CoUDA: Continual Unsupervised Domain Adaptation for Industrial Fault Diagnosis Under Dynamic Working Conditions

Bojian Chen, Xinmin Zhang, *Member, IEEE*, Changqing Shen, *Senior Member, IEEE*, Qi Li, *Graduate Student Member, IEEE*, and Zhihuan Song

***Abstract*—Unsupervised domain adaptation (UDA) has recently gained attention in fault diagnosis due to its ability to address domain shift problems arising from changes in working conditions. However, when faced with the continual domain shift problem inherent in real-world industries with dynamic working conditions, UDA often suffers from catastrophic forgetting. To address this challenge, we propose a novel replay-free continual UDA framework, CoUDA, for fault diagnosis under dynamic working conditions. In CoUDA, prototype contrastive learning is employed in source domain pre-training in order to improve the model generalization ability in preparation for the adaptation to the subsequent target domains. Then, source discriminator constraint is employed to ensure that the acquired source domain knowledge serves as an anchor, and source feature knowledge distillation is applied to prevent catastrophic forgetting without replay in sequential target domain adaptation. In addition, for better domain adaptation, local domain alignment and information entropy minimization are utilized to achieve fine-grained domain alignment. Experimental results demonstrate the superiority of the proposed CoUDA in achieving robust fault diagnosis under dynamic working conditions.**

***Index Terms*—Catastrophic forgetting, continual learning, fault diagnosis, unsupervised domain adaptation (UDA).**

I. Introduction

Intelligent fault diagnosis is crucial for ensuring the safety and reliability of industries [1]. With the advent of Industry 4.0, modern industries have accumulated vast amounts of data, and data-driven deep learning methods have been widely applied in intelligent fault diagnosis. [2], [3]. Typically, data-driven methods rely on historical data for offline modeling. However, when dealing with streaming data collected under complex and dynamic conditions in actual industries, they face serious challenges: the distribution shifts in both data and labels. Data distribution shift is the difference in distribution between the source domain of training and the target domain of testing, also known as the domain shift problem. Addressing the distribution shift problem requires the evolution and revolution of learning paradigms, from isolated learning to adaptive learning and ultimately to continual learning [4]. Currently, adaptive learning paradigms, such as unsupervised domain adaptation (UDA), have been widely applied to solve the domain shift problem [5]. The vanilla UDA methods are typically designed for a single target domain. However, in real industries with dynamic working conditions, the continuous influx of unlabeled fault data streaming brings the continual domain shift problem, which cannot be addressed by vanilla UDA methods.

Continual learning is an advanced learning paradigm, and the key challenge of which is to address the catastrophic forgetting [6]. An inherent problem in deep learning is catastrophic forgetting, a phenomenon in which a deep neural network model forgets the knowledge it learned from previous tasks when learning a new task. In the field of fault diagnosis, there have been some studies on continual learning for fault diagnosis [7], [8], [9]. However, these studies mainly focus on the addition of fault types, that is, class-incremental learning, without considering continual domain shift problems.

In recent years, continual UDA that integrates continual learning and UDA has gained attention by addressing the continual domain shift problems [10], [11], [12]. As illustrated in Fig. 1, the vanilla UDA model is trained on the source and target domains to minimize the domain discrepancy, and tested on the target domain. Compared with the vanilla UDA, the continual UDA model is pretrained on the source domain, then sequentially adapted to the target domains, and finally tested on all seen domains. The continual UDA enables the model

Received 25 November 2024; revised 26 December 2024; accepted 14 January 2025. Date of publication 20 February 2025; date of current version 21 April 2025. This work was supported in part by the Key Research and Development Program of Ningbo under Grant 2024T011, in part by the National Key Research and Development Program of China under Grant 2022ZD0120003, in part by the National Natural Science Foundation of China under Grant 62473103, and in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions under Grant 22KJD460006. Paper no. TII-24-6291. *(Corresponding author: Xinmin Zhang.)*

Bojian Chen, Xinmin Zhang, and Zhihuan Song are with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: bjchen@zju.edu.cn; xinminzhang@zju.edu.cn; songzhihuan@zju.edu.cn).

Changqing Shen is with the School of Rail Transportation, Soochow University, Suzhou 215131, China (e-mail: cqshen@suda.edu.cn).

Qi Li is with the State Key Laboratory of Tribology in Advanced Equipment, Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China (e-mail: liq22@tsinghua.org.cn).

This article has supplementary downloadable material available at http://doi.org/10.1109/JBHI.2021.3083187, provided by the authors.

Digital Object Identifier 10.1109/TII.2025.3538135

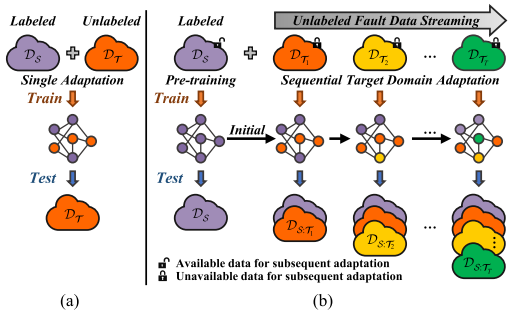

Fig. 1. Comparison of vanilla UDA and continual UDA. (a) Vanilla UDA: The model is trained on the source and target domains to minimize the domain discrepancy, and tested on the target domain. (b) Continual UDA: The model is pretrained on the source domain, then adapted to the target domains sequentially, and finally tested on all seen domains. Remind that the source domain data are always available.

to effectively adapt to a series of target domains without catastrophically forgetting previously learned knowledge. In the field of industrial fault diagnosis, the dynamic working conditions also lead to the continual domain shift problem. Recently, Chen et al. [13] introduced continual UDA into the field of fault diagnosis, attempting to address catastrophic forgetting during the adaptation process to successive target domains. However, subsequent related research is quite limited [14], [15]. These methods are based on a replay mechanism [16], which stores an amount of data from the historical domain and then uses the replay of the historical domain data to address the catastrophic forgetting problem in the process of adapting to the current target domain. Unfortunately, the replay mechanism does not strictly uphold the privacy of the historical data, and thus may produce more serious data privacy issues. Since the instances with reliable labels from the source domain are usually industrial fault data from laboratory simulations, while the instances from the target domain are usually from real industrial systems, there are strong data privacy issues. Thus, domain adaptation must be achieved while upholding the privacy of historical data and preserving knowledge acquired from all previous domains. Consequently, an ideal continual UDA method for fault diagnosis should address the following challenges:

1) Better domain adaptation: The model can effectively adapt to a new domain even if the domain shift is large because of dynamic working conditions;
2) Less catastrophic forgetting: The model should adapt to the current target domain without forgetting the knowledge acquired from previous domains;
3) Data privacy constraints: The model should not access the historical target domain data when adapting to the current target domain.

To address the above challenges, this work proposes a novel continual UDA framework, CoUDA, for industrial fault diagnosis under dynamic working conditions. The CoUDA framework strictly adheres to data privacy constraints, achieves better domain adaptation, and addresses catastrophic forgetting from the perspective of metric and representation learning. CoUDA mainly consists of two stages: source domain pretraining and sequential target domain adaptation. In the source domain pretraining stage, a natural idea is that the generalization of the model should be improved to prepare for the adaptation to the subsequent target domains. A contrastive representation learning strategy, prototype contrastive learning (PCL), is employed to improve the generalization performance of the source domain model.

In the sequential target domain adaptation, to prevent catastrophic forgetting without replay, the source discriminator constraint (SDC) is employed to freeze the classifier of source domain pretraining to constrain source discriminator performance, ensuring that the acquired source domain knowledge serves as an anchor. Then, the source features knowledge distillation (FKD) is applied to maintain the distribution information of the invisible historical domains under the premise of strictly adhering to data privacy constraints. Further, for better domain adaptation, two distance metric methods, namely, local domain alignment (LDA) and information entropy minimization (IEM), are leveraged to achieve fine-grained domain alignment between source and target domains. The source domain plays both the role of a proxy for overcoming the forgetting of historical domains and an anchor point for domain adaptation. The main contributions of this work can be summarized as follows.

