the information loss introduced when approximating one probability distribution with another, making the direction of comparison a critical factor due to its asymmetric nature. To guide the follower networks toward the prediction behavior of the leader network, the prediction divergences L_D^{13} and L_D^{23} are calculated according to (6.4) and (6.5). Furthermore, the JS divergence is applied in calculating L_D^{12} based on (6.6) due to its symmetric and bounded nature to ensure balanced comparison of the output prediction between the two follower networks.*

Overall Loss Function

The overall loss functions of the three networks (i.e., L_1, L_2, and L_3) are given by:

$$L_1 = \alpha_1 L_C^1 + \beta_1 L_D^{13} + \gamma_1 L_D^{12}, \tag{6.7}$$

$$L_2 = \alpha_2 L_C^2 + \beta_2 L_D^{23} + \gamma_2 L_D^{12}, \tag{6.8}$$

$$L_3 = L_C^3, \tag{6.9}$$

where α_1, α_2, β_1, β_2, γ_1, and γ_2 are hyper-parameters for balancing the contributions of the classification loss and the prediction divergences.

6.2.3 Training Scheme

Fig. 6.2 demonstrates the training scheme of the RD-LNL approach.

Pre-training of Leader Network

In the proposed RD-LNL approach, the feature extraction and representation abilities of the leader network are significantly important. Thanks to its generalization and knowledge

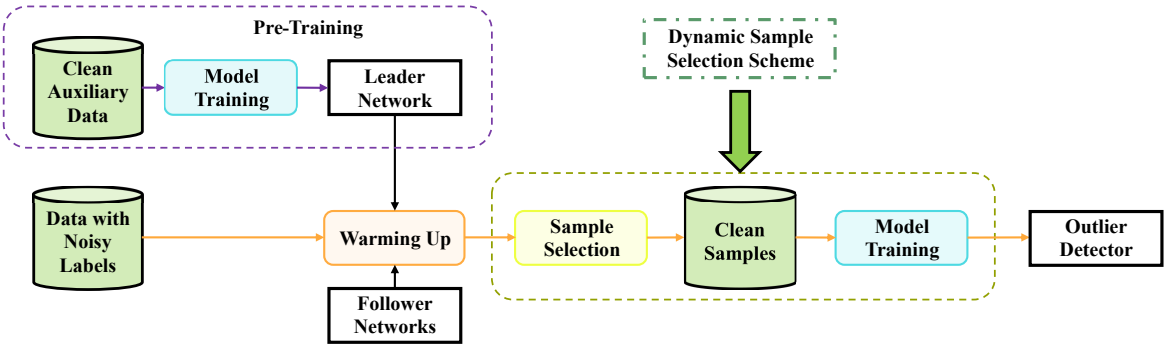


Fig. 6.2 The training process of the RD-LNL approach

transfer capability, transfer learning becomes an effective solution to obtain the leader network using a clean auxiliary data set. The leader network is pre-trained on the clean source domain to learn transferable representations for the downstream noisy task.

Joint Training on Noisy Labeled Data

After obtaining the leader network, the follower networks are trained jointly with the leader network. The joint training process consists of two main stages, namely warm-up and sample selection. Specifically, the leader network is initialized using the weights obtained through pre-training, and the follower networks are initialized randomly. In the warm-up stage, the leader network and the follower networks are trained on the noisy data for several epochs to ensure a stable initialization and learn initial representations. During warm-up, the leader network is fine-tuned to adapt its pre-trained knowledge to the noisy target domain. In the sample selection stage, the clean samples are selected using the adaptive selection scheme in each epoch and are then used to update three networks.

The Adaptive Selection scheme

In RD-LNL, the "small-loss criterion" is utilized for sample selection. To be specific, a selection metric is introduced for selecting potential clean samples based on three networks, where the loss values of each sample from three networks during the training are leveraged for distinguishing clean samples from noisy ones. The selection metric of each sample $L_S(i)$ can be calculated by:

$$L_S(i) = \mu_1 L_1(i) + \mu_2 L_2(i) + \mu_3 L_3(i), \tag{6.10}$$

where $L_1(i)$, $L_2(i)$, and $L_3(i)$ are per-sample losses of three networks; and μ_1, μ_2, and μ_3 are three hyper-parameters for balancing the contributions of each term.

The CSSR $R(e)$ is adjusted adaptively during the training process based on the training epoch and selection metric, which is calculated by:

$$R(e) = \max\left\{\left(1 - \frac{e}{E} \cdot \hat{r}\right) \cdot \frac{\sigma_e}{\sigma_0},\ 1 - \hat{r}\right\}, \tag{6.11}$$

where e and E are the current and total training epochs, respectively; σ_e and σ_0 denote the standard deviation of values of selection metric across all samples in the mini-batch at the eth epoch and the initial training epoch, respectively; and $\hat{r}$ is the estimated noise ratio derived from the validation set.

Training Procedure

The training process of the RD-LNL approach is detailed in Algorithm 5. After the training process, the leader network is then utilized for outlier detection tasks.

6.3 WAAM Outlier Detection

6.3.1 Data Pre-Processing

A sliding window method is applied to segment each data set, with a window size of 10 and a stride of 5. The label for each segment is determined based on the labels of its constituent instances: a segment is considered "Normal" only if all included instances are labeled as such; otherwise, it is marked as an "Outlier". For classification purposes, all labels are converted into one-hot encoded vectors. Each processed data set is subsequently divided into training and testing subsets in a 70 : 30 ratio. To ensure scale consistency and enhance model performance, min-max normalization is applied to both the training sets and the testing sets.

Algorithm 5 Training procedure of the RD-LNL approach

Require: Noisy training data set $\tilde{\mathcal{D}}$, clean auxiliary data set $\mathcal{D}$, pre-training epochs E_P, warm-up epochs E_W, training epochs E, batch size B, hyper-parameters μ_1, μ_2, μ_3
Ensure: Model parameters of three networks $\theta_1, \theta_2, \theta_3$

Randomly initialize θ_1
for $e = 1$ to E_P **do**
  for all mini-batches $\mathcal{B} \subset \mathcal{D}$ with $|\mathcal{B}| = B$ **do**
    Compute L_3 according to (6.9)
    Update θ_1 by minimizing L_3
  end for
end for
Retain θ_1 and randomly initialize θ_2 and θ_3
for $e = 1$ to E_W **do**
  for all mini-batches $\tilde{\mathcal{B}} \subset \tilde{\mathcal{D}}$ with $|\tilde{\mathcal{B}}| = B$ **do**
    Compute L_1, L_2, L_3 according to (6.7), (6.8) and (6.9)
    Update $\theta_1, \theta_2, \theta_3$
  end for
end for
Retain θ_1, θ_2 and θ_3
for $e = 1$ to E **do**
  for all mini-batches $\tilde{\mathcal{B}} \subset \tilde{\mathcal{D}}$ with $|\tilde{\mathcal{B}}| = B$ **do**
    Compute selection metric $L_S(i)$ according to (6.10)
    Compute selection ratio $R(e)$ according to (6.11)
    Obtain selected clean subset $\tilde{\mathcal{B}}_C$ from $\tilde{\mathcal{B}}$ using $L_S(i)$ and $R(e)$
    Compute L_1, L_2, L_3 on $\tilde{\mathcal{B}}_C$ according to (6.7), (6.8) and (6.9)
    Update $\theta_1, \theta_2, \theta_3$
  end for
end for

6.3.2 Implementation Details

Model Configurations

The Transformer encoder of the leader network consists of 3 encoder layers. The CNNs of the two leader networks both consist of 5 convolutional modules. The classifier of each network is a 3-layer multi-layer perceptron. To ensure fair and robust evaluation, all hyper-parameters are selected through grid search on a separate validation set. The hyper-parameters are tuned within predefined ranges based on the best validation performance, and the same selection procedure is applied consistently across all selected approaches. To be specific, the hyper-parameters α_1, α_2, β_1, β_2, γ_1 and γ_2 in the loss functions are set as 0.35, 0.35, 0.7,

0.7, 0.25, and 0.25, respectively. The hyper-parameters μ_1, μ_1 and μ_1 in the selection metric are set to be 0.6, 0.2 and 0.2, respectively. The Adam optimizer is adopted and the learning rate is configured as 0.0001. The epoch numbers of the pre-training process, the warm-up process and the training process are 30, 10, and 50, respectively.

Experimental Settings

All experiments are performed under a consistent environment: Ubuntu 20.04.6, PyTorch 2.5.1, Python 3.9.21, and CUDA 12.3, using hardware with an NVIDIA RTX A6000 GPU (48 GB) and an Intel Xeon Silver 4214R processor to ensure fair comparisons. Each experiment is repeated five times, and the mean values of all evaluation metrics are presented to minimize the effect of random variation.

To verify the performance of the developed RD-LNL approach, two standard DL-based outlier detection approaches and two representative LNL approaches (e.g., Co-teaching and JoCoR) are selected for comparison. Specifically, in two standard DL-based outlier detection approaches, the CNN and the Transformer are employed as the backbone and connected with two identical classifiers, respectively.

Four evaluation metrics (e.g., accuracy, precision, recall, and F1-score) are applied to evaluate the effectiveness of the proposed RD-LNL approach for outlier detection under label noise. In the experiments, a clean auxiliary data set is selected for pre-training the leader network, and the other four data sets are used as the noisy training data. Extensive experiments are conducted under various noise ratios denoted by ϕ to evaluate the outlier detection performance under different levels of label noise. Each data set is evaluated under noise ratios ϕ of 30%, 40%, and 50%.

