This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

Mitigating Catastrophic Forgetting in Cross-Domain Fault Diagnosis: An Unsupervised Class Incremental Learning Network Approach

Yifan Zhan, Rui Yang, *Senior Member, IEEE, Yong Zhang, Zidong Wang, Fellow, IEEE*

***Abstract*—While deep learning has found widespread application in fault diagnosis, it continues to face three primary challenges. Firstly, it assumes that training and test datasets adhere to the same distribution, which is often not the case in industries with varying conditions. Secondly, it relies heavily on the availability of abundant labeled data for training, overlooking the reality that newly collected data is frequently unlabeled. Thirdly, neural networks frequently encounter catastrophic forgetting, a critical concern in dynamic industrial settings with emerging faults. Therefore, this paper proposes an unsupervised class incremental learning network (UCILN), to mitigate catastrophic forgetting in cross-domain fault diagnosis, particularly in situations where the target domain lacks labeled data. A memory module and a semi-frozen and semi-updated incremental strategy are designed to balance the retention of old knowledge with the acquisition of new information. Test results obtained from the CWRU and PU datasets demonstrate the exceptional performance of UCILN.**

***Index Terms*—Catastrophic forgetting, class incremental learning, fault diagnosis, unsupervised domain adaptation, unsupervised transfer learning**

I. Introduction

ROTATING machinery in industrial environments is exposed to challenges such as high speeds, heavy loads, and elevated temperatures, making it susceptible to failure. The inaccurate or untimely detection and diagnosis of faults may lead to severe equipment breakdowns or financial losses [8], [18], [19], [24], [31]. Therefore, the accurate diagnosis of faults in real industrial scenarios is of significant importance [3], [7], [20], [28]. In recent years, various deep learning methods have been introduced for fault diagnosis, facilitating an automated end-to-end process from raw signals to diagnosis [9], [33], [35], [37]. Despite the widespread application of deep learning in fault diagnosis, these methods encounter three primary challenges.

Firstly, traditional deep learning-based methods assume that the training and test datasets follow an identical distribution. However, in industrial settings, working conditions such as temperature and rotation speed may vary greatly between tasks, which requires models trained on a dataset from one working condition to successfully transfer to the test dataset from another working condition. However, the variability in working conditions makes it challenging for models trained in the source domain to directly generalize or transfer to the target domain, leading to domain shift [3], [5], [30]. To mitigate this challenge, various transfer learning methods have been applied in fault diagnosis. These techniques facilitate the transfer of knowledge from one working condition (source domain) to another working condition (target domain) [6], [32], [36].

Secondly, traditional deep learning methods also presuppose the existence of ample labeled data for training [3], [23]. Nonetheless, newly acquired data from diverse working conditions often lacks labels, posing challenges in the labeling process that may be unfeasible in certain scenarios. Hence, the application of unsupervised deep transfer learning (UDTL), or unsupervised domain adaptation, becomes pertinent. This approach makes predictions for unlabeled data within the target domain given labeled data from the source domain by leveraging similar features across different application scenarios or working conditions [36].

Thirdly, gradient-based neural networks often face the challenge of catastrophic forgetting [11]. Catastrophic forgetting refers to the phenomenon where, as the model learns new tasks, it adjusts its weights and parameters to accommodate the new data, which may result in a decline in performance on previously learned tasks or even complete forgetting of previously acquired knowledge [13]. In dynamic industrial settings, the emergence of new fault types is unpredictable, necessitating model continuous updates to adapt to emerging faults. However, the updating process may lead to a significant decline in the model's diagnostic capability for old fault types, resulting in catastrophic forgetting. Consequently, this could cause production interruptions, equipment damage, and human safety issues. Thus, mitigating the impact of catastrophic forgetting is imperative to ensure the robustness and reliability of deep learning in fault diagnosis.

The emerging research direction of class incremental learning aims to equip models with the capability to continuously

This work is partially supported by the Jiangsu Provincial Qinglan Project (2021), the Research Development Fund of XJTLU (RDF-20-01-18), XJTLU Research Enhancement Fund (REF-23-01-008) and the Suzhou Science and Technology Programme (SYG202106).

Yifan Zhan is with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China, 215123, and also with the School of Electrical Engineering, Electronics and Computer Science, University of Liverpool, Liverpool, L69 3BX, United Kingdom (email: Yifan.Zhan22@student.xjtlu.edu.cn).

Rui Yang is with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, 215123, China (email: R.Yang@xjtlu.edu.cn).

Yong Zhang is with the Department of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan 430081, China (email: zhangyong77@wust.edu.cn).

Zidong Wang is with the Department of Computer Science, Brunel University London, Uxbridge, Middlesex UB8 3PH, United Kingdom (email: Zidong.Wang@brunel.ac.uk).

Corresponding author: Rui Yang

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

learn new classes while preserving discriminative abilities for previously learned classes within the open and dynamic environment, thereby mitigating catastrophic forgetting [26], [39]. Through the employment of methodologies such as knowledge distillation and data replay, class incremental learning enables models to maintain effectiveness in earlier tasks throughout the continuous learning process [34], [41]. However, no fault diagnosis method currently exists that can effectively mitigate the impact of catastrophic forgetting while simultaneously tackling domain shift and unlabeled data challenges.

Inspired by the above discussions, the purpose of this study is to mitigate the impact of catastrophic forgetting in cross-domain fault diagnosis, especially in scenarios where the data from the target domain lacks labels. To address this, a novel unsupervised class incremental learning network (UCILN) is proposed in this paper for cross-domain fault diagnosis. Unlike existing unsupervised class incremental learning methods in other fields [14], [21], UCILN processes both the labeled source domain data and the unlabeled target domain data in a streaming setting during the incremental phase (i.e., the samples are fed into UCILN one by one, allowing the model to timely capture new data information) [13]. This learning strategy mirrors the continuous acquisition and assimilation of information by the brain from the environment. To mitigate forgetting, UCILN integrates a feature replay mechanism to continuously replay data features and maintain a balance between old and new information. Besides, product quantization (PQ) is utilized to obtain a compact representation with minimal loss in reconstruction. Drawing from the hippocampal indexing theory [29], this compression representation method simulates the human process of compressing and encoding newly acquired information for efficient storage and retrieval. Moreover, semi-frozen and semi-updated incremental strategy is employed in this paper, which maintains high recognition accuracy for all fault types over long-term use.

In the absence of class labels for target domain data, UCILN leverages UDTL based on discrepancy to assess domain disparities. It effectively aligns the fault features from different domains by minimizing the discrepancy in feature distributions between the source and target domains [36]. This enables UCILN to effectively transfer fault diagnosis knowledge learned in the source domain to the target domain, ensuring high diagnostic accuracy for new faults even in the absence of labeled data in the target domain.

The following outlines the primary contributions of this paper:

1) Mitigating the impact of catastrophic forgetting in cross-domain fault diagnosis in the absence of labels for the target domain data, by employing unsupervised deep transfer learning and class incremental learning techniques;
2) Proposing a novel unsupervised class incremental learning network (UCILN) for incrementally predicting unlabeled faults within the target domain, which incorporates real feature replay to continuously review old information, and implements a semi-frozen and semi-updated incremental strategy to balance the retention of old knowledge with the acquisition of new information.
3) Designing a memory module to encode and preserve data distribution information within the deep feature space, which mimics the memory mechanism of the human brain and upholds data privacy standards in comparison to directly storing old class data.
4) Evaluating the UCILN model's effectiveness on the CWRU and PU bearing datasets, with the results demonstrating outstanding performance characterized by high incremental accuracy and low incremental forgetting rates.

The subsequent sections of the paper are structured as follows. Section II provides a detailed exposition of the overall architecture of UCILN and explains UCILN-based fault diagnosis process. Section III introduces the experimental setup and presents an analysis of the results obtained in this study. Finally, section IV offers concluding remarks on the study and outlines potential avenues for future research.

II. Methodology

To mitigate the issue of catastrophic forgetting in class incremental learning when target domain data is unlabeled in cross-domain fault diagnosis, this study proposes UCILN, depicted with the detailed structure in Figure 1(a) and the semi-fixed and semi-updated incremental strategy as illustrated in Figure 1(b). Notably, the model maintains the same structure across each phase, with changes in data illustrated in the figure using phase 2 as an illustration.

In terms of the input, during each phase, the model receives new fault type data from both the source and target domains and undergoes corresponding training and feature encoding. In other words, each phase corresponds to a fault diagnosis task, with newly introduced fault classes in each task being entirely novel. $D_s^p = \{x_i^s, y_i^s\}_{i=1}^{N_s}$ and $D_t^p = \{x_i^t\}_{i=1}^{N_t}$ represent labeled source domain fault dataset and unlabeled target domain fault dataset in phase p respectively, where N_s and N_t denote the number of fault samples introduced in the source and target domain in phase p respectively. It is important to note that the fault data classes introduced in each phase do not overlap with those from previous phases. Consequently, the model consistently encounters new fault classes that it has not encountered before. Simultaneously, features from previous data are encoded and preserved for future replay. The encoded source and target domain fault features of phase p are denoted as $A_s^p = \{a_i^s, y_i^s\}_{i=1}^{N_s}$ and $A_t^p = \{a_i^t\}_{i=1}^{N_t}$, where a_i^s and a_i^t represent the encoded feature representation of x_i^s and x_i^t respectively. After the training in each phase, all seen class data from the target domain will be employed to assess the model, to validate the model's fault diagnosis capabilities across both old and new classes.

A. Structure of the Proposed UCILN

In the detailed structure of UCILN (Figure 1(a)), the model maintains a consistent structure across each phase, primarily consisting of a feature extractor, memory module, and classifier. In the figure, the input and storage of data are illustrated using phase 2 as an example. The feature extractor

