Mechanical Systems and Signal Processing 226 (2025) 112317

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

# EverAdapt: Continuous adaptation for dynamic machine fault diagnosis environments

Edward [a,1], Mohamed Ragab [b,c,*,1], Min Wu [b], Yuecong Xu [b], Zhenghua Chen [b], Abdulla Alseiari [c], Xiaoli Li [a,b]

[a] Centre for Frontier AI Research, A*STAR, Singapore 138632, Singapore
[b] Institute for Infocomm Research, A*STAR, Singapore 138632, Singapore
[c] Propulsion and Space Research Center, Technology Innovation Institute, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Communicated by D. Wang

*Keywords:*
Domain adaptation
Fault diagnosis
Continual learning
Dynamic environments

ABSTRACT

Unsupervised Domain Adaptation (UDA) has emerged as a key solution in data-driven fault diagnosis, addressing domain shift where models underperform in changing environments. However, under the realm of continually changing environments, UDA tends to underperform on previously seen domains when adapting to new ones - a problem known as catastrophic forgetting. To address this limitation, we introduce the EverAdapt framework, specifically designed for continuous model adaptation in dynamic environments. Central to EverAdapt is a novel Continual Batch Normalization (CBN), which leverages source domain statistics as a reference point to standardize feature representations across domains. EverAdapt not only retains statistical information from previous domains but also adapts effectively to new scenarios. Complementing CBN, we design a class-conditional domain alignment module for effective integration of target domains, and a Sample-efficient Replay strategy to reinforce memory retention. Experiments on real-world datasets demonstrate EverAdapt superiority in maintaining robust fault diagnosis in dynamic environments. Our code is available here: EverAdapt-Code.

## 1. Introduction

Data-driven fault diagnosis has emerged as a critical discipline, serving as an indispensable tool for minimizing maintenance expenditure, maximizing equipment reliability, and ensuring operational safety across diverse industrial sectors [1]. Deep learning (DL) has demonstrated superior performance in classifying different health states through its automatic feature learning capabilities [2]. However, DL only works under the assumption that training samples and testing samples originate from the same distribution. As such, DL performance significantly deteriorates when encountering testing distributions that differ from training distributions [3]. This phenomenon is known as the domain shift problem, where training (i.e., source domain) and testing (i.e., target domain) data are sampled from different distributions. In machine fault diagnosis, such problems are prevalent as changing working conditions are a key property in real-world environments [4].

Unsupervised domain adaptation (UDA) has emerged as a promising solution to address distribution shift challenges in fault diagnosis. By leveraging labeled data from a source domain, UDA facilitates knowledge transfer to unlabeled data in a target

* Corresponding author at: Propulsion and Space Research Center, Technology Innovation Institute, Abu Dhabi, United Arab Emirates.
*E-mail address:* mohamedr002@e.ntu.edu.sg (M. Ragab).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.ymssp.2025.112317
Received 19 July 2024; Received in revised form 30 November 2024; Accepted 3 January 2025
Available online 16 January 2025

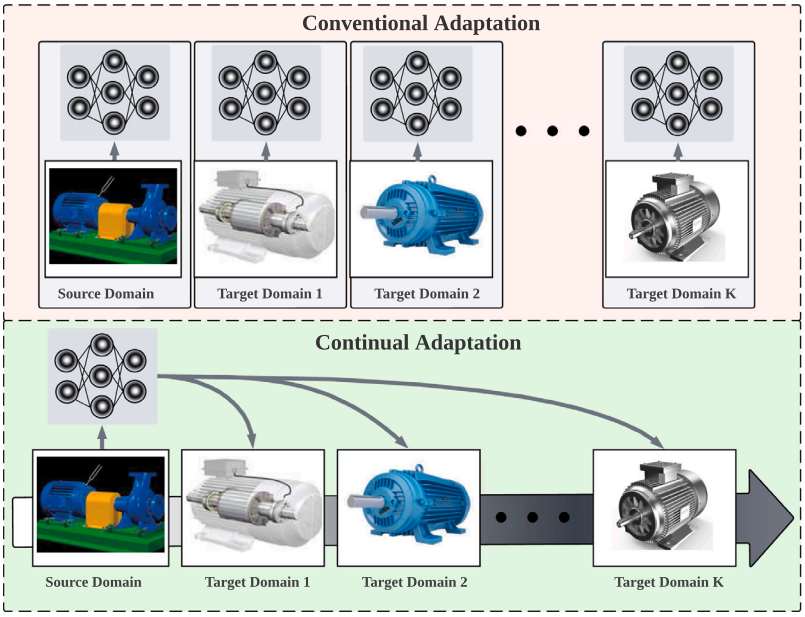



**Fig. 1.** Comparison of Conventional and Continual Adaptation Approaches in Domain Adaptation. **Top**: the conventional adaptation approach, where individual models are independently trained for each new target domain. This often results in a scalability issue as the number of target domains increases, necessitating separate model and training phases for each domain. **Bottom**: the continual adaptation strategy, which employs a singular model that is sequentially adapted across multiple target domains. This method maintains knowledge from previous domains, effectively mitigating catastrophic forgetting and promoting model adaptation across a series of domain shifts.

