Data Privacy Preserving for Centralized Robotic Fault Diagnosis with Modified Dataset Distillation

Tao Wang[a, b], Yu Huang[a, b], Ying Liu[c] and Chong Chen[a*]

[a] *Guangdong Provincial Key Laboratory of Cyber-Physical System, Guangdong University of Technology, Guangzhou 510006, China*

[b] *School of Automation, Guangdong University of Technology, Guangzhou 510006, China.*

[c] *Department of Mechanical Engineering, School of Engineering, Cardiff University, Cardiff CF24 3AA, UK*

[*] *Corresponding author. E-mail: chenc2021@gdut.edu.cn*

Abstract

Industrial robots generate monitoring data rich in sensitive information, often making enterprises reluctant to share, which impedes the use of data in fault diagnosis modeling. Dataset distillation (DD) is an effective approach to condense large dataset into smaller, synthesized forms, focusing solely on fault-related features, which facilitates secure and efficient data transfer for diagnostic purposes. However, the challenge of achieving satisfactory fault diagnosis accuracy with distilled data stems from the computational complexity in data distillation process. To address this problem, this paper proposes a Modified KernelWarehouse (MKW) network-based DD method to achieve accurate fault diagnosis with the distilled dataset. In this algorithm, DD first generates distilled training and testing dataset, followed by the training of an MKW-based network based on these distilled datasets. Specifically, MKW reduces network complexity through the division of static kernels into disjoint kernel cells, which are then computed as linear mixtures from a shared warehouse. An experimental study based on the real-world robotic dataset reveals the effectiveness of the proposed approach. The experimental results indicate that the proposed method can achieve a fault diagnosis accuracy of 86.3% when only trained with distilled data.

***Keywords**:* Data privacy; Dataset distillation; Fault diagnosis; Industrial robot.

1 Introduction

The advance of Industry 4.0 has revolutionized industrial asset maintenance through breakthroughs in the Industrial Internet-of-Thing, cyber-physical systems, and artificial intelligence [1]. These assets have become more automated, intelligent, and intricate, presenting new challenges in fault diagnosis [2]. Fault diagnosis in these advanced systems is crucial because it aids in early anomaly detection and helps prevent potential downtimes, which could result in significant financial losses and safety risks [3, 4]. With the growing complexity of industrial robots, the methods for fault diagnosis have evolved, necessitating advanced analytical techniques to manage these intricate systems effectively. Therefore, to ensure the safety and efficiency of production, it is necessary to conduct intelligent fault diagnosis on industrial robots.

Fault diagnosis involves the evaluation and interpretation of measurement signals to monitor the condition of mechanical equipment, either under static conditions or during operation [5, 6]. In industrial robot fault diagnosis, the signals collected from reducers and motors under different axes are typically transformed into time-frequency data using Continuous Wavelet Transform (CWT) [7]. However, the time-frequency data might include detailed insights into production methods, machinery performance, and productivity figures, all crucial for maintaining a company’s competitive advantage. Moreover, as IIoT applications become more prevalent, they pose risks of privacy breaches. Challenges include data leakage during cloud storage and sharing, compounded by variable access rights and lack of access constraints [8]. Companies resist consolidating their data in a single, centralized spot for collaborative model training. This creates a paradoxical situation where companies are inclined to engage in a shared data ecosystem, they are reluctant to exchange their unprocessed commercial data with others [9]. Hence, it is important to draw more attention to industrial privacy-preserving.

Recent advancements in privacy protection for fault diagnosis have become increasingly crucial in the field of industrial data processing [10]. These advancements aim to preserve sensitive information inherent in monitoring data. DD is a centralized privacy-preserving approach, aiming to lessen storage and transmission loads from large data sets and streamline model training, ensuring the distilled data maintains essential information for effective classification [11]. By employing DD, enterprises can condense and encrypt large volumes of data into more manageable forms, ensuring the preservation of sensitive information [12]. This method enhances data security and transfer efficiency and improves data processing, making it useful for managing large, complex industrial datasets. The main challenge for DD lies in distilling a smaller dataset and achieving high fault diagnosis accuracy. The performance of the distilled dataset on a larger network is significantly worse than that of simple networks. This is because the distilled dataset is small in size, while complex networks are hard to be trained on such a limited dataset. However, when the size of the distilled dataset grows, it will greatly increase the computational burden of the dataset distillation algorithm. Therefore, investigating a lightweight and efficient network becomes a potential solution [13].

Lightweight Convolutional Neural Networks (CNNs) have evolved to address the need for efficient deep learning models that are optimized for performance with limited computational resources. Nonetheless, current lightweight CNN still struggles with design complexity, optimization, and scalability challenges. KernelWarehouse (KW), as a general form of dynamic convolution, represents another special form of reducing the number of network parameters [14]. The KW method innovates by redefining kernel concepts and assembly in dynamic convolution, using smaller but more numerous kernels. The ‘warehouse’ could store the parameters and share them within the convolution layers. It enhances convolutional parameter dependencies within the same layer and across successive layers by dividing the kernel and sharing resources. However, this approach encounters challenges in optimization, especially when the number of kernel cells is considerably large. In these situations, the KW often lacks stability, causing variable training accuracy and convergence problems, which can prolong the network fine-tuning time needed to reach the desired performance.

To address these challenges, this paper proposes an optimized method named MKW, which improves the attention mechanism, enabling networks equipped with KW to effectively integrate with DD. This optimization strategy aims to enhance the network’s stability and computational efficiency and increase its accuracy and effectiveness in processing noisy data. The main contribution can be concluded: (1) A lightweight CNN-based method for industrial robot time-frequency data for fault diagnosis is proposed. (2) The MKW method effectively reduces training instability and slow training times in the KW approach by incorporating a more stable and effective attention function. (3) An experimental study was implemented based on the industrial robot, which reveals that the proposed approach shows merits in comparison with other CNN architecture models. The remainder of this article is organized as follows: Section

2 reviews the related works on recent advances in the construction of Differential Privacy and lightweight CNNs. The methodology of this paper is given in Section 3; The experimental setup is presented in Section 4 and the results are demonstrated in Section 5; Finally, Section 6 discusses the experimental results and Section 7 provides the conclusion.

2 Literature Review

With the rapid advancements of IIoT technology and the ongoing deepening of global network information, privacy protection has become a key issue in industrial intelligent manufacturing. Despite significant progress in this field over the past decade, there remains a substantial research gap in the effectiveness of privacy data in the context of Differential Privacy (DP) and lightweight neural networks. This literature review aims to explore and evaluate existing research in these areas, revealing not only the developmental trajectory and main points of contention in the field but also identifying the potential challenges faced by this research. By reviewing related literature, this review will provide an overview of the current state and shortcomings of these complex technological areas, as well as the challenges to be faced by this research.

2.1 Recent advances in Privacy Preserving

The concept of DP was first proposed by Dwork et al. [15]. This technique guarantees the privacy and utility of a dataset with a rigorous theoretical foundation [16]. The algorithm aims to promote the use of privacy-preserving methods in healthcare and other sectors, providing support to researchers and practitioners as they navigate the complex challenges involved in achieving broad implementation. Cheu et al. [17] proposed the analytical exploration of a shuffled model for distributed differentially private algorithms, positioned between the local and central models. The Sparse Vector Technique is key in maintaining DP and has a unique ability to provide certain query responses without apparently compromising privacy.