1) A novel continual UDA framework, CoUDA, is designed to address the continual domain shift problem for industrial fault diagnosis under dynamic working conditions;
2) The generalization performance of the source domain pretraining is considered for the first time in continual UDA, and a contrastive representation learning strategy PCL is employed in CoUDA to improve it to prepare for subsequent adaptation;
3) CoUDA strictly adheres to data privacy constraints and prevents catastrophic forgetting without replay by SDC and FKD in the sequential target domain adaptation;
4) Efficient and stable domain adaptation is achieved in CoUDA by two distance metric methods, LDA and IEM, which align fine-grained class distribution between source and target domains;
5) Through well-designed experiments, the superiority of CoUDA in achieving robust fault diagnosis under dynamic working conditions is validated.

The rest of this article is organized as follows. Section II reviews the related studies. Section III presents the details of the proposed CoUDA. Section IV provides the case studies and experimental results analysis. Finally, Section V concludes this article.

II. Related Work

A. Unsupervised Domain Adaptation

In the field of fault diagnosis, UDA has been widely used to address domain shift problems. The commonly used UDA methods include domain adversarial neural networks (DANN) [17], maximum mean discrepancy (MMD) [18], correlation alignment (CORAL) [19], and local MMD (LMMD) [20]. These vanilla UDA methods are typically designed for a single target domain. In real industries with dynamic working conditions, these methods encounter limitations when dealing with a sequence of

TABLE I
COMPARISON OF PROBLEM SETTINGS BETWEEN COUDA AND RELATED PARADIGMS

Settings	Data		Dynamic	Historical
	Source	Target	$(t>1)$	Accuracy
Fine-tuning	✗	$x^{\mathcal{T}_t}, y^{\mathcal{T}_t}$	✗	✗
UDA	$x^{\mathcal{S}}, y^{\mathcal{S}}$	$x^{\mathcal{T}_t}$	✗	✗
Continual Learning	✗	$x^{\mathcal{T}_t}, y^{\mathcal{T}_t}$	✓	✓
CoUDA	$x^{\mathcal{S}}, y^{\mathcal{S}}$	$x^{\mathcal{T}_t}$	✓	✓

target domains, both in terms of large domain discrepancy and catastrophic forgetting. These limitations underscore the need for novel methods to facilitate adaptation to sequential dynamic domains while preserving knowledge from previous domains.

B. Metric and Representation Learning

Metric and representation learning methods improve the performance of the model by constraining the similarity of instances in the feature space to learn good mappings. The common approach for constraining similarity is information-noise contrastive estimation (infoNCE) [21], which also refers to contrastive learning. Notably, methods such as MMD and CORAL are also forms of metric learning, which reduce domain discrepancy by minimizing the distance between source and target domains.

C. Continual Unsupervised Domain Adaptation

Continual UDA is an emerging research direction that aims to address the continual domain shift problem as compared with other paradigms in Table I. Existing methods are mainly categorized into two types: replay-based methods and regularization-based methods.

1) Replay-Based Methods: The replay mechanism has proven instrumental in addressing catastrophic forgetting when adapting to new domains [10]. It is remarkably similar to the way humans quickly recall memories via tips, and is achieved by storing a certain amount of data from historical domains [15] or generating synthetic data by generative models [22].

2) Regularization-Based Methods: Regularization-based methods try to constrain the updating of the model parameters to prevent catastrophic forgetting by adding regularization terms to the loss function. The most commonly used regularization method in continual UDA is knowledge distillation, which constrains the output of the new model to be as consistent as possible with the old model [12]. In addition, knowledge distillation is often combined with replay to constrain the old and new models to be as consistent as possible in feature extraction or output on historical data to further prevent forgetting [11], [14].

In the field of fault diagnosis, early attempts at continual UDA have been made in [13], but subsequent related research is scarce. Recently, Li et al. [14] proposed dynamic weight aggregation to simultaneously address supervised class-incremental and domain-incremental problems in rotating machinery. Ragab et al. [15] designed the EverAdapt framework to achieve CoUDA for dynamic scenarios. These methods all rely on storing historical domain data for replay, which may lead to data privacy issues. In contrast, we adhere to stricter privacy constraints and propose a novel CoUDA framework without replay, namely, CoUDA, to achieve fault diagnosis under dynamic working conditions.

III. METHODOLOGY

A. Problem Formulation and Notations

The definition of the continual domain shift problem in industrial fault diagnosis is first introduced. We follow the standard settings of CoUDA [10], as illustrated in Fig. 1 and Table I. The model is pretrained on the full-labeled source domain $\mathcal{S}$ and then adapted to the T unlabeled target domains $\{\mathcal{T}_t\}_{t=1}^{T}$ sequentially. After the adaptation, the model is tested on all seen domains. For convenience, let $\mathcal{D}_{\mathcal{S}} = \{(x_i^{\mathcal{S}}, y_i^{\mathcal{S}})\}_{i=1}^{N_{\mathcal{S}}}$ denote the source domain with $N_{\mathcal{S}}$ labeled instances, where $y_i^{\mathcal{S}} \in \{0, 1, \ldots, C-1\}$ represents the source domain that has instances with C kinds of different labels, and $x^{\mathcal{S}}$ follows a distribution $\mathcal{P}(x^{\mathcal{S}})$. Similarly, on the tth target domain, there is an unlabeled dataset $\mathcal{D}_{\mathcal{T}_t} = \{(x_j^{\mathcal{T}_t})\}_{j=1}^{N_{\mathcal{T}_t}}$ with $N_{\mathcal{T}_t}$ instances, which shares the same label space with the source domain. Due to the dynamic working conditions, the data distribution between the source and target domains is different, i.e., $\mathcal{P}(x^{\mathcal{T}_t}) \neq \mathcal{P}(x^{\mathcal{S}})$, and the data distribution between target domains is also different, i.e., $\mathcal{P}(x^{\mathcal{T}_t}) \neq \mathcal{P}(x^{\mathcal{T}_{t'}})$ $(t \neq t')$.

To deal with the continual domain shift problem in industrial fault diagnosis, this study adopts a model $\Theta(x) = g(f(x))$ with parameter θ, which consists of two parts: a feature extractor $f_\theta : x \to \mathbb{R}^m$ and a classifier $g_\theta : \mathbb{R}^m \to \mathbb{R}^C$, where m is the output dimension of the feature extractor. Remind that the feature map extracted by the feature extractor f_θ is denoted as $h = f_\theta(x)$, and h can be normalized to a unit vector in a unit hypersphere feature space $\mathcal{F}^m$, which is by L2 normalization, i.e., $\|h\|_2 = 1$. The classifier g_θ is a cosine similarity classifier, which is composed by C prototypes $P_{g_\theta} = \{p_c\}_{c=0}^{C-1}$ and $\|p_c\|_2 = 1$. The unit hypersphere feature space $\mathcal{F}^m$ is divided into C regions by prototypes P_{g_θ}, and the prediction of the model $z_c = h \cdot p_c$ is the cosine similarity between the feature map h and the prototype p_c. Therefore, the feature distribution $\mathcal{P}(h)$ in the unit hypersphere feature space is determined by f_θ, and the classification score is determined by the prototypes P_{g_θ} of the classifier g_θ.

In this study, we proposed a novel continual UDA framework, CoUDA, to address the continual domain shift problem in industrial fault diagnosis under dynamic working conditions. The goal of CoUDA is to train the model not only to adapt to each target domain but also to prevent catastrophic forgetting of the knowledge acquired from the previous domains while observing data privacy.