domain [5]. However, as UDA requires simultaneous access to both source and target data, privacy concerns may arise regarding source data. To address this, source-free domain adaptation approaches have been proposed, enabling adaptation to the target domain while preserving source data privacy [6,7]. Despite these advancements, both UDA and source-free domain adaptation methods typically assume the existence of a single target domain—a limitation that does not align with real-world scenarios. In dynamic systems, operating conditions can change over time, necessitating models capable of adapting to multiple and evolving target domains, as illustrated in Fig. 1. Progressive domain adaptation methods can address the multi-targets adaptation via progressively adapting from easy target subsets to harder target subsets [8,9]. However, this approach overlooks the catastrophic forgetting problem, a well-known issue where models trained on multiple tasks sequentially may forget previously learned knowledge [10,11]. To mitigate this, continual unsupervised domain adaptation methods have gained attention for their ability to adapt to new domains while retaining knowledge from prior ones [12,13]. However, the majority of existing methods are designed for computer vision applications, which may fail to perform well on time series data in machine fault diagnosis applications. A naive solution to this issue is to train a separate model for each working condition. However, this approach is impractical and resource-intensive. Therefore, there is a need for models that can continually adapt to new domains without losing their effectiveness in earlier ones [11].

Recently, continual unsupervised domain adaptation methods have gained traction by allowing models to adapt to new domains without forgetting previous ones [12,13]. However, the majority of existing methods are designed for computer vision applications, which may fail to perform well on time series data in machine fault diagnosis applications. Further, we argue conventional batch normalization (BN) can be detrimental to knowledge retention when adapting to new domains in fault diagnosis applications. Specifically, BN adjusts the model to the current domain's statistics, overlooking those from previous domains. This causes the model to specialize in the latest domain, impairing its performance on previously seen domains. To address this issue, the "EverAdapt" framework is designed for continual model adaptation across diverse domains while addressing the catastrophic forgetting problem. The framework features a class-conditional domain alignment (CCA) module for integrating new domains, aligning them with the source domain at the class-wise level. This ensures effective domain adaptation by addressing class misalignment, crucial for consistent performance across different conditions. To address the catastrophic forgetting problem, we develop a novel Continual Batch Normalization (CBN), which standardizes the batch statistics across different domains using fixed statistics from the source domain. This process ensures consistent feature representation, significantly reducing the risk of forgetting when adapting to new domains. However, resetting target domains to source statistics in CBN can lead to training instability due to domain distribution shifts. To counter this, we reduce the uncertainty of the learned features by minimizing their conditional entropy. This approach helps mitigate the instability caused by the adaptation of batch statistics from various domains to the source statistics. Beyond adapting batch statistics across domains, our approach augments CBN with simple self-training using replay samples to align fine-grained classes between domains. Notably, integrating CBN significantly cuts down the number of replay samples required for effective self-training.

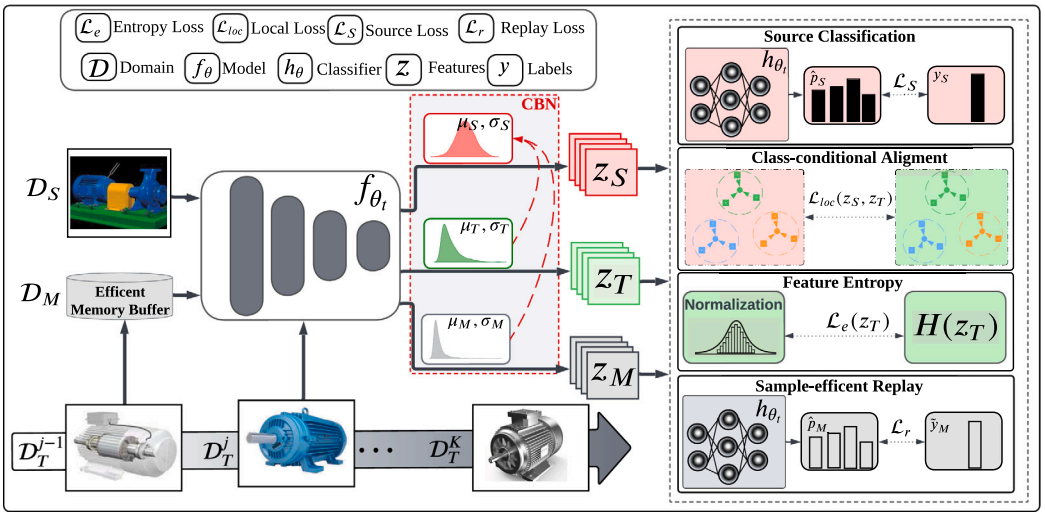



**Fig. 2.** EverAdapt Framework Overview: incorporates input source samples, input target samples from the current target domain, and input memory samples to the feature extractor. It applies conditional entropy loss on the feature space of the target samples, cross-entropy loss on the input source samples, self-training with pseudo-labels on the memory samples, and local alignment loss between the source and target features.