6.3.3 Results and Discussion

The outlier detection results of the selected approaches on four WAAM data sets with various noise ratios are summarized in Table 6.1. As shown by the results, the RD-LNL approach shows competitive performance across four tasks under various noise ratios. Detailed analysis of the results is provided below based on the noise ratio.

Table 6.1 Outlier detection results on noisy WAAM data sets with different noise ratios

Approaches	Metrics (%)	Task 1			Task 2			Task 3			Task 4		
		30%	40%	50%	30%	40%	50%	30%	40%	50%	30%	40%	50%
Co-teaching	Accuracy	96.77	89.01	74.76	95.22	88.43	72.11	95.12	88.82	73.57	93.80	88.65	73.52
	Precision	96.10	90.82	50.33	94.26	82.04	52.19	95.86	82.33	53.81	88.54	78.93	48.90
	Recall	92.05	90.37	77.62	93.64	88.59	76.36	93.06	88.97	69.31	94.22	79.90	70.55
	F1 Score	94.28	90.38	62.28	93.05	86.87	58.15	94.11	87.76	60.31	90.45	70.47	49.73
JoCoR	Accuracy	97.13	93.98	79.00	96.09	93.90	75.50	95.63	93.12	77.20	94.63	92.01	69.58
	Precision	97.29	93.50	48.55	96.00	93.01	46.01	96.74	91.49	45.75	89.91	79.16	34.01
	Recall	92.30	89.18	66.41	95.12	93.43	79.85	95.53	92.61	90.08	96.42	94.55	95.53
	F1 Score	94.57	91.39	56.16	95.68	92.65	60.59	96.01	92.41	61.09	93.12	87.12	50.16
Standard (CNN)	Accuracy	87.24	61.90	39.23	85.46	58.92	25.07	73.72	53.34	30.64	71.14	56.98	10.63
	Precision	83.29	62.79	55.53	80.48	75.82	36.90	69.53	56.28	42.27	69.24	59.14	26.49
	Recall	88.09	85.34	31.71	93.12	45.49	46.81	90.79	78.26	45.96	72.62	36.87	18.94
	F1 Score	90.02	72.35	40.37	87.45	56.84	41.21	81.71	66.41	43.72	80.64	53.67	22.24
Standard (Transformer)	Accuracy	89.16	63.36	41.65	87.66	60.96	27.19	75.71	55.34	32.24	72.39	58.35	12.74
	Precision	85.76	64.44	56.93	82.15	77.83	38.44	70.44	58.56	43.71	70.00	61.52	28.25
	Recall	89.40	86.11	34.05	94.70	47.73	47.69	91.70	79.54	47.33	73.41	38.94	20.94
	F1 Score	92.09	74.33	42.77	89.92	58.52	42.55	82.27	67.56	45.38	81.79	55.74	24.39
RD-LNL (Ours)	Accuracy	96.98	94.20	85.52	95.40	94.07	81.80	95.85	92.85	82.49	95.16	93.85	78.90
	Precision	96.17	93.85	86.96	96.95	94.04	86.80	96.82	92.95	88.60	96.70	93.15	87.96
	Recall	97.02	93.02	87.64	95.25	92.11	81.71	95.66	94.33	85.85	96.66	92.27	88.37
	F1 Score	96.37	95.09	86.40	96.28	94.23	84.22	96.14	92.70	86.55	96.29	93.22	87.78

In the experiments under a noise ratio of 30%, the selected LNL approaches as well as two standard approaches all exhibit satisfactory performance on four data sets. It should be noticed that Tasks 1 and 2, the JoCoR reaches higher accuracy than the RD-LNL. On the other two tasks, the RD-LNL approach outperforms the selected approaches in terms of detection accuracy and F1 score. At a noise level of 40%, both standard methods exhibit noticeable performance drops. According to the results, the selected LNL approaches still show reliable performance. On task 3, the accuracy of the JoCoR is higher than RD-LNL. Nevertheless, the RD-LNL approach maintains superior performance across the remaining tasks, which confirms its effectiveness in detecting outliers under noisy labels.

In the scenario with a 50% noise ratio, the RD-LNL approach also demonstrates strong performance, while all selected approaches suffer significant degradation across the four data sets. In particular, the accuracy and F1 scores of the selected approaches do not exceed 80% and 63%, respectively. In contrast, RD-LNL achieves results on all data sets, highlighting its robustness in the presence of moderate label noise.

6.4 Conclusion

In this chapter, a novel RD-LNL approach has been developed for outlier detection on sufficient industrial data under noisy labels. An LFSS strategy has been proposed to train the outlier detector, where a leader network and two follower networks are trained jointly for sample selection by leveraging the domain knowledge of the leader network. A selection metric has been designed for identifying clean samples. In addition, an adaptive selection scheme has been put forward based on the characteristics of the selection metric for adjusting the CSSR. Experimental results for outlier detection on real-world data sets have demonstrated satisfactory performance of the developed approach. Future work can be summarized into the following four aspects: 1) designing label correction strategies to improve the utilization of unselected samples; 2) applying the optimization techniques for hyper-parameter selection; 3) deploying the proposed approach to other tasks under label noise; and 4) developing advanced transfer learning framework to make use of the domain knowledge.

Chapter 7

Conclusions and Future Research

In this chapter, we first provide a comprehensive summarization of our work in this thesis, and the contributions of each chapter are also presented in Section 7.1. Then, we point out some future research directions that follow from this thesis in Section 7.2.

7.1 Summarization

In many real-world scenarios, data imperfections are unavoidable due to various factors such as hardware constraints, human errors, and uncontrollable external conditions, which pose significant challenges to outlier detection and ensuring reliable data-driven decision making. In this context, we aim to put forward novel frameworks for outlier detection on WAAM data with different imperfections to better accommodate real-world conditions and deliver consistent, reliable performance.

In this thesis, we have considered three major imperfections in real-world WAAM data and developed specific outlier detection frameworks for each scenario, in order to ensure accurate detection and satisfactory performance in practical applications. An optimized FCM-based outlier detection method has been designed for analyzing the unlabeled WAAM data (Chapter 3). A PSO-embedded DTL framework has been proposed for outlier detection on limited-sample and imbalanced WAAM data (Chapter 4). A robust-training-based LNL framework named TFCCL and a sample-selection-based LNL framework named RD-LNL have been developed for WAAM outlier detection under label noise (Chapter 5 and Chapter 6). In the following paragraphs, the research outputs in each chapter are summarized.

In Chapter 3, an optimized FCM-based outlier detection method has been proposed to analyze the unlabeled WAAM data. The ASRPPSO has been developed to optimize the initial locations of the cluster centroids in the FCM algorithm. An AWU strategy has been designed in the ASRPPSO to adaptively adjust the acceleration coefficients, and the Gaussian white noise has been added to the velocity updating equation based on the evolutionary states. Besides, a switching strategy has been employed to balance the global and local searches. Under this parameter updating strategy, the particles could obtain strong search ability, and they could avoid being trapped in the local optima. Experimental results have shown the superiority of the ASRPPSO over some existing PSO algorithms in terms of convergence rate and solution quality. The outlier detection performance on the unlabeled WAAM data has also shown the effectiveness of the developed ASRPPSO-based FCM algorithm.

In Chapter 4, a PSO-embedded DTL framework has been developed for outlier detection on WAAM data that suffer from the small-sample problem. A novel DDA strategy has been designed to minimize the cross-domain discrepancies of both the marginal distribution and the conditional distribution. Two separate coefficients have been introduced to adjust the conditional domain discrepancies of normal instances and outliers to alleviate the data imbalance problem. The PSO algorithm has been adopted to adjust the hyper-parameters of the proposed method. Experimental results have shown satisfactory and promising performance in the detection accuracy of the proposed approach on small-sample WAAM data.

In Chapter 5, a novel robust-training-based LNL framework named TFCCL has been developed for WAAM outlier detection under label noise and limited-data conditions. The FCCL strategy has been introduced to update the Transformer encoder, where a UFCM algorithm has been proposed for identifying positive and negative sample pairs. A dynamic two-stage training scheme has been designed to train the outlier detector, incorporating a joint learning strategy to enhance its robustness against noisy labels. A dynamic updating strategy has been employed to update the UFCM algorithm throughout the training process. A label-consistency regularization term has also been introduced to minimize the discrepancy between the outputs of the outlier detector and the UFCM algorithm to further improve model reliability. Experimental results have demonstrated the effectiveness of the developed TFCCL framework on noisy labeled WAAM data.

In Chapter 6, a sample-selection-based LNL approach named RD-LNL has been proposed for WAAM outlier detection with noisy labels under sufficient-data settings. An LFSS strategy has been proposed to train the outlier detector. A selection metric has been designed for identifying clean samples. In addition, an adaptive selection scheme has been put forward based on the selection metric for adjusting the CSSR. Experimental results for outlier detection on WAAM data with label noise have demonstrated satisfactory performance of the proposed approach.

It should be noted that the WAAM data used in this thesis cannot be publicly released because they are collected from real-world manufacturing processes and are subject to confidentiality and intellectual property restrictions. For the same reason, the data-dependent implementation code cannot be publicly released either. However, the source code of the ASRPPSO algorithm, which is independent of the WAAM data, is publicly available on GitHub (https://github.com/jisu33162/Adaptive-Switching-Randomly-Perturbed-Particle-Swarm-Optimization-Algorithm-ASRPPSO).