Federated learning (FL) has gained considerable interest in recent years, the key concern in FL is the safeguarding of privacy. Stacey et al. [18] proposed an FL system which has the ability to protect against inference on the messages shared during training and the final trained model, while also ensuring that the resultant model possesses satisfactory predictive accuracy. Liu et al. [19] introduced a Privacy-Enhanced Federated Learning framework, which utilizes homomorphic encryption as its core technology. This framework allows the server to penalize malicious actors by effectively extracting gradient data through the logarithmic function. Wei et al. [20] designed the principle of Local Differential Privacy and suggested a User-Level Differential Privacy algorithm. This approach is designed to tackle the issue of a curious server attempting to deduce private information from the shared models uploaded by Mobile Terminals. Wu et al. [21] introduced a DP mechanism to resist various background knowledge attacks. In order to protect users' privacy and improve the test accuracy of FL, Zhao et al. [22] designed a new normalization method with the enforcement of DP on the extracted features. Additionally, to attract more customers to participate in the crowdsourcing FL task, they designed an incentive mechanism to award participants. Metha et al. [23] proposed an FL framework to address data scarcity and privacy concerns in semantic segmentation for additive manufacturing, and innovated in privacy protection by implementing FL for semantic segmentation in additive manufacturing, enabling collaborative model training without direct data sharing, thus preserving data confidentiality. Chen et al. [24] devised the Decentralized Wireless Privacy Preserving Federated Learning algorithm, primarily aimed at improving wireless IoT networks. This algorithm addresses the key issues found in traditional FL architectures, such as limited fault tolerance, high communication overhead, and difficulties in accessing private data. By organizing workers in a peer-to-peer and server-less structure and enabling the parallel exchange of privacy-protected data through analog transmission over wireless channels, the algorithm enhances both efficiency and privacy in FL systems. Wu et al. [25] introduced a framework for privacy-preserving data mining in edge computing, utilizing private random decision trees. This framework is designed to provide strong privacy guarantees while maintaining a reasonable data utility level. The algorithm further increased data utility while still providing strong privacy preservation. This improvement is crucial for the practical applicability of the framework in real-world scenarios.

DD keeps the model fixed and instead attempts to distill the knowledge from a large training dataset into a small one [11]. Dong et al. [12] highlighted the role of DD in enhancing data privacy protection, presenting it as a means to prevent accidental data breaches. They integrated DD methods into the privacy sector and provided a theoretical analysis of its relationship with differential privacy. Chen et al. [26] further employed DD to create high-dimensional data under DP assurances, enabling private data sharing with reduced memory and computational requirements. Significant progress has been made to approve the performance of DD. Zhou et al. [27] proposed an algorithm by utilizing neural Feature Regression with Pooling to improve the effectiveness of the distilled dataset. Timothy et al. [28] introduced a meta-learning algorithm named Kernel Inducing Points, which significantly improved previous DD methods. Zhao et al. [29] optimized the distillation method by matching gradient matching loss. They also proposed a novel dataset condensation method based on distribution matching, improving efficiency and potential [30]. This method identified two major issues in traditional approaches: imbalanced feature numbers and unvalidated embeddings for distance computation. To address these issues, they designed three innovative techniques: partitioning and expansion augmentation, efficient and enriched model sampling, and class-aware distribution regularization. Cazenavette et al. [31] introduced a novel approach in DD by matching training trajectories, and refining distilled data to steer networks towards performance comparable to training with real data. This technique involves multiple training iterations using the distilled data, followed by optimization based on the disparity between parameters trained synthetically and those trained with real dataset.

2.2 The research on lightweight convolutional neural networks

In the field of image recognition, the application of CNNs is becoming increasingly widespread, the fundamental of CNN involves applying convolution extraction to localized regions within an image [32]. The core principle of CNNs involves the use of convolutional filters to extract features from localized image regions, a technique that has been significantly refined over the years. Over the past decade, the fields of deep learning and computer vision have experienced significant development. Many groundbreaking CNN architectures such as AlexNet [33] have emerged, greatly enhancing performance on the ImageNet dataset. The increasing complexity of CNN models presents a challenge for devices with limited processing capabilities, necessitating the development of more efficient, lightweight models. The drive for smaller, faster, yet accurate CNNs aims to reduce computational demands and bring advanced image recognition to mobile and edge devices. This endeavor promises to broaden the applications of CNNs, ensuring better performance and quicker responsiveness across various platforms, from smartphones to self-driving cars. So, there is an ongoing effort to create more compact neural network models. The goal is to maintain the precision of these models, making them smaller and faster. To enhance power efficiency and ensure portability for embedded platforms. Some small-size CNN architectures were proposed, like MobileNet [34], ShuffleNet [35] and SqueezeNet [36]. Zhu et al. [34] explored network lightweighting and performance optimization by using the MobileNet. They adapted the structure for one-dimensional signal processing, incorporating wavelet convolution to improve feature extraction and robustness, and demonstrated effective noise-resistant classification performance in experiments on gears and bearings. ShuffleNet enhances efficiency and reduces parameters using pointwise group convolution and channel shuffle, allowing for complex structures with multiple convolutional layers but lower complexity and enhanced feature encoding. Luo et al. [35] utilized a combination of multiple features and an enhanced version of ShuffleNet V2 that achieved high fault recognition accuracy for rolling bearings at variable speeds, while maintaining a moderate model size, effectively enhancing accuracy without significantly increasing the model’s size. Wang et al. [37] introduced a novel approach for thorough intelligent diagnosis, utilizing the ShuffleNet Lightweight Convolution Neural Network (SLCNN). The SLCNN not only excels in various performance measures but also proves effective in precise and reliable fault diagnosis of power equipment. Moreover, a comparison of feature maps revealed that insulation defects are less distinct in appearance and boundaries compared to mechanical faults, indicating greater challenges in diagnosing insulation issues. SqueezeNet was first proposed by Iandola et al. [36]. This architecture achieved AlexNet-level accuracy with significantly fewer parameters, making it highly efficient for deployment in environments with limited computational resources. Zhong et al. [38] designed a Self-Attention Ensemble Lightweight Model with Transfer Learning (SLTL) method. This method addressed the challenges of creating a lightweight model and reducing the reliance on large training datasets in the context of bearing fault classification using deep learning.

2.3 A brief summary

The main concept of DD involves creating a smaller dataset from a larger original one to achieve comparable performance. The existing research on DD indicates that constructing a distilled dataset could be challenging. The process of generating a distilled dataset is still bonded with limited network architecture. Large and complex models usually perform worse than simple networks. In order to establish DD with a wider application towards different networks and datasets. The lightweight convolution networks could be a potential solution to it.

Existing lightweight CNN architectures have the potential to become smaller, faster, and more accurate compared to typical large CNNs, aiming to fulfill the requirements of limited computational resources. However, the current lightweight networks are often challenged in balancing the complexity and performance of the model. Hence, the challenges for constructing DD with a lightweight model are: (1) When performing a dataset distillation task, the network architecture is limited to perform an effective distillation task; (2) The existing lightweight convolutional network may not be able to handle the balance between complexity and performance of the model, which may lead to bad performance to the distilled dataset. Therefore, it is worthwhile to investigate an MKW approach that takes advantage of lightweight to achieve DD.

3 Methodology

In this section, the overall flow of constructing DD and the proposed MKW algorithm is elaborated. The overall flowchart of the construction is shown in Figure 1. First, CWT is used to convert the time-series data collected from industrial robots into time-frequency images. The transformed data could hardly be reversed into time-frequency signals due to the potential information loss and computational complexity. For the collected original time-series data, CWT adjusts to different scales by scaling and shifts the same mother wavelet function, offering better time resolution at high frequencies and better frequency resolution at low frequencies. These images contain key information related to manufacturer privacy, such as operational patterns. To protect these private data, the paper employs the DD algorithm. Specifically, a CNN is used to generate a distilled dataset for training and testing. The distilled dataset is specially designed to retain diagnostic features from the original time-frequency images while encrypting the private information. Then, we use a model based on the proposed MKW method to perform a fault diagnosis with the distilled data. Thus, model training can solely rely on the distilled dataset, without needing to access the real-time frequency data that contain private data.