In summary, EverAdapt presents a scalable and efficient framework adept at navigating the dynamic complexities of machine fault diagnosis. The primary contributions of this approach are summarized as follows:

- Forgetting Prevention Module: Introducing a novel CBN technique via standardizing batch statistics across domains using fixed statistics from the source domain. This approach preserves consistent feature representation and substantially mitigates the risk of forgetting.
- Flexible EverAdapt Framework: The adaptable design of the EverAdapt framework supports a variety of adaptation and replay techniques, enabling its application beyond fault diagnosis to more general domain adaptation scenarios.
- Empirical Validation: Demonstrated superiority of the proposed approach through experiments on real-world datasets, showcasing significant improvements over state-of-the-art methods and substantial mitigation of the forgetting issue.

## 2. Related works

### 2.1. Domain adaptation for fault diagnosis

In the field of machine fault diagnosis, domain adaptation has emerged as a vital solution for adapting models to diverse industrial environments. Early studies focused on aligning feature distributions using techniques like Maximum Mean Discrepancy (MMD) [14]. Adversarial networks were later introduced for improved distribution alignment [15]. Recent advancements include class-conditional alignment methods [16], which align not only feature distributions but also class-related information between domains. Some techniques leverage multiple source domains through weighting schemes [17]. While these approaches are effective in static environments with a single target domain, they encounter limitations when dealing with dynamic environments where models encounter multiple domains sequentially. Notably, as models adapt to new domains, they often suffer from the drawback of forgetting knowledge about previously encountered domains. This limitation underscores the need for novel methods to facilitate adaptation to sequential, dynamic domains while preserving knowledge from previous domains.

### 2.2. Continual domain adaptation

Continual adaptation to new domains while retaining knowledge of previous domains is a crucial challenge in computer vision applications. Existing methods have primarily focused on mitigating catastrophic forgetting when adapting to new domains. Feature replay has proven instrumental in addressing this problem, either through subsamples from previous domains [12,18] or synthetic data generated by generative models [13,19]. Another approach involves parameter and weight regularization, achieved by either regularizing domain-specific features [20], domain-specific neurons [21], or domain-specific weights [22]. While these methods have been effective in vision applications, they may not be directly applicable to signal data in machine fault diagnosis. Moreover, these approaches often overlook the contribution of Batch Normalization (BN) to the forgetting problem in previously seen domains. In contrast, we introduce a novel approach tailored to machine fault diagnosis. We present a simple yet effective Continual Batch Normalization (CBN) technique that addresses BN limitations and significantly reduces forgetting on previously seen domains

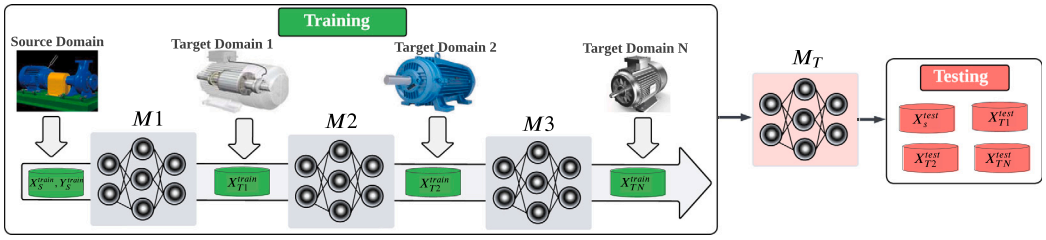



**Fig. 3.** The training and testing procedure for the continual adaptation scenario is as follows: the model is first trained on the source data and then continually adapted to different target domains, updating progressively with each domain. Finally, the last model, denoted as $M_T$, is tested on the testing data of all previously encountered domains to assess both forgetting and adaptability.

## 3. Methodology

### 3.1. Problem definition

In *continual domain adaptation*, the objective is to train a model $f_\theta$ to predict labels across a sequence of target domains $D_T = \{D_T^1, D_T^2, \ldots, D_T^K\}$, where each target domain has a unique marginal distribution $P_T^i(x)$, distinct from the source domain $D_S$. The source dataset $D_S = \{x_S^i, y_S^i\}_{i=1}^{n_S}$, consisting of labeled samples, is available during the adaptation to each target domain. The conditional distributions $P(y|x)$ are assumed to remain invariant across the source and target domains. The challenge is to enable $f_\theta$ to adapt to the distinct characteristics of each target domain while retaining knowledge of previously seen domains, including the source. This requires addressing *catastrophic forgetting*, where the model's performance on earlier domains deteriorates as it adapts to new ones. Testing is conducted on all historical domains, including the source, to verify the model's ability to generalize and retain knowledge effectively, as illustrated in Fig. 3 . shows the training and testing

### 3.2. Overview of EverAdapt

EverAdapt integrates two key components: Class-Conditional Alignment (CCA), and Continual Batch Normalization (CBN) complemented with self-training, as illustrated in Fig. 2. Specifically, CCA effectively addresses domain shifts by maintaining fine structures during adaptation. Continual Batch Normalization, which normalizes incoming target domain data using source domain statistics, in conjunction with self-training of replay samples, ensures that the model retains knowledge of its previously learned domains without forgetting. The detailed algorithm is presented in Algorithm 1, and the subsequent sections provide a thorough discussion of each component.