B. Overview of the Proposed CoUDA

The overview of the proposed CoUDA for industrial fault diagnosis under dynamic working conditions is shown in Fig. 2. CoUDA mainly consists of two stages: source domain pretraining and sequence target domain adaptation. In the source domain pretraining stage (0th stage), a natural idea is that the generalization of the model should be improved to prepare for

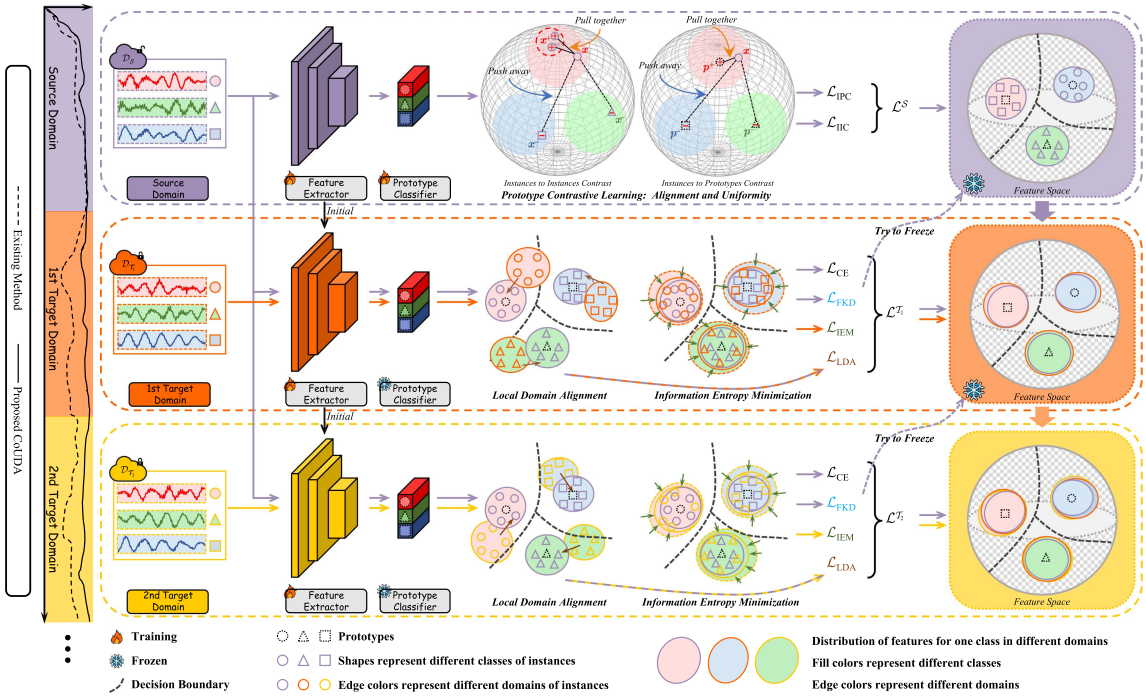

Fig. 2. Overview of the proposed CoUDA. CoUDA consists of two stages: Source domain pretraining and sequence target domain adaptation. In the source domain pretraining stage, the model is trained on the source domain by PCL to improve the generalization ability. In the sequence target domain adaptation stage, the model is adapted to the sequence of target domains by LDA and IEM, and prevents catastrophic forgetting by SDC and FKD without replaying historical data.

the adaptation to the subsequent target domains. We employ a contrastive representation learning strategy, PCL, to improve the generalization performance of the source domain model. In sequential target domain adaptation, to prevent catastrophic forgetting without replay, we first employ SDC to freeze the classifier of source domain pretraining to constrain source discriminator performance, ensuring that the acquired source domain knowledge serves as an anchor. Then, source FKD is applied to maintain the distribution information of the invisible historical domains under the premise of strictly adhering to data privacy constraints. Further, for better domain adaptation, we employ distance metric methods, LDA and IEM, to achieve fine-grained domain alignment between source and target domains. The source domain serves as a constant anchor in the whole process, drawing all target domains toward it to ensure that knowledge is not lost during the sequence target domain adaptation process.

C. Source Domain Pretraining

Source domain pretraining is a crucial step in the CoUDA framework. The goal of this stage is to improve the generalization of the source domain model to prepare for the adaptation to the subsequent target domains. This stage can be considered as a single domain generalization problem [23].

C. Prototype Contrastive Learning: Inspired by [24], a contrastive representation learning, PCL, is employed during source domain pretraining to improve the generalization performance of the source domain model. As illustrated in Fig. 3, the principle of PCL can be visualized in a unit hypersphere feature space $\mathcal{F}^m$. PCL consists of two parts: instance-to-instance contrast (IIC) and instance-to-prototype contrast (IPC). IIC is achieved by supervised infoNCE loss [21] to encourage the instances from the same class to be close to each other, while IPC is achieved by cross-entropy (CE) loss to pull the instances toward the prototype with the same class. For x_i with label y_i, IIC and IPC are calculated as follows:

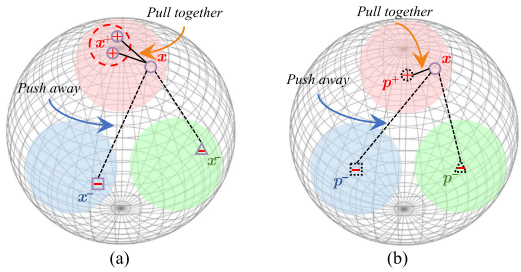

Fig. 3. Prototype contrastive learning. (a) Instance-to-instance contrast. (b) Instance-to-prototype contrast.

$$\begin{aligned}\mathrm{IIC}(x_i) &= \\ &-\log\frac{\exp\left(\frac{f(x_i)^\top f(x^+)}{\tau}\right)}{\exp\left(\frac{f(x_i)^\top f(x^+)}{\tau}\right)+\sum_{y^-\neq y_i}\exp\left(\frac{f(x_i)^\top f(x^-)}{\tau}\right)} \\ &\approx \underbrace{-\frac{f(x_i)^\top f(x^+)}{\tau}}_{\text{instance alignment}} + \underbrace{\log\sum_{y^-\neq y_i}\exp\left(\frac{f(x_i)^\top f(x^-)}{\tau}\right)}_{\text{instance uniformity}} \end{aligned} \tag{1}$$

$$\begin{aligned}\mathrm{IPC}(x_i) &= -\log \frac{\exp\left(f(x_i)^\top p_{y_i}^+\right)}{\exp\left(f(x_i)^\top p_{y_i}^+\right) + \sum_{c\neq y_i}\exp\left(f(x_i)^\top p_c^-\right)} \\ &\approx \underbrace{-f(x_i)^\top p_{y_i}^+}_{\text{prototypical alignment}} + \underbrace{\log \sum_{c\neq y_i}\exp\left(f(x_i)^\top p_c^-\right)}_{\text{prototypical uniformity}} \end{aligned} \tag{2}$$

where x^+ is a positive instance with label y_i, x^- is a negative instance with label $y^- \neq y_i$, $p_{y_i}^+$ is a positive prototype belonging to class y_i, and p_c^- is a negative prototype belonging to class $c \neq y_i$, respectively. To simplify formulas and layout beautifully, we use f to represent the feature extractor f_{θ_S} in the source domain pretraining. τ is a temperature parameter to determine how much attention the contrast loss pays to difficult negative instances. As discussed in [24] and [25], assuming that the features in the unit hypersphere feature space follow a mixed von Mises–Fisher (vMF) distribution and the number of instances are large enough, IIC and IPC can be approximately decomposed into two parts: alignment and uniformity.

The whole loss function of the source domain pretraining is defined as follows:

$$\begin{aligned}\mathcal{L}^{\mathcal{S}} &= \mathcal{L}_{\mathrm{IPC}} + \alpha \cdot \mathcal{L}_{\mathrm{IIC}} \\ &= \mathop{\mathbb{E}}_{x_i\in\mathcal{D}_{\mathcal{S}}}\left\{\sum_{i=1}^{N_{\mathcal{S}}}\left[\mathrm{IPC}(x_i) + \alpha\cdot\sum_{y^+=y_i}\mathrm{IIC}(x_i)\right]\right\}\end{aligned} \tag{3}$$

where α is a hyperparameter to balance the two parts of the loss function.

Each element in IIC and IPC is a dot product of two vectors, which is a cosine similarity. In the feature space $\mathcal{F}^m$, IIC is used to align the instances in the same class and to homogenize instances in different categories, which means that clusters of each class are compact and separated from each other uniformly. The alignment and uniformity of instances on the feature space can help the model to learn domain-invariant representations, which can improve the generalization performance of the source domain model. IPC is used to align the instances to the prototypes of the same class and to homogenize instances to the prototypes of different classes, which means that each instance is close to the prototype of its own class and far from the prototypes of other classes. The data features from the cth class should gather around the cth prototype. Therefore, the prototype p_c can be treated as an anchor of the cth class, which contains domain-invariant representations in the cth class. In CoUDA, these reliable domain-invariant representations can be preserved and serve as an anchor to prevent catastrophic forgetting.