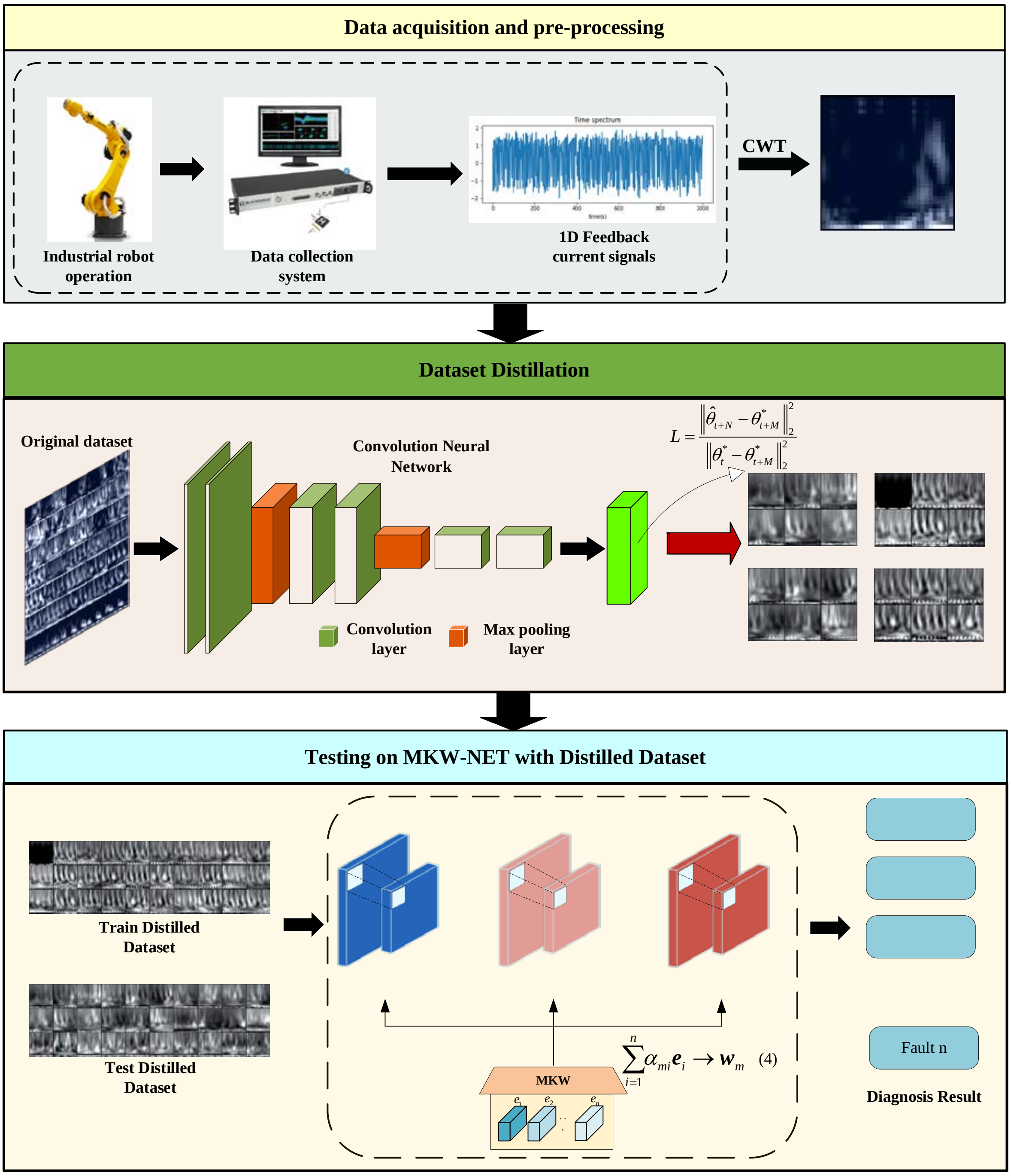

Data acquisition and pre-processing
Industrial robot operation
Data collection system
Time spectrum
1D Feedback current signals
CWT
Dataset Distillation
Original dataset
Convolution Neural Network
$L = \frac{\left\| \hat{\theta}_{t+N} - \theta^*_{t+M} \right\|_2^2}{\left\| \theta^*_t - \theta^*_{t+M} \right\|_2^2}$
Convolution layer
Max pooling layer
Testing on MKW-NET with Distilled Dataset
Train Distilled Dataset
Test Distilled Dataset
MKW
e_1
e_2
e_n
$\sum_{i=1}^{n} \alpha_{mi} \boldsymbol{e}_i \rightarrow \boldsymbol{w}_m$ (4)
Fault n
Diagnosis Result

Figure 1. The overall flow of the methodology.

After generating the distilled dataset, the distilled dataset is trained in a new lightweight CNN named MKW-NET, the architecture is based on the MKW method for fault diagnosis.

Compared to traditional CNNs, this network has a simpler structure but fewer parameters, making it more suitable for efficient industrial fault diagnosis while protecting privacy. The model trained on the distilled dataset can ultimately output fault prediction results and their accuracy.

3.1 MKW-Network architecture

The fundamental concept of MKW redefines the fundamental concepts of ‘kernel’ and ‘assembling kernel’ in dynamic convolutions from the perspectives of reducing kernel dimensions and significantly increasing the number of kernels. Specifically, MKW first divides the static kernels of any convolutional layer of a CNN into m disjoint kernel units with the same dimensions, then computes each kernel unit as a linear mix based on a predefined ‘warehouse’ composed of n kernel units. This warehouse is also shared across multiple adjacent convolutional layers. The static kernel is finally replaced by sequentially assembling the corresponding m mixtures, thereby generating a high degree of freedom to fit the desired parameter budget. The main structure of MKW is shown in Figure 2.

Dynamic convolution differs from regular convolution in that it learns a mixed convolutional kernel made from a linear blend of n static kernels, weighted by their sample-related attention. However, existing designs generally have issues with parameter efficiency due to increasing the number of dynamic convolutional kernels by a factor of n. MKW addresses this by cleverly using kernel partition and warehouse sharing. This enhances the dependency of convolutional parameters within the same layer and across consecutive layers while reducing the number of kernels in a typical dynamic convolutional network.

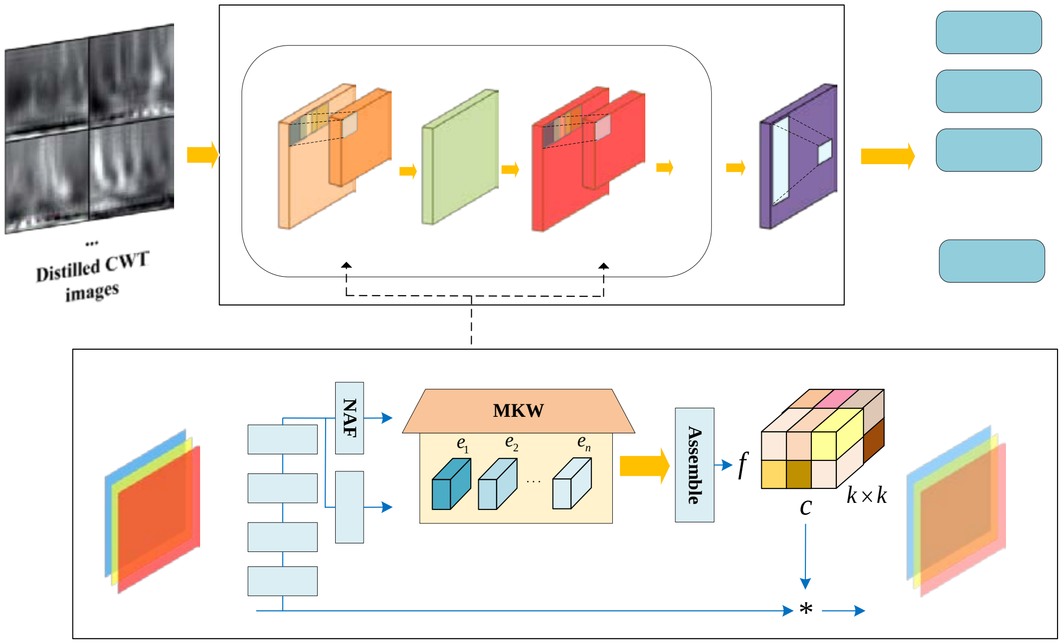


Figure 2. The main structure of MKW.

In MKW, a larger number of kernel cells are linearly mixed from a shared warehouse, which serves not only a single convolutional layer but also spans across $l-1$ other convolutional layers in the network at the same hierarchical level. Consequently, the selection of an optimal attention function becomes pivotal. The attention function in MKW plays a crucial role in dynamically generating weights for kernel cells stored in a 'warehouse'. This function operates by normalizing sets of feature logits, which are derived from the network's second fully connected (FC) layer, in parallel. By adjusting each kernel cell's importance based on input data characteristics, it enables targeted feature extraction. This process streamlines convolutional layer focus, improving efficiency in recognizing and processing key information. The ideal attention mechanism should simultaneously distribute diverse attention to all linear mixtures, thereby empowering the l convolutional layers to extract meaningful features effectively and hierarchically. To address this intricate optimization problem, we propose an innovative attention function. In this function, the attention allocated to the i^{th} kernel cell in the static kernel $\mathbf{W}$ is informed by the feature logits $z_{i1}, \dots, z_{in}$ derived from the network's second FC layer.

$$\alpha_{ij} = \frac{z_{ij}}{\sum_{p=1}^{n} |z_{ip}|} \text{ , } j \in \{1, \dots, n\} \tag{1}$$

where $\frac{z_{ij}}{\sum_{p=1}^{n} |z_{ip}|}$ is a linear normalization function which can have negative attention outputs that are essential to encourage the network to learn adversarial attention relationships among all linear mixtures sharing the same warehouse.