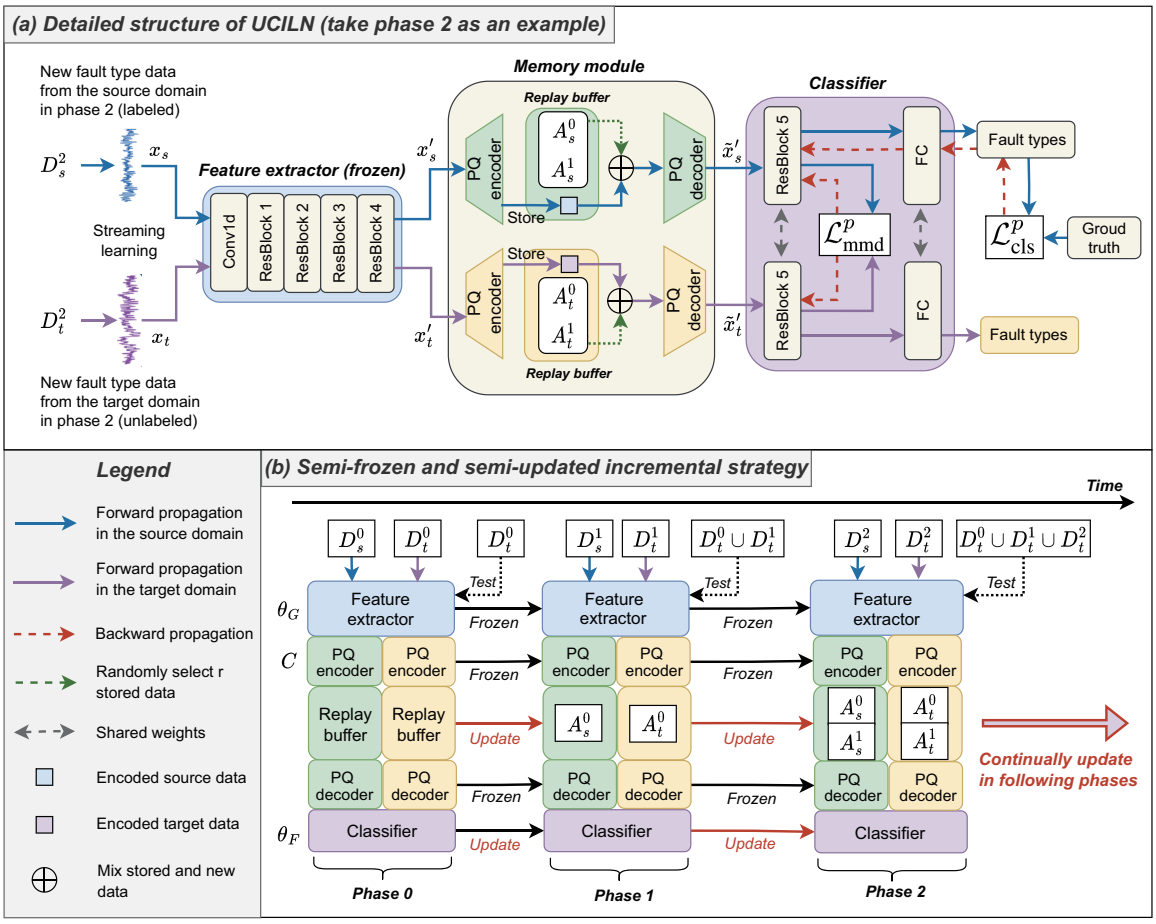

Fig. 1. Structure Diagram of UCILN Approach

is composed of the initial 15 convolutional layers from a one-dimensional ResNet-18 (1D ResNet-18). Its primary role is to extract fault features, which are subsequently stored and utilized for later replay.

Subsequently, the memory module incorporates a PQ encoder, a replay buffer, and a PQ decoder in each domain, tasked with indexing, storing, and reconstructing features, respectively. Efficient processing and storage of these features are achieved through PQ [15], a compression method highly suitable for storing and replaying encoded features. The detailed process will discussed in the next section. For implementing domain adaptation, UCILN designs independent PQ encoder, replay buffer and PQ decoder to separately store and process fault feature information from the source and target domains, addressing the differences in fault characteristics across various operating conditions. This design not only enhances the model's cross-domain adaptability but also allows it to independently learn and store fault features for each environment when dealing with multiple industrial settings. Regarding mitigating forgetting, by storing fault features from various stages, the replay buffer ensures that the model can retain and review known fault characteristics through a replay mechanism while learning new fault types, effectively preventing the model from forgetting previously learned fault types.

Finally, the classifier is designed to predict the types of fault signals from the target domain, with both the source and target domain classifiers sharing the same weight. It consists of the remaining 3 layers of 1D ResNet-18 (comprising 2 convolutional layers and 1 fully connected layer (FC)). Notably, using a 1D ResNet-18 model offers advantages such as enhanced storage efficiency, minimized space requirements, and simplified saving and deployment processes. The detailed parameter information of UCILN is shown in Table I.

TABLE I
DETAILED STRUCTURE OF UCILN

Layer	Channels × kernel size	Output size
Input	/	(1, 1024)
Conv1d	64×7, stride = 2	(64, 512)
ResBlock 1	$(64 \times 3) \times 4$, stride = 1	(64, 512)
ResBlock 2	$(128 \times 3) \times 4$, stride = 1	(128, 256)
ResBlock 3	$(256 \times 3) \times 4$, stride = 1	(256, 128)
ResBlock 4	$(512 \times 3) \times 2$, stride = 1	(512, 64)
PQ encoder	/	(64, 32)
Replay buffer	/	(r+1, 64, 32)
PQ decoder	/	(r+1, 64, 512)
ResBlock 5	$(512 \times 3) \times 2$, stride = 1	(r+1, 512, 64)
FC	/	(r+1, number of classes)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

B. Semi-frozen and Semi-updated Incremental Strategy

As depicted in Figure 1(b), new fault-type data from two domains is continuously introduced to update the model, while the compressed features of previously trained data are stored in replay buffers to mitigate catastrophic forgetting.

To achieve a balance between retaining old knowledge and acquiring new information, UCILN employs a semi-frozen and semi-updated incremental strategy. Specifically, during training, the replay buffer and classifier are updated to adapt to new class data, while the PQ encoder/decoder and feature extractor remain frozen after phase 0 to preserve the effectiveness of stored information. Consequently, the learning strategy varies between phase 0 and subsequent incremental phases.

1) Learning strategy in phase 0: In phase 0, the optimization of the feature extractor and classifier is conducted independently from the optimization of the PQ encoder and decoder. The feature extractor θ_G and classifier θ_F are firstly trained. The multi-kernel maximum mean discrepancy loss is applied to minimize the distribution difference between source and target domains:

$$\begin{aligned}\mathcal{L}^0_{\text{mmd}} =& \frac{1}{N_s^2}\sum_{i=1}^{N_s}\sum_{j=1}^{N_s}k(x_i^s,x_j^s) - \frac{2}{N_sN_t}\sum_{i=1}^{N_s}\sum_{j=1}^{N_t}k(x_i^s,x_j^t)\\ &+ \frac{1}{N_t^2}\sum_{i=1}^{N_t}\sum_{j=1}^{N_t}k(x_i^t,x_j^t)\end{aligned} \tag{1}$$

where $k(\cdot,\cdot)$ is the Gaussian kernel function [12]. Alongside $\mathcal{L}^0_{\text{mmd}}$, the standard cross-entropy loss $\mathcal{L}^0_{\text{cls}}$ is calculated within the source domain to measure the difference between the predicted probabilities and the actual labels:

$$\mathcal{L}^0_{\text{cls}} = -\sum_{i=1}^{N_s} y_i^s \log \hat{y}_i^s \tag{2}$$

where $\hat{y}_i^s$ represents the model's predicted classification probability for samples from the source domain. Therefore, the overall optimization objective for θ_G and θ_F is:

$$\min_{\theta_G,\theta_F} \mathcal{L}^0 = \min_{\theta_G,\theta_F}(\mathcal{L}^0_{\text{cls}} + \lambda\cdot\mathcal{L}^0_{\text{mmd}}) \tag{3}$$

where λ represents the trade-off parameter.

Following the training of θ_G and θ_F in phase 0, their parameters remain fixed to train the PQ codebook, a pivotal component of the PQ encoder/decoder (considering that the data processing in the memory module remains consistent across both domains, the relevant illustration does not differentiate between domains for the sake of simplicity and clarity.). Specifically, after feeding data into the feature extractor, the set of extracted feature $X' = \{x'_i\}_{i=1}^N$ is obtained. In the PQ encoder, X' is transformed into a matrix form and then partitioned into M subspaces:

$$X' = \begin{bmatrix} x'_{11} & x'_{12} & \dots & x'_{1M} \\ x'_{21} & x'_{22} & \dots & x'_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x'_{N1} & x'_{N2} & \dots & x'_{NM} \end{bmatrix} \tag{4}$$

where x'_{ij} represents the i-th tensor of the j-th subspace. Next, the K-Means clustering algorithm [15] is applied to determine K cluster centers of each subspace. For M subspaces, a cluster center matrix C is obtained as follows:

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1M} \\ c_{21} & c_{22} & \dots & c_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ c_{K1} & c_{K2} & \dots & c_{KM} \end{bmatrix} \tag{5}$$

where c_{zj} represents the z-th cluster center of the j-th subspace. Through iterative optimization, update C by minimizing the Euclidean distance between x'_{ij} and its assigned cluster center c_{zj}, which is shown as follows:

$$\min_C \mathcal{L}_{pq} = \min_C \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{z=1}^{K}\|x'_{ij} - c_{zj}\|^2 \tag{6}$$

Therefore, the comprehensive optimization objective for phase 0 encompasses both components:

$$\min_{\theta_G,\theta_F,C} \mathcal{L}^0_{\text{total}} = \min_{\theta_G,\theta_F}(\mathcal{L}^0_{\text{cls}} + \lambda\cdot\mathcal{L}^0_{\text{mmd}}) + \min_C \mathcal{L}_{pq} \tag{7}$$

After the optimization of phase 0 is completed, the data features from phase 0 are encoded and stored in the replay buffer. Specifically, for the PQ encoding process, each subvector x'_{ij} is mapped to the cluster index l_{ij} of its nearest cluster center c_{zj} in C (trained by (6)) as follows:

$$l_{ij} = \text{argmin}_{1\le z\le K}\|x'_{ij} - c_{zj}\|^2 \tag{8}$$

Consequently, x'_i is encoded into M labels, which is denoted as (9):

$$a_i = [l_{i1}, l_{i2}, \dots, l_{iM}] \tag{9}$$

Therefore, after the completion of phase 0, $A^0_s = \{a_i^s, y_i^s\}_{i=1}^{N_s}$ and $A^0_t = \{a_i^t\}_{i=1}^{N_t}$ are respectively stored in the replay buffers of the source and target domains. After phase 0, θ_G and C are frozen, while θ_F and replay buffers are updated in the subsequent incremental phases.

Remark 1: In light of the requirement for class incremental learning networks to adapt to a continuous stream of data, a maximum memory threshold is established for the replay buffer. To elaborate, the buffer retains quantized indices up to its maximum capacity. Upon reaching this limit, samples are randomly removed from the class that has accumulated the most instances.

2) Learning strategy in following phases: After the completion of phase 0, UCILN employs streaming learning techniques [10], [13] to learn new types of samples one by one, allowing it to timely capture new patterns and information.

After feature extraction, the PQ encoder encodes the feature of newly introduced data using (8) and (9), which is subsequently stored in the replay buffer. Simultaneously, in each domain, r encoded representations are randomly selected from the stored data in previous phases $(A^{p-1}, A^{p-2}, \dots, A^0)$ and mixed with newly encoded representation for reconstruction. Therefore, a total of $r+1$ representations are reconstructed in each domain per replay. In the PQ decoder, the encoded representation a_i is decoded to obtain an approximate representation $\tilde{x}'_i$ of the original feature x'_i. This process involves locating the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

corresponding cluster center c_{zj} based on the cluster index l_{ij} stored in a_i, and then using this cluster center to approximate the original feature. The corresponding formula is shown as follows:

$$\tilde{x}'_{ij} = c_{zj}, \quad \text{where } z = l_{ij} \tag{10}$$

where $\tilde{x}'_{ij}$ represents the decoded representation of x'_{ij}. Finally, these decoded features are then fed into the classifier to predict fault labels.