D. Prevent Catastrophic Forgetting Without Replay

CoUDA is designed to continually adapt to new working conditions without replay. Replay, which tries to use instances from previous target domains to ensure that the features and results of the old model are as close as possible to those of the currently trained model, has proven instrumental in preventing catastrophic forgetting in continual learning [10]. Obviously, the extent to which replay-based methods mitigate catastrophic forgetting depends on the number of instances retained. However, retaining instances for replay brings potential risks, including data privacy leakage and the escalation of storage costs.

We believe that replay is not necessary for continual UDA. It is perfectly possible to use always-accessible source data instead of replay samples, as long as certain conditions are met. To achieve continual UDA without replay, we first discuss the importance of maintaining the discriminative ability of source features. As mentioned in [11], the discriminative ability of source features should be maintained by screening source data without increasing the classification loss on the source domain to improve the adaptation to the target domains. However, Tang et al. [11] ignored the important point that a reduction of discriminative ability in the source domain can result in the loss of discriminative ability in the learned target domains, since adaptation is achieved by the distribution of features between source and target domains being aligned, i.e., $P(h_t^{\mathcal{S}}) \approx P(h_t^{\mathcal{T}_t})$, where $h_t^{\mathcal{S}}$ and $h_t^{\mathcal{T}_t}$ are the features of domains $\mathcal{S}$ and $\mathcal{T}_t$ in the tth stage. Therefore, if $P(h_t^{\mathcal{S}}) \not\approx P(h_{t-1}^{\mathcal{S}})$, it may lead to a chain reaction of the learned domain $\mathcal{T}_{1:t-1}$ forgetting, which is particularly evident in the adaptation with long sequences and large changes in distribution.

1) Source Discriminator Constraint: Inspired by source hypothesis transfer [26], the source classifier (hypothesis) encodes the domain-invariant representations in the source domain pretraining. The fine-tuning of feature extractor by $\mathcal{D}_{\mathcal{S}}$ when freezing source classifier, i.e., $g_{\theta_t} = g_{\theta_S}$, will produce features with a similar distribution as in the source domain pretraining. Therefore, the additional computation used to screen the instances that do not harm source domain discriminative ability in [11] is not needed. The CE loss is utilized for fine-tuning of feature extractor by $\mathcal{D}_{\mathcal{S}}$, which is defined as follows:

$$\mathcal{L}_{\mathrm{CE}} = \mathop{\mathbb{E}}_{x_i,y_i\in\mathcal{D}_{\mathcal{S}}}\sum_{c=0}^{C-1} -\left[\widetilde{y}_{i,c}\log\sigma(f_{\theta_t}(x_i)^\top p_c)\right] \tag{4}$$

where $\widetilde{y}_{i,c}$ is the cth element of the ith instance one-hot label in the source domain, and $f_{\theta_t}(x_i)$ is the feature map of the source domain data x_i extracted by the model in tth stage.

By SDC, the source domain serves as a constant anchor, i.e., $P(h_0^{\mathcal{S}}) \approx P(h_1^{\mathcal{S}}) \approx \cdots \approx P(h_{t-1}^{\mathcal{S}}) \approx P(h_t^{\mathcal{S}})$, preserving reliable domain-invariant representations from the source domain pretraining and implicitly reducing the forgetting of historical domains. However, SDC does not strongly constrain the unforgettable updating of model parameters, and the old model parameters are not fully utilized. Ideally, we assume that the adaptation of each target domain is successful, i.e., $P(h_t^{\mathcal{S}}) \approx P(h_t^{\mathcal{T}_t})$. Under this assumption, we can further use the source domain features as a proxy to replace replay data to explicitly reduce the forgetting of historical domains by knowledge distillation.

2) Feature Knowledge Distillation: Knowledge distillation is often used in continual learning to prevent catastrophic forgetting by constraining the output of the current model (student) to be as consistent as possible with the old model (teacher). In CoUDA, FKD is employed to maintain the distribution information of the invisible historical domains under the premise of strictly adhering to data privacy constraints, which can be

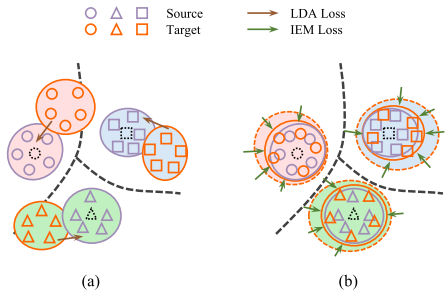

Fig. 4. Efficient and stable adaptation. (a) LDA. (b) IEM.

defined as follows:

$$\mathcal{L}_{\text{FKD}} = 1 - \mathop{\mathbb{E}}_{x_i \in \mathcal{D}_S} \left[f_{\theta_t}(x_i)^\top f_{\theta_{t-1}}(x_i) \right] \tag{5}$$

where $f_{\theta_{t-1}(x_i)}$ is the feature map of the source domain data x_i extracted by the old model from $(t-1)$th stage. FKD is achieved by measuring and maximizing the cosine similarity $f_{\theta_t}(x_i)^\top f_{\theta_{t-1}}(x_i)$ between the source domain features of the current model and the old model, thus ensuring that the model does not forget the knowledge of the old domain when adapting to the new target domain. To ensure the effectiveness of FKD by meeting the assumption that the adaptation of each target domain is successful, effective and stable adaptation to target domains should be achieved.

E. Efficient and Stable Adaptation

One of the key tasks in CoUDA is to minimize the domain distribution discrepancy between the source and target domains. The commonly utilized distance metrics, such as CORAL and MMD, primarily concentrate on the average distribution distance between domains. While these metrics are effective to a certain extent, they often neglect the fine-grained class distribution within individual domains, which may hinder their ability to fully address the intricacies of continual UDA. Such an oversight may result in the misalignment of similar classes across domains, negatively impacting the adaptation performance of the model.

1) Local Domain Alignment: To address this challenge, as illustrated in Fig. 4(a), we employ the distance metric method, LDA, to focus on aligning the class distribution between domains more granularly. The LDA loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{LDA}} &= \frac{1}{C}\sum_{c=0}^{C-1} d\left(\mathop{\mathbb{E}}_{\substack{x_i \in \mathcal{D}_S \\ y_i = c}} \left[f_{\theta_t}(x_i) \right], \mathop{\mathbb{E}}_{\substack{x_j \in \mathcal{D}_{\mathcal{T}_t} \\ \widehat{y}_j = c}} \left[f_{\theta_t}(x_j) \right] \right) \\ &= \frac{1}{C}\sum_{c=0}^{C-1} \left\| \mathop{\mathbb{E}}_{\substack{x_i \in \mathcal{D}_S \\ y_i = c}} \left[\phi(f_{\theta_t}(x_i)) \right] - \mathop{\mathbb{E}}_{\substack{x_j \in \mathcal{D}_{\mathcal{T}_t} \\ \widehat{y}_j = c}} \left[\phi(f_{\theta_t}(x_j)) \right] \right\|_{\mathcal{H}}^2 \end{aligned} \tag{6}$$

where $\widehat{y}_j = \arg\max_c [f_{\theta_t}(x_j)^\top p_c]$ is the pseudolabel of the target domain data x_j. The distance metric $d(,)$ is MMD. ϕ is a mapping function to a reproducing kernel Hilbert space $\mathcal{H}$, and $\|\cdot\|_{\mathcal{H}}$ is the norm in $\mathcal{H}$. Once the pseudolabels have been assigned, the distribution of the source and target domain features can be aligned for each class by LDA. LDA is used to align fine-grained class distribution between domains, which is more effective than global domain alignment methods but less stable because it is sensitive to pseudolabels.

The alignment of the class distribution between domains is difficult to achieve due to the complex distribution of the data. The uncertainty of pseudolabels may lead to changes in the distribution of class features, including changes in the number and location of features on the feature space. The failure alignment of one class may lead to the misalignment of other classes, which may consequently affect the stability of LDA and hinder the adaptation performance of the model.