For a convolutional layer, let x $\in \mathbb{R}^{h\times w\times c}$ and $y \in \mathbb{R}^{h\times w\times f}$ represent the input with c feature channels of resolution $h \times w$ and the output with f feature channels of the same resolution respectively. The output of normal convolutional layers is computed as follows:

$$\mathrm{y} = \mathbf{W} \cdot \mathrm{x} \tag{2}$$

The normal convolution uses a single static kernel composed of f convolutional filters of size $k \times k$. In contrast, dynamic convolution $\mathbf{W}$ replaces this static kernel $\alpha_{\mathrm{n}}\mathbf{W}_{\mathrm{n}}$, a linear mix of n static kernels of the same dimensions, weighted by their input-related scalar attentions α_1 ... α_m. MKW differs from the existing dynamic convolution method in that it applies this attention mixing to densely local-scaled static kernels through kernel partitioning and warehouse sharing, rather than to a single, large-scale kernel.

The basic idea of kernel partition is to explicitly enhance parameter dependencies within the same convolutional layer, reducing kernel dimensions and increasing the number of kernels. First, the static convolutional kernel $\mathbf{W}$ in a regular convolutional layer is divided into m

disjoint parts $\mathrm{w}_1 \dots \mathrm{w}_m$, referred to as 'kernel cells', each having the same dimensions. Kernel partition can be defined as follows:

$$\mathbf{W} = \mathbf{w}_1 \cup \dots \cup \mathbf{w}_m \tag{3}$$

After kernel partition, the kernel cells $\mathrm{w}_1 \dots \mathrm{w}_m$ are treated as 'local kernels' $E = \{e_1 \dots e_m\}$, and a 'warehouse' containing n kernel units is defined, where each unit $e_1 \dots e_m$ has the same dimensions as $\mathbf{w}_1 \dots \mathbf{w}_m$.

Then, the 'warehouse' $E = \{e_1 \dots e_m\}$ is shared within the same convolutional layer. The authors represent each kernel cell $\mathrm{w}_1 \dots \mathrm{w}_m$ as follows:

$$\mathrm{w}_i = \alpha_{i1} e_1 + \cdots + \alpha_{in} e_\mathrm{n} \tag{4}$$

where $i \in \{1, \dots, m\}$, the scalar attention α_in is generated by the attention module $\phi(x)$ based on the input x. Finally, in a regular convolutional layer, the static convolutional kernel **W** is replaced by sequentially assembling its corresponding m linear mixtures.

Building on the basic objective of kernel partition, the main goal of warehouse sharing is to explicitly enhance parameter dependencies between consecutive convolutional layers. This approach aims to further improve MKW's parameter efficiency and representational capability.

Specifically, in a convolution neural network, a single warehouse $E = \{e_1 \dots e_m\}$ is shared among l adjacent convolutional layers in the same stage building block to represent each kernel unit. This allows for the use of a larger n setting in kernel partitioning. This is easy to implement, as modern convolution neural networks often adopt a modular design scheme, where the static convolutional kernels in the same stage layers usually have the same dimensions.

Let n be the number of kernel units in a ‘warehouse’ shared by l convolutional layers of a convolutional network and let the total number of kernel units across these convolutional layers be (when $l = 1, m_t = m$). Then, it can serve as a scaling factor, indicating the convolutional parameter budget of MKW relative to regular convolution. In this case, the authors do not consider the number of parameters in the attention module ϕ, which generates n scalar attention, because it is much smaller than the total number of parameters in regular convolutions across the convolutional layers.

3.2 Dataset Distillation by Matching Training Trajectories

The DD method, which matches the training trajectory, is based on the teacher-student network structure. The architecture of the teacher-student network is similar to the principles of KD. In this setup, the teacher network undergoes initial training with the original dataset and then retains its parameters, which serve to direct the process of distilling the dataset. The teacher network is trained on the original dataset and captures the training trajectory in its parameters. The parameters guide the student network in effectively minimizing gradient loss to achieve the best distilled dataset. The purpose of guiding the DD by the teacher trajectory is to match the parameters of the trained student network on the distillation dataset $\mathcal{D}_{\text{distill}}$ with the parameters of the teacher network on the original dataset $\mathcal{D}_{\text{original}}$ to achieve better training results. The method is mainly divided into training of the teacher-student network structure, DD using parameter pruning, and generation of the optimized distillation dataset.

For the parameter training of the teacher-student network structure, there are N teacher networks are first pre-trained on $\mathcal{D}_{\text{original}}$ and their snapshot parameters are saved in each epoch. The teacher parameters are defined as the time series of parameters $\{\theta_i\}_0^I$. Meanwhile, the student parameters are defined as $\tilde{\theta}_i$, and they are trained on the distilled dataset at each training step i. At each distillation step, we first draw parameters from one of the teacher parameters of random step i and use it to initialize the student parameters as $\tilde{\theta}_i = \theta_i$. The number of updates to the student parameters and teacher parameters are set to J and K, where $J \ll K$. For each student's update parameter j, we extract a minibatch $b_{i,j}$ from the distilled dataset as follows.

$$b_{i,j} \sim \mathcal{D}_{\text{distill}} \tag{5}$$

Once the student network is set up, the network updates its parameters through N rounds of gradient descent, focusing on minimizing the classification loss from synthetic data. The update process for the student parameters is described by the following equation:

$$\hat{\theta}_{t+n+1} = \hat{\theta}_{t+n} - \alpha \nabla \ell(A(\mathcal{D}_{\text{distill}}); \hat{\theta}_{t+n}) \tag{6}$$

The classification loss is calculated by the cross-entropy. Cross-entropy is a measure used in machine learning to quantify the difference between two probability distributions, typically the true distribution of labels in a dataset and the distribution predicted by a model. It is commonly used as a loss function for classification problems, where a lower cross-entropy value indicates that the model's predictions are closer to the true labels.

$$H(y, \hat{y}) = -\sum_i (y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)) \tag{7}$$

Subsequently, after updating the parameters of the student network, the parameters that are difficult to match are constructed, and the final loss L between the ending student and teacher parameters is calculated as follows.

$$L = \frac{\left\|\hat{\theta}_{t+N} - \theta^*_{t+M}\right\|_2^2}{\left\|\hat{\theta}_t - \theta^*_{t+M}\right\|_2^2} \tag{8}$$

4 Experimental Study

4.1 Distilled Dataset Generation

In this paper, a brand of time-frequency data for a six-axis industrial robot was verified, which includes multiple normal/abnormal robot drive feedback current data. The data samples were collected every second from reducers and motors on different robot axes, covering seven operating states, with specific faults detailed in Table 1.

Table 1. The specific fault data.

Label	axis	Data volume	Status	Label
0	1 ~ 6 axis	180,000	1 axis reducer, 2 axis motor fault	Class 0
1	1 ~ 6 axis	180,000	1 axis reducer, 3 axis motor fault	Class 1
2	1 ~ 6 axis	180,000	3 axis reducer, 4 axis motor fault	Class 2
3	1 ~ 6 axis	180,000	3 axis reducer fault	Class 3
4	1 ~ 6 axis	180,000	2 axis motor fault	Class 4
5	1 ~ 6 axis	180,000	4 axis reducer fault	Class 5
6	1 ~ 6 axis	300,000	Normal	Class 6

In the experiment, 180,000 data points from seven different operating states were used, and the image dataset generated through CWT was shuffled and reorganized. To be specific, label 0 to 2 represent various types of compound faults, indicating scenarios where multiple issues occur simultaneously. Conversely, label 3 to 5 are assigned to single fault conditions, each depicting a single type of fault in motor. Importantly, the label 6 is designated for normal data. The dataset was used along with a three-layer CNN to generate the distilled training and testing dataset. The neural network architecture used for generating a distilled dataset primarily followed the simple CNN architecture designed by Gidaris and Komodakis [39]. This architecture consists of multiple convolutional blocks, each containing a 3 × 3 convolutional layer and three 2 × 2 average pooling layers, with functions like filter, instance normalization, and ReLU, with a stride of 2. After the convolutional blocks, a single linear layer generates logs. The size of the images is 32 × 32, and ZCA whitening was applied to the dataset. ZCA whitening is a linear transformation used to decorrelate source signals and an important preprocessing step for reducing the redundancy of the input data.