To achieve domain adaptation, the multi-kernel maximum mean discrepancy is applied to the output features of the classifier, which is defined as:

$$\mathcal{L}^p_{\text{mmd}} = \frac{1}{(r+1)^2}\sum_{i=1}^{r+1}\sum_{j=1}^{r+1}[(k(\tilde{x}_i^{'s},\tilde{x}_j^{'s}) - 2k(\tilde{x}_i^{'s},\tilde{x}_j^{'t}) + k(\tilde{x}_i^{'t},\tilde{x}_j^{'t})] \tag{11}$$

where $x_i^{'s}$ and $x_i^{'t}$ represent reconstructed features per replay in the source and target domain respectively. Besides, the standard cross-entropy loss on reconstructed data $\mathcal{L}^p_{\text{cls}}$ is calculated in the source domain:

$$\mathcal{L}^p_{\text{cls}} = -\sum_{i=1}^{r+1} y_i^s \log \hat{y}_i^s \tag{12}$$

where y_i^s represents the actual labels of reconstructed features in the source domain and $\hat{y}_i^s$ represents the corresponding predicted classification probability. Therefore, the overall optimization objective for incremental learning after phase 0 is defined as follows:

$$\min_{\theta_F} \mathcal{L}^p_{\text{total}} = \min_{\theta_F}(\mathcal{L}^p_{\text{cls}} + \lambda \cdot \mathcal{L}^p_{\text{mmd}}) \tag{13}$$

C. Fault Diagnosis Process

The UCILN-based unsupervised cross-domain class incremental fault diagnosis process consists of the following five steps:

Step 1. Data collection and preprocessing in the initialization phase: Collect data from faulty equipment in both the source and target domains, segment the long sequence data into 1024-sample segments, and preprocess it using min-max normalization.

Step 2. Model initialization: Initialize the feature extractor θ_G, classifier θ_F, and codebooks. Encode and store the features extracted by the feature extractor in the replay buffer through the PQ encoder. After training, save the parameters of the feature extractor θ_G and classifier θ_F, as well as the codebooks and replay buffer data in the memory module of two domains.

Step 3. Data collection and preprocessing in incremental phase: Collect new fault type data in two domains and apply the same preprocessing as shown in step 1.

Step 4. Incremental training: Restore the parameters of the feature extractor θ_G and classifier θ_F saved in the previous phase. Additionally, reinstate the codebooks and replay buffer data in the memory module. Perform streaming learning on the incremental datasets by extracting features using feature extractor and compressing and storing features with PQ encoder. Mix the newly encoded representation with randomly selected fault representation from replay buffer. Finally, update the memory module and classifier θ_F, and save the updated parameters of the feature extractor θ_G and classifier θ_F, as well as the codebooks and replay buffer data in the memory module.

Step 5. Target domain fault diagnosis: Apply the trained feature extractor θ_G and classifier θ_F on D_t^p, and output the diagnosis results for target domain data.

When new fault types appear, steps 3, 4 and 5 are repeated for target domain fault diagnosis. To illustrate the fault diagnosis process clearly, the pseudo-code of UCILN is provided in Algorithm 1.

Algorithm 1 UCILN-based Fault Diagnosis Process

Input: Labeled source domain fault data $D_s^p = \{x_i^s, y_i^s\}_{i=1}^{N_s}$ and unlabeled target domain fault data $D_t^p = \{x_i^t\}_{i=1}^{N_t}$
Output: Diagnosis results for target domain data

Optimization in phase 0**:**
for *each batch* in D_s^0 and D_t^0 **do**
1. Train feature extractor θ_G and classifier θ_F by (3);
2. Train the PQ codebooks by (6);
3. Encode and save data of phase 0 by (8) and (9);
4. Save θ_G, θ_F, and codebooks.
end for

Optimization in subsequent phases:
for *each sample* in D_s^p and D_t^p of current phase **do**
1. Restore θ_G, θ_F, codebooks and replay buffer data stored in previous phase $p-1$;
2. Extract features by the feature extractor θ_G;
3. Compress and store features with PQ encoder by (8) and (9);
4. Mix the newly encoded representation with r randomly selected representation;
5. Decompress the encoded representation by (10);
6. Predict fault labels by the classifier θ_F;
7. Update θ_F by (13);
8. Save θ_G, θ_F, and codebooks.
end for

Target domain fault diagnosis:
1. Apply the trained feature extractor θ_G and classifier θ_F on D_t^p;
2. Output the diagnosis results for target domain data.

III. Experimental Results and Analysis

To validate the effectiveness of UCILN, this study compares it with seven class incremental learning models on two datasets. In addition, five ablation experiments are conducted to further validate the reliability of UCILN.

A. Dataset Description

UCILN is assessed on a widely used bearing fault dataset from Case Western Reserve University (CWRU), as shown in Fig. 2. The dataset includes drive-end faults covering health (H), inner race fault (IR), outer race fault (OR), and ball fault (BF). The faults are sampled at 12 kHz, generated through

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

an electro-discharge machine with fault diameters of 7 mils, 14 mils, 21 mils, and 28 mils. The bearings experience four loads ranging from 0 to 3 horsepower, leading to diverse data distributions corresponding to each load condition.

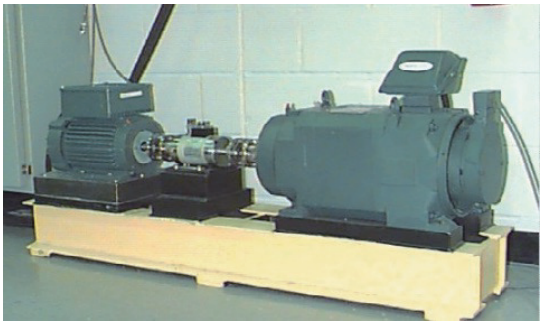

Fig. 2. The Testbed of the CWRU dataset

TABLE II
SETTING OF FAULT TYPES IN DIFFERENT PHASES OF THE CWRU DATASET

Label	Condition	Diameter	Phase
0	H	0	0
1	IR	21	
2	OR	21	
3	BF	21	
4	BF	28	1
5	IR	28	2
6	BF	7	3
7	BF	14	4
8	IR	7	5
9	OR	7	6

In this experiment, data obtained under the 3-horsepower condition is used as the source domain data, whereas data from the 1-horsepower condition is designated as the target domain data. Each domain includes ten fault classes with the setting of fault types in different phases outlined in Table II. The long sequential data undergoes preprocessing and segmentation into 1024-sample segments using min-max normalization, forming 800 training samples and 200 testing samples for each fault class within each domain. The experiment starts by selecting data from the four fault classes in each domain as the initial set in phase 0. Subsequently, in each subsequent incremental phase, one new fault class from both domains is introduced into the model for incremental training.

UCILN is also evaluated using a dataset from Paderborn University (PU), as illustrated in Figure 3. This dataset consists of vibration signals from 6203-type ball bearings, captured using piezoelectric accelerometers attached to the bearing housings, with a sampling frequency of 64 kHz. It comprises vibration and motor current signals from 32 ball bearings, including 12 with artificially manufactured damage, 14 with real damage samples generated from accelerated lifetime tests, and 6 in normal condition. Four failure modes are IR, OR, H, and certain compound faults (IR-OR). The severity of faults is determined by the extent of damage, specifically, the length of the damaged surface in the rolling direction (0–2 mm, 2–4.5 mm, >4.5 mm).

The dataset consists of four operational conditions, determined by varying the rotational speed of the drive system, the radial forces on the bearings, and the load torque on the drive system. In this experiment, data from the condition with a radial force of 1000 N is considered as the source domain data, while data from the condition with a radial force of 400 N is considered as the target domain data. Each domain comprises 10 fault classes, and the setting of fault types in different phases is provided in Table III. Preprocessing of the long sequence data is conducted similarly to that for the CWRU dataset.

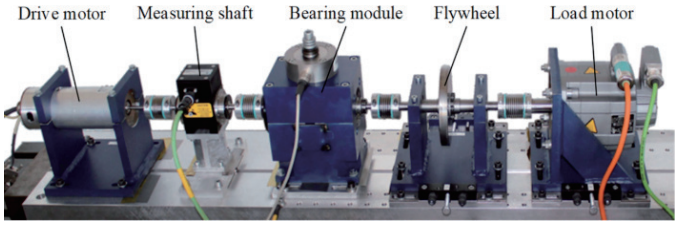

Fig. 3. The Testbed of the PU dataset

TABLE III
SETTING OF FAULT TYPES IN DIFFERENT PHASES OF THE PU DATASET

Label	Bearing Code	Condition	Phase
0	K001	H	0
1	KI15	IR	
2	KA15	OR	
3	KI17	IR	
4	KA16	OR	1
5	KA04	OR	2
6	K004	H	3
7	K002	H	4
8	K003	H	5
9	KI04	IR	6

B. Evaluation Metrics

This study employs metrics from [39] to evaluate the methods, with a focus on incremental accuracy and incremental forgetting rate. Besides, a new metric (memory rate) is proposed in this paper to provide a comprehensive evaluation of the model. These metrics play a crucial role in assessing the effectiveness of class incremental learning methods, where an ideal model is characterized by a high average incremental accuracy and a minimal average incremental forgetting rate. Table IV provides a summary of the evaluation metrics involved in the experiment with detailed explanations as follows:

1) Acc_p evaluates the classification accuracy of model in phase p for all seen classes.
2) For_p is used to evaluate the degree of forgetting of previously learned knowledge by the model in phase p. Following the completion of the task in phase p, the incremental forgetting rate for the task in phase i is calculated as follows:

$$f_p^i = \max_{t \in 0,\ldots,p-1} (c_{t,i} - c_{p,i}), \forall i < p \tag{14}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

TABLE IV
EVALUATION METRICS

Evaluation Metrics	Formula
Incremental accuracy	Acc_p
Incremental forgetting rate	$For_p = \frac{1}{p}\sum_{i=0}^{p-1} f_p^i$
Memory rate	$MR_p = \frac{1}{2}(Acc_p + 1 - For_p)$
Average incremental accuracy	$\overline{Acc} = \frac{1}{P}\sum_{i=1}^{P} Acc_i$
Average incremental forgetting rate	$\overline{For} = \frac{1}{P}\sum_{i=1}^{P} For_i$
Average memory rate	$\overline{MR} = \frac{1}{P}\sum_{i=1}^{P} MR_i$

where $c_{t,i}$ denotes the classification accuracy of the model on the task in phase i after learning the task in phase t. Consequently, the forgetting rate for the phase p can be computed as:

$$For_p = \frac{1}{p}\sum_{i=0}^{p-1} f_p^i \tag{15}$$

3) A new metric MR_p is introduced in this paper to comprehensively evaluate the model of phase p:

$$MR_p = \frac{1}{2}(Acc_p + 1 - For_p) \tag{16}$$

4) The average incremental accuracy of the model is:

$$\overline{Acc} = \frac{1}{P}\sum_{i=1}^{P} Acc_i \tag{17}$$

where P represents the total number of incremental phases after the initial phase.