2) Information Entropy Minimization: To ensure the stability of the adaptation, we employ IEM to minimize the information entropy of classification probability. The information entropy can measure the uncertainty of classification, and the lower the entropy the lower the uncertainty. IEM is defined as follows:

$$\mathcal{L}_{\text{IEM}} = \mathop{\mathbb{E}}_{x_j \in \mathcal{D}_{\mathcal{T}_t}} \sum_{c=0}^{C-1} - \left[\sigma(f_{\theta_t}(x_j)^\top p_c) \log \sigma(f_{\theta_t}(x_j)^\top p_c) \right] \tag{7}$$

where σ is the softmax function. As shown in Fig. 4(b), in the feature space, IEM narrows down the target domain features to the corresponding class prototypes to ensure the closeness of the cluster and avoid training fluctuations or even collapse caused by uncertainty.

F. Sequential Target Domain Adaptation

CoUDA optimizes multiple objectives to adapt to the sequence of target domains without forgetting the knowledge acquired from the previous domains. These objectives include maintaining the discriminative ability of the source features ($\mathcal{L}_{\text{CE}}$) and preserving the distribution information of the invisible historical domains ($\mathcal{L}_{\text{FKD}}$) to prevent catastrophic forgetting, aligning the class distribution between domains ($\mathcal{L}_{\text{LDA}}$) to achieve fine-grained domain alignment, and minimizing the information entropy of classification probability ($\mathcal{L}_{\text{IEM}}$) to ensure the stability of the adaptation. The whole loss function of the tth target domain adaptation in the CoUDA framework is defined as follows:

$$\mathcal{L}^{\mathcal{T}_t} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{FKD}} + \beta(e) \cdot \mathcal{L}_{\text{IEM}} + (1 - \beta(e)) \cdot \mathcal{L}_{\text{LDA}} \tag{8}$$

where $\beta(e) = \frac{e}{E}$ is a tradeoff to balance $\mathcal{L}_{\text{LDA}}$ and $\mathcal{L}_{\text{IEM}}$, and e is the current epoch in a total number of E epochs. As training progresses, we gradually shift the focus of training from $\mathcal{L}_{\text{LDA}}$ to $\mathcal{L}_{\text{IEM}}$ by $\beta(e)$. The reason for this design is that in the early stage of training, a larger $\mathcal{L}_{\text{LDA}}$ weight can help achieve the initial alignment of feature distributions, while in the later stage of training, a larger $\mathcal{L}_{\text{IEM}}$ weight is needed to reduce the classification uncertainty of some difficult samples and thus ensure the stability of training.

The algorithm of the CoUDA framework is summarized in Supplementary Material.

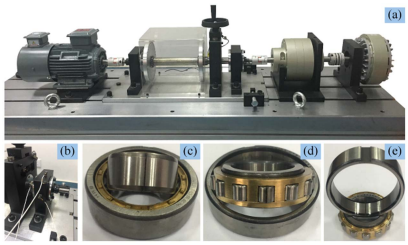

Fig. 5. SDUST dataset. (a) Experimental setup. (b) Acceleration sensor. (c) Inner race fault. (d) Roller fault. (e) Outer race fault.

TABLE II
SDUST DATASET SIGNAL DESCRIPTION

Bearing	Speed and load range	Location	Damage degree	Class
N205EU	0-60 N 500-3000 rpm/min	N/A	No Damage	0
		Inner	0.2mm	1
		Roller	0.2mm	2
		Outer	0.2mm	3
		Inner	0.4mm	4
		Roller	0.4mm	5
		Outer	0.4mm	6
		Inner	0.6mm	7
		Roller	0.6mm	8
		Outer	0.6mm	9

IV. EXPERIMENTS

A. Datasets Description

The Shandong University of Science and Technology (SDUST) dataset [27] with various working conditions is adopted to verify the superiority of the proposed CoUDA framework. The experimental setup, as depicted in Fig. 5(a),

consists of a gearbox, a motor, three shaft couplings, two rotors, two bearing seats, and a brake. The vibration acceleration signals are collected by three vibration sensors set in three directions of the test bearing seat, as illustrated in Fig. 5(b). The bearing type is N205EU, and the faults simulated include inner race fault, roller fault, and outer race fault, which are shown in Fig. 5(c)–(e). The SDUST dataset contains ten classes of bearing faults, each type of fault has three different damage dimensions. The detailed data information is shown in Table II . The data sampling frequency is 25.6 kHz. The rotating speed ranges from 500 to 3000 r/min, i.e., 500, 1000, 1500, 2000, 2500, and 3000 r/min. The load ranges from 0 to 60 N, i.e., 0, 20, 40, and 60 N. Therefore, there are 24 working conditions in total. We employed a moving window technique with a fixed window size of 1024 and a stride length of 1024 to segment the data, ensuring that the resulting segments were distinct and nonoverlapping, which is critical for model training.

B. Domain Scenarios Protocol

To simulate the CoUDA scenario, we have selected six distinct working conditions from the 24 working conditions in the SDUST dataset. As we all know, the speed change is more challenging than the load change in the UDA fault diagnosis task. To increase the challenge of the model evaluation and to demonstrate the superiority of our proposed method, the rotating speeds of the selected working conditions are always different, and their loads change alternatively, as shown in Table III.

TABLE III
SIX WORKING CONDITIONS OF SDUST DATASETS

Working condition ID	W1	W2	W3	W4	W5	W6
Load (N)	20	0	20	40	60	40
Rotating speed (rpm)	500	1000	1500	2000	2500	3000

TABLE IV
SIX EXPERIMENT SCENARIOS OF SDUST DATASETS

Scenario	$\mathcal{D}_S$	$\mathcal{D}_{\mathcal{T}_1}$	$\mathcal{D}_{\mathcal{T}_2}$	$\mathcal{D}_{\mathcal{T}_3}$	$\mathcal{D}_{\mathcal{T}_4}$	$\mathcal{D}_{\mathcal{T}_5}$
S1	W1	W2	W3	W4	W5	W6
S2	W2	W1	W3	W4	W5	W6
S3	W1	W3	W2	W4	W5	W6
S4	W1	W2	W4	W3	W5	W6
S5	W1	W2	W3	W5	W4	W6
S6	W1	W2	W3	W4	W6	W5

* Underlined fonts represent the order of exchange.

The selected six working conditions are alternately ordered to obtain six scenarios, as shown in Table IV. This approach enhances the reliability of the results by preventing any bias toward specific scenarios that might favor certain methods.

C. Evaluation Metrics

We assume that each domain corresponds to a single training phase, and the model is tested on all domains after training at each phase. In these settings, an accuracy matrix can be obtained to evaluate the model performance on all domains. The accuracy matrix is represented as $\mathbf{R} \in \mathbb{R}^{T+1}$, where $R_{i,j}$ denotes the accuracy of the model trained on the ith domain and tested on the jth domain. The existing evaluation metrics in [11] are named as average accuracy (ACC) $\text{ACC} = \frac{1}{T}\sum_{t=1}^{T} R_{T,t}$ and average backward transfer (BWT) $\text{BWT} = \frac{1}{T-1}\sum_{t=1}^{T-1} R_{T,t} - R_{t,t}$, which are used to evaluate the model performance in preventing catastrophic forgetting. We also introduce the average adaptation (AAD) $\text{AAD} = \frac{1}{T}\sum_{t=1}^{T} R_{t,t}$ to evaluate the adaptation ability of the model. Remind that these three metrics are not considered the source domain performance, because the source domain is always available for retraining the model.

D. Comparison Methods

To evaluate the performance of the proposed CoUDA, it is compared with recent domain adaptation methods proposed for fault diagnosis, image classification, and other tasks. We have reimplemented all the baselines within our framework, ensuring equivalence in terms of the backbone network and training protocols. Overall, the compared baseline methods used in the experiments can be divided into two categories: vanilla UDA and continual UDA.

1) Vanilla UDA: The vanilla UDA methods are designed to adapt the model to a single target domain. The methods used in the experiments include the following:

a) DANN [17] leverages the gradient reversal layer to adversarially train a domain discriminator network against an encoder network;
b) Improved DANN (IDANN) [28] combines multifeature fusion scheme with DANN;
c) Hierarchical deep domain adaptation (HDDA) [19] leverages CORAL to align the second-order statistics of the source and target distributions in order to effectively minimize the shift between the two domains;
d) MMD [18] minimizes the domain discrepancy between the source and target domains by matching the distributions of two domains in the reproducing kernel Hilbert space;
e) Deep subdomain adaptation network (DSAN) [20] aligns the relevant subdomain distributions across different domains based on LMMD;
f) Improved MMD (IMMD) [29]: Combines LMMD and CORAL with entropy minimization to effectively address the domain shift issue.