In the experiment, distillation training was conducted for 10,000 iterations, aiming to increase the quantity of the distilled dataset training for more stable network training. The training subset of the distilled dataset comprised data saved from iterations 9,000 to 10,000. For testing purposes, the distilled dataset included data from the later iterations, specifically 9,700 to 10,000. The final distilled dataset used for verification maintained a ratio of 2:8 between the testing and training dataset. It utilized a batch of 50 images per class, resulting in the distillation of 350 images in each iteration. This process was repeated across all seven categories present within the dataset.

4.2 Experimental setup

To verify the effectiveness of DD on the industrial dataset, this experiment utilized Accuracy to express the effect of lightweight network classification with the following equation.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{9}$$

where TP represents the number of correctly identified categories, FP represents the number of other categories incorrectly identified as such, and FN represents the number of categories misidentified as other categories.

The model employs a three-layer MKW architecture for the convolutional neural network, which adheres to the concept of MKW’s shared convolutional layers for enhanced feature extraction. A three-layer MKW architecture for a convolutional neural network, adhering to the concept of MKW’s shared convolutional layers. This design substantially reduces parameter count compared to traditional networks, boosting efficiency. The convolutional layers of the MKW-NET share convolutional parameters following the idea of MKW, with each block containing a 3 × 3 convolutional layer, batch normalization layer, ReLU activation function, and three 2 × 2 average pooling layers. In this experiment, the effectiveness of MKW-NET was validated via two sets of experiments. The first experiment verifiable the distilled industrial time-frequency dataset by MKW-NET. The learning rate, batch size and layer number are adjusted in each trial. Specifically, the mean and standard deviation of the outcomes from the 5-fold experiments were recorded as the result. The hyperparameters of MKW-NET are detailed in Table 2.

Table 2. The details of hyperparameters of MKW-NET.

Hyperparameter	Value
Learning rate	0.005
Batch size	128
Number of epochs	100
Convolutional kernel size for patch embedding	7*7
Average pooling size	2

In the second experiment, a benchmarking experiment was set up to reveal the effectiveness of MKW-NET. Five current mainstream algorithms were used as benchmarking algorithms, where the parameters of the networks, such as learning rate, training epoch, and batch size, were determined after multiple trials. The network structures used in the experiment are detailed below:

1. **Compact Convolutional Transformer (CCT)** [40]: A lightweight Transformer for image classification. In this experiment, the number of convolutional layers and the number of encoders were both set to 2.
2. **ViT-Tiny** [41]: ViT-Tiny has far fewer parameters than the standard ViT. In this experiment, ViT's patch size was set to 4, and the network depth to 2.
3. **ConvNext** [42]: An improved network based on the foundational architecture of convolutional neural networks and ViT, with the network depth set to 4 layers in this experiment.
4. **MobilenetV2** [43]: Designed for computer vision applications on mobile and edge devices, MobilenetV2 uses the original 6 residual blocks.
5. **Resnet** [44]: ResNet addresses deep neural network training difficulties, gradient vanishing, and explosion by introducing residual blocks.

In each trial, the experiment was conducted five times, and the mean of the results was marked. These tests were carried out on a server running Ubuntu 16.0, equipped with an Intel i9-10920X 3.50Ghz CPU and an Nvidia GeForce RTX 3090 graphics card. The setup included Python 3.8.12 and the Pytorch 1.10.1 package for algorithm development. The evaluation focused on overall accuracy, encompassing both individual and compound fault categories, to assess the algorithm's effectiveness. Additionally, the size of the model and its FLOPs (floating-point operations per second) were used as benchmarks to measure the computational efficiency of the experiment.

5 Experimental Result

Figure 3 illustrates the performance of MKW-NET in terms of accuracy over 100 training epochs with different learning rates ranging from 0.001 to 0.1. Observations reveal that a learning rate of 0.005 achieves the most steady and robust improvement in accuracy, stabilizing around 85% after about 30 epochs. In comparison, a learning rate of 0.01 also trends towards a similar accuracy level but with greater fluctuations, suggesting less stability. Notably, the network experiences increased volatility with a learning rate of 0.05, leading to significant variations in accuracy across epochs. The most pronounced instability is seen with a learning rate of 0.1, where the accuracy shows sharp declines at several points, underscoring the detrimental impact of higher learning rates on model performance. Table 3 delineates the relationship between the convolutional network depth and its performance metrics. As the number of layers increases from two to six, there is a progressive improvement in test accuracy, indicating the network's enhanced learning capacity with additional layers. Especially, the test accuracy peaks at five layers with a value of 90.97%. However, the training time also escalates with each additional layer, reaching a substantial duration of 1328.71 minutes for the six-layer configuration. Despite the longer training times, the six-layer network does not perform better in test accuracy than the five-layer one, suggesting diminishing returns with further complexity.

Table 3. The mean accuracy of different convolutional layers.

Numbers of layer	#Params (M)	FLOPs(M)	Training time (min)	Test accuracy
2	0.013	0.45	113.53	80.12%
3	0.043	0.56	136.03	86.46%
4	0.13	0.67	177.57	89.43%
5	0.44	1.01	412.07	90.97%
6	0.79	8.70	1328.71	86.67%