7.2 Future Work

In this thesis, we have investigated outlier detection on imperfect real-world WAAM data. Although the developed frameworks have been successfully applied to real-world WAAM data, there is still much room to extend our work from the perspectives of the algorithm and application. It is worth mentioning that unlabeled data and small-sample data have been widely studied in recent years, and various existing approaches have achieved remarkably good performance in addressing these challenges. As such, in future research, our attention will be directed toward the noisy label problem, as it remains insufficiently explored and holds substantial potential for further study. Below are some future research directions relevant to the research work in this thesis.

7.2.1 Physics-Informed Learning

In many LNL approaches, label noise is primarily perceived by the model through statistical training signals, such as per-sample losses and confidence scores, which reflect the underlying learning information of the clean and noisy samples during the training process [71, 221].

Nevertheless, existing LNL approaches typically adopt static strategies, which neglect the variability across different domains. Furthermore, the statistical training signals may sometimes become ambiguous in some real-world scenarios, especially under high label noise conditions.

To address the gap in LNL research, it is essential to develop adaptive frameworks that can effectively distinguish clean and noisy samples across diverse data domains. Physics-informed learning offers a complementary perspective by embedding prior physical knowledge into the learning objective to enhance the model's robustness and generalization ability [77]. By utilizing additional physical patterns and enforcing consistency with known physical laws, the model is capable of suppressing the influence of noisy labels and producing physically consistent predictions. Besides, tracking and analyzing the training dynamics can also help the model identify clean and noisy samples based on the learning behaviors of samples across different domains adaptively. To be specific, integrating advanced dynamic learning techniques such as online meta-learning, temporal noise modeling, and Bayesian uncertainty estimation can enable models to adapt to diverse training behaviors, thereby enhancing the model's robustness against label noise.

7.2.2 Instance-Dependent Label Noise

Comparing with the CDLN (which assumes label noise to be independent and uniformly distributed), the probability of label corruption in the IDLN depends on both the ground truth and the corresponding input features, which can be seen as a more realistic noise model [54, 167]. Specifically, the noisy labels usually arise from minor uncertainties in the input data or context-dependent factors that affect human annotators. The samples under IDLN typically exhibit high heterogeneity and complexity, and the extracted features are often similar to those of clean samples, which makes it particularly challenging to distinguish between noisy and clean samples using existing LNL approaches designed for CDLN problems.

So far, only a few LNL approaches have been proposed to handle the IDLN problem [59, 163, 201], and most of which rely on sample selection strategies to identify and leverage clean samples to mitigate the complexities arising from the IDLN. It is worth pointing out that sample selection strategies may discard potentially valuable samples, which can reduce the quantity and diversity of the training data, and thereby negatively affect the

overall performance. As such, there still remains considerable research potential for further investigating IDLN scenarios by improving the sample utilization efficiency via developing adaptive weighting or label correction mechanisms. Besides, designing hybrid approaches that combine sample selection with advanced noise modeling techniques is also a potential solution to enhance the robustness and generalization ability of the model under IDLN.

7.2.3 Hyper-parameter Selection

During the past few years, numerous LNL approaches have been developed by adjusting the model architectures or modifying the loss functions, which inevitably introduce a variety of hyper-parameters. Hyper-parameter selection plays an important role in: 1) determining the network architecture (e.g., number of units, number of layers, etc.) which affects the model complexity; and 2) balancing the contribution of each term in the loss function, which affects the performance of the model such as model robustness, generalization ability, accuracy and so on. It should be noted that the values of hyper-parameters can directly influence model training, thereby affecting the model performance. Normally, the selection process of hyper-parameters relies primarily on manual and experience-based methods, which proves to be time-consuming. Such manual selection may lead to suboptimal model performance and limited scalability, especially with constrained time and resources [118].

It is worth mentioning that selecting optimal hyper-parameters involves systematically exploring various combinations of parameter values and evaluating their impact on a chosen performance metric such as the loss value or the model accuracy, which can be formulated as an optimization problem. To further enhance the effectiveness of the LNL approaches, employing the optimization techniques (e.g., evolutionary computation, Bayesian optimization, neural architecture search, and reinforcement learning-based strategies) to automate the hyper-parameter selection process is a feasible solution with broad research potential for future exploration. Besides, designing adaptive hyper-parameter selection strategies also offers the potential to dynamically adjust model parameters in real time during the model training process, which would further enhance the overall model performance in real-world applications.

7.2.4 Budget-Aware Evaluation and Method Selection

The methods developed in this thesis have mainly focused on improving the robustness and detection performance of outlier detection models under imperfect industrial data conditions. Nevertheless, computational budget is also an important factor that should be considered in practical WAAM monitoring applications. Different methods may require different levels of computational resources due to differences in model complexity, optimization strategy, training scheme, and hyper-parameter selection process. For example, deep learning-based approaches and optimization-embedded frameworks may achieve strong detection performance, but they may also introduce higher computational costs than simpler clustering-based or shallow learning methods.

Therefore, future research could further investigate budget-aware and resource-aware comparisons of different outlier detection methods under varying levels of computational resource availability. Specifically, the trade-off between detection performance and computational cost could be systematically evaluated by considering factors such as training time, number of fitness evaluations, memory usage, and hardware requirements. Such analysis would help identify lightweight methods for resource-limited industrial environments and more complex methods for scenarios where sufficient computational resources are available.

7.2.5 Data Security

With the widespread adoption of cloud-based technologies, data security has become a critical concern in various data analysis tasks, especially in finance, transportation, and manufacturing [70]. Security risks (such as data tampering, data corruption, data poisoning, or adversarial attacks) may result in the occurrence of noisy labels, which would mislead the learning process and degrade the model performance. It is worth mentioning that such noisy labels often arise from deliberate adversarial actions and are carefully generated to resemble clean labels, which brings significant challenges for model training. Furthermore, existing LNL approaches are typically developed based on assumptions of random or static noise, which may fail to address the data security challenges effectively.

To tackle the noisy label problem caused by data security issues, future research could focus on developing integrated LNL approaches that incorporate security-aware mechanisms

to detect and mitigate adversarially induced label noise. Specifically, such approaches should be capable of distinguishing between naturally occurring and intentionally introduced label noise, adapting dynamically to evolving attack patterns, and maintaining robustness even under targeted data manipulation. Promising directions include integrating adversarial defense strategies with noise modeling and designing hybrid methods that jointly consider data integrity and label correctness.

References

[1] A. Smiti, A critical overview of outlier detection methods, *Computer Science Review*, vol. 38, art. no. 100306, 2020.

[2] S. Ait-Aoudia, E. Guerrout and R. Mahiou, Medical image segmentation using particle swarm optimization, In: *Proceedings of the 2014 International Conference on Information Visualisation*, Paris, France, Nov. 2014, pp. 287-291.

[3] K. S. Akshat and B. Prabodh, Swarm intelligence based optimal sizing of solar PV, fuel cell and battery hybrid system, In: *Proceedings of the International Conference on Power and Energy Systems*, Hong Kong, China, Aug. 2012, pp. 467-473.

[4] P. Albert, E. Arazo, T. Krishna, N. O'Connor and K. McGuinness, Is your noise correction noisy? PLS: Robustness to label noise with two stage detection, In: *Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, USA, Jan. 2023, pp. 118-127.

[5] W. Al-Hassan, M. B. Fayek and S. I. Shaheen, PSOSA: An optimized particle swarm technique for solving the urban planning problem, In: *Proceedings of the 2006 International Conference on Computer Engineering and Systems*, Cairo, Egypt, Nov. 2006, pp. 401-405.

[6] H. Alimohammadi and S. N. Chen, Performance evaluation of outlier detection techniques in production timeseries: A systematic review and meta-analysis, *Expert Systems with Applications*, vol. 191, art. no. 116371, 2022.

[7] W. Al-Saedi, S. W. Lachowicz and D. Habibi, An optimal current control strategy for a three-phase grid-connected photovoltaic system using particle swarm optimization, In: *Proceedings of the 2011 IEEE Power Engineering and Automation Conference*, Wuhan, China, Sept. 2011, pp. 286-290.

[8] R. Andonie, Hyperparameter optimization in learning systems, *Journal of Membrane Computing*, vol. 1, pp. 279-291, 2019.

[9] E. Arazo, D. Ortego, P. Albert, N. O'Connor and K. Mcguinness, Unsupervised label noise modeling and loss correction, In: *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, Jun. 2019, pp. 312-321.

[10] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio and S. Lacoste-Julien, A closer look at memorization in deep networks, In: *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, Jun. 2017, pp. 233-242.

[11] M. Azab, Optimal power point tracking for stand-alone PV system using particle swarm optimization, In: *Proceedings of the 2010 IEEE International Symposium on Industrial Electronics*, Bari, Italy, Nov. 2010, pp. 969-973.

[12] N. A. A. Aziz, Z. Ibrahim, M. Mubin, S. W. Nawawi and M. S. Mohamad, Improving particle swarm optimization via adaptive switching asynchronous–synchronous update, *Applied Soft Computing*, vol. 72, pp. 298-311, 2018.

[13] P. Bajpai and S. N. Singh, Fuzzy adaptive particle swarm optimization for bidding strategy in uniform price spot market, *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 2152-2160, 2007.

[14] J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jadon and A. Abraham, Inertia weight strategies in particle swarm optimization, In: *Proceedings of the 2011 Third World Congress on Nature and Biologically Inspired Computing*, Salamanca, Spain, Oct. 2011, pp. 633-640.

[15] G. Q. Bao and K. F. Mao, Particle swarm optimization algorithm with asymmetric time varying acceleration coefficients, In: *Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics*, Guilin, China, Dec. 2009, pp. 2134-2139.