These methods are representative UDA methods applied to fault diagnosis, including the classical adversarial transfer learning-based DANN and its upgraded version, and the classical distributional difference metric-based methods CORAL, MMD, and their corresponding upgraded versions.

2) Continual UDA: The continual UDA methods are designed to adapt the model to multiple target domains sequentially. The methods used in the experiments include the following:

Basic replay methods:
a) Continual unsupervised adaptation (CUA) [10] leverages replay loss to address catastrophic forgetting and distance loss to minimize the domain discrepancy between the source and target domains;
b) CUA-MMD extends CUA by incorporating MMD to minimize the domain discrepancy between the source and target domains;
c) EverAdapt [15] introduces continual batch normalization combined with a replay mechanism to preserve model performance across varying domains and mitigate catastrophic forgetting.

Replay methods combined with knowledge distillation:
a) Gradient regularized contrastive learning (GRCL) [11] constraints the gradient not to increase the loss on old target domains to prevent catastrophic forgetting;
b) Dynamic weight aggregation (DCTLN-DWA) [14] combines techniques from adversarial domain adaptation and replay distillation loss.

Only knowledge distillation without replay mechanism:
a) Multihead Distillatio (MuHDi) [12] performs distillation at multiple levels from the previous model and uses source data as a proxy to prevent forgetting in the semantic segmentation task. We have reimplemented it into fault diagnosis tasks.

Without loss of generality, we have selected seven representative methods in the fields of fault diagnosis, image classification, and semantic segmentation, including basic replay methods (CUA, CUA-MMD, and EverAdapt), replay methods combined with knowledge distillation (GRCL and DCTLN-DWA), and the method without replay (MuHDi).

E. Implementation Details

To avoid unfair comparisons that can arise from variations in the data augment, backbone, base training parameter, and other factors, we reimplemented all the compared methods within the same framework. Following the suggested settings in [30], the data was converted into the frequency domain by fast Fourier transform (FFT), and then its dimension was reshaped from 1024 to 32×32. The backbone network was ResNet-14, comprising 1 initial convolution layer and 3 residual blocks. Each block is composed of four convolution layers with 3×3 kernels. The number of filters starts from 16 and is doubled every next block. Subsequent to these blocks, there is an average-pooling layer to compress the output feature maps to a feature embedding.

The base training parameter settings were uniform across different methods. The optimizer was SGD with momentum m of 0.9, the origin learning rate η was 0.1, and the batch size b was 128. The total number of epochs E was 40, and the learning rate was reduced by a factor of 0.1 at the 20th and 30th epochs. We used a grid search strategy, which is detailed in the Supplementary Material, to determine the hyperparameters of the proposed CoUDA method. The temperature τ was set to 0.07, and the balance parameter α was set to 0.1. For the baseline methods, the other hyperparameters are all the default settings in the open-source code or the original papers. In addition, the herding algorithm [31] is used for all methods with a replay to select the most representative instances from the previous domain. The number of replay instances is set to 1% , and the total number of replay instances increases continually with the number of target domains.

To validate the robustness of the models, each method underwent five runs with different random seed values, ensuring the reliability of the performance to seed variation. We present the results of our method based on the average and standard deviation from six different scenarios.

F. Comparison With Baselines

We use three metrics, ACC, BWT, and AAD, to evaluate the overall performance, degree of forgetting, and adaptation ability of continual UDA, respectively. The comparative performance of CoUDA and baseline methods on the SDUST dataset across six domain scenarios is detailed in Table V .

The proposed CoUDA has demonstrated state-of-the-art performance, achieving the highest ACC, AAD, and BWT across all six domain scenarios. Specifically, CoUDA achieves an ACC of 97.61% , which is 3.49% higher than the best baseline method, EverAdapt, with most baseline methods scoring below 80% . In terms of BWT, CoUDA has the highest score of -1.01% , indicating minimal forgetting. All vanilla UDA methods exhibit low BWT, showing that catastrophic forgetting is a common issue that has been overlooked in these methods. In comparison, all continual UDA methods consider catastrophic forgetting, thereby achieving high BWT. In addition, CoUDA also achieves the highest AAD of 98.62% , demonstrating superior adaptation

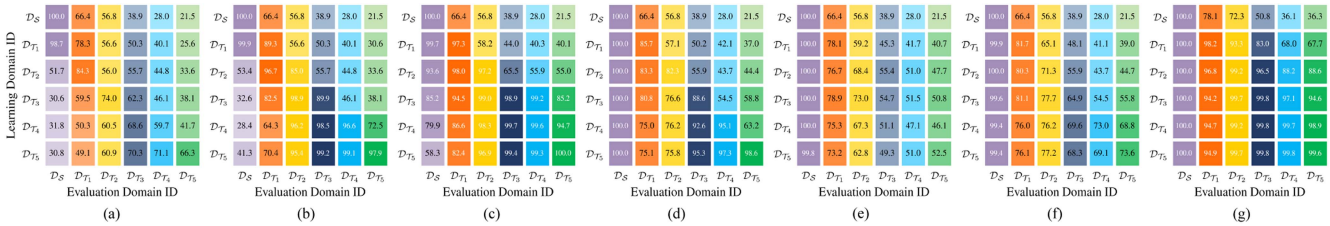

Fig. 6. Accuracy matrices. For each compared CoUDA method, the accuracy matrix is shown in the domain scenario S1.

TABLE V
COMPARATIVE PERFORMANCE OF COUDA AND BASELINE METHODS ON THE SDUST DATASET ACROSS SIX DOMAIN SCENARIOS

Methods	Evaluation Metrics		
	ACC (%)	BWT (%)	AAD (%)
DANN† [17]	40.77 ± 10.71	–6.75 ± 4.53	47.52 ± 7.75
IDANN† [28]	52.18 ± 8.26	–6.12 ± 3.82	58.30 ± 6.97
HDDA† [19]	48.07 ± 5.86	–8.30 ± 3.45	56.37 ± 5.94
MMD† [18]	79.12 ± 4.24	–11.29 ± 2.01	90.40 ± 3.73
DSAN† [20]	80.16 ± 8.88	–10.93 ± 1.63	91.09 ± 9.43
IMMD† [29]	75.39 ± 7.05	–11.37 ± 1.94	86.76 ± 7.62
CUA‡ [10]	64.30 ± 8.09	–1.87 ± 4.71	66.17 ± 5.31
CUA-MMD‡	89.48 ± 3.54	–1.55 ± 1.97	91.03 ± 2.54
EverAdapt‡ [15]	94.12 ± 4.39	–3.44 ± 3.55	97.56 ± 1.36
GRCL‡ [11]	81.35 ± 3.19	–1.43 ± 2.20	85.04 ± 3.83
DCTLN-DWA‡ [14]	66.78 ± 7.40	–1.66 ± 2.36	68.44 ± 6.22
MuHDi [12]	75.35 ± 9.41	–1.11 ± 2.13	75.46 ± 8.00
CoUDA	**97.61 ± 1.43**	**–1.01 ± 0.84**	**98.62 ± 0.95**

† represents the methods re-implemented in the continual UDA scenario.
‡ represents the continual UDA methods using replay with 1% instances.
* **Bold** and underlined fonts represent the best and second-best results, respectively.