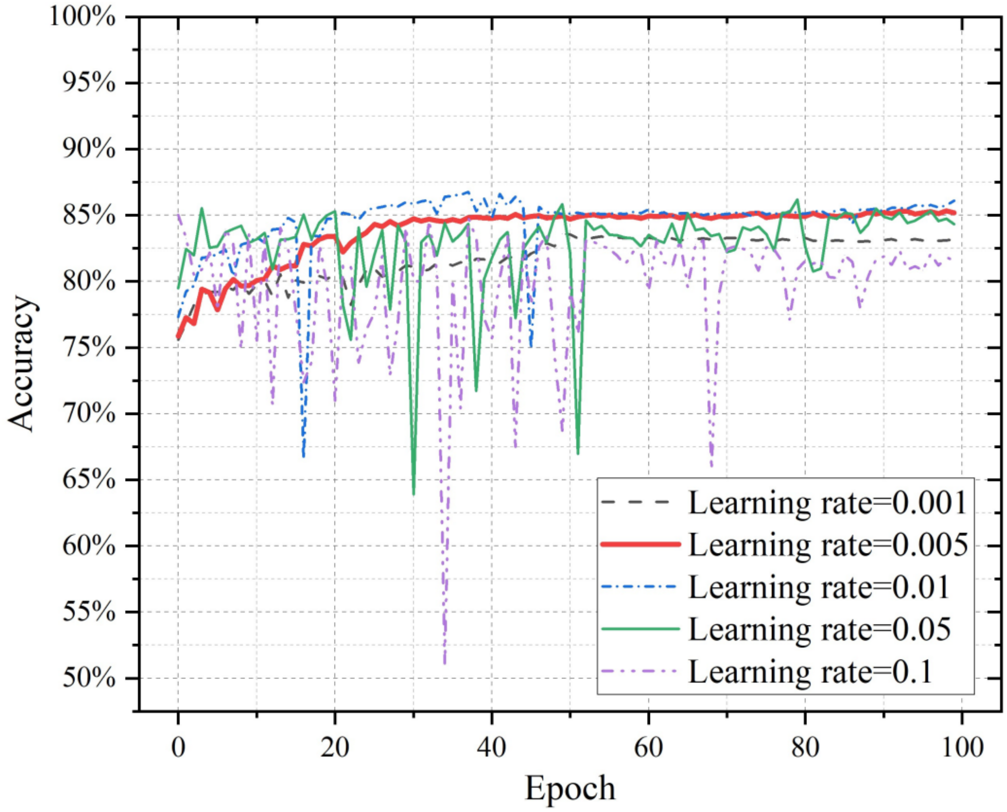


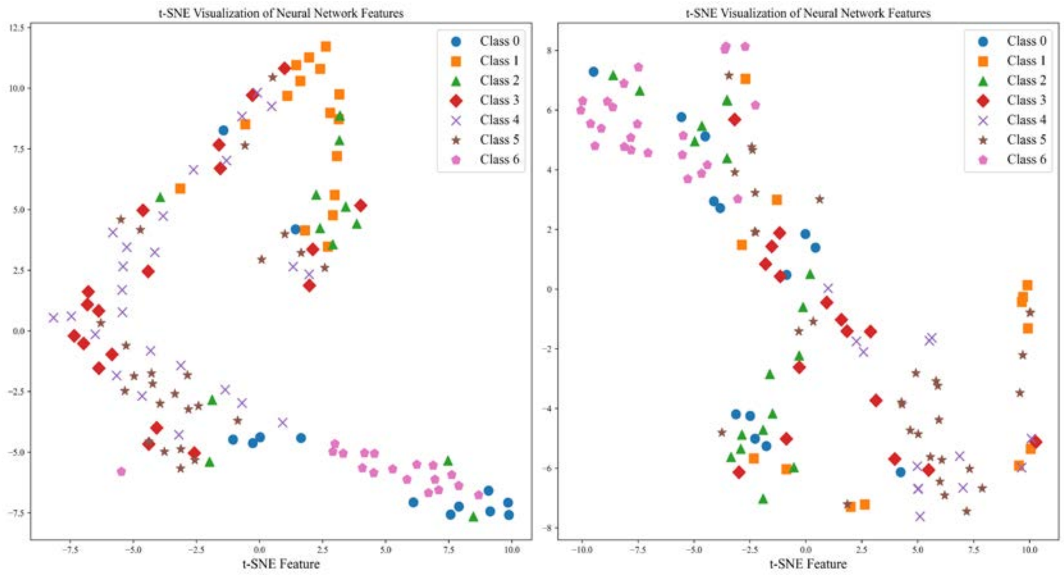
Figure 3. The accuracy of fault diagnosis with different learning rates.

Table 4 displays the model size, FLOPs, and classification results for MKW-NET in this experiment. Overall, due to MKW's kernel sharing mechanism and the proposed optimized attention function, MKW-NET has the least network parameters, FLOPs, and the highest classification accuracy. MKW-NET's model size is significantly smaller, with only 0.043M parameters, which is substantially less than most of the other algorithms listed. This indicates a highly efficient model that requires fewer resources for storage and quicker processing during inference. The FLOPs of MKW-NET are impressively low at just 0.56M, which suggests that it is computationally less expensive to run, making it suitable for environments with limited computational capacity. Also, the net also achieved an optimal balance between computational cost and algorithmic performance. This balance is crucial for practical applications where both efficiency and accuracy are valued. It is clear that MKW-NET achieves the highest overall accuracy on the distilled dataset, outperforming the second-best algorithm by 1.86%. Its model size is just 0.043 million parameters, considerably smaller than others. In contrast, MobilenetV2's tiny version requires 0.13 million parameters, and CCT is approximately 0.28 million. While similar in size to tiny-VIT, MKW-NET is significantly more efficient with 4.2% fewer FLOPs, and tiny-VIT's classification accuracy is 63.09%. On the distilled dataset, MobilenetV2 and CCT, with relatively smaller parameter counts, have overall accuracies of 78.64% and 61.28%, respectively. Among these algorithms, CCT performs the worst. In comparison to MKW-NET, Convnext has a much larger model size, with FLOPs reaching 1722.58M, which far exceeds other algorithms.

Table 4. The comparison of computational cost and algorithm performance.

Algorithms	#Params (M)	FLOPs(M)	Test accuracy
MKW-NET(Proposed)	**0.043**	**0.56**	**87.40%**
CCT [40]	0.28	50.70	61.28%
ViT-tiny [41]	0.13	13.46	63.09%
Convnext [42]	36.43	1722.58	75.02%
MobilenetV2 [43]	2.23	6.67	78.64%
ResNet [44]	11.18	37.22	85.54%

Figure 4 provides a comparative t-SNE visualization between the original and the distilled dataset. Within the original dataset, the visual patterns of classes 1, 2, and 3 are characterized by a dispersed arrangement, which implies that the boundaries between these classes are not clearly defined. This scattered distribution could make it challenging for classification algorithms to distinguish between the classes accurately. On the other hand, the distilled dataset presents a stark contrast, with each class demonstrating a more cohesive and distinct clustering indicative of well-defined separations. The reduction in overlap between different classes in the distilled dataset is significant, suggesting that the feature space has been effectively condensed and refined. Such clarity in the feature representation is likely to result in a more precise classification model for the lightweight networks, as the clearer demarcation of class boundaries aids in reducing misclassification errors and improves the model's ability to generalize from the distilled data to new, unseen instances. This enhanced separation and definition of classes could prove to be beneficial for applications where precise and reliable classification is critical.



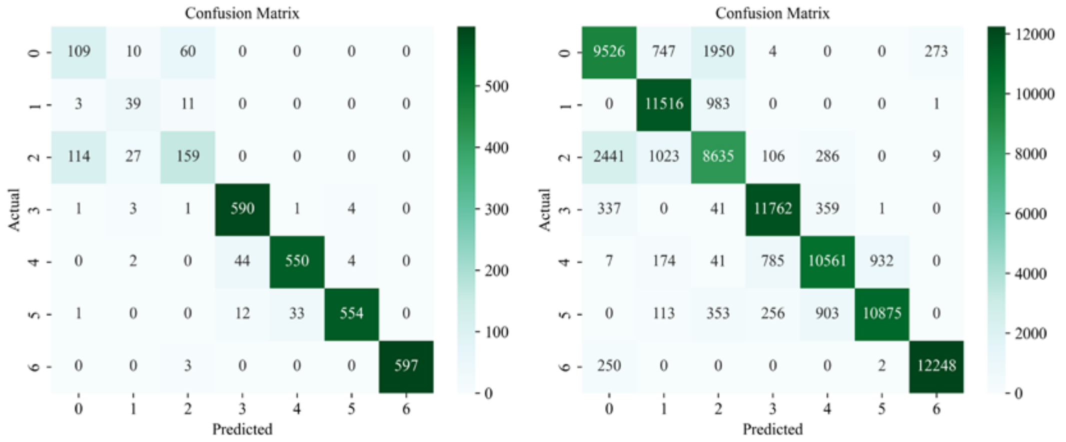
(a)Original dataset

(b)Distilled dataset

Figure 4. The t-SNE between the original dataset and the distilled dataset.

Figure 5 presents the confusion matrices for each dataset. In the confusion matrix of the original dataset, category 1 has 43 samples correctly classified but also has 8 samples misclassified into other categories, mainly category 2. For category 2, there are 124 samples correctly classified, but 140 samples are misclassified as category 0, indicating issues with recognizing category 2. The performance for category 3 to 6 is relatively good, which has a high number of correct classifications. The confusion between categories is relatively low, particularly from category 3 to 6, suggesting that the model distinguishes these categories with relative accuracy. Overall, the model performs reasonably well on the original dataset, particularly in certain categories.

However, there is lower accuracy in recognizing category 2 and some confusion between categories 1 and 2. The confusion matrix for the distilled dataset indicates an increase in the number of correctly classified samples and an improvement in the efficacy of sample classification after distillation. The model correctly classifies 9652 samples for category 0, with 704 samples misclassified as category 1. For category 2, there are 9164 correct classifications, but 1907 samples are incorrectly classified as category 1, suggesting that while the recognition rate for category 2 is high, there is a relative abundance of confusion with category 1. Categories 3 and 6 show a high number of correct classifications, with 11955 and 12250 samples respectively. However, confusion between categories in certain areas is quite pronounced, especially between categories 1 and 2, as well as between categories 4 and 5. Figure 6 provides a comparison of data before and after distillation. It is clear that the structure of the original data has been greatly changed, including the privacy information.



(a)Original dataset

(b)Distilled dataset

Figure 5. The confusion matrices between the original dataset and the distilled dataset.