[16] M. Bashir and J. Sadeh, Size optimization of new hybrid stand-alone renewable energy system considering a reliability index, In: *Proceedings of the 11th International Conference on Environment and Electrical Engineering*, Venice, Italy, Jun. 2009, pp. 989-994.

[17] M. Braik, A. F. Sheta and A. Ayesh, Image enhancement using particle swarm optimization, In: *Proceedings of the World Congress on Engineering*, London, UK, Jul. 2007, vol. 1, pp. 978-988.

[18] A. Castellani, S. Schmitt and B. Hammer, Estimating the electrical power output of industrial devices with end-to-end time-series classification in the presence of label noise, In: *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Bilbao, Spain, Sep. 2021, pp. 469-484.

[19] A. Chatterjee and P. Siarry, Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization, *Computers & Operations Research*, vol. 33, no. 3, pp. 859-871, 2006.

[20] J. Chen, S. Deng, D. Teng, D. Chen, T. Jia and H. Wang, APPN: An attention-based pseudo-label propagation network for few-shot learning with noisy labels, *Neurocomputing*, vol. 602, art. no. 128212, 2024.

[21] K. Chen, F. Zhou, Y. Wang and L. Yin, An ameliorated particle swarm optimizer for solving numerical optimization problems, *Applied Soft Computing*, vol. 73, pp. 482-496, 2018.

[22] K. Chen, F. Zhou, L. Yin, S. Wang, Y. Wang and F. Wan, A hybrid particle swarm optimizer with sine cosine acceleration coefficients, *Information Sciences*, vol. 422, pp. 218-241, 2018.

[23] L. Chen, Y. Zhang, T. Huang, L. Su, Z. Lin, X. Xiao, X. Xia and T. Liu, Erase: Error-resilient representation learning on graphs for label noise tolerance, In: *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, Boise, USA, Oct. 2021, pp. 270-280.

[24] M. Chen, Y. Zhao, B. He, Z. Han, J. Huang, B. Wu and J. Yao, Learning with noisy labels over imbalanced subpopulations, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 4, pp. 6544-6555, 2024.

[25] Y. Chen, W. Peng and M. Jian, Particle swarm optimization with recombination and dynamic linkage discovery, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 6, pp. 1460-1470, 2007.

[26] C. Cheng, X. Liu, B. Zhou and Y. Yuan, Intelligent fault diagnosis with noisy labels via semisupervised learning on industrial time series, *IEEE Transactions on Industrial Informatics*, vol. 19, no. 6, pp. 7724-7732, 2023.

[27] C. Cheng, X. Yu, H. Wen, J. Sun, G. Yue, Y. Zhang and Z. Wei, Exploring the robustness of in-context learning with noisy labels, *arXiv preprint arXiv:2404.18191*, 2024.

[28] G. Cheng, W. Jing and C. Gao, Recovery trajectory planning for the reusable launch vehicle, *Aerospace Science and Technology*, vol. 117, art. no. 106965, 2021.

[29] S. Cheng, Y. Shi and Q. Qin, Promoting diversity in particle swarm optimization to solve multimodal problems, In: *Proceedings of the International Conference on Neural Information Processing*, Berlin, Germany, Nov. 2011, pp. 228-237.

[30] H. W. Cho, S. J. Shin, G. J. Seo, D. B. Kim and D. H. Lee, Real-time anomaly detection using convolutional neural network in wire arc additive manufacturing: Molybdenum material, *Journal of Materials Processing Technology*, vol. 302, art. no. 117495, 2022.

[31] Y. Cho, W. Kim, S. Hong and S. Yoon, Part-based pseudo label refinement for unsupervised person re-identification, In: *Proceedings of the 2022 IEEE/CVF Conference*

on Computer Vision and Pattern Recognition (CVPR), New Orleans, USA, Jun. 2022, pp. 7298-7308.

[32] G. Ciuprina, D. Ioan and I. Munteanu, Use of intelligent-particle swarm optimization in electromagnetics, *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 1037-1040, 2002.

[33] D. Chong, J. Hong and C. D. Manning, Detecting label errors by using pre-trained language models, *arXiv preprint arXiv:2205.12702*, 2022.

[34] M. Clerc, The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization, In: *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, USA, Jul. 1999, pp. 1951-1957.

[35] M. Clerc and J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58-73, 2002.

[36] L. S. Coelho and C. S. Lee, Solving economic load dispatch problems in power systems using chaotic and Gaussian particle swarm optimization approaches, *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 5, pp. 297-307, Jun. 2008

[37] F. Cordeiro, R. Sachdeva, V. Belagiannis, I. Reid and G. Carneiro, LongReMix: Robust learning with high confidence samples in a noisy label environment, *Pattern Recognition*, vol. 133, art. no. 109013, 2023.

[38] R. Croft, M. A. Babar and H. Chen, Noisy label learning for security defects, In: *Proceedings of the 2022 IEEE/ACM 19th International Conference on Mining Software Repositories (MSR)*, Pittsburgh, USA, May. 2022, pp. 435-447.

[39] S. Dai, S. He, X. Chen and X. Jin, Adaptive leader–follower formation control of nonholonomic mobile robots with prescribed transient and steady-state performance, *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3662-3671, 2020.

[40] Y. Dai, Z. Mei, J. Li, Z. Li, K. Wei, M. Ding, S. Guo and W. Chen, Clustering-based contrastive learning for fault diagnosis with few labeled samples, *IEEE Transactions on Instrumentation and Measurement*, vol. 73, art. no. 2504913, 2023.

[41] S. Das, A. Abraham and A. Konar, Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm, *Pattern Recognition Letters*, vol. 29, no. 5, pp. 688-699, 2008.

[42] Y. Del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. Hernandez and R. G. Harley, Particle swarm optimization: Basic concepts, variants and applications in power systems, *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171-195, 2004.

[43] E. Di Mario, I. Navarro and A. Martinoli, A distributed noise-resistant particle swarm optimization algorithm for high-dimensional multi-robot learning, In: *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, Seattle, USA, May 2015, pp. 5970-5976.

[44] K. Ding, J. Shu, D. Meng and Z. Xu, Improve noise tolerance of robust loss via noise-awareness, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 7, pp. 13189-13203, 2025.

[45] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, In: *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, Nagoya, Japan, Oct. 1995, pp. 39-43.

[46] R. Eberhart and Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, In: *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, USA, Jul. 2000, pp. 84-88.

[47] C. Fang, L. Cheng, Y. Mao, D. Zhang, Y. Fang, G. Li, H. Qi and L. Jiao, Separating noisy samples from tail classes for long-tailed image classification with label noise, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 11, pp. 16036-16048, 2024.

[48] K. Fatras, B. Damodaran, S. Lobry, R. Flamary, D. Tuia and N. Courty, Wasserstein adversarial regularization for learning with label noise, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 7296-7306, 2022.

[49] C. Feng, Y. Ren and X. Xie, OT-Filter: An optimal transport filter for learning with noisy labels, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 16164-16174.

[50] C. Feng, G. Tzimiropoulos and I. Patras, SSR: An efficient and robust framework for learning with unknown label noise, *arXiv preprint arXiv:2111.11288*, 2022.

[51] C. Feng, G. Tzimiropoulos and I. Patras, NoiseBox: Towards more efficient and effective learning with noisy labels, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 11, pp. 11914-11928, 2024.

[52] Y. Feng, G. Teng, A. Wang and Y. Yao, Chaotic inertia weight in particle swarm optimization, variants and applications in power systems, In: *Proceedings of the 2nd International Conference on Innovative Computing, Informatio and Control*, Kumamoto, Japan, Sep. 2007, pp. 475-478.

[53] F. Fooladgar, M. N. N. To, P. Mousavi and P. Abolmaesumi, Manifold DivideMix: A semi-supervised contrastive learning framework for severe label noise, In: *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops)*, Seattle, USA, Jun. 2024, pp. 4012-4021.

[54] B. Frenay and M. Verleysen, Classification in the presence of label noise: A survey, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845-869, 2014.

[55] J. A. Fries, P. Varma, V. S. Chen, K. Xiao, H. Tejeda, P. Saha, J. Dunnmon, H. Chubb, S. Maskatia, M. Fiterau, S. Delp, E. Ashley, C. Ré and J. R. Priest, Weakly supervised classification of aortic valve malformations using unlabeled cardiac MRI sequences, *Nature Communications*, vol. 10, art. no. 3111, 2019.

[56] Z. L. Gaing, A particle swarm optimization approach for optimum design of PID controller in AVR system, *IEEE Transactions on Energy Conversion*, vol. 19, no. 2, pp. 384-391, 2004.

[57] W. Gao, Y. Zhang, D. Ramanujan, K. Ramani, Y. Chen, C. B. Williams, C. Wang, Y. Shin, S. Zhang and P. Zavattieri, The status, challenges, and future of additive manufacturing in engineering, *Computer-Aided Design*, vol. 69, pp. 65-89, 2015.

[58] Y. Gao, X. An and J. Liu, A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation, In: *Proceedings of the 2008 International Conference on Computational Intelligence and Security*, Suzhou, China, Dec. 2008, pp. 61-65.

[59] A. Garg, C. Nguyen, R. Felix, T. Do and G. Carneiro, Instance-dependent noisy label learning via graphical modelling, In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Hawaii, USA, Jan. 2023, pp. 2288-2298.

[60] S. Garg, G. Ramakrishnan and V. Thumbe, Towards robustness to label noise in text classification via noise modeling, In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, Queensland, Australia, Nov. 2021, pp. 3024-3028.