### 3.3. Pretraining on source domain

The source model architecture consists of a feature extractor $f_{\theta_s} : \mathcal{X} \to \mathcal{Z}$, which maps the input space to the feature space $\mathcal{Z} \in \mathbb{R}^d$, and a classifier $h_{\theta_s} : \mathcal{Z} \to \mathcal{Y}$, responsible for mapping the feature space to class predictions. To train the source model, we utilize the standard cross-entropy loss $\mathcal{L}_{ce}^s$, which is defined as:

$$\mathcal{L}_{ce}^s = -\sum_{c=1}^{C} \tilde{y}_{s,c} \log\left(p_{s,c}\right) \tag{1}$$

where, $p_{s,c} = \sigma(h_{\theta_s}(f_{\theta_s}(x_S)))$ is the $c$th element of the softmax output, $\sigma(\cdot)$ represents the softmax function.

Once the source model is trained, we transfer its weights and batch normalization statistics to the target domains to obtain $f_{\theta_t}$ and $h_{\theta_t}$. This transfer sets the stage for training the target model to adapt sequentially to the incoming multiple target domains.

### 3.4. Class-conditional Alignment (CCA)

One of the key tasks in continual domain adaptation is the alignment of data distributions across different domains. However, conventional alignment methods primarily focus on aligning feature distributions between the source and target domains. While effective to some extent, they often overlook the fine-grained class distribution within each domain. This oversight can lead to a misalignment of similar classes across domains, negatively impacting the model's adaptation performance. To address this challenge, inspired by local maximum mean discrepancy (LMMD) loss [23], we introduce our CCA module Our approach addresses the issue of unlabeled target samples by leveraging robust pseudo-labeling, where pseudo-labels are assigned based on the class with the highest predicted probability. This ensures that target samples are grouped into their most likely classes, facilitating more accurate alignment between the source and target domains. The pseudo-label for a target sample $x_T^j$ is given by:

$$\hat{y}_T^j = \arg\max \sigma(f_{\theta_t}(z_T^j)), \tag{2}$$

where $\hat{y}_T^i$ is the pseudo-label for the $i$th target sample, and $f_{\theta_t}$ represents the encoder model applied to current target domain time $D_T^t$. Once pseudo-labels are assigned, we align the class distributions by minimizing a class-level loss. This loss aims to reduce the discrepancy between the source and target distributions for each class. The class-level alignment loss $\mathcal{L}_{loc}$ can be expressed as:

$$\mathcal{L}_{loc} = \min_{\theta} \sum_{c=1}^{C} d(Z_S^c, Z_T^c), \tag{3}$$

where $C$ denotes the number of classes, $Z_S^c$ and $Z_T^c$ are the latent features for class $c$ in the source and target domains, respectively. $d(\cdot,\cdot)$ is a distance metric measuring the discrepancy between the two domains. Here, the Maximum Mean Discrepancy (MMD) is employed as the distance between similar classes across domains, which defined as:

$$\mathrm{d}(Z_S^c, Z_T^c) = \left\| \mathbb{E}_{Z_S}[\zeta(Z_S^c)] - \mathbb{E}_{Z_T}[\zeta(Z_T^c)] \right\|. \tag{4}$$

In the above equation, $\zeta$ is a feature map transforming the samples into a Reproducing Kernel Hilbert Space (RKHS) with a characteristic kernel $k$, and $\|\cdot\|$ denotes the norm in this space. The kernel function $k$ is defined by the inner product in the RKHS: $k(\cdot,\cdot) = \langle \zeta(\cdot), \zeta(\cdot) \rangle$.

This design of our CCA Module is particularly robust in scenarios with highly imbalanced classes and noisy pseudo-labels. For imbalanced classes, direct alignment of individual pseudo-labeled samples may overemphasize dominant classes and underrepresent minority ones. By clustering and aligning at the class level, our method ensures that each class contributes proportionally to the alignment process. Additionally, incorrect pseudo-labels have minimal impact, as alignment is driven by the majority of correctly pseudo-labeled samples within each cluster.