TABLE VI
PERFORMANCE OF COUDA ON SIX DOMAIN SCENARIOS

Scenarios	Evaluation Metrics		
	ACC (%)	BWT (%)	AAD (%)
S1	98.36 ± 0.27	–0.62 ± 0.46	98.98 ± 0.59
S2	94.98 ± 1.24	–2.41 ± 1.00	97.39 ± 0.33
S3	98.17 ± 0.24	–0.56 ± 0.70	98.72 ± 0.80
S4	97.63 ± 1.41	–0.88 ± 0.42	98.51 ± 1.56
S5	98.23 ± 0.53	–0.70 ± 0.28	98.93 ± 0.61
S6	98.30 ± 0.34	–0.87 ± 0.38	99.17 ± 0.11

TABLE VII
ABLATION STUDY OF COUDA ON THE SDUST DATASET ACROSS SIX DOMAIN SCENARIOS

Ablation Study	Evaluation Metrics		
	ACC (%)	BWT (%)	AAD (%)
CoUDA	**97.61 ± 1.43**	**–1.01 ± 0.84**	**98.62 ± 0.95**
CoUDA w/o PCL	94.17 ± 8.13	–2.00 ± 3.75	96.17 ± 5.88
CoUDA w/o SDC	94.65 ± 6.23	–3.15 ± 5.02	97.80 ± 6.22
CoUDA w/o FKD	94.46 ± 7.07	–3.71 ± 5.22	98.17 ± 1.96
CoUDA w/o IEM	95.17 ± 6.71	–2.13 ± 3.34	97.30 ± 4.14

Bold and underlined fonts represent the best and second-best results, respectively.

ability. The variance of the baseline methods is significantly higher than that of CoUDA. This is because continual UDA is a challenging task, and poor performance in any one domain will impact the performance of other domains before and after. The results demonstrate that CoUDA exhibits greater robustness due to the stability of domain-invariant representations and the alignment of target domain features with these invariant features.

The superiority of CoUDA is further illustrated in the accuracy matrices of the compared continual UDA methods in the domain scenario S1, as shown in Fig. 6. The accuracy matrices of the compared methods are presented in the form of heatmaps. The color intensity represents the accuracy of the model on the domain, with darker colors indicating higher accuracy. The decline of source discriminator performance in CUA, CUA-MMD, and EverAdapt has a significant impact on the forgetting of the first target domain $\mathcal{D}_{\mathcal{T}_1}$. However, the learning of subsequent domains is somewhat enhanced by supervised replay training, which in turn improves the performance of some history domains, resulting in high BWT for these three basic replay methods. The performance of GRCL, DCTLN-DWA, and MuHDi is relatively stable in BWT, but they lack the ability to adapt to new target domains, resulting in low AAD and ACC. In contrast, CoUDA achieves excellent adaptation performance across all target domains while maintaining high accuracy on old target domains.

G. Performance of CoUDA on Six Domain Scenarios

To further evaluate the performance of CoUDA, we present the results of CoUDA on six domain scenarios, as shown in Table VI. The scenario S2 is the most challenging scenario, with the lowest scores of ACC, BWT, and AAD. There is no significant difference in the difficulty of the other scenarios. Compared with S1, S2 swaps the order of the source and first target domains, i.e., the adaptation order changes from W1 →W2 to W2 →W1. The low performance of S2 suggests that the adaptation of W2 →W1 is not satisfactory, which in turn affects the whole results. This further illustrates the complexity of the CoUDA task.

H. Ablation Study

An ablation study is conducted to assess the efficacy of each component in the CoUDA framework. The results are presented in Table VII. All three evaluation metrics decline in the ablation method. Compared with the full model, the removal of SDC and FKD resulted in a 2.14% and 2.70% reduction in BWT, respectively, indicating that both SDC and FKD are effective in mitigating catastrophic forgetting. Although the removal of IEM does not have the greatest impact on any of the three

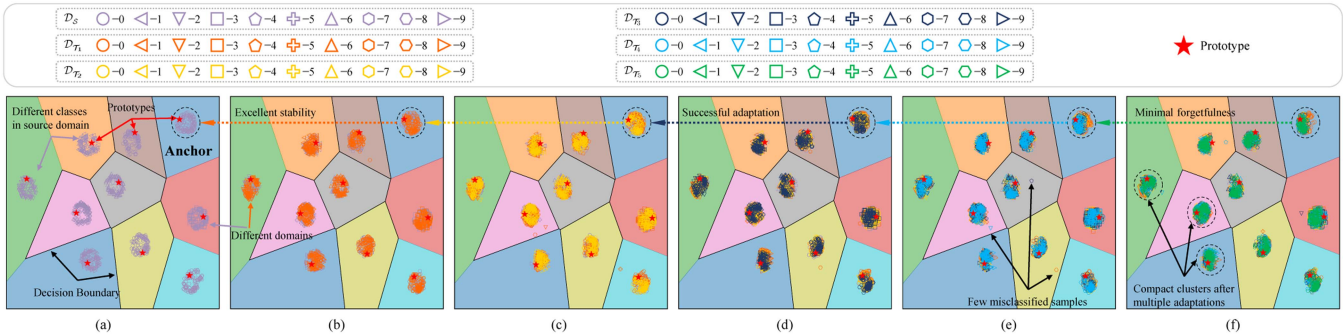

Fig. 7. U-map visualization of the feature distribution on different phases in the domain scenario S1.

TABLE VIII
COMPUTATIONAL COSTS OF COUDA AND BASELINE METHODS

	CUA	CUA-MMD	GRCL	DCTLN-DWA	EverAdapt	MuHDi	CoUDA
Storage memory (M)	8.75	8.75	8.75	8.75	8.75	8.55	8.52
Training time (s)	66.36	67.71	77.64	75.27	77.04	72.66	87.55

metrics, it results in a significant reduction in all three metrics, suggesting that reducing the uncertainty of instances is beneficial for model adaptation and preventing forgetting. The removal of PCL reduces the generalization ability of the source model, leading to a reduction in its ability of both adaptation and antiforgetting, more detailed analysis is shown in the Supplementary Material.

I. Visualization of Feature Distribution

To further validate the effectiveness of CoUDA, we visualize the feature distribution of the model across different phases in the domain scenario S1, as shown in Fig. 7. The U-map visualization demonstrates the feature distribution of the model across different phases in the domain scenario S1. CoUDA employs the source domain as an anchor, effectively aligning all target domains towards it, thereby mitigating catastrophic forgetting. Throughout all phases, CoUDA has demonstrated commendable performance

J. Computational Complexity Analysis

To show the computational cost of the proposed CoUDA, we conducted computational complexity experiments and compared it with the baseline methods of continual UDA methods. The computational complexity of CoUDA is evaluated in terms of training time and storage requirements. The experimental results are presented in Table VIII. As shown in Table VIII, the storage memory of CoUDA is 8.52 M, which is lower than that of all baseline methods. This is mainly because the CoUDA framework fixes the classifier and abandons the replay mechanism, which contributes to the reduction in storage memory. For training time, CoUDA completes the training in 87.55 s, which is higher than most baseline methods. The increase in time overhead of CoUDA is due to the introduction of additional metric computations such as PCL, FKD, LDA, and IEM. Consequently, although CoUDA has a long training time, its superior performance and lowest storage memory justify the increase in time overhead.

V. CONCLUSION

In this study, we propose a novel continual UDA framework, CoUDA, for fault diagnosis under dynamic working conditions from the perspective of metric and representation learning. CoUDA can achieve the learning of domain-invariant representations, implicit and explicit maintenance of the historical feature distribution, alignment of the local domain features, and reduction of feature uncertainty, respectively. The source domain plays both the role of a proxy for overcoming the forgetting of historical domains and an anchor point for domain adaptation. The experimental results on the SDUST dataset, across six domain scenarios, demonstrate that CoUDA has outstanding performance in continual UDA for fault diagnosis tasks. It is worth noting that although the designed CoUDA framework can greatly improve the fault diagnosis performance compared with the baseline methods through the continual UDA strategy, this will increase the model training time. Therefore, exploring other optimization strategies to reduce the training time will be an important research direction in the future. In addition, it is also interesting to explore continual source-free UDA methods to further enhance data privacy constraints.