[61] P. Ghamisi, M. S. Couceiro, F. M. L. Martins and J. A. Benediktsson, Multilevel image segmentation based on fractional-order Darwinian particle swarm optimization, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 5, pp. 2382-2394, 2014.

[62] I. Gibson, D. Rosen and B. Stucker, *Additive Manufacturing Technologies*, Switzerland: Springer, 2021.

[63] J. Goldberger and E. Ben-Reuven, Training deep neural-networks using a noise adaptation layer, In: *Proceedings of the 10th International Conference on Learning Representations*, Toulon, France, Apr. 2017.

[64] M. Gong, Y. Zhao, H. Li, A. Qin, L. Xing, J. Li, Y. Liu and Y. Liu, Deep fuzzy variable C-means clustering incorporated with curriculum learning, *IEEE Transactions on Fuzzy Systems* , vol. 31, no. 12, pp. 4321-4335, 2023.

[65] A. Gorai and A. Ghosh, Gray-level image enhancement by particle swarm optimization, In: *Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing*, Coimbatore, India, Dec. 2009, pp. 72-77.

[66] J. Guo and S. Tang, An improved particle swarm optimization with re-initialization mechanism, In: *Proceedings of the 2009 International Conference on Intelligent Human-Machine Systems and Cybernetics*, Hangzhou, China, Aug. 2009, vol. 1, pp. 437-441.

[67] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott and D. Huang, CurriculumNet: Weakly supervised learning from large-scale web images, In: *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, Sep. 2018, pp. 135-150.

[68] W. Guo, G. Chen and X. Feng, A new strategy of acceleration coefficients for particle swarm optimization, In: *Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design*, Nanjing, China, May 2006, pp. 1-5.

[69] M. Gupta, J. Gao, C. C. Aggarwal and J. Han, Outlier detection for temporal data: A survey, *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2250-2267, 2014.

[70] E. Hallaji, R. Razavi-Far, M. Saif and E. Herrera-Viedma, Label noise analysis meets adversarial training: A defense against label poisoning in federated learning, *Knowledge-Based Systems*, vol. 266, art. no. 110384, 2023.

[71] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang and M. Sugiyama, Co-teaching: Robust training of deep neural networks with extremely noisy labels, In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS 2018)*, Montreal, Canada, Dec. 2018, vol. 31.

[72] D. Hao, L. Zhang, J. Sumkin, A. Mohamed and S. Wu, Inaccurate labels in weakly-supervised deep learning: Automatic identification and correction and their impact on classification performance, *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 9, pp. 2701-2710, 2020.

[73] F. He, L. Yuan, H. Mu, M. Ros, D. Ding, Z. Pan and H. Li, Research and application of artificial intelligence techniques for wire arc additive manufacturing: A state-of-the-art review, *Robotics and Computer-Integrated Manufacturing*, vol. 82, art. no. 102525, 2023.

[74] K. He, H. Fan, Y. Wu, S. Xie and R. Girshick, Momentum contrast for unsupervised visual representation learning, In: *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, Jun. 2020, pp. 9726-9735.

[75] S. He, W. K. Ao and Y. -Q. Ni, A unified label noise-tolerant framework of deep learning-based fault diagnosis via a bounded neural network, *IEEE Transactions on Instrumentation and Measurement*, vol. 73, art. no. 3513515, 2024.

[76] D. Hendrycks and K. Gimpel, Gaussian error linear units (gelus), *arXiv preprint arXiv:1606.08415*, 2016.

[77] B. Huang and J. Wang, Applications of physics-informed neural networks in power systems-a review, *IEEE Transactions on Power Systems*, vol. 38, no. 1, pp. 572-588, 2023.

[78] H. Huang, FPGA-based parallel metaheuristic PSO algorithm and its application to global path planning for autonomous robot navigation, *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3, pp. 475-488, 2014.

[79] J. Huang, L. Qu, R. Jia and B. Zhao, O2U-Net: A simple noisy label detection approach for deep neural networks, In: *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 3325-3333.

[80] Z. Huang, J. Zhang and H. Shan, Twin contrastive learning with noisy labels, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 11661-11670.

[81] A. Iscen, J. Valmadre, A. Arnab and C. Schmid, Learning with neighbor consistency for noisy labels, In: *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, USA, Jun. 2022, pp. 4662-4671.

[82] K. Ishaque, Z. Salam, M. Amjad and S. Mekhilef, An improved particle swarm optimization (PSO)–based MPPT for PV with reduced steady-state oscillation, *IEEE Transactions on Power Electronics*, vol. 27, no. 8, pp. 3627-3638, 2012.

[83] K. Ishaque, Z. Salam and A. Shamsudin, Application of particle swarm optimization for maximum power point tracking of PV system with direct control method, In: *Proceedings of the 37th Annual Conference of the IEEE Industrial Electronics Society*, Melbourne, Australia, Apr. 2011, pp. 1214-1219.

[84] M. Javaid and A. Haleem, Additive manufacturing applications in medical cases: A literature based review, *Alexandria Journal of Medicine*, vol. 54, no. 4, pp. 411-422, 2018.

[85] J. Jiang, G. Han, L. liu, L. Shu and M. Guizani, Outlier detection approaches based on machine learning in the internet-of-things, *IEEE Wireless Communications*, vol. 27, no. 3, pp. 53-59, 2020.

[86] J. Jiang, J. Ma and X. Liu, Multilayer spectral-spatial graphs for label noisy robust hyperspectral image classification, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 839-852, 2022.

[87] L. Jiang, Z. Zhou, T. Leung, L. Li and F. Li, Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels, In: *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, Jul. 2018, pp. 2304-2313.

[88] I. Jindal, D. Pressel, B. Lester and M. Nokleby, An effective label noise model for DNN text classification, *arXiv preprint arXiv:1903.07507*, 2019.

[89] A. R. Jordehi, Time varying acceleration coefficients particle swarm optimisation (TVACPSO): A new optimisation algorithm for estimating parameters of PV cells and modules, *Energy Conversion and Management*, vol. 129, pp. 262-274, 2016.

[90] L. Ju, X. Wang, L. Wang, D. Mahapatra, X. Zhao, Q. Zhou, T. Liu and Z. Ge, Improving medical images classification with label noise using dual-uncertainty estimation, *IEEE Transactions on Medical Imaging*, vol. 41, no. 6, pp. 1533-1546, 2022.

[91] Y. T. Kao, E. Zahara and I. W. Kao, A hybridized approach to data clustering, *Expert Systems with Applications*, vol. 34, no. 3, pp. 1754-1762, 2008.

[92] N. Karim, M. N.Rizve, N. Rahnavard, A. Mian and M. Shah, UNICON: Combating label noise through uniform selection and contrastive learning, In: *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, USA, Jun. 2022, pp. 9666-9676.

[93] J. Kennedy, The particle swarm: Social adaptation of knowledge, In: *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation*, Indianapolis, USA, Apr. 1997, pp. 303-308.

[94] J. Kennedy, Stereotyping: Improving particle swarm performance with cluster analysis, In: *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, USA, Jul. 2000, vol. 2, pp. 1507-1512.

[95] J. Kennedy and R. Eberhart, Particle swarm optimization, In: *Proceedings of the International Conference on Neural Networks*, Perth, Austrilia, Nov. 1995, pp. 1942-1948.

[96] A. Khare, and S. Rangnekar, A review of particle swarm optimization and its applications in solar photovoltaic system, *Applied Soft Computing*, vol. 13, no. 5, pp. 2997-3006, 2013.

[97] L. Kong, C. Lian, D. Huang, Z. Li, Y. Hu and Q. Zhou, Breaking the dilemma of medical image-to-image translation, In: *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS 2021)*, online, Dec. 2021, vol. 34.

[98] S. Kong, Y. Shen and L. Huang, Resolving training biases via influence-based data relabeling, In: *Proceedings of the 10th International Conference on Learning Representations*, online, Apr. 2022.

[99] R. Kundu, S. Das, R. Mukherjee and S. Debchoudhury, An improved particle swarm optimizer with difference mean based perturbation, *Neurocomputing*, vol. 129, pp. 315-333, 2014.

[100] C. Li and Z. Mao, A label noise filtering method for regression based on adaptive threshold and noise score, *Expert Systems with Applications*, vol. 228, art. no. 120422, 2023.

[101] J. Li, R. Socher and S. C. H. Hoi, DivideMix: Learning with noisy labels as semi-supervised learning, *arXiv preprint arXiv:2002.07394*, 2020.

[102] S. Li, X. Xia, S. Ge and T. Liu, Selective-supervised contrastive learning with noisy labels, In: *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, USA, Jun. 2022, pp. 316-325.

[103] Y. Li, L. Guo and Z. Zhou, Towards safe weakly supervised learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 334-346, 2021.

[104] Y. Li, H. Han, S. Shan and X. Chen, DISC: Learning from noisy labels via dynamic instance-specific selection and correction, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 24070-24079.

[105] J. Liang, A. Qin, P. N. Suganthan and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281-295, 2006.

[106] L. Liu, W. Liu, and D. A. Cartes, Particle swarm optimization-based parameter identification applied to permanent magnet synchronous motors, *Engineering Applications of Artificial Intelligence*, vol. 21, no. 7, pp. 1092-1100, 2008.

[107] M. Liu, F. Wang, X. Wang, Y. Wang and A. K. Roy-Chowdhury, A two-stage noise-tolerant paradigm for label corrupted person re-identification, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 7, pp. 4944-4956, 2024.

[108] S. Liu, J. Niles-Weed, N. Razavian and C. Fernandez-Granda, Early-learning regularization prevents memorization of noisy labels, In: *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*, online, Dec. 2020, vol. 33.