5) The average incremental forgetting rate of the model is:

$$\overline{For} = \frac{1}{P}\sum_{i=1}^{P} For_i \tag{18}$$

6) The average memory rate of the model is:

$$\overline{MR} = \frac{1}{P}\sum_{i=1}^{P} MR_i \tag{19}$$

C. Comparative Experiments

In this experiment, the effectiveness of UCILN is compared with seven other class incremental learning models— Fine-tuning, iCaRL [27], PASS [40], MAFDRC [2], AANets [22], MEMO [38], and GSAEMA [25]—utilizing the CWRU and PU dataset. UCILN innovatively considers both the data distribution differences between old and new classes and the differences between the source and target domain data distributions. The goal of this experiment is to compare different class incremental learning methods to evaluate the effectiveness of UCILN in handling these two types of distribution differences. Since existing comparison methods do not comprehensively address both types of differences, this experiment ensures a fair comparison by training all models with simultaneous input from both source and target domains. Additionally, it employs the same unsupervised domain adaptation techniques and 1D ResNet-18 backbone as UCILN to ensure that both types of differences are accounted for in the comparison.

UCILN utilizes Faiss PQ [16] for the quantization-decoding operation, with 32 codebooks and a codebook size of 256. The size of the replay buffer for each domain is set to 80% of the total training samples (i.e., 6400 samples). The specific hyperparameters in the experiments are detailed in Table V. Based on Table I and Table V, the complexity metrics of UCILN are as shown in Table VI. "FLOPs (both domains)" indicates the floating-point operations required across both domains to measure computational complexity. "Total params" shows the total number of parameters in the model, while "Trainable params" refers to the number of parameters that can be adjusted during training. Additionally, the table provides the average time for codebook training and the storage space required for the codebook, as well as for parameters θ_G and θ_F, and for data of specific shapes.

TABLE V
THE HYPERPARAMETERS OF THE EXPERIMENTS FOR UCILN

Hyperparameters	Value
Epochs	150
Batch size	128
The size of each codebook	256
The number of codebooks	32
Reconstructed old instances per replay	50
Replay buffer size in each domain	6400
Initial learning rate	0.01
Final learning rate	0.001

Each method undergoes training via the stochastic gradient descent algorithm with a separate learning rate decay strategy for each class [26], [42]. The learning rate decay factor is determined as below, with the initial and final learning rates (η_i and η_f) set as 0.01 and 0.001, respectively:

$$\gamma = e^c, \quad \text{where} \quad c = s \cdot \ln\left(\frac{\eta_f}{\eta_i}\right) \tag{20}$$

where s represents the decay step size, signifying that decay occurs after each processing of a sample of length s.

*1) **Case Study on CWRU Dataset**:* The experimental results of different methods are shown in Table VII. Since all methods initialize the 1D ResNet-18 in the same way, there is no difference in the test results in phase 0, with an accuracy of 99.50% in the target domain. Therefore, the result of phase 0 is omitted in Table VII.

In order to clearly present the results in the table, the incremental accuracy and incremental forgetting rate curve of UCILN and compared methods in different phases are shown in Figure 4(a) and 4(b) respectively. UCILN consistently attains the highest incremental accuracy throughout all phases and exhibits the lowest incremental forgetting rate compared to the other methods. UCILN significantly mitigates catastrophic forgetting, with an average incremental accuracy of 97.04% and an average incremental forgetting rate of 0.91%, effectively supporting incremental learning of knowledge and the retention of knowledge from existing tasks. As the memory

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

TABLE VI
COMPLEXITY OF UCILN

Phase	Input shape	FLOPs (both domains)	Total params	Trainable params	Average codebook training time	Codebook storage space (both domains)	Storage space for θ_G and θ_F	Storage space for (1, 1024)
Phase 0	(1,1024)	0.70G	3.85M	3.85M	28s	1MB	6.04 MB	46.12 byte
Incremental phases	(1,1024)	10.80G	3.85M	1.58M	/	1MB	6.04 MB	46.12 byte

TABLE VII
RESULTS OF COMPARED EXPERIMENTS IN THE TARGET DOMAIN ON THE CWRU DATASET (%)
(**BOLD** AND UNDERLINE REPRESENT THE BEST AND SECOND BEST RESULTS RESPECTIVELY.)

Phase	Fine-tuning	PASS [40]	MEMO [38]	GSAEMA [25]	iCaRL [27]	MAFDRC [2]	AANets [22]	UCILN
Incremental Accuracy (%)								
1	78.30 ± 0.00	79.59 ± 4.63	78.70 ± 0.40	80.70 ± 1.23	84.90 ± 0.60	93.63 ± 0.53	98.20 ± 0.31	$\mathbf{98.40 \pm 0.25}$
2	66.08 ± 0.00	51.19 ± 1.68	67.58 ± 2.67	76.90 ± 0.45	91.77 ± 0.23	89.76 ± 0.76	91.75 ± 0.24	$\mathbf{98.91 \pm 0.27}$
3	43.57 ± 0.00	50.92 ± 0.08	70.86 ± 1.35	74.00 ± 3.00	82.02 ± 3.98	89.21 ± 1.06	85.71 ± 0.30	$\mathbf{98.29 \pm 0.44}$
4	43.94 ± 0.00	54.50 ± 1.01	70.19 ± 2.12	76.80 ± 2.67	78.20 ± 4.60	84.33 ± 1.16	83.38 ± 0.33	$\mathbf{95.06 \pm 0.41}$
5	26.00 ± 0.00	52.67 ± 1.08	73.44 ± 0.95	76.10 ± 1.89	74.54 ± 3.40	82.32 ± 1.44	85.72 ± 0.19	$\mathbf{96.05 \pm 0.21}$
6	25.80 ± 0.00	49.83 ± 0.17	75.95 ± 2.25	77.30 ± 2.34	69.78 ± 2.38	79.06 ± 1.15	81.20 ± 0.20	$\mathbf{95.50 \pm 0.21}$
$\overline{Acc}$	47.28 ± 0.00	56.45 ± 1.44	72.79 ± 1.62	76.97 ± 1.93	80.20 ± 2.53	86.39 ± 1.02	87.66 ± 0.26	$\mathbf{97.04 \pm 0.31}$
Incremental Forgetting Rate (%)								
1	26.50 ± 0.00	73.04 ± 1.46	1.50 ± 0.67	9.70 ± 0.57	17.50 ± 1.38	5.17 ± 1.78	1.75 ± 0.34	$\mathbf{1.49 \pm 0.18}$
2	62.44 ± 0.00	39.88 ± 4.44	12.92 ± 2.00	4.80 ± 2.12	5.81 ± 0.06	5.43 ± 1.84	11.38 ± 0.57	$\mathbf{0.75 \pm 0.09}$
3	81.33 ± 0.00	53.42 ± 2.63	9.83 ± 0.17	8.77 ± 1.34	16.29 ± 4.55	6.18 ± 0.19	17.50 ± 0.53	$\mathbf{0.62 \pm 0.27}$
4	78.03 ± 0.00	52.25 ± 1.13	16.13 ± 2.00	13.23 ± 0.98	17.44 ± 7.54	10.50 ± 0.36	16.22 ± 0.42	$\mathbf{1.56 \pm 0.40}$
5	82.85 ± 0.00	55.90 ± 1.78	11.13 ± 1.67	14.60 ± 2.76	15.13 ± 1.63	11.71 ± 1.93	8.18 ± 0.65	$\mathbf{0.32 \pm 0.07}$
6	84.58 ± 0.00	66.25 ± 1.25	10.24 ± 1.77	18.85 ± 1.45	18.75 ± 2.22	14.26 ± 1.94	14.63 ± 0.74	$\mathbf{0.70 \pm 0.18}$
$\overline{For}$	69.29 ± 0.00	56.79 ± 2.28	10.29 ± 1.38	11.66 ± 1.54	15.15 ± 2.90	8.87 ± 1.34	11.61 ± 0.54	$\mathbf{0.91 \pm 0.20}$
Memory Rate (%)								
1	75.90 ± 0.00	53.28 ± 1.59	88.60 ± 0.14	85.50 ± 0.33	83.70 ± 0.39	94.23 ± 0.63	98.23 ± 0.02	$\mathbf{98.46 \pm 0.04}$
2	51.82 ± 0.00	55.66 ± 1.38	77.33 ± 0.34	86.05 ± 0.84	92.98 ± 0.09	92.17 ± 0.54	90.19 ± 0.17	$\mathbf{99.08 \pm 0.09}$
3	31.12 ± 0.00	48.75 ± 1.27	80.51 ± 0.59	82.62 ± 0.83	82.87 ± 0.29	91.52 ± 0.44	84.11 ± 0.12	$\mathbf{98.84 \pm 0.00}$
4	32.95 ± 0.00	51.13 ± 0.56	77.03 ± 0.06	81.79 ± 0.85	80.38 ± 1.47	86.92 ± 0.40	83.58 ± 0.05	$\mathbf{96.75 \pm 0.02}$
5	21.57 ± 0.00	48.38 ± 0.35	81.15 ± 0.36	80.75 ± 0.44	79.71 ± 0.89	85.30 ± 0.25	88.77 ± 0.23	$\mathbf{97.84 \pm 0.17}$
6	20.61 ± 0.00	41.79 ± 0.54	82.86 ± 0.24	79.23 ± 0.45	75.52 ± 0.08	82.40 ± 0.40	83.29 ± 0.27	$\mathbf{97.40 \pm 0.02}$
$\overline{MR}$	39.00 ± 0.00	49.83 ± 0.95	81.25 ± 0.12	82.65 ± 0.20	82.52 ± 0.53	88.76 ± 0.44	88.03 ± 0.14	$\mathbf{98.06 \pm 0.06}$

rate (MR_p) offers a comprehensive perspective on incremental accuracy and incremental forgetting rate, UCILN still consistently outperforms the other methods as depicted in Figure 5.

Fine-tuning and PASS experience catastrophic forgetting, with incremental forgetting rates averaging 69.29% and 56.79%, respectively. Both methods show a significant drop in incremental accuracy. Fine-tuning's incremental accuracy declines from 78.30% in phase 1 to 25.80% in phase 6, while PASS decreases from 79.59% to 49.83% over the same period, indicating that neither model can maintain classification accuracy on all previously seen data. From Figure 4, it can be seen that the initial accuracy of MEMO and GSAEMA is relatively low, but as the training stages progress, the incremental accuracy shows an increasing trend. This indicates that the models may require more training in the initial stages to overcome catastrophic forgetting. iCaRL, MAFDRC and AANets demonstrate relatively high incremental accuracy in the first few phases. However, in terms of incremental forgetting rate, these models exhibit instability. In phase 6, iCaRL's incremental forgetting rate reaches 18.75%, MAFDRC reaches 14.26%, AANets reaches 14.63%, while UCILN is only at 0.70%.