REFERENCES

[1] S. Fan, X. Zhang, and Z. Song, "Imbalanced sample selection with deep reinforcement learning for fault diagnosis," *IEEE Trans. Ind. Inform.*, vol. 18, no. 4, pp. 2518–2527, Apr. 2022.
[2] S. Yin, X. Li, H. Gao, and O. Kaynak, "Data-based techniques focused on modern industry: An overview," *IEEE Trans. Ind. Electron.*, vol. 62, no. 1, pp. 657–667, Jan. 2015.
[3] X. Zhang, H. Zhang, and Z. Song, "Feature-aligned stacked autoencoder: A novel semisupervised deep learning model for pattern classification of industrial faults," *IEEE Trans. Artif. Intell.*, vol. 4, no. 4, pp. 592–601, Aug. 2023.
[4] Z. Yang and Z. Ge, "On paradigm of industrial Big Data analytics: From evolution to revolution," *IEEE Trans. Ind. Inform.*, vol. 18, no. 12, pp. 8373–8388, Dec. 2022.
[5] L. Guo, Y. Lei, S. Xing, T. Yan, and N. Li, "Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabeled data," *IEEE Trans. Ind. Electron.*, vol. 66, no. 9, pp. 7316–7325, Sep. 2019.
[6] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, 2017.
[7] B. Chen, C. Shen, D. Wang, L. Kong, L. Chen, and Z. Zhu, "A lifelong learning method for gearbox diagnosis with incremental fault types," *IEEE Trans. Instrum. Meas.*, vol. 71, 2022, Art. no. 3514010.
[8] A. Ding, Y. Qin, B. Wang, X. Cheng, and L. Jia, "An elastic expandable fault diagnosis method of three-phase motors using continual learning for class-added sample accumulations," *IEEE Trans. Ind. Electron.*, vol. 71, no. 7, pp. 7896–7905, Jul. 2024.

[9] C. Sun, L. Gao, X. Li, P. Zheng, and Y. Gao, "An incremental knowledge learning framework for continuous defect detection," *IEEE Trans. Instrum. Meas.*, vol. 73, 2024, Art. no. 3505211.
[10] A. Bobu, E. Tzeng, J. Hoffman, and T. Darrell, "Adapting to continuously shifting domains," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–4.
[11] S. Tang, P. Su, D. Chen, and W. Ouyang, "Gradient regularized contrastive learning for continual domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 2665–2673.
[12] A. Saporta, A. Douillard, T.-H. Vu, P. Pérez, and M. Cord, "' for continual unsupervised domain adaptation in semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 3751–3760.
[13] B. Chen, C. Shen, L. Li, J. Shi, W. Huang, and Z. Zhu, "Continual unsupervised domain adaptation for bearing fault diagnosis under variable working conditions," in *Proc. Int. Conf. Elect. Inf. Technol. Rail Transp.*, 2023, pp. 395–403.
[14] J. Li, R. Huang, Z. Chen, G. He, K. C. Gryllias, and W. Li, "Deep continual transfer learning with dynamic weight aggregation for fault diagnosis of industrial streaming data under varying working conditions," *Adv. Eng. Inform.*, vol. 55, 2023, Art. no. 101883.
[15] Edward, M. Ragab, M. Wu, Y. Xu, Z. Chen, A. Alseiari, and X. Li, "EverAdapt: Continuous adaptation for dynamic machine fault diagnosis environments," *Proc. Mech. Syst. Signal*, vol. 226, 2025, Art. no. 112317.
[16] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 8, pp. 5362–5383, Aug. 2024.
[17] Y. Wang, X. Sun, J. Li, and Y. Yang, "Intelligent fault diagnosis with deep adversarial domain adaptation," *IEEE Trans. Instrum. Meas.*, vol. 70, 2021, Art. no. 2503509.
[18] B. Yang, Y. Lei, F. Jia, N. Li, and Z. Du, "A polynomial kernel induced distance metric to improve deep transfer learning for fault diagnosis of machines," *IEEE Trans. Ind. Electron.*, vol. 67, no. 11, pp. 9747–9757, Nov. 2020.
[19] X. Wang, H. He, and L. Li, "A hierarchical deep domain adaptation approach for fault diagnosis of power plant thermal system," *IEEE Trans. Ind. Inform.*, vol. 15, no. 9, pp. 5139–5148, Sep. 2019.
[20] Y. Zhu et al., "Deep subdomain adaptation network for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 4, pp. 1713–1722, Apr. 2021.
[21] H. Xu et al., "Enhancing information maximization with distance-aware contrastive learning for source-free cross-domain few-shot learning," *IEEE Trans. Image Process.*, vol. 33, pp. 2058–2073, 2024.
[22] S. Rakshit, A. Mohanty, R. Chavhan, B. Banerjee, G. Roig, and S. Chaudhuri, "Frida — generative feature replay for incremental domain adaptation," *Comput. Vis. Image Understanding*, vol. 217, 2022, Art. no. 103367.
[23] L. Li et al., "Progressive domain expansion network for single domain generalization," in *Proc. 2021 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 224–233.
[24] Z. Huang, J. Chen, J. Zhang, and H. Shan, "Learning representation for clustering via prototype scattering and positive sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 7509–7524, Jun. 2023.
[25] C. Du, Y. Wang, S. Song, and G. Huang, "Probabilistic contrastive learning for long-tailed visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 9, pp. 5890–5904, Sep. 2024.
[26] J. Liang, D. Hu, Y. Wang, R. He, and J. Feng, "Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 8602–8617, Nov. 2022.
[27] S. Jia, J. Wang, B. Han, G. Zhang, X. Wang, and J. He, "A novel transfer learning method for fault diagnosis using maximum classifier discrepancy with marginal probability distribution adaptation," *IEEE Access*, vol. 8, pp. 71475–71485, 2020.
[28] D. Zhang and L. Zhang, "A multi-feature fusion-based domain adversarial neural network for fault diagnosis of rotating machinery," *Measurement*, vol. 200, 2022, Art. no. 111576.
[29] M. Azamfar, X. Li, and J. Lee, "Deep learning-based domain adaptation method for fault diagnosis in semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 33, no. 3, pp. 445–453, Aug. 2020.
[30] Z. Zhao et al., "Deep learning algorithms for rotating machinery intelligent diagnosis: An open source benchmark study," *ISA Trans.*, vol. 107, pp. 224–255, 2020.
[31] Y. Liu, B. Schiele, and Q. Sun, "Adaptive aggregation networks for class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2544–2553.

Bojian Chen received the B.S. degree in vehicle engineering and the M.S. degree in pattern recognition and intelligent systems from Soochow University, Suzhou, China, in 2020 and 2023, respectively. He is currently working toward the Ph.D. degree in control science and engineering with Zhejiang University, Hangzhou, China.

His research interests include prognostic and health management, continual learning, fault diagnosis, and foundation models.

Xinmin Zhang (Member, IEEE) received the Ph.D. degree in system science from Kyoto University, Kyoto, Japan, in 2019.

From April 2019 to December 2019, he was a Postdoctoral Research Fellow with the Department of Systems Science, Kyoto University. From 2020 to 2023, he was an Associate Professor with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China, where he is currently a Professor. His research interests include process control, process data analysis, machine learning and industrial Big Data, fault diagnosis, and soft sensors with application to industrial processes.

Changqing Shen (Senior Member, IEEE) received the B.S. and Ph.D. degrees in instrument science and technology from the University of Science and Technology of China, Hefei, China, in 2009 and 2014, respectively, and the Ph.D. degree in systems engineering and engineering management from the City University of Hong Kong, Hong Kong, in 2014.

He is currently Professor with the School of Rail Transportation, Soochow University, Suzhou, China. His research interests include signal processing and machine learning-based fault diagnosis.

Dr. Shen is an Associate Editor for IEEE OPEN JOURNAL OF INSTRUMENTATION AND MEASUREMENT.

Qi Li (Graduate Student Member, IEEE) received the B.Eng. and M.Eng. degrees in electrical engineering and control theory and control engineering from Soochow University, Suzhou, China, in 2019 and 2022, respectively. He is currently working toward the Ph.D. degree in mechanical engineering with Tsinghua University, Beijing, China.

His research interests include the intersection of trustworthy AI, reliable prognostic and health management, and foundation models.

Zhihuan Song received the B.Eng. and M.Eng. degrees in industrial automation from the Hefei University of Technology, Hefei, China, in 1983 and 1986, respectively, and the Ph.D. degree in industrial automation from Zhejiang University, Hangzhou, China, in 1997.

Since 1997, he has been with the College of Control Science and Engineering, Zhejiang University, where he was first a Postdoctoral Research Fellow, then an Associate Professor, and currently a Professor. He has authored or coauthored more than 200 papers in journals and conference proceedings. His research interests include the modeling and fault diagnosis of industrial processes, analytics and applications of industrial Big Data, and advanced process control technologies.