[109] W. Liu, Z. Wang, X. Liu, N. Zeng and D. Bell, A Novel particle swarm optimization approach for patient clustering from emergency departments, *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 632-644, 2019.

[110] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu and F. E. Alsaadi, A survey of deep neural network architectures and their applications, *Neurocomputing*, vol. 234, pp. 11-26, 2017.

[111] W. Liu, Z. Wang, Y. Yuan, N. Zeng, K. Hone and X. Liu, A Novel Sigmoid-Function-Based Adaptive Weighted Particle Swarm Optimizer, *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 1085-1093, 2021.

[112] W. Liu, Z. Wang, N. Zeng, Y. Yuan, F. E. Alsaadi and X. Liu, A novel randomised particle swarm optimizer, *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 2, pp. 529-540, 2021.

[113] Z. Liu, P. Ma, D. Chen, W. Pei and Q. Ma, Scale-teaching: Robust multi-scale training for time series classification with noisy labels, In: *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, New Orleans, USA, Dec. 2023, vol. 36.

[114] M. Long, J. Wang, G. Ding, S. J. Pan and P. S. Yu, Adaptation regularization: A general framework for transfer learning, *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1076-1089, 2014.

[115] M. Long, H. Zhu, J. Wang and M. I. Jordan, Deep transfer learning with joint adaptation networks, In: *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, Aug. 2017, pp. 2208-2217.

[116] W. Luo, S. Chen, T. Liu, B. Han, G. Niu, M. Sugiyama, D. Tao and C. Gang, Estimating per-class statistics for label noise learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 1, pp. 305-322, 2024.

[117] Y. Lyu and I. W. Tsang, Curriculum loss: Robust learning and generalization against label corruption, *arXiv preprint arXiv:1905.10045*, 2020.

[118] G. Ma, Z. Wang, W. Liu, J. Fang, Y. Zhang, H. Ding and Y. Yuan, Estimating the state of health for lithium-ion batteries: A particle swarm optimization-assisted deep domain adaptation approach, *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 7, pp. 1530-1543, 2023.

[119] E. Malach and S. Shalev-Shwartz, Decoupling "when to update" from "how to update", In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)*, Long Beach, USA, Dec. 2017, vol. 30.

[120] R. F. Malik, T. A. Rahman, S. Z. M. Hashim, and R. Ngah, New particle swarm optimizer with sigmoid increasing inertia weight, *International Journal of Computer Science and Security*, vol. 1, no. 2, pp. 35-44, 2007.

[121] G. Mattera, J. Polden, A. Caggiano, L. Nele, Z. Pan and J. Norrish, Semi-supervised learning for real-time anomaly detection in pulsed transfer wire arc additive manufacturing, *Robotics and Computer-Integrated Manufacturing*, vol. 128, pp. 84-94, 2024.

[122] M. Miyatake, M. Veerachary, F. Toriumi, N. Fujii and H. Ko, Maximum power point tracking of multiple photovoltaic arrays: A PSO approach, *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 1, pp. 367-380, 2011.

[123] F. Montevecchi, G. Venturini, A. Scippa and G. Hui, Finite element modelling of wire-arc-additive-manufacturing process, *Procedia CIRP*, vol. 55, pp. 109-114, 2016.

[124] A. Navaeefard, S. M. M. Tafreshi, M. Barzegari and A. J. Shahrood, Optimal sizing of distributed energy resources in microgrid considering wind energy uncertainty with respect to reliability, In: *Proceedings of the 2010 IEEE International Energy Conference*, Manama, Bahrain, Dec. 2010, pp. 820-825.

[125] T. D. Ngo, A. Kashani, G. Imbalzano, K. T. Q. Nguyen and D. Hui, Additive manufacturing (3D printing): A review of materials, methods, applications and challenges, *Composites Part B: Engineering*, vol. 143, pp. 172-196, 2018.

[126] D. T. Nguyen, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, L. Beggel and T. Brox, Self: Learning to filter noisy labels with self-ensembling, *arXiv preprint arXiv:1910.01842*, 2019.

[127] T. Niknam and B. Amiri, An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis, *Applied Soft Computing*, vol. 10, no. 1, pp. 183-197, 2010.

[128] T. Niu, Z. Wang, W. Li, K. Li, Y. Li, G. Xu and B. Li, Learning trustworthy model from noisy labels based on rough set for surface defect detection, *Applied Soft Computing*, vol. 165, art. no. 112138, 2024.

[129] A. Oord, Y. Li and O. Vinyals, Representation learning with contrastive predictive coding, *arXiv preprint arXiv:1807.03748*, 2018.

[130] D. Ortego, E. Arazo, P. Albert, N. E. O'Connor and K. McGuinness, Multi-objective interpolation training for robustness to label noise, In: *Proceedings of the 2021*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, USA, Jun. 2021, pp. 6602-6611.

[131] W. Ou, G. L. Knapp, T. Mukherjee, Y. Wei and T. DebRoy, An improved heat transfer and fluid flow model of wire-arc additive manufacturing, *International Journal of Heat and Mass Transfer*, vol. 167, art. no. 120835, 2021.

[132] S. J. Pan, I. W. Tsang, J. T. Kwok and Q. Yang, Domain adaptation via transfer component analysis, *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199-210, 2011.

[133] G. Pang, C. Shen, L. Cao and A. V. D. Hengel, Deep learning for anomaly detection: A review, *ACM Computing Surveys*, vol. 54, no. 2, pp. 1-38, 2021.

[134] B. K. Panigrahi, V. R. Pandi and S. Das, Adaptive particle swarm optimization approach for static and dynamic economic load dispatch, *Energy Conversion and Management*, vol. 49, no. 6, pp. 1407-1415, 2008.

[135] J. Park, Y. Jeong, J. Shin and K. Y. Lee, An improved particle swarm optimization for nonconvex economic dispatch problems, *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 156-166, 2010.

[136] S. Park, S. Han, S. Kim, D. Kim, S. Park, S. Hong and M. Cha, Improving unsupervised image clustering with robust learning, In: *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, USA, Jun. 2021, pp. 12273-12282.

[137] D. Patel and P. S. Sastry, Adaptive sample selection for robust learning under label noise, In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Hawaii, USA, Jan. 2023, pp. 3932-3942.

[138] G. Patrini, A. Rozza, A. K. Menon, R. Nock and L. Qu, Making deep neural networks robust to label noise: A loss correction approach, In: *Proceedings of the*

2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA, Jul. 2017, pp. 2233-2241.

[139] H. Pham, T. Le, D. Tran, D. Ngo and H. Nguyen, Interpreting chest X-rays via CNNs that exploit hierarchical disease dependencies and uncertainty labels, *Neurocomputing*, vol. 437, pp. 186-194, 2019.

[140] E. S. Pires, J. T. Machado, P. B. de Moura Oliveira, J. B. Cunha and L. Mendes, Particle swarm optimization with fractional-order velocity, *Nonlinear Dynamics*, vol. 61, no. 1, pp. 295-301, 2010.

[141] C. Pizzuti, Evolutionary computation for community detection in networks: A review, *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 464-483, 2018.

[142] J. Pugh, A. Martinoli and Y. Zhang, Particle swarm optimization for unsupervised robotic learning, In: *Proceedings of the 2005 IEEE Swarm Intelligence Symposium*, Pasadena, USA, Jun. 2005, pp. 92-99.

[143] J. Pugh and A. Martinoli, Multi-robot learning with particle swarm optimization, In: *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Hakodate, Japan, May 2006, pp. 441-448.

[144] X. Qin, P. Yao, M. Liu, X. Cheng, F. Shi and L. Guo, Robust classification of incomplete time series with noisy labels, In: *Proceedings of the 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Tianjin, China, May. 2024, pp. 2620-2625.

[145] A. Queguineur, G. Rückert, F. Cortial and J. Y. Hascoët, Evaluation of wire arc additive manufacturing for large-sized components in naval applications, *Welding in the World*, vol. 62, no. 2, pp. 259-266, 2018.

[146] P. Raja and S. Pugazhenthi, Optimal path planning of mobile robots: A review, *International Journal of Physical Sciences*, vol. 7, no. 9, pp. 1314-1320, 2012.

[147] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240-255, 2004.

[148] R. Reisch, T. Hauser, B. Lutz, M. Pantano, T. Kamps and A. Knoll, Distance-based multivariate anomaly detection in wire arc additive manufacturing, In: *Proceedings of the 19th IEEE International Conference on Machine Learning and Applications*, Miami, USA, Dec. 2020, pp. 659-664.

[149] N. A. Rosli, M. R. Alkahari, M. F. Abdollah, S. Maidin, F. R. Ramli and S. G. Herawan, Review on effect of heat input for wire arc additive manufacturing process, *Journal of Materials Research and Technology*, vol. 11, pp. 2127-2145, 2021.

[150] P. J. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53-65, 1987.

[151] T. A. Runkler and C. Katz, Fuzzy clustering by particle swarm optimization, In: *Proceedings of the 2006 IEEE International Conference on Fuzzy Systems*, Vancouver, Canada, Jul. 2006, pp. 601-608.

[152] A. Y. Saber, T. Senjyu, A. Yona and T. Funabashi, Unit commitment computation by fuzzy adaptive particle swarm optimisation, *IET Generation, Transmission & Distribution*, vol. 1, no. 3, pp. 456-465, 2007.