To assess the classification accuracy of the methods for all fault categories after completing all incremental tasks, the confusion matrix for phase 6 is depicted in Figure 6. Both matrices of the Fine-tuning and PASS reveal catastrophic forgetting, indicating a significant reduction in the model's ability to classify old data when new tasks are introduced. Specifically, following the completion of the task in phase 6, both models demonstrate 100% discriminative capability towards the newly introduced data (label 9). However, for labels 3, 4, 5, and 6, the model demonstrates a complete absence of discriminative capability with the accruacy of 0%. Both methods fail to effectively retain information about old classes of fault signals. In contrast, MEMO achieves an accuracy of 94% for predicting label 9, but only 44% for label 1; GSAEMA achieves an accuracy of 99% for predicting label 9, but only 59% for label 1; iCaRL achieves an accuracy of 98% in predicting label 4, but only 34% in predicting label 7; MAFDRC and AANets are able to retain most of the information from the old class data, but their overall classification accuracy is inferior to that of UCILN. UCILN demonstrates the capability to achieve a prediction accuracy of 96% on label 9, while retaining a perfect accuracy of 100% on previously encountered data with labels 1, 4 and 5.

*2) **Case Study on PU Dataset**:* To further evaluate the performance, UCILN is tested on the PU dataset and compared

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

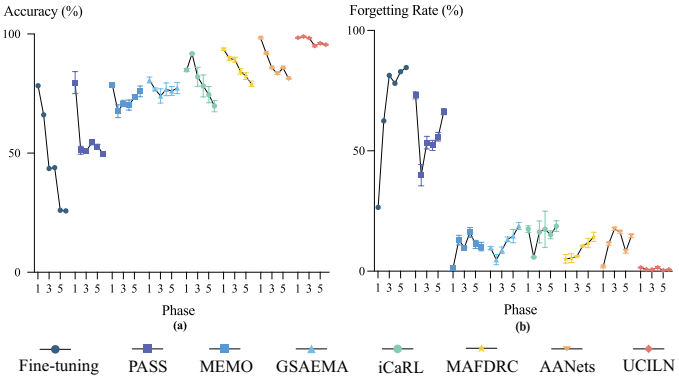

Fig. 4. Incremental Accuracy (a) and Incremental Forgetting Rate (b) of UCILN and Compared Methods in Different Phases on the CWRU Dataset

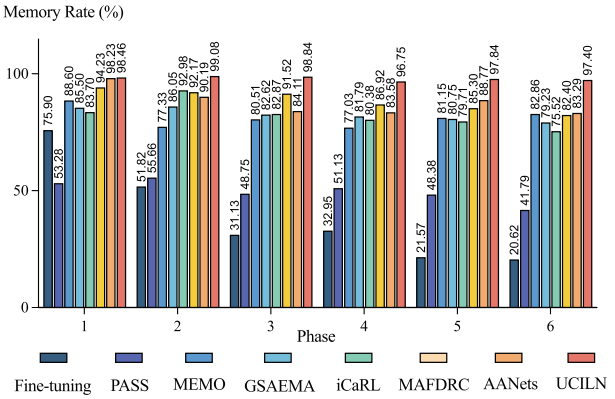

Fig. 5. Memory Rate of UCILN and Compared Methods in Different Phases on the CWRU Dataset

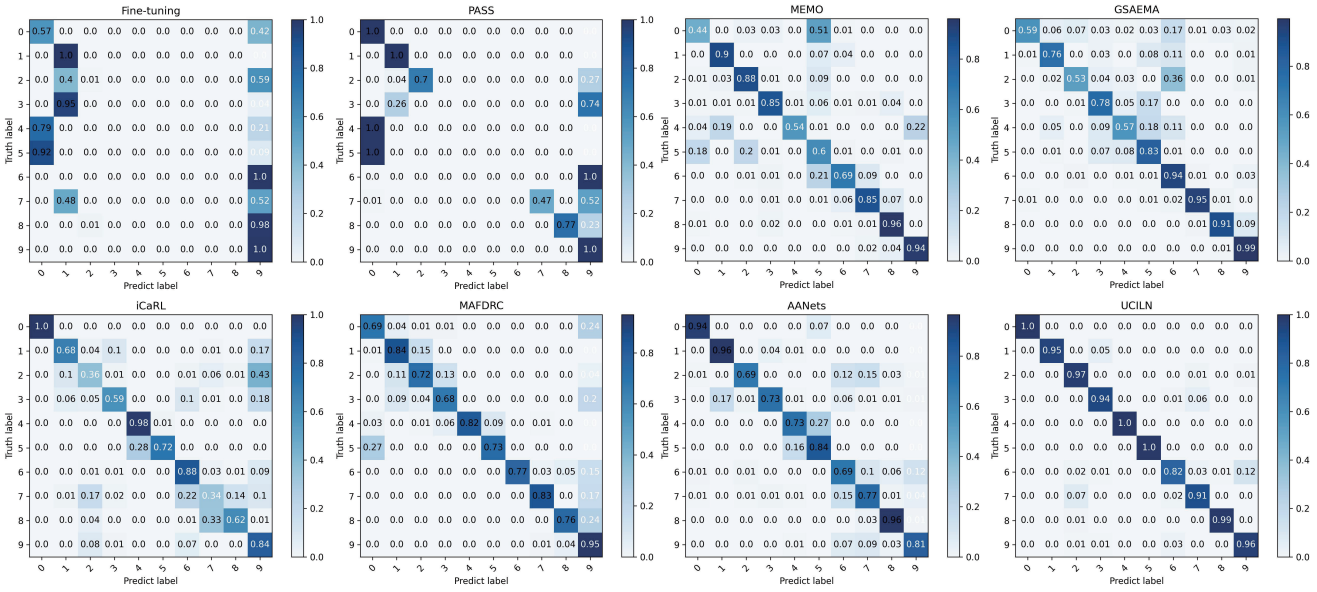

Fig. 6. Confusion Matrices for Comparison Methods in Phase 6 (the last phase) on the CWRU Dataset

with seven other methods in terms of incremental accuracy, incremental forgetting rate, and memory rate. The experimental results are shown in Table VIII. Since all methods initialize the 1D ResNet-18 in the same way, there is no difference in the test results during phase 0, with an accuracy of 99.75% in the target domain. Therefore, the result of phase 0 is omitted in the table. For clear visualization, the incremental accuracy and incremental forgetting rate curve are shown in Figure 7(a) and 7(b) respectively.

From the table and figure, it can be seen that, firstly, UCILN demonstrates the highest incremental accuracy across all phases among all methods. This indicates that UCILN maintains a high discriminative ability towards previously learned classes while progressively learning new classes. Secondly, in terms of incremental forgetting rate, UCILN also outperforms the other methods, with the lowest forgetting rates across all phases. Particularly noteworthy is that UCILN's incremental forgetting rates are even negative in the second (-0.84%) and third (-2.08%) stages, as well as in the fourth (-1.98%) and fifth (-1.61%) stages, suggesting a reinforcement of memory for old classes while learning new ones. Lastly, as depicted in Figure 8, UCILN achieves the highest memory rates across all phases among all models, showing its effectiveness in retaining old data while learning new ones.

D. Ablation Experiments

Ablation experiments are performed to evaluate the validity of UCILN by analyzing the impact of different components and settings. Table IX provides an overview of the five ablation experiments along with corresponding descriptions.

The results of the ablation experiments on the CWRU dataset are summarized in Table X, while Figure 9 shows the incremental accuracy and incremental forgetting rate curves of the ablation experiments. In experiment A1, the removal of the memory module leads to a significant decrease in the average incremental accuracy to 14.10%. Simultaneously, there is a remarkably high average incremental forgetting rate of 99.79%, indicating the essential role of the memory module in preserving model performance and preventing forgetting.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

TABLE VIII
RESULTS OF COMPARED EXPERIMENTS IN THE TARGET DOMAIN ON THE PU DATASET (%)
(**BOLD** AND UNDERLINE REPRESENT THE BEST AND SECOND BEST RESULTS RESPECTIVELY.)

Phase	Fine-tuning	PASS [40]	iCaRL [27]	MAFDRC [2]	GSAEMA [25]	MEMO [38]	AANets [22]	UCILN
Incremental Accuracy (%)								
1	73.08 ± 0.00	67.34 ± 2.69	76.77 ± 0.92	81.80 ± 2.85	71.89 ± 1.87	90.50 ± 0.08	91.02 ± 1.14	$\mathbf{95.35 \pm 0.45}$
2	67.53 ± 0.00	56.68 ± 0.24	67.04 ± 0.54	76.40 ± 7.16	83.47 ± 2.41	81.67 ± 1.89	89.40 ± 0.68	$\mathbf{93.19 \pm 0.56}$
3	43.16 ± 0.00	53.33 ± 0.13	67.50 ± 0.64	75.91 ± 5.20	80.35 ± 0.62	74.14 ± 2.10	90.37 ± 1.29	$\mathbf{92.91 \pm 0.66}$
4	34.01 ± 0.00	50.01 ± 1.29	66.56 ± 0.38	72.40 ± 6.06	72.56 ± 2.95	76.12 ± 1.52	83.38 ± 0.43	$\mathbf{92.92 \pm 0.70}$
5	26.15 ± 0.00	47.83 ± 1.03	67.35 ± 0.49	68.77 ± 4.85	71.04 ± 1.14	68.39 ± 0.52	81.12 ± 1.01	$\mathbf{93.46 \pm 0.15}$
6	25.36 ± 0.00	46.17 ± 0.67	71.20 ± 0.30	65.34 ± 5.33	64.25 ± 0.33	71.85 ± 1.24	78.60 ± 0.17	$\mathbf{91.12 \pm 0.33}$
$\overline{Acc}$	44.88 ± 0.00	53.56 ± 1.01	69.40 ± 0.55	73.44 ± 5.24	73.93 ± 1.55	77.11 ± 1.22	86.81 ± 0.79	$\mathbf{93.16 \pm 0.48}$
Incremental Forgetting Rate (%)								
1	29.70 ± 0.00	74.52 ± 3.52	14.25 ± 0.50	16.16 ± 4.85	15.63 ± 0.78	$\mathbf{2.21 \pm 0.54}$	8.26 ± 0.76	3.91 ± 0.47
2	63.14 ± 0.00	41.91 ± 2.71	24.30 ± 1.02	11.56 ± 4.66	11.79 ± 2.47	15.52 ± 2.40	13.17 ± 0.28	$\mathbf{-0.84 \pm 1.09}$
3	84.26 ± 0.00	50.79 ± 0.69	15.33 ± 0.17	10.31 ± 3.93	22.68 ± 1.92	20.29 ± 1.40	12.02 ± 0.75	$\mathbf{-2.08 \pm 2.08}$
4	77.68 ± 0.00	55.39 ± 1.10	11.49 ± 0.73	11.41 ± 4.05	22.84 ± 2.63	21.30 ± 1.21	10.47 ± 0.62	$\mathbf{-1.98 \pm 1.48}$
5	82.04 ± 0.00	61.64 ± 0.36	11.58 ± 1.13	15.99 ± 1.13	26.76 ± 1.35	25.31 ± 2.53	12.05 ± 0.20	$\mathbf{-1.61 \pm 1.16}$
6	84.54 ± 0.00	67.34 ± 4.27	6.85 ± 0.63	18.32 ± 0.94	26.39 ± 0.89	24.93 ± 0.77	17.34 ± 0.74	$\mathbf{1.46 \pm 2.27}$
$\overline{For}$	70.23 ± 0.00	58.60 ± 2.11	13.97 ± 0.69	13.96 ± 3.26	21.02 ± 1.67	18.26 ± 1.47	12.22 ± 0.56	$\mathbf{-0.19 \pm 1.43}$
Memory Rate (%)								
1	71.69 ± 0.00	46.41 ± 0.42	81.26 ± 0.21	82.82 ± 1.00	78.13 ± 0.55	94.15 ± 0.23	91.38 ± 0.19	$\mathbf{95.72 \pm 0.01}$
2	52.20 ± 0.00	57.39 ± 1.24	71.37 ± 0.24	82.42 ± 1.25	85.84 ± 0.03	83.07 ± 0.25	88.12 ± 0.20	$\mathbf{97.02 \pm 0.27}$
3	29.45 ± 0.00	51.27 ± 0.28	76.08 ± 0.24	82.80 ± 0.64	78.84 ± 0.65	76.92 ± 0.35	89.18 ± 0.27	$\mathbf{97.50 \pm 0.71}$
4	28.17 ± 0.00	47.31 ± 0.10	77.54 ± 0.18	80.50 ± 1.01	74.86 ± 0.16	77.41 ± 0.16	86.46 ± 0.10	$\mathbf{97.45 \pm 0.39}$
5	22.06 ± 0.00	43.10 ± 0.34	77.89 ± 0.32	76.39 ± 1.86	72.14 ± 0.11	71.54 ± 1.00	84.54 ± 0.41	$\mathbf{97.54 \pm 0.50}$
6	20.41 ± 0.00	39.42 ± 1.80	82.18 ± 0.17	73.51 ± 2.19	68.93 ± 0.28	73.46 ± 0.24	80.63 ± 0.29	$\mathbf{94.83 \pm 0.97}$
$\overline{MR}$	37.33 ± 0.00	47.48 ± 0.69	77.72 ± 0.22	79.74 ± 1.32	76.46 ± 0.06	79.43 ± 0.13	86.72 ± 0.24	$\mathbf{96.68 \pm 0.48}$