**Algorithm 1** Continual Domain Adaptation Algorithm

**Require:** Source dataset $\mathcal{D}_S$, sequence of target domains $\{D_T^1, D_T^2, \ldots, D_T^K\}$.
**Ensure:** Adapted model $f_\theta^K$ for the last target domain, performance metrics for the current domain, and backward transfer (forgetability) metrics for previous domains.
1: Pretrain the model $f_\theta$ on the source dataset $\mathcal{D}_S$.
2: **for** each target domain $t$ in $\{1, \ldots, K\}$ **do**
3: Input a batch of source samples from $\mathcal{D}_S$ into model $f_{\theta_t}$.
4: Input buffer samples from the previous target domain $D_T^{t-1}$ into model $f_{\theta_t}$.
5: Input a batch of current target data from $D_T^t$ into model $f_{\theta_t}$.
6: Normalize the batch statistics of the current target domain and memory samples with respect to source statistics (refer to Eq. (10)).
7: Compute the source classification loss used during pre-training (refer to Eq. (1)).
8: Compute the conditional entropy loss by minimizing the uncertainty of the target feature representation (refer to Eq. (11)).
9: Compute the class-level alignment loss by minimizing the discrepancy between the source and target distributions for each class (refer to Eq. (3)).
10: Optimize the models $f_{\theta_t}, h_{\theta_t}$ by minimizing the overall loss (refer to Eq. (14)).
11: Assess performance on the current domain $D_T^t$ post-adaptation.
12: **if** $t > 1$ **then**
13: Measure backward transfer (forgetability) on previous domains $\{D_T^1, \ldots, D_T^{t-1}\}$.
14: **end if**
15: **end for**
16: Evaluate the overall performance across all domains.

### 3.5. Preventing catastrophic forgetting

A major challenge in continual adaptation is mitigating performance degradation on previously learned domains after adapting to new domains, a phenomenon known as catastrophic forgetting. In this work, we posit that batch normalization (BN) contributes significantly to this forgetting. To address this, we introduce a simple yet effective approach that adapts BN for sequentially arriving domains. We first discuss conventional BN to identify the underlying causes of forgetting. Subsequently, we present our CBN technique, designed specifically to overcome the issue of catastrophic forgetting in dynamic learning environments.

#### 3.5.1. Batch normalization

Batch Normalization (BN) is an essential technique in neural networks, aimed at addressing internal covariate shift. It normalizes the inputs of each layer to have zero mean and unit variance, contributing to the stabilization of the training process. For a mini-batch $\mathcal{B}$, BN normalizes each input $x_i$ as:

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}. \tag{5}$$

Here, $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ are the mean and variance of the mini-batch, respectively, calculated by:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2. \tag{6}$$

The normalized input $\hat{x}_i$ is then linearly transformed using learnable parameters $\gamma$ and $\beta$:

$$y_i = \gamma \hat{x}_i + \beta. \tag{7}$$

A fundamental limitation of conventional BN in continual learning arises from its domain-specific normalization approach. BN normalizes inputs based on the current domain's statistics as:

$$\mathrm{BN}(x;\mu_{\mathrm{domain}},\sigma^2_{\mathrm{domain}}) = \gamma\left(\frac{x-\mu_{\mathrm{domain}}}{\sqrt{\sigma^2_{\mathrm{domain}}+\epsilon}}\right)+\beta \tag{8}$$

In this context, $\mu_{\mathrm{domain}}$ and $\sigma^2_{\mathrm{domain}}$ are the mean and variance computed from the current domain's data. While this approach is effective for static data distributions, it can be problematic for continual learning. Rapid adaptation to the new domain's statistics ($\mu_{\mathrm{domain}},\sigma^2_{\mathrm{domain}}$) may lead to a loss of information about previous domains' statistical properties, posing a challenge for models that need to perform well across diverse and evolving data streams.

### 3.5.2. Continual Batch Normalization (CBN)

To overcome the limitations of conventional BN in continual learning scenarios, we introduce CBN. This technique aims to preserve knowledge from previously learned domains while effectively adapting to new data, mitigating catastrophic forgetting. Unlike conventional BN, which recalculates mean and variance for each target domain, CBN standardizes the normalization process using statistics from the source domain.

During the source pretraining stage, we obtain running source statistics, including mean $\mu_{\mathrm{EMA}}$ and variance $\sigma^2_{\mathrm{EMA}}$, from each batch using Exponential Moving Average (EMA):

$$\begin{aligned}\mu_{\mathrm{EMA}} &= (1-\alpha)\cdot\mu_{\mathrm{EMA}}+\alpha\cdot\mu_S,\\ \sigma^2_{\mathrm{EMA}} &= (1-\alpha)\cdot\sigma^2_{\mathrm{EMA}}+\alpha\cdot\sigma^2_S.\end{aligned} \tag{9}$$

Using these estimated source statistics, we standardize the batches of the all the incoming target domain:

$$\hat{x}_T = \frac{x_T-\mu_{\mathrm{EMA}}}{\sqrt{\sigma^2_{\mathrm{EMA}}+\epsilon}}. \tag{10}$$

By normalizing target domain data relative to the fixed statistics from the source domain, CBN maintains a consistent feature distribution across domains. This consistency ensures that knowledge from the source domain is preserved as the model adapts to new target domains, enhancing its generalization capabilities in continual domain adaptation tasks.