[153] S. Sengupta, S. Basak and R. A. Peters, Data clustering using a hybrid of fuzzy C-Means and quantum-behaved particle swarm optimization, In: *Proceedings of the 2018 IEEE 8th Annual Computing and Communication Workshop and Conference*, Las Vegas, USA, Jan. 2018, pp. 137-142.

[154] Y. Shen and S. Sanghavi, Learning with bad training data via iterative trimmed loss minimization, In: *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, Jun. 2019, pp. 5739-5748.

[155] X. Shi, H. Su, F. Xing, Y. Liang, G. Qu and L. Yang, Graph temporal ensembling based semi-supervised convolutional neural network with noisy labels for histopathology image analysis, *Medical Image Analysis*, vol. 60, art. no. 101624, 2020.

[156] Y. Shi and R. Eberhart, Parameter selection in particle swarm optimization, In: *Proceedings of the International Conference on Evolutionary Programming*, San Diego, USA, Mar. 1998, pp. 591-600.

[157] Y. Shi and R. Eberhart, A modified particle swarm optimizer, In: *Proceedings of the 1998 Congress on Evolutionary Computation*, Anchorage, USA, May 1998, pp. 69-73.

[158] Y. Shi and R. Eberhart, Empirical study of particle swarm optimization, In: *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, DC, USA, Jul. 1999, pp. 1945-1950.

[159] Y. Shi and R. Eberhart, Tracking and optimizing dynamic systems with particle swarms, In: *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul, Korea (South), May 2001, vol. 1, pp. 94-100.

[160] Y. Shi and R. Eberhart, Fuzzy adaptive particle swarm optimization, In: *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul, Korea (South), May 2001, vol. 1, pp. 101-106.

[161] J. Shin, J. Won, H. Lee and J. Lee, A review on label cleaning techniques for learning with noisy labels, *ICT Express*, vol. 10, no. 6, pp. 1315-1330, 2024.

[162] T. M. Silva Filho, B. A. Pimentel, R. M. Souza and A. L. Oliveira, Hybrid methods for fuzzy clustering based on fuzzy c-means and improved particle swarm optimization, *Expert Systems with Applications*, vol. 42, no. 17-18, pp. 6315-6328, 2015.

[163] B. Smart and G. Carneiro, Bootstrapping the relationship between images and their clean and noisy labels, In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Hawaii, USA, Jan. 2023, pp. 5344-5354.

[164] B. Song, Z. Wang and L. Zou, On global smooth path planning for mobile robots using a novel multimodal delayed PSO algorithm, *Cognitive Computation*, vol. 9, no. 1, pp. 5-17, 2017.

[165] B. Song, Z. Wang and L. Zou, An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve, *Applied Soft Computing*, vol. 100, art. no. 106960, 2021.

[166] B. Song, Z. Wang, L. Zou, L. Xu and F. E. Alsaadi, A new approach to smooth global path planning of mobile robots with kinematic constraints, *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 1, pp. 107-119, 2019.

[167] H. Song, M. Kim, D. Park, Y. Shin and J. Lee, Learning from noisy labels with deep neural networks: A survey, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 8135-8153, 2023.

[168] H. Song, C. Li, Y. Fu, R. Li, H. Zhang and G. Wang , A two-stage unsupervised approach for surface anomaly detection in wire and arc additive manufacturing, *Computers in Industry*, vol. 151, art. no. 103994, 2023.

[169] J. Soon and K. S. Low, Optimizing photovoltaic model parameters for simulation, In: *Proceedings of the 2012 IEEE International Symposium on Industrial Electronics*, Hangzhou, China, May 2012, pp. 1813-1818.

[170] H. Sun, Q. Wei, L. Feng, Y. Hu, F. Liu, H. Fan and Y. Yin, Variational rectification inference for learning with noisy labels, *International Journal of Computer Vision*, vol. 133, pp. 652-671, 2024.

[171] J. Sun, B. Feng and W. Xu, Particle swarm optimization with particles having quantum behaviour, In: *Proceedings of the 2004 Congress on Evolutionary Computation*, Portland, USA, Jun. 2004, vol. 1, pp. 325-331.

[172] J. Sun, W. Xu and B. Feng, A global search strategy of quantum-behaved particle swarm optimization, In: *Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems*, Singapore, Dec. 2004, vol. 1, pp. 111-116.

[173] Z. Sun, H. Liu, Q. Wang, T. Zhou, Q. Wu and Z. Tang, Co-ldl: A co-training-based label distribution learning method for tackling label noise, *IEEE Transactions on Multimedia*, vol. 24, pp. 1093-1104, 2022.

[174] C. Tan, J. Xia, L. Wu and S. Li, Co-learning: Learning from noisy labels with self-supervision, In: *Proceedings of the 29th ACM International Conference on Multimedia*, China, Oct. 2021, pp. 1405-1413.

[175] D. Tan, Y. Chen, T. Chen and W. Chen, Trustmae: A noise-resilient defect classification framework using memory-augmented auto-encoders with trust regions, In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Hawaii, USA, Jan. 2021, pp. 276-285.

[176] D. Tanaka, D. Ikami, T. Yamasaki and K. Aizawa, Joint optimization framework for learning with noisy labels, In: *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, USA, Jun. 2018, pp. 5552-5560.

[177] Y. Tang, Z. Wang and J. Fang, Parameters identification of unknown delayed genetic regulatory networks by a switching particle swarm optimization algorithm, *Expert Systems with Applications*, vol. 38, no. 3, pp. 2523-2535, 2011.

[178] F. Thabtah, S. Hammoud, F. Kamalov and A. Gonsalves, Data imbalance in classification: Experimental evaluation, *Information Sciences*, vol. 513, pp. 429-441, 2020.

[179] D. Tian, X. Zhao and Z. Shi, Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization, *Swarm and Evolutionary Computation*, vol. 51, art. no. 100573, 2019.

[180] C. Y. Tsai and I. W. Kao, Particle swarm optimization with selective particle regeneration for data clustering, *Expert Systems with Applications*, vol. 38, no. 6, pp. 6565-6576, 2011.

[181] Y. Tu, B. Zhang, Y. Li, L. Liu, J. Li, Y. Wang, C. Wang and C. Zhao, Learning from noisy labels with decoupled meta label purifier, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 19934-19943.

[182] Y. Tu, B. Zhang, Y. Li, L. Liu, J. Li, J. Zhang, Y. Wang, C. Wang and C. R. Zhao, Learning with noisy labels via self-supervised adversarial noisy masking, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 16186-16195.

[183] B. Tudu, S. Majumder, K. K. Mandal and N. Chakraborty, Comparative performance study of genetic algorithm and particle swarm optimization applied on off-grid renewable hybrid energy system, In: *Proceedings of the International Conference on Swarm, Evolutionary, and Memetic Computing*, Visakhapatnam, India, Dec. 2011, pp. 151-158.

[184] F. van den Bergh and A. P. Engelbrecht, A cooperative approach to particle swarm optimization, *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225-239, 2004.

[185] T. Velmurugan and T. Santhanam, Performance evaluation of K-means and fuzzy C-means clustering algorithms for statistical distributions of input data points, *European Journal of Scientific Research*, vol. 46, no. 3, pp. 320-330, 2010.

[186] P. Verma, M. Sinha and S. Panda, Fuzzy C-means clustering-based novel threshold criteria for outlier detection in electronic nose, *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1975-1981, 2021.

[187] T. A. A. Victoire and A. E. Jeyakumar, Reserve constrained dynamic dispatch of units with valve-point effects, *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1273-1282, 2005.

[188] G. Wang, X. Liu, C. Li, Z. Xu, J. Ruan, H. Zhu, T. Meng, K. Li, N. Huang and S. Zhang, A noise-robust framework for automatic segmentation of COVID-19 pneumonia lesions from CT images, *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, pp. 2653-2663, 2020.

[189] H. Wang, C. Li, P. Ding, S. Li, T. Li, C. Liu, X. Zhang and Z. Hong, A novel transformer-based few-shot learning method for intelligent fault diagnosis with noisy labels under varying working conditions, *Reliability Engineering & System Safety*, vol. 251, art. no. 110400, 2024.

[190] H. Wang, B. Liu, C. Li, Y. Yang and T. Li, Learning with noisy labels for sentence-level sentiment classification, *arXiv preprint arXiv:1909.00124*, 2019.

[191] H. Wang, R. Xiao, Y. Dong, L. Feng and J. Zhao, Promix: Combating label noise via maximizing clean sample utility, *arXiv preprint arXiv:2207.10276*, 2022.

[192] J. Wang, Y. Chen, S. Hao, W. Feng and Z. Shen, Balanced distribution adaptation for transfer learning, In: *Proceedings of the 2017 IEEE International Conference on Data Mining*, New Orleans, USA, Nov. 2017, pp. 1129-1134.

[193] R. Wang, X. Wu, T. Xu, C. Hu and J. Kittler, U-SPDNet: An SPD manifold learning-based neural network for visual classification, *Neural Networks*, vol. 161, pp. 382-396, 2023.

[194] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song and S. Xia, Iterative learning with open-set noisy labels, In: *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, USA, Jun. 2018, pp. 8688-8696.

[195] H. Wei, L. Feng, X. Chen and B. An, Combating noisy labels by agreement: A joint training method with co-regularization, In: *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, Jun. 2020, pp. 13723-13732.

[196] Q. Wei, L. Feng, H. Sun, R. Wang, C. Guo and Y. Yin, Fine-grained classification with noisy labels, In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, Jun. 2023, pp. 11651-11660.

[197] K. Weiss, T. M. Khoshgoftaar and D. D. Wang, A survey of transfer learning, *Journal of Big Data*, vol. 3, art. no. 9, 2016.