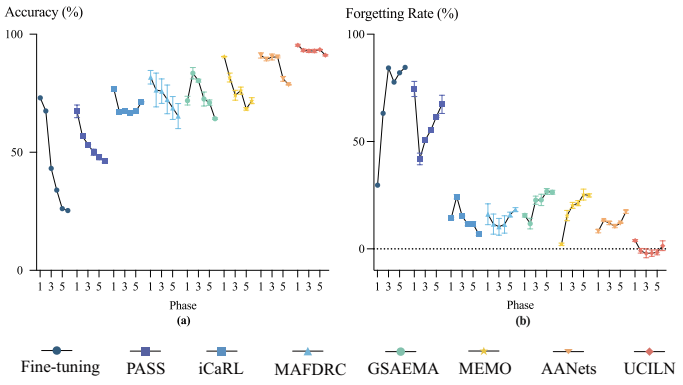

Fig. 7. Incremental Accuracy (a) and Incremental Forgetting Rate (b) of UCILN and Compared Methods in Different Phases on the PU Dataset

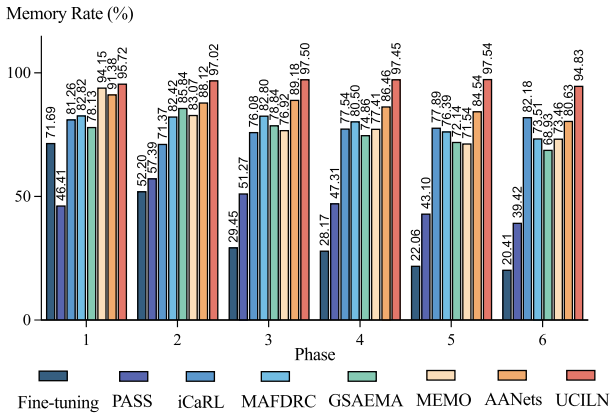

Fig. 8. Memory Rate of UCILN and Compared Methods in Different Phases on the PU Dataset

TABLE IX
SUMMARY OF THE ABLATION EXPERIMENTS

Experiments	Description
A1	The Memory module is removed.
A2	All convolutional parameters are trainable.
A3	The streaming setting is removed.
A4	The size of replay buffer in each domain is set to 25% (2000 samples) of all training samples.
A5	The number of reconstructed instances r is set to 20.

In experiment A2, all convolutional parameters are allowed to be trained, with the result of an average incremental accuracy of 39.68% and a relatively high incremental forgetting rate of 73.82%. This suggests that by freezing the parameters of the feature extractor, information from old data can be preserved. Additionally, freezing parameters ensure that the features of old data preserved by the model remain effective across incremental phases [13]. Therefore, the semi-frozen and semi-updated incremental strategy plays a crucial role in the effective operation of the model.

In experiment A3, the removal of the streaming learning setting leads to noticeable fluctuations in the model's incremental accuracy. Specifically, a negative incremental forgetting rate (-5.73%) is observed in phase 4, indicating that the model unexpectedly reinforces the memory of old tasks during the learning of new tasks. This indicates that the removal of the streaming learning setting causes the model's learning

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

TABLE X
RESULTS OF ABLATION EXPERIMENTS IN THE TARGET DOMAIN (%)
(**BOLD** AND UNDERLINE REPRESENT THE BEST AND SECOND BEST RESULTS RESPECTIVELY.)

Phase	A1	A2	A3	A4	A5	UCILN
Incremental Accuracy (%)						
1	20.00 ± 0.00	40.09 ± 0.10	79.50 ± 0.00	98.40 ± 0.25	98.40 ± 0.25	$\mathbf{98.40 \pm 0.25}$
2	16.67 ± 0.00	41.41 ± 0.21	66.28 ± 0.13	98.25 ± 0.75	98.75 ± 0.09	$\mathbf{98.91 \pm 0.27}$
3	14.29 ± 0.00	46.50 ± 0.43	40.86 ± 0.36	97.42 ± 0.22	97.78 ± 0.41	$\mathbf{98.29 \pm 0.44}$
4	12.50 ± 0.00	45.75 ± 0.09	37.32 ± 0.05	94.31 ± 0.81	94.93 ± 0.26	$\mathbf{95.06 \pm 0.41}$
5	11.11 ± 0.00	39.22 ± 0.19	22.67 ± 0.39	95.22 ± 0.75	95.50 ± 0.13	$\mathbf{96.05 \pm 0.21}$
6	10.00 ± 0.00	25.09 ± 0.50	27.67 ± 0.48	94.59 ± 0.18	95.30 ± 0.32	$\mathbf{95.50 \pm 0.21}$
$\overline{Acc}$	14.10 ± 0.00	39.68 ± 0.25	45.72 ± 0.24	96.37 ± 0.49	96.77 ± 0.24	$\mathbf{97.04 \pm 0.31}$
Incremental Forgetting Rate (%)						
1	99.50 ± 0.00	74.25 ± 0.12	25.13 ± 0.00	1.49 ± 0.18	1.49 ± 0.18	$\mathbf{1.49 \pm 0.18}$
2	99.75 ± 0.00	65.12 ± 0.22	12.62 ± 0.00	0.75 ± 0.24	0.93 ± 0.02	$\mathbf{0.75 \pm 0.09}$
3	99.83 ± 0.00	73.83 ± 1.19	17.90 ± 0.56	0.87 ± 0.16	0.70 ± 0.21	$\mathbf{0.62 \pm 0.27}$
4	99.87 ± 0.00	68.53 ± 0.73	$\mathbf{-5.73 \pm 0.21}$	1.81 ± 0.56	1.58 ± 0.48	1.56 ± 0.40
5	99.90 ± 0.00	74.82 ± 0.41	14.80 ± 0.20	1.25 ± 0.25	0.77 ± 0.08	$\mathbf{0.32 \pm 0.07}$
6	99.91 ± 0.00	86.35 ± 0.01	9.03 ± 0.39	2.83 ± 0.43	2.06 ± 0.37	$\mathbf{0.70 \pm 0.18}$
$\overline{For}$	99.79 ± 0.00	73.82 ± 0.45	12.29 ± 0.23	1.50 ± 0.30	1.26 ± 0.22	$\mathbf{0.91 \pm 0.20}$
Memory Rate (%)						
1	10.25 ± 0.00	32.92 ± 0.01	77.19 ± 0.00	98.46 ± 0.04	98.46 ± 0.04	$\mathbf{98.46 \pm 0.04}$
2	8.46 ± 0.00	38.15 ± 0.01	76.83 ± 0.07	98.75 ± 0.26	98.91 ± 0.04	$\mathbf{99.08 \pm 0.09}$
3	7.23 ± 0.00	36.34 ± 0.38	61.48 ± 0.10	98.28 ± 0.03	98.54 ± 0.10	$\mathbf{98.84 \pm 0.00}$
4	6.32 ± 0.00	38.61 ± 0.32	71.53 ± 0.08	96.25 ± 0.13	96.68 ± 0.11	$\mathbf{96.75 \pm 0.02}$
5	5.61 ± 0.00	32.20 ± 0.11	53.94 ± 0.10	96.99 ± 0.25	97.37 ± 0.03	$\mathbf{97.84 \pm 0.17}$
6	5.05 ± 0.00	19.37 ± 0.25	59.32 ± 0.05	95.88 ± 0.13	96.62 ± 0.03	$\mathbf{97.40 \pm 0.02}$
$\overline{MR}$	7.15 ± 0.00	32.93 ± 0.18	66.71 ± 0.06	97.43 ± 0.14	97.76 ± 0.06	$\mathbf{98.06 \pm 0.06}$

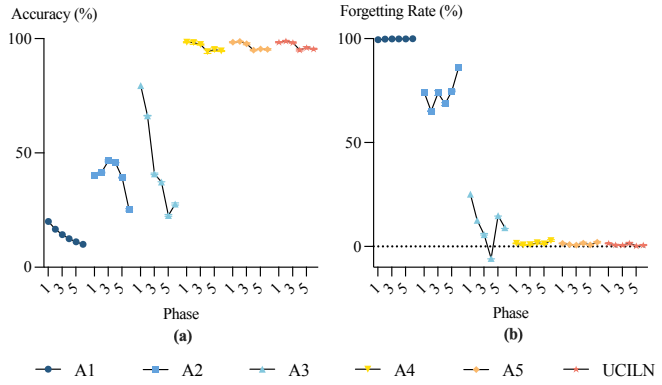

Fig. 9. Incremental Accuracy (a) and Incremental Forgetting Rate (b) in Ablation Experiments in Different Phases

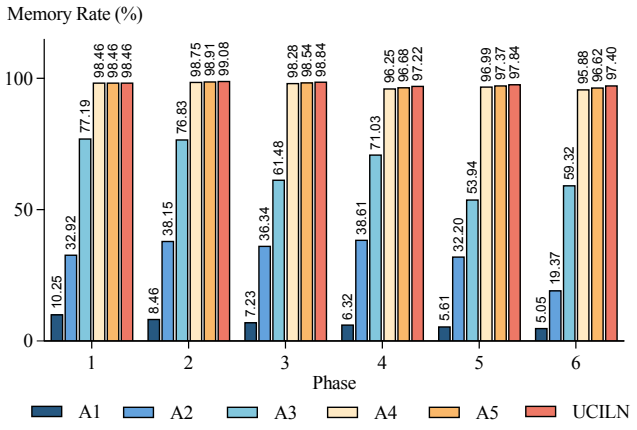

Fig. 10. Memory Rate in Ablation Experiments in Different Phases

behavior to excessively emphasize the memory of old data at certain stages, rather than effectively adapting to the learning of new data. This phenomenon underscores the importance of streaming learning settings in maintaining model balance and adaptability.