### 3.5.3. Minimizing features entropy

Resetting different target domains to the source statistics can cause instability in the training performance of CBN due to the distribution shift between domains. To address this, we aim to reduce the uncertainty of the learned features by minimizing their conditional entropy. This differs from popular entropy minimization techniques in domain adaptation which applies entropy minimization in the label space [24] rather than the feature space. This approach helps mitigate the instability caused by the differing adaptation of batch statistics from various domains to the source statistics. We formulate this process as follows: Given the target domain features $z_T = f_{\theta_t}(x_T)$, we normalize these features to obtain $\hat{z}_T = \mathrm{Norm}(z_T)$. Finally, our objective is to minimize the conditional entropy of the normalized features, which can be expressed as:

$$L_e = \min_{\theta} H(\hat{z}_T|x_T), \tag{11}$$

where $H(\hat{z}_T|x_T)$ represents the conditional entropy, which quantifies the average uncertainty in the normalized feature set $\hat{z}_T$ given the observed target data $x_T$. By minimizing $L_e$, we aim to reduce this uncertainty, thereby enhancing the features sharpness and, consequently, stabilizing the training process amidst varying domain-specific data distributions.

### 3.5.4. Sample-efficient replay

While CBN can significantly reduce forgetting by referencing the batch statistics of incoming target domains to the statics of the source domain, there still exists a risk of forgetting due to variations in class distribution across different domains. To counter this, we employed an efficient yet effective sample replay strategy [25]. Particularly, a replay buffer stores data points sampled randomly with a uniform distribution and corresponding predictions from each target domain. These replay samples are denoted as $x_M$, with $z_M$ representing their extracted features. We utilized the cross-entropy loss $\mathcal{L}_{ce}$ for self-training the model using the predicted pseudo-labels from these replay samples. While more effective replay selection and loss functions exist.

$$\mathcal{L}_r = \min_{\theta} \mathcal{L}_{ce}(h_\theta(f_\theta(x_M)),\tilde{y}_m) \tag{12}$$

Here, $\mathcal{L}_r$ represents the replay loss, $h_\theta(f_\theta(x_M))$ are the predictions from the classification network for the features $f_\theta(x_M)$, and $\tilde{y}_m$ are the corresponding pseudo labels for these replay samples.

### 3.6. Overall objective

EverAdapt optimizes multiple objectives to facilitate adaptation to new domains while retaining knowledge from previous ones. These objectives include minimizing the conditional entropy of target features ($\mathcal{L}_e$), aligning source and target features with consideration for class information ($\mathcal{L}_{\text{loc}}$), self-training using memory samples ($\mathcal{L}_{\text{m}}$), and maintaining source classification performance ($\mathcal{L}_s$). However, balancing the minimization of entropy and class-conditional alignment (CCA) poses challenges, as excessive entropy reduction can result in prediction collapse into a single class, counteracting CCA's goal of precise class alignment across domains. To navigate this, we employ an adaptive weighting strategy. Initially, we prioritize entropy minimization ($\mathcal{L}_e$) with lesser emphasis on CCA loss ($\mathcal{L}_{\text{loc}}$). As training progresses, we gradually shift the focus, reducing entropy weight and enhancing the emphasis on CCA

The overall objective of EverAdapt is formalized as:

$$\mathcal{L}_{\text{Overall}} = \alpha(t)\mathcal{L}_e(z_T) + (1-\alpha(t))\mathcal{L}_{\text{loc}}(z_s, z_T) + \beta\mathcal{L}_{\text{m}}(x_M) + \mathcal{L}_s(x_s, y_s), \tag{13}$$

$$\alpha(t) = \alpha^t \tag{14}$$

where the parameter $\alpha(t)$ is a decay factor which decreases exponentially after each epoch, $t$. $\alpha$ controls the transition between the entropy loss $\mathcal{L}_e$ and the local loss $\mathcal{L}_{\text{loc}}$. A higher value of $\alpha$ means that the model focuses on minimizing the entropy loss for a longer duration before transitioning to the local loss, which helps refine alignment between source and target domain features. The parameter $\beta$ is a scaling factor that governs the contribution of the memory loss $\mathcal{L}_m$ in the overall objective. This term leverages self-training using memory samples to enhance model generalization and stability.

## 4. Experimental settings

### 4.1. Dataset

We validated our method using the Paderborn University (PU) bearing dataset and the University of Ottawa (UO) bearing dataset, which are ideal for testing a CDA setting due to its various working conditions. Details regarding each dataset will be discussed in the next section. Following the approach suggested by Zhao et al. [26], we used data segmentation to increase the size of both dataset and simplify the model's input requirements. Specifically, we applied a moving window technique with a window size and stride length of 1024 to segment the data, ensuring that the resulting data segments are distinct and non-overlapping for model training.