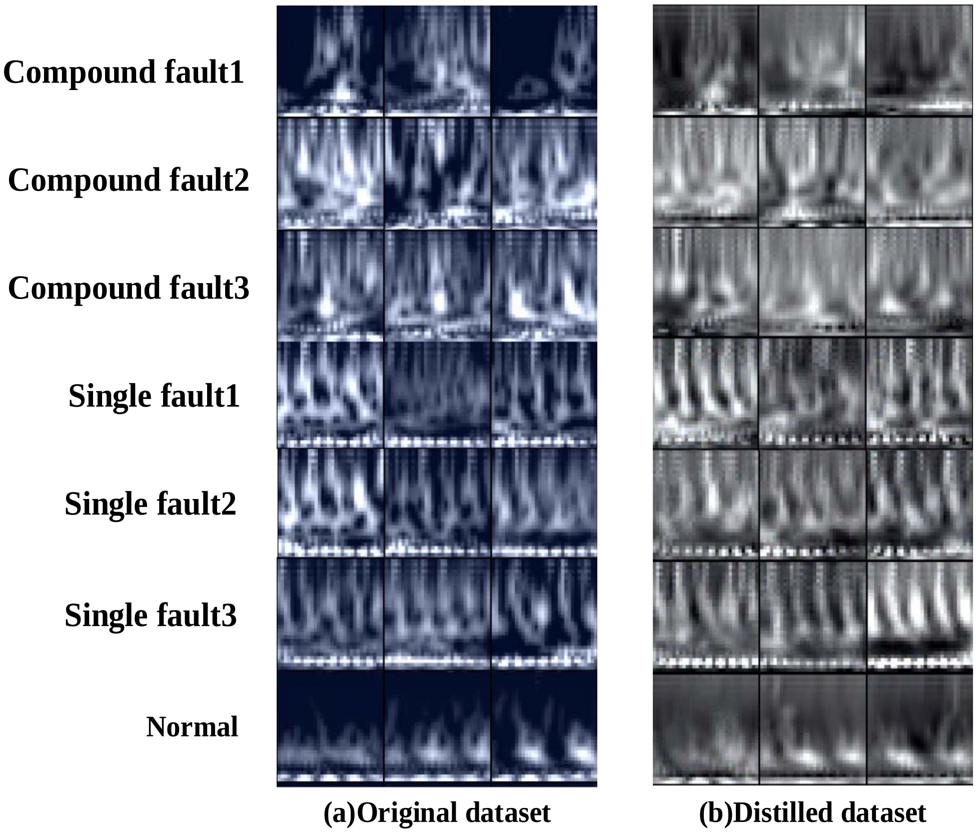


Figure 6. (a) original dataset (0 iteration); (b) distilled dataset (10000 iteration)

6 Discussion

The MKW-Net offers a way to decrease the number of network parameters, making it useful in dataset distillation. This network excels in fault diagnosis within distilled dataset, offering precise detection and analysis of anomalies. The comparison of the original and distilled time-frequency dataset reveals that the distillation process effectively encrypts most time-frequency data containing sensitive private information. At the same time, it retains crucial information for classification, striking a balance between preserving data utility and addressing privacy concerns by eliminating unnecessary details while maintaining key features needed for fault diagnosis. The t-SNE visualization reveals that the distilled dataset displays a more defined clustering by category, indicating effective feature extraction. This suggests that the distillation process not only refines the dataset by removing outliers but also retains essential information, thus optimizing it for machine learning applications. Figure 5 compares confusion matrices from the original and distilled dataset. The original dataset shows decent classification performance, with categories demanding the single faults or normal demonstrating particularly high accuracy. However, category 2 frequently gets incorrectly labeled as category 0, and there is a noticeable mix-up between categories 1 and 2. CWT provides variable time-frequency resolution for the original dataset, enhancing time resolution at high frequencies and frequency resolution at low frequencies. Yet, this trade-off might make features in some frequency bands less clear, affecting classification. Additionally, boundary effects may lead to inaccurate representations at signal edges, adversely impacting accuracy, particularly when signal lengths differ or edge information is crucial. Simpler images, characterized by less noise and more distinct features, decrease the likelihood of misclassification and facilitate the extraction of distinctive features from time-frequency data, thereby improving differentiation between categories. From the confusion matrix it could easily depicted that the compound faults are easy to trigger the confusion, as the rest of original dataset trigger less confusion while classifying. In contrast, the distillation process has modified the original classification pattern, making it much easier to classify categories 0,1 and 2, which are now more distinct and separable, and to a lesser extent between categories 4 and 5. Figure 6 clearly shows the comparison between the original and distilled dataset. Since the challenges in reversing the CWT transformed time-frequency, the distilled dataset could hardly perform a restore for its condensed and differentiated information. Moreover, the distilled dataset is not the traditional time-frequency dataset. Based on the above discussion, the privacy information has been successfully protected. These experimental results indicate that the distilled dataset significantly alters the label distribution of the original dataset, demonstrating that the privacy features in the original dataset have also been correspondingly altered. However, there are still several issues to be resolved with the dataset distillation method and our proposed MKW network, which we have identified and verified through our experiments. In this paper, we adjust variables such as learning rate, the number of network convolutional layers, and the quantity of convolutional kernels. Also, to comprehensively assess our method, we compare it with current popular lightweight architectures as well as transformer-based architectures. Figure 3 shows that a moderate learning rate, especially 0.005, results in stable and reliable accuracy. In contrast, learning rates that are too high or too low could result in unstable training results, impacting the network's ability to classify distilled data. For example, while a higher learning rate such as 0.01 can achieve accuracy comparable to 0.005, it tends to result in less stable training. From the results of Table 3, as the number of network layers increases, the corresponding training time for the network also rises. However, as the network expands to six layers, the training duration significantly increases. This suggests that although MKW has enhanced the network's stability, the training efficiency for configurations with multiple layers and numerous kernel cell still requires improvement. Moreover, Table 4 shows that complex networks or those needing a lot of computing power usually perform worse on the distilled datasets, as for the similar performance, the MKW-Net has a lower compute complexity and fewer network parameters. It can be seen that the performance of MKW-Net shows merits in comparison with other algorithms for its less complexity and network parameters. This implies that the synthesized features of the distilled dataset might not fit well with these complex models. It highlights ongoing issues with the quality and stability of distilled data, as well as the poor training performance of large networks on the dataset.

Future research in DD should focus on improving methods to work better with complex neural networks like Transformers. The main challenge of DD is to find a balance between the network complexity and the performance of the distilled data. For DD, it's important to enhance how well it works with different types of networks by improving gradient matching during the distillation process. Moreover, the latest neural network should not only focus on the performance of dataset classification but also pay more attention to computational requirements, as complex networks are not particularly widely used in industrial fault diagnosis. The final goal is to develop a distillation method that fits well with the wide range of time-frequency data and can be used in different fault diagnosis scenarios. This approach could be highly beneficial for companies seeking to safeguard sensitive information such as production processes, machinery efficiency, and output statistics, ensuring both confidentiality and competitive advantage in their industry.

7 Conclusions

As the era of Industry 4.0 accelerates, protecting the process privacy information contained within industrial datasets has become a critical priority, essential for preserving a company's competitive edge and ensuring operational security. Addressing the challenge of encrypting an industrial robot dataset, this paper introduces a novel, lightweight CNN-based DD approach that leverages the MKW method. This approach adeptly condenses a voluminous dataset into a manageable number of images per class, all while capturing the essential features necessary for accurate classification. Our experimental findings affirm the superiority of our approach, which outperforms larger neural networks in classifying a distilled dataset. Future work should focus on developing refined DD methods suitable for complex neural network structures, especially Transformers. The focus will be on developing networks that are lighter and use less computing power while still maintaining good performance.

Acknowledgement

Our work is supported by multiple funds in China, including the Key Program of NSFC-Guangdong Joint Funds (U2001201), the National key R & D project (2021YFB3301802), the National Natural Science Foundation of China (62302103). Our work is also supported by Guangdong Provincial Key Laboratory of Cyber-Physical System (2020B1212060069).

References

[1] Parto, M., Urbina Coronado, P. D., Saldana, C., and Kurfess, T., 2021, "Cyber-Physical System Implementation for Manufacturing With Analytics in the Cloud Layer," Journal of Computing and Information Science in Engineering, 22(1).