In experiment A4, the size of the replay buffer is reduced; while in experiment A5, the number of reconstructed instances per replay is decreased. In both experiments, UCILN exhibits high incremental accuracy and low incremental forgetting rates in each phase. This is because the setting of streaming learning allows a much larger number of old samples to be involved in each training iteration than new samples, enabling the model to recover information from the old data. The memory rate of the ablation experiments is depicted in Figure 10, indicating the crucial role of the replay mechanism, frozen feature extractor, and streaming learning in UCILN.

E. Performance Evaluation on Different Transfer Tasks

To further evaluate the performance, UCILN is assessed on six different transfer tasks on the CWRU dataset, namely T_{30}, T_{31}, T_{32}, T_{20}, T_{21}, and T_{23}. For instance, task T_{30} represents a transfer scenario where the source domain data is collected under a load of 3 horsepower, while the target domain data is collected under a load of 0 horsepower. The experimental results are shown in Table XI.

TABLE XI
RESULTS OF DIFFERENT TRANSFER EXPERIMENTS IN THE TARGET DOMAIN (%)

Phase	T_{20}	T_{21}	T_{23}	T_{30}	T_{31}	T_{32}
Incremental Accuracy (%)						
0	98.12 ± 0.00	97.50 ± 0.00	99.75 ± 0.00	97.50 ± 0.00	99.50 ± 0.00	99.62 ± 0.00
1	97.60 ± 0.10	98.30 ± 0.00	99.90 ± 0.10	97.20 ± 0.10	98.40 ± 0.25	99.80 ± 0.10
2	98.67 ± 0.17	98.42 ± 0.00	99.92 ± 0.00	98.08 ± 0.67	98.91 ± 0.27	99.92 ± 0.08
3	97.79 ± 0.43	94.29 ± 1.14	98.36 ± 0.36	95.86 ± 0.00	98.29 ± 0.44	98.71 ± 0.71
4	92.75 ± 0.19	93.88 ± 0.81	96.44 ± 0.69	92.44 ± 0.00	95.06 ± 0.41	97.87 ± 0.37
5	94.67 ± 0.44	96.11 ± 0.61	97.33 ± 0.00	93.39 ± 0.00	96.05 ± 0.21	97.56 ± 0.44
6	95.00 ± 0.25	96.10 ± 0.55	97.35 ± 0.10	94.10 ± 0.45	95.50 ± 0.21	98.50 ± 0.00
$\overline{Acc}$	96.08 ± 0.20	96.18 ± 0.25	98.22 ± 0.06	95.18 ± 0.02	97.04 ± 0.31	98.73 ± 0.10
Incremental Forgetting Rate (%)						
1	0.50 ± 0.12	-0.50 ± 0.00	-0.12 ± 0.12	0.00 ± 0.13	1.49 ± 0.18	-0.25 ± 0.00
2	-0.81 ± 0.12	-0.12 ± 0.00	-0.06 ± 0.00	-1.44 ± 1.19	0.75 ± 0.09	-0.37 ± 0.31
3	-0.25 ± 0.12	2.17 ± 0.79	0.13 ± 0.17	0.50 ± 0.42	0.62 ± 0.27	0.46 ± 0.58
4	1.28 ± 0.50	1.50 ± 0.34	0.81 ± 0.78	1.94 ± 0.44	1.56 ± 0.40	0.53 ± 0.28
5	-0.75 ± 0.35	0.42 ± 0.73	-0.02 ± 0.40	1.50 ± 0.22	0.32 ± 0.07	0.90 ± 0.13
6	-0.42 ± 0.08	0.46 ± 0.48	-0.17 ± 0.31	0.92 ± 0.19	0.70 ± 0.18	0.33 ± 0.12
$\overline{For}$	-0.07 ± 0.52	0.65 ± 0.39	0.09 ± 0.00	0.57 ± 0.02	0.91 ± 0.20	0.27 ± 0.20
Memory Rate (%)						
1	98.55 ± 0.01	99.40 ± 0.00	100.01 ± 0.01	98.60 ± 0.01	98.46 ± 0.04	100.03 ± 0.05
2	99.74 ± 0.02	99.27 ± 0.00	99.99 ± 0.00	99.76 ± 0.26	99.08 ± 0.09	100.15 ± 0.11
3	99.02 ± 0.15	96.06 ± 0.18	99.12 ± 0.10	97.68 ± 0.21	98.84 ± 0.00	99.13 ± 0.07
4	95.73 ± 0.16	96.19 ± 0.23	97.81 ± 0.05	95.25 ± 0.22	96.75 ± 0.02	98.67 ± 0.05
5	97.71 ± 0.05	97.84 ± 0.06	98.68 ± 0.20	95.94 ± 0.11	97.84 ± 0.17	98.33 ± 0.16
6	97.71 ± 0.08	97.82 ± 0.04	98.76 ± 0.11	96.59 ± 0.13	97.40 ± 0.02	99.08 ± 0.06
$\overline{MR}$	98.08 ± 0.09	97.76 ± 0.07	99.06 ± 0.03	97.30 ± 0.14	98.06 ± 0.06	99.23 ± 0.05

To provide a comprehensive evaluation of the model's performance and effectiveness, this study presents detailed experimental results from the source domain across the six transfer tasks mentioned above, as shown in Table XII. The figures show that UCILN performs exceptionally well across different transfer learning tasks. In the target domain, the average incremental accuracy remains above 95%, while in the source domain, it stays above 99%. This demonstrates the model's strong transfer learning capabilities, and the ability to mitigate forgetting in both domains.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/tim.2024.3500047 IEEE Transactions on Instrumentation and Measurement

TABLE XII
RESULTS OF DIFFERENT TRANSFER EXPERIMENTS IN THE SOURCE DOMAIN (%)

Phase	T_{20}	T_{21}	T_{23}	T_{30}	T_{31}	T_{32}
Incremental Accuracy (%)						
0	100.00 ± 0.00	100.00 ± 0.00	99.62 ± 0.00	100.00 ± 0.00	98.62 ± 0.00	99.87 ± 0.00
1	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.80 ± 0.00
2	100.00 ± 0.08	100.00 ± 0.00	99.92 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
3	100.00 ± 0.00	99.79 ± 0.21	100.00 ± 0.00	100.00 ± 0.07	99.86 ± 0.00	99.86 ± 0.02
4	99.56 ± 0.06	99.87 ± 0.12	99.69 ± 0.06	99.81 ± 0.00	99.31 ± 0.00	99.50 ± 0.03
5	99.83 ± 0.00	99.94 ± 0.06	99.83 ± 0.00	99.94 ± 0.06	99.83 ± 0.02	99.61 ± 0.01
6	99.85 ± 0.00	99.95 ± 0.00	99.90 ± 0.00	99.95 ± 0.05	99.80 ± 0.03	99.50 ± 0.02
$\overline{Acc}$	99.87 ± 0.02	99.93 ± 0.01	99.89 ± 0.01	99.95 ± 0.01	99.80 ± 0.02	99.71 ± 0.03
Incremental Forgetting Rate (%)						
1	0.00 ± 0.00	0.00 ± 0.00	-0.38 ± 0.00	0.00 ± 0.00	-1.38 ± 0.00	0.00 ± 0.00
2	0.00 ± 0.00	0.00 ± 0.00	-0.19 ± 0.00	0.00 ± 0.00	-0.69 ± 0.00	-0.31 ± 0.00
3	0.00 ± 0.17	0.00 ± 0.00	-0.29 ± 0.00	0.00 ± 0.00	-0.38 ± 0.00	-0.21 ± 0.00
4	0.16 ± 0.00	-0.37 ± 0.62	-0.19 ± 0.19	0.00 ± 0.10	0.00 ± 0.53	-0.03 ± 0.25
5	-0.27 ± 0.03	-0.40 ± 0.40	-0.43 ± 0.13	-0.20 ± 0.00	-0.72 ± 0.23	-0.13 ± 0.00
6	-0.23 ± 0.02	-0.33 ± 0.25	-0.44 ± 0.17	-0.17 ± 0.08	-0.60 ± 0.19	0.15 ± 0.08
$\overline{For}$	-0.06 ± 0.04	-0.18 ± 0.21	-0.32 ± 0.08	-0.06 ± 0.03	-0.63 ± 0.16	-0.09 ± 0.03
Memory Rate (%)						
1	100.00 ± 0.00	100.00 ± 0.00	100.19 ± 0.00	100.00 ± 0.00	100.69 ± 0.00	99.90 ± 0.00
2	100.00 ± 0.04	100.00 ± 0.00	100.05 ± 0.00	100.00 ± 0.00	100.34 ± 0.00	100.16 ± 0.00
3	100.00 ± 0.08	99.89 ± 0.11	100.15 ± 0.00	100.00 ± 0.04	100.12 ± 0.00	100.03 ± 0.04
4	99.70 ± 0.03	100.12 ± 0.25	99.94 ± 0.06	99.91 ± 0.00	99.66 ± 0.17	99.77 ± 0.09
5	100.05 ± 0.01	100.17 ± 0.17	100.13 ± 0.06	100.07 ± 0.02	100.28 ± 0.08	99.87 ± 0.03
6	100.04 ± 0.01	100.14 ± 0.12	100.17 ± 0.08	100.06 ± 0.02	100.20 ± 0.09	99.68 ± 0.01
$\overline{MR}$	99.97 ± 0.01	100.06 ± 0.10	100.10 ± 0.03	100.01 ± 0.01	100.21 ± 0.06	99.90 ± 0.00

TABLE XIII
SUMMARY OF FOUR SIGNAL AUGMENTATION TECHNIQUES

Augmentation	Description
Gaussian SNR noise	Add Gaussian noise to the signal by randomly choosing an SNR value from a range of 3 to 30 dB.
Mask	Mask a segment of the signal with a length selected randomly to obscure part of the data.
Shift	Shift the signal forward or backward by a randomly chosen number of steps to vary its temporal position.
Vertical flip	Vertically flip the signal to reverse its amplitude values.