#### 4.1.1. Paderborn University dataset

The Paderborn University dataset [27] contains vibration signals from an electric motor, with a total of 32 sets of signals, each representing a different bearing. Out of these, 6 bearings are healthy, 12 have artificial damage, and 14 have real damage from actual working conditions. Each bearing was tested under four different working conditions. Two dataset, named PU Artificial and PU Real, were created using combining signals from healthy bearings and artificially damaged bearings or bearings with real damage. Both subsets include a combination of healthy and faulty signals, as detailed in a table referred to as Table 1. In these datasets, the type of bearing is used as the class label and the different working conditions under which the bearings were tested are considered as different domains.

#### 4.1.2. University of Ottowa dataset

The University of Ottowa (UO) dataset [28] comprises vibration signals from bearings operating under varying health conditions and rotational speeds. A total of 36 set of signals are included, each corresponding to one of 12 experimental conditions derived from combinations of three bearing health states (healthy, inner race defect, outer race defect) and four rotational speed patterns (increasing speed, decreasing speed, increasing then decreasing speed, and decreasing then increasing speed). For each condition, three trials were conducted to ensure data reliability. In the UO dataset, the state of the bearing's health is used as a class label, and the different rotational speed patterns are considered as separate domains.

### 4.2. Domain scenarios

We present the results of our method based on the average from three different scenarios for each dataset, as detailed in Table 2. This approach enhances the reliability of our results by preventing any bias towards specific scenarios that might favor certain methods.

**Table 1**
PU dataset signal description.

| Bearing | Damage level | Damage type | Location | Damage code | Type |
|---|---|---|---|---|---|
| K001 | 0 | None | N/A | No Damage | Healthy |
| KA01 | 1 | EDM | Outer | O-L1-EDM | Artificial |
| KA03 | 2 | Engraving | Outer | O-L2-Engraving | Artificial |
| KA05 | 1 | Engraving | Outer | O-L1-Engraving | Artificial |
| KA07 | 1 | Drilling | Outer | O-L1-Drilling | Artificial |
| KI01 | 1 | EDM | Outer | O-L1-EDM | Artificial |
| KI03 | 1 | Engraving | Inner | I-L1-Engraving | Artificial |
| KI07 | 2 | Engraving | Inner | I-L2-Engraving | Artificial |
| KA04 | 1 | EDM | Outer | O-L1-EDM | Real |
| KB23 | 2 | Engraving | Inner | I-L2-Engraving | Real |
| KB27 | 1 | Engraving | Outer | O-L1-Engraving | Real |
| KI04 | 1 | Drilling | Inner | I-L1-Drilling | Real |

**Table 2**
Domain sequence used for each dataset.

| Dataset | Scenario | Source | Target 1 | Target 2 | Target 3 |
|---|---|---|---|---|---|
| PU Artificial | 1 | A1 | A2 | A3 | A4 |
| | 2 | A1 | A3 | A2 | A4 |
| | 3 | A1 | A2 | A4 | A3 |
| PU Real | 1 | R1 | R2 | R3 | R4 |
| | 2 | R1 | R3 | R2 | R4 |
| | 3 | R1 | R2 | R4 | R3 |
| UO | 1 | U1 | U2 | U3 | U4 |
| | 2 | U2 | U1 | U3 | U4 |
| | 3 | U4 | U1 | U2 | U3 |

**Table 3**
Four working conditions of PU datasets, A/R denotes domains from PU Artificial and PU Real.

| Domain | Rotating speed (rpm) | Load torque (Nm) | Radial force (N) |
|---|---|---|---|
| A1/R1 | 1500 | 0.7 | 1000 |
| A2/R2 | 900 | 0.7 | 1000 |
| A3/R3 | 1500 | 0.1 | 1000 |
| A4/R4 | 1500 | 0.7 | 400 |

### 4.3. Evaluation metrics

We introduce three key metrics to assess a model's performance when adapting to multiple target domains. The first metric, average Accuracy (ACC), evaluates the model's overall performance across all observed domains. The second metric, average Backward Transfer (BWT), measures how well the model maintains its performance on previously adapted domains. The third metric, average Adaptation (ADAPT), assesses the model's effectiveness in adapting to unseen domains. To calculate these metrics, we introduce $R$ matrix, where each element $R_{i,j}$ signifies the test accuracy on domain $D_j$ after the model has adapted to domain $D_i$. Here, $N$ represents the total number of domains. Using the R matrix, we can calculate our proposed metrics as follows (see Table 3):

$$\text{ACC} = \frac{1}{N}\sum_{i=1}^{N} R_{N,i} \tag{15}$$

$$\text{BWT} = \frac{1}{N-1}\sum_{i=1}^{N-1}(R_{N,i} - R_{i,i}) \tag{16}$$

$$\text{ADAPT} = \frac{1}{N-1}\sum_{i=1}^{N} R_{i,i} \tag{17}$$

### 4.4. Implementation details

To ensure a fair comparison, all models, including EverAdapt, were assessed using a standardized feature encoder and classifier. The feature encoder comprises three convolution blocks, following the structure suggested by [29]. Key components of each block include a 1D convolution layer, batch normalization, a ReLU layer, and a max pooling layer. The first block features a 128-channel CNN layer with a kernel size of 5 and a dropout layer (dropout probability: 0.5). The second block doubles the channels, using a kernel size of 8, while the third block returns to 128 channels, also with a kernel size of 8. An adaptive layer then condenses the outputs to a length, leading to a fully connected classification layer. We conducted a hyperparameter sweep for all methods