[2] Li, W., Zhong, X., Shao, H., Cai, B., and Yang, X., 2022, "Multi-mode data augmentation and fault diagnosis of rotating machinery using modified ACGAN designed with new framework," Advanced Engineering Informatics, 52, p. 101552.

[3] Xiao, Y., Shao, H., Wang, J., Yan, S., Liu, B. J. M. S., and Processing, S., 2024, "Bayesian Variational Transformer: A generalizable model for rotating machinery fault diagnosis," 207, p. 110936.

[4] Luo, J., Shao, H., Lin, J., Liu, B. J. R. E., and Safety, S., 2024, "Meta-learning with elastic prototypical network for fault transfer diagnosis of bearings under unstable speeds," p. 110001.

[5] Chen, C., Wang, T., Liu, C., Liu, Y., Cheng, L. J. I. T. o. I., and Measurement, 2023, "Lightweight Convolutional Transformers Enhanced Meta Learning for Compound Fault Diagnosis of Industrial Robot."

[6] Shao, H., Zhou, X., Lin, J., and Liu, B. J. I. I. o. T. J., 2024, "Few-Shot Cross-Domain Fault Diagnosis of Bearing

Driven By Task-Supervised ANIL."

[7] Chen, C., Liu, C., Wang, T., Zhang, A., Wu, W., and Cheng, L., 2023, "Compound fault diagnosis for industrial robots based on dual-transformer networks," Journal of Manufacturing Systems, 66, pp. 163-178.

[8] Lee, H., Finke, D., and Yang, H., 2023, "Privacy-preserving Neural Networks for Smart Manufacturing," Journal of Computing and Information Science in Engineering, pp. 1-20.

[9] Ranathunga, T., McGibney, A., Rea, S., and Bharti, S., 2022, "Blockchain-Based Decentralized Model Aggregation for Cross-Silo Federated Learning in Industry 4.0," IEEE Internet of Things Journal, 10(5), pp. 4449-4461.

[10] Shi, M., Ding, C., Chang, S., Wang, R., Huang, W., and Zhu, Z., 2023, "Cross-domain privacy-preserving broad network for fault diagnosis of rotating machinery," Advanced Engineering Informatics, 58, p. 102157.

[11] Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A., 2018, "Dataset distillation," arXiv preprint arXiv:1811.10959.

[12] Dong, T., Zhao, B., and Lyu, L., "Privacy for free: How does dataset condensation help privacy?," Proc. International Conference on Machine Learning, PMLR, pp. 5378-5396.

[13] Yun, J., Jiang, D., Liu, Y., Sun, Y., Tao, B., Kong, J., Tian, J., Tong, X., Xu, M., and Fang, Z., 2022, "Real-time target detection method based on lightweight convolutional neural network," Frontiers in Bioengineering and Biotechnology, 10, p. 861286.

[14] Li, C., and Yao, A., 2023, "KernelWarehouse: Towards Parameter-Efficient Dynamic Convolution," arXiv preprint arXiv:2308.08361.

[15] Dwork, C., "Differential privacy," Proc. International colloquium on automata, languages, and programming, Springer, pp. 1-12.

[16] Dwork, C., McSherry, F., Nissim, K., and Smith, A., "Calibrating noise to sensitivity in private data analysis," Proc. Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3, Springer, pp. 265-284.

[17] Cheu, A., Smith, A., Ullman, J., Zeber, D., and Zhilyaev, M., "Distributed differential privacy via shuffling," Proc. Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38, Springer, pp. 375-403.

[18] Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R., and Zhou, Y., "A hybrid approach to privacy-preserving federated learning," Proc. Proceedings of the 12th ACM workshop on artificial intelligence and security, pp. 1-11.

[19] Liu, X., Li, H., Xu, G., Chen, Z., Huang, X., and Lu, R., 2021, "Privacy-enhanced federated learning against poisoning adversaries," IEEE Transactions on Information Forensics and Security, 16, pp. 4574-4588.

[20] Wei, K., Li, J., Ding, M., Ma, C., Su, H., Zhang, B., and Poor, H. V., 2021, "User-level privacy-preserving federated learning: Analysis and performance optimization," IEEE Transactions on Mobile Computing, 21(9), pp. 3388-3401.

[21] Wu, X., Zhang, Y., Shi, M., Li, P., Li, R., and Xiong, N. N., 2022, "An adaptive federated learning scheme with differential privacy preserving," Future Generation Computer Systems, 127, pp. 362-372.

[22] Zhao, Y., Zhao, J., Jiang, L., Tan, R., Niyato, D., Li, Z., Lyu, L., and Liu, Y., 2020, "Privacy-preserving blockchain-based federated learning for IoT devices," IEEE Internet of Things Journal, 8(3), pp. 1817-1829.

[23] Mehta, M., and Shao, C., 2022, "Federated learning-based semantic segmentation for pixel-wise defect detection in additive manufacturing," Journal of Manufacturing Systems, 64, pp. 197-210.

[24] Chen, S., Yu, D., Zou, Y., Yu, J., and Cheng, X., 2022, "Decentralized wireless federated learning with differential privacy," IEEE Transactions on Industrial Informatics, 18(9), pp. 6273-6282.

[25] Wu, X., Qi, L., Gao, J., Ji, G., and Xu, X., 2022, "An ensemble of random decision trees with local differential privacy in edge computing," Neurocomputing, 485, pp. 181-195.

[26] Chen, D., Kerkouche, R., and Fritz, M., 2022, "Private set generation with discriminative information," Advances in Neural Information Processing Systems, 35, pp. 14678-14690.

[27] Zhou, Y., Nezhadarya, E., and Ba, J., 2022, "Dataset distillation using neural feature regression," Advances in Neural Information Processing Systems, 35, pp. 9813-9827.

[28] Nguyen, T., Chen, Z., and Lee, J., 2020, "Dataset meta-learning from kernel ridge-regression," arXiv preprint arXiv:2011.00050.

[29] Zhao, B., Mopuri, K. R., and Bilen, H., 2020, "Dataset condensation with gradient matching," arXiv preprint arXiv:2006.05929.

[30] Zhao, G., Li, G., Qin, Y., and Yu, Y., "Improved distribution matching for dataset condensation," Proc. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7856-7865.

[31] Cazenavette, G., Wang, T., Torralba, A., Efros, A. A., and Zhu, J.-Y., "Dataset distillation by matching training trajectories," Proc. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4750-4759.

[32] Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J., 2021, "A survey of convolutional neural networks: analysis, applications, and prospects," IEEE transactions on neural networks and learning systems.

[33] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2012, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, 25.

[34] Zhu, F., Liu, C., Yang, J., and Wang, S., 2022, "An Improved MobileNet Network with Wavelet Energy and Global Average Pooling for Rotating Machinery Fault Diagnosis," Sensors, 22(12), p. 4427.

[35] Luo, Z., Tan, H., Dong, X., Zhu, G., and Li, J., 2022, "A fault diagnosis method for rotating machinery with variable speed based on multi-feature fusion and improved ShuffleNet V2," Measurement Science and Technology, 34(3), p. 035110.

[36] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K., 2016, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size," arXiv preprint arXiv:1602.07360.

[37] Wang, Y., Yan, J., Sun, Q., Zhao, Y., and Liu, T., 2021, "ShuffleNet□based comprehensive diagnosis for insulation and mechanical faults of power equipment," High Voltage, 6(5), pp. 861-872.

[38] Zhong, H., Lv, Y., Yuan, R., and Yang, D., 2022, "Bearing fault diagnosis using transfer learning and self-attention ensemble lightweight convolutional neural network," Neurocomputing, 501, pp. 765-777.

[39] Gidaris, S., and Komodakis, N., "Dynamic few-shot visual learning without forgetting," Proc. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4367-4375.

[40] Hassani, A., Walton, S., Shah, N., Abuduweili, A., Li, J., and Shi, H., 2021, "Escaping the big data paradigm with compact transformers," arXiv preprint arXiv:2104.05704.

[41] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., and Gelly, S., 2020, "An image is worth 16x16 words: Transformers for image recognition at scale," arXiv preprint arXiv:2010.11929.

[42] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S., "A convnet for the 2020s," Proc. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 11976-11986.

[43] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C., "Mobilenetv2: Inverted residuals and linear bottlenecks," Proc. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520.

[44] He, K., Zhang, X., Ren, S., and Sun, J., "Deep residual learning for image recognition," Proc. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778.