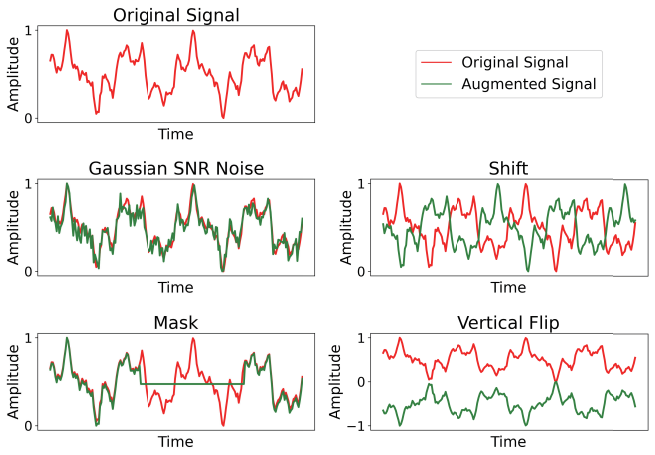

Fig. 11. Visualization of Four Augmentation Techniques on Bearing Fault Signals

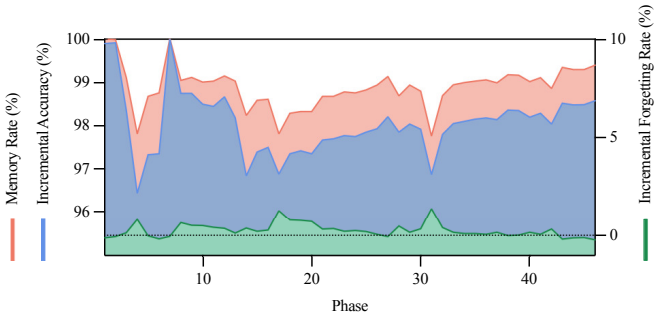

Fig. 12. Results for 46 Phases on the CWRU Dataset

F. Plasticity Evaluation

To assess the plasticity of UCILN, this experiment applies four signal augmentation techniques to perturb the input data. The model's plasticity and stability are evaluated by analyzing its performance under these perturbations, including incremental accuracy, incremental forgetting rate, and memory rate. In the study of [4], eight signal augmentation methods are proposed to expand the training data and improve the model's robustness. In this experiment, four of these methods are adopted, as shown in Table XIII and Figure 11.

This experiment is conducted on the 12 kHz drive-end fault data from the CWRU dataset, using data from the 2-horsepower condition as the source domain and data from the 3-horsepower condition as the target domain. The experiment begins with incremental training on the original signals as outlined in Table II, encompassing 6 phases. Subsequently, Gaussian noise, shift, masking, and vertical flip are sequentially applied to the original signals. After each augmentation, incremental training is performed across all 10 fault types. The results for all 46 phases are shown in Figure 12. After 46 incremental phases, UCILN achieves an average incremental accuracy of 98.06%, an average incremental forgetting rate of only 0.26%, and an average memory rate of 98.90%. These results demonstrate its ability to adapt to new data and maintain memory of old data.

IV. CONCLUSION

This paper proposes a novel unsupervised class incremental learning network (UCILN) to mitigate catastrophic forgetting in cross-domain fault diagnosis, particularly in situations where the target domain data lacks labels. UCILN processes both labeled source domain data and unlabeled target domain data in a streaming setting, which enables the model to timely diagnose new fault types in both domains. Besides, UCILN integrates a real feature replay mechanism for replaying data features to maintain a balance between the acquisition and retention of knowledge. A memory module is designed to encode and preserve data distribution in the deep feature space. In the absence of labels of target domain data, UCILN utilizes unsupervised deep transfer learning based on discrepancy to measure the differences between domains for domain adaptation. This study compares UCILN's performance with seven other class incremental learning models using the CWRU and PU datasets. Its performance is also evaluated across six transfer tasks, and its plasticity is tested under various perturbations. The experimental results highlight UCILN's exceptional accuracy and robustness in diagnosing faults. While UCILN exhibits promising outcomes, maintaining effective learning and diagnostic capabilities with a limited number of samples remains challenging. Future research should focus on few-shot class-incremental learning, especially given the limited labeled data for new fault classes in industrial settings.

REFERENCES

[1] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," *in Proc. Eur. Conf. Comput. Vis. (ECCV)*, pp. 532–547, 2018.

[2] X. Chen and X. Chang, "Dynamic residual classifier for class incremental learning," *in Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 18743-18752, 2023.

[3] X. Chen, R. Yang, Y. Xue, M. Huang, R. Ferrero, and Z. Wang, "Deep transfer learning for bearing fault diagnosis: A systematic review since 2016," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–21, 2023.

[4] X. Chen, R. Yang, Y. Xue, B. Song, and Z. Wang, "TFPred: Learning discriminative representations from unlabeled data for few-label rotating machinery fault diagnosis," *Control Eng. Pract.*, 146, 105900, 2024.

[5] Z. Chen, R. Yang, M. Huang, F. Li, G. Lu, and Z. Wang, "EEGProgress: A fast and lightweight progressive convolution architecture for EEG classification", *Comput. Biol. Med.*, 2024.

[6] Z. Chen, R. Yang, M. Huang, Z. Wang, and X. Liu, "Electrode domain adaptation network: Minimizing the difference across electrodes in single-source to single-target otor Imagery Classification", *IEEE Trans Emerg. Top. Comput. Intell.*, 2024.

[7] A. Dong, A. Starr and Y. Zhao, "Neural network-based parametric system identification: a review," *Int. J. Syst. Sci.*, vol. 54, no. 13, pp. 2676–2688, 2023.

[8] J. Dou and Y. Song, "An improved generative adversarial network with feature filtering for imbalanced data," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 4, art. no. 100017, Dec. 2023.

[9] S. Feng, X. Li, S. Zhang, Z. Jian, H. Duan and Z. Wang, "A review: state estimation based on hybrid models of Kalman filter and neural network," *Syst. Sci. Control Eng.*, vol. 11, no. 1, art. no. 2173682, 2023.

[10] J. Gama, R. Sebastio, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Mach. Learn.*, vol. 90, no. 3, pp. 317-346, 2013.

[11] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," *in Proc. Int. Conf. Learn. Represent.*, vol. 84, no. 12, pp. 1387–91, 2013.

[12] A. Gretton, D. Sejdinovic, H. Strathmann, et al., "Optimal kernel choice for large-scale two-sample tests," *Adv. Neural Inf. Process. Syst.*, vol. 25, 2012.

[13] T. L. Hayes, K. Kafle, R. Shrestha, M. Acharya, and C. Kanan, "REMIND your neural network to prevent catastrophic forgetting," *in Proc. Eur. Conf. Comput. Vis. (ECCV)*, pp. 466–483, 2020.

[14] J. He and F. Zhu, "Unsupervised continual learning via pseudo labels," *in International Workshop on Continual Semi-Supervised Learning*, pp. 15–32, 2021.

[15] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, 2011.

[16] J. Johnson, M. Douze, and H. Jegou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 03, pp. 535–547, 2021.

[17] C. Lessmeier, J.K. Kimotho, D. Zimmer and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification," *In Proc. Eur. Conf. Progn. Health Manag. Soc. (PHME16)*, vol. 3, no. 1, pp. 5-8, 2016.

[18] T. Li, Z. Zhao, C. Sun, R. Yan, and X. Chen, "Domain adversarial graph convolutional network for fault diagnosis under variable working conditions," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–10, 2021.

[19] W. Li and F. Yang, "Information fusion over network dynamics with unknown correlations: An overview," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 2, art. no. 100003, Jun. 2023.

[20] X. Li, M. Li, P. Yan, G. Li, Y. Jiang, H. Luo and S. Yin, "Deep learning attention mechanism in medical image analysis: Basics and beyonds," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 1, pp. 93–116, Mar. 2023.

[21] H. Lin, Y. Zhang, Z. Qiu, et al., "Prototype-guided continual adaptation for class-incremental unsupervised domain adaptation," *in Proc. Eur. Conf. Comput. Vis. (ECCV)*, pp. 351–368, 2022.

[22] Y. Liu, B. Schiele, and Q. Sun, "Adaptive aggregation networks for class-incremental learning," *in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 2544–2553, 2021.

[23] Z. Lu and G. Guo, "Control and communication scheduling co-design for networked control systems: a survey," *Int. J. Syst. Sci.*, vol. 54, no. 1, pp. 189–203, 2023.

[24] S. Ma and Y. Li, "Adaptive fuzzy fault-tolerant control for active seat suspension systems with full-state constraints," *Syst. Sci. Control Eng.*, vol. 11, no. 1, art. no. 2153391, 2023.

[25] N. Michel, M. Wang, L. Xiao, et al, "Rethinking momentum knowledge distillation in online continual learning," *arXiv preprint* arXiv:2309.02870, 2023.

[26] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural networks*, vol. 113, pp. 54–71, 2019.

[27] S. A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," *in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 2001–2010, 2017.

[28] J. Tan, Y. Li, Q. Xie and X. Wang, "Modeling and causality analysis of human sensorimotor control system based on NVAR method," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 4, art. no. 100014, Dec. 2023.

[29] T.J. Teyler, and J.W. Rudy, "The hippocampal indexing theory and episodic memory: updating the index," *Hippocampus*, vol. 17, no. 12, pp. 1158–1169, 2007.

[30] C. Wang, Z. Wang, W. Liu, Y. Shen, and H. Dong, "A novel deep offline-to-online transfer learning framework for pipeline leakage detection with small samples," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[31] Y.-A. Wang, B. Shen, L. Zou and Q.-L. Han, "A survey on recent advances in distributed filtering over sensor networks subject to communication constraints," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 2, art. no. 100007, Jun. 2023.

[32] Y. Wang, H.-J. Liu and H.-L. Tan, "An overview of filtering for sampled-data systems under communication constraints," *Int. J. Netw. Dyn. Intell.*, vol. 2, no. 3, art. no. 100011, Sep. 2023.

[33] Y. Xue, R. Yang, X. Chen, Z. Tian, and Z. Wang, "A novel local binary temporal convolutional neural network for bearing fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[34] S. Yan, J. Xie, and X. He, "DER: Dynamically expandable representation for class incremental learning," *in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 3014–3023, 2021.

[35] Z. Zhao, L. Xia, L. Jiang, Q. Ge and F. Yu, "Distributed bandit online optimisation for energy management in smart grids," *Int. J. Syst. Sci.*, vol. 54, no. 16, pp. 2957–2974, 2023.

[36] Z. Zhao, Q. Zhang, X. Yu, et al., "Applications of unsupervised deep transfer learning to intelligent fault diagnosis: A survey and comparative study," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–28, 2021.

[37] M. Zhong, X. Zhu, T. Xue and L. Zhang, "An overview of recent advances in model-based event-triggered fault detection and estimation," *Int. J. Syst. Sci.*, vol. 54, no. 4, pp. 929–943, 2023.

[38] D. W. Zhou, Q. W. Wang, H. J. Ye, and D. C. Zhan, "A model or 603 exemplars: Towards memory-efficient class-incremental learning," *arXiv preprint* arXiv:2205.13218, 2022.

[39] F. Zhu, X. Y. Zhang, and C. L. Liu, "Class incremental learning: A review and performance evaluation," *Acta Automatica Sinica*, vol. 49, p. 3, 2023.

[40] F. Zhu, X. Y. Zhang, C. Wang, F. Yin, and C. L. Liu, "Prototype augmentation and self-supervision for incremental learning," *in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 5871–5880, 2021.

[41] K. Zhu, W. Zhai, Y. Cao, J. Luo, and Z. J. Zha, "Self-sustaining representation expansion for non-exemplar class-incremental learning," *in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 9296–9305, 2022.

[42] M. Zinkevich, M. Weimer, A. J. Smola, and L. Li, "Parallelized stochastic gradient descent," *Adv. Neural Inf. Process. Syst.*, vol. 23, 2010.