[198] B. Wu, Z. Pan, D. Ding, D. Cuiuri, H. Li, J. Xu and J. Norrish, A review of the wire arc additive manufacturing of metals: Properties, defects and quality improvement, *Journal of Manufacturing Processes*, vol. 35, pp. 127-139, 2018.

[199] D. Wu, V. J. Lawhern, S. Gordon, B. J. Lance and C. T. Lin, Driver drowsiness estimation from EEG signals using online weighted adaptation regularization for regression (OwARR), *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1522-1535, 2017.

[200] H. Wu, B. Jia and G. Sheng, Early-late dropout for DivideMix: Learning with noisy labels in deep neural networks, In: *Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN)*, Yokohama, Japan, Jul. 2024, pp. 1-8.

[201] S. Wu, T. Zhou, Y. Du, J. Yu, B. Han and T. Liu, A time-consistency curriculum for learning from instance-dependent noisy labels, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 7, pp. 4830-4842, 2024.

[202] J. Xia, H. Lin, Y. Xu, C. Tan, L. Wu, S. Li and S. Z. Li, Gnn cleaner: Label cleaner for graph structured data, *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 2, pp. 640-651, 2024.

[203] S. Xia, S. Zheng, G. Wang, X. Gao and B. Wang, Granular ball sampling for noisy label classification or imbalanced classification, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 4, pp. 2144-2155, 2023.

[204] X. Xia, B. Han, Y. Zhan, J. Yu, M. Gong, C. Gong and T. Liu, Combating noisy labels with sample selection by mining high-discrepancy examples, In: *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, Oct. 2023, pp. 1833-1843.

[205] X. Xia, P. Lu, C. Gong, B. Han, J. Yu, J. Yu and T. Liu, Regularly truncated M-estimators for learning with noisy labels, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3522-3536, 2024.

[206] K. Xia, L. Wang, S. Zhou, G. Hua and W. Tang, Learning from noisy pseudo labels for semi-supervised temporal action localization, In: *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, Oct. 2023, pp. 10126-10135.

[207] T. Xiao, T. Xia, Y. Yang, C. Huang and X. Wang, Learning from massive noisy labeled data for image classification, In: *Proceedings of the 2015 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, USA, Jun. 2015, pp. 2691-2699.

[208] X. Xiao, E. R. Dow, R. Eberhart, Z. B. Miled and R. J. Oppelt, Gene clustering using self-organizing maps and particle swarm optimization, In: *Proceedings of the International Parallel and Distributed Processing Symposium*, Nice, France, Apr. 2003, pp. 10-19.

[209] J. Xin, G. Chen and Y. Hai, A particle swarm optimizer with multi-stage linearly-decreasing inertia weight, In: *Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization*, Sanya, China, Apr. 2009, pp. 505-508.

[210] Y. Xu, Z. Li, W. Li, Q. Du, C. Liu, Z. Fang and L. Zhai, Dual-channel residual network for hyperspectral image classification with noisy labels, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, art. no. 5502511, 2022.

[211] C. Xue, Q. Dou, X. Shi, H. Chen and P. A. Heng, Robust learning at noisy labeled medical images: Applied to skin lesion classification, In: *Proceedings of the 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, Venice, Italy, Apr. 2019, pp. 1280-1283.

[212] B. Xue, M. Zhang and W. N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656-1671, 2013.

[213] Y. Xue, B. Xue and M. Zhang, Self-adaptive particle swarm optimization for large-scale feature selection in classification, *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 5, pp. 1-27, 2019.

[214] T. Yamaguchi and K. Yasuda, Adaptive particle swarm optimization; self-coordinating mechanism with updating information, In: *Proceedings of the 2006 IEEE International Conference on Systems, Man and Cybernetics*, Taipei, Taiwan, Oct. 2006, pp. 2303-2308.

[215] J. Yan, L. Luo, C. Deng and H. Huang, Adaptive hierarchical similarity metric learning with noisy labels, *IEEE Transactions on Image Processing*, vol. 32, pp. 1245-1256, 2023.

[216] J. Yao, J. Wang, I. W. Tsang, Y. Zhang, J. Sun, C. Zhang and R. Zhang, Deep learning from noisy image labels with quality embedding, *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1909-1922, 2019.

[217] Q. Yao, H. Yang, B. Han, G. Niu and J. T. Kwok, Searching to exploit memorization effect in learning with noisy labels, In: *Proceedings of the 37th International Conference on Machine Learning*, online, Jul. 2020, pp. 10789-10798.

[218] M. Ye, H. Li, B. Du, J. Shen, L. Shao and S. C. H. Hoi, Collaborative refining for person re-identification with label noise, *IEEE Transactions on Image Processing*, vol. 31, pp. 379-391, 2022.

[219] M. Ye and P. C. Yuen, PurifyNet: A robust person re-identification model with noisy labels, *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2655-2666, 2020.

[220] N. Yin, L. Shen, M. Wang, X. Luo, Z. Luo and D. Tao, OMG: Towards effective graph classification against label noise, *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 12873-12886, 2023.

[221] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang and M. Sugiyama, How does disagreement help generalization against label corruption?, In: *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, Jun. 2019, pp. 7164-7173.

[222] J. Yuan, X. Luo, Y. Qin, Y. Zhao, W. Ju and M. Zhang, Learning on graphs under label noise, In: *Proceedings of the 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, Jun. 2023, pp. 1-5.

[223] C. Yue and N. K. Jha, Ctrl: Clustering training losses for label error detection, *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 8, pp. 4121-4135, 2024.

[224] N. Zeng, Z. Wang, Y. Li, M. Du and X. Liu, A hybrid EKF and switching PSO algorithm for joint state and parameter estimation of lateral flow immunoassay models, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 2, pp. 321-329, 2012.

[225] N. Zeng, Z. Wang, W. Liu, H. Zhang, K. Hone and X. Liu, A dynamic neighbourhood-based switching particle swarm optimization algorithm, *IEEE Transactions on Cybernetics*, vol. 52, no. 9, pp. 9290-9301, 2022.

[226] N. Zeng, Z. Wang, H. Zhang and F. E. Alsaadi, A novel switching delayed PSO algorithm for estimating unknown parameters of lateral flow immunoassay, *Cognitive Computation*, vol. 8, no. 2, pp. 143-152, 2016.

[227] N. Zeng, H. Zhang, Y. Chen B. Chen and Y. Liu, Path planning for intelligent robot based on switching local evolutionary PSO algorithm, *Assembly Automation*, vol. 36, no. 2, pp. 120-126, 2016.

[228] Z. Zhan, J. Zhang, Y. Li and H. S. Chung, Adaptive particle swarm optimization, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 6, pp. 1362-1381, 2009.

[229] C. Zhang, S. Bengio, M. Hardt, B. Recht and O. Vinyals, Understanding deep learning (still) requires rethinking generalization, *Communications of the ACM*, vol. 64, no. 3, pp. 107-115, 2021.

[230] J. Zhang, B. Song, H. Wang, B. Han, T. Liu, L. Liu and M. Sugiyama, BadLabel: A robust perspective on evaluating and enhancing label-noise learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 6, pp. 4398-4409, 2024.

[231] K. Zhang, B. Tang, L. Deng, Q. Tan and H. Yu, A fault diagnosis method for wind turbines gearbox based on adaptive loss weighted meta-ResNet under noisy labels, *Mechanical Systems and Signal Processing*, vol. 161, art. no. 107963, 2021.

[232] Y. Zhang, W. Deng, M. Wang, J. Hu, X. Li, D. Zhao and D. Wen, Global-local GCN: Large-scale label noise cleansing for face recognition, In: *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, Jun. 2020, pp. 7728-7737.

[233] Y. Zhang, C. Wang, X. Ling and W. Deng, Learn from all: Erasing attention consistency for noisy label facial expression recognition, In: *Proceedings of the European Conference on Computer Vision (ECCV)*, Tel Aviv, Israel, Oct. 2022, pp. 418-434.

[234] Z. Zhang, W. Chen, C. Fang, Z. Li, L. Chen, L. Lin and G. Li, RankMatch: Fostering confidence and consistency in learning with noisy labels, In: *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, Oct. 2023, pp. 1644-1654.

[235] Z. Zhang, H. Zhang, S. Ö. Arik, H. Lee and T. Pfister, Distilling effective supervision from severe label noise, In: *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, Canada, Jun. 2020, pp. 9291-9300.

[236] Y. Zhao, J. Zhan, Y. Zhang, D. Wang and B. Zou, The optimal capacity configuration of an independent Wind/PV hybrid power supply system based on improved PSO algorithm, In: *Proceedings of the 8th International Conference on Advances in Power System Control, Operation and Management*, Hong Kong, China, Nov. 2009, pp. 1-7.

[237] J. Zhong, N. Li, W. Kong, S. Liu, T. Li and G. Li, Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection, In: *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, USA, Jun. 2019, pp. 1237-1246.

[238] X. Zhou, X. Liu, C. Wang, D. Zhai, J. Jiang and X. Ji, Learning with noisy labels via sparse regularization, In: *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, Canada, Oct. 2021, pp. 72-81.

[239] Z. Zhou, A brief introduction to weakly supervised learning, *National Science Review*, vol. 5, no. 1, pp. 44-53, 2018.

[240] D. Zhu, M. A. Hedderich, F. Zhai, D. I. Adelani and D. Klakow, Is BERT robust to label noise? A study on learning with noisy labels in text classification, *arXiv preprint arXiv:2204.09371*, 2022.