to ensure a fair comparison between our method, EverAdapt, and the baseline methods. For each method, we randomly searched 50 sets of hyperparameters and selected the best selection that yielded the best overall accuracy on the validation set. Parameter settings were uniform across methods: a learning rate of $1 \times 10^{-3}$, weight decay of $1 \times 10^{-4}$, 40 epochs, and a batch size of 256. To validate robustness, each model underwent five runs with different random seed values, ensuring the reliability of the performance to seed variation.

## 5. Results and discussions

### 5.1. Baseline methods

To evaluate the performance of our model, we compare it We assessed the efficacy of our EverAdapt technique by comparing it with recent domain adaptation methods proposed for fault diagnosis. We re-implement all the baselines in our framework, while ensuring the same backbone network and training schemes. Overall, the compared methods are as follows:

- Conditional adversarial DA with discrimination embedding (CADA-DE) [30]: utilized a conditional adversarial alignment by integrating task-specific knowledge with the features during the alignment step for the different domains.
- Hierarchical deep domain adaptation (HDDA) [31]: aligns the second-order statistics of the source and target distributions in order to effectively minimize the shift between the two domains.
- Improved Domain Adversarial Neural Network (IDANN) [32]: leverages gradient reversal layer to adversarially train a domain discriminator network against an encoder network.
- Minimum Discrepancy Estimation for Deep Domain Adaptation (MMDA) [33]: combines the MMD and correlation alignment with entropy minimization to effectively address the domain shift issue.
- Subdomain adaptation transfer learning network (SATLN) [16]: leverages gradient reversal layer to adversarially train a domain discriminator network against an encoder network.

In addition to the leading domain adaptation methods, we have assessed EverAdapt against CDA methods proposed in other fields which includes:

- Continuous unsupervised adaptation (CUA) [34]: leverage replay sample loss to address catastrophic forgetting
- Continual Unsupervised Domain Adaptation (CONDA) [12] build upon the work of [35] by incorporating it with sample replay with an appropriate sample replay manager to append new target domain samples with class-representative samples.
- (DCTLN-DWA) [36]: combines techniques from adversarial domain adaptation and replay sample loss, which are selected through a herding algorithm to obtain class-representative samples.

### 5.2. Comparison with baselines

We evaluated EverAdapt's performance against various established domain adaptation methods, utilizing both the PU datasets and UO dataset. The comparative results, presented in Tables 4, 5, and 6, are averaged over three distinct scenarios. We found that EverAdapt demonstrated state-of-the-art performance for all datasets, achieving the highest accuracy and the BWT scores across all three datasets while merely trailing behind in adaptation performance. Specifically,

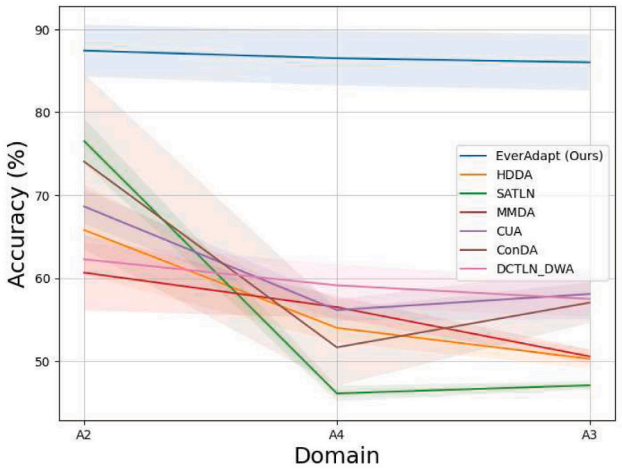
- PU Artificial Dataset: EverAdapt demonstrated superior accuracy, outperforming the best baseline methods by 8.67%. It also led in BWT scores by 1.29%. In terms of adaptation performance, EverAdapt was ahead by 2.17%.
- PU Real Dataset: EverAdapt exceeded the top baseline methods in accuracy by 2.36% and in BWT scores by 0.48%. However, it lagged slightly in adaptation performance, trailing by 0.46%.
- UO Dataset: EverAdapt continued to show excellent performance, surpassing the best baseline methods in accuracy by 0.93% and in BWT scores by 0.33%. In adaptation performance, it was behind by 0.94%.

These results indicate that EverAdapt is highly effective in retaining previously learned knowledge while adapting to new tasks which contrasts the baseline CDA methods such as CUA and DCTLN-DWA which demonstrated remarkable BWT scores at the expense of Adapt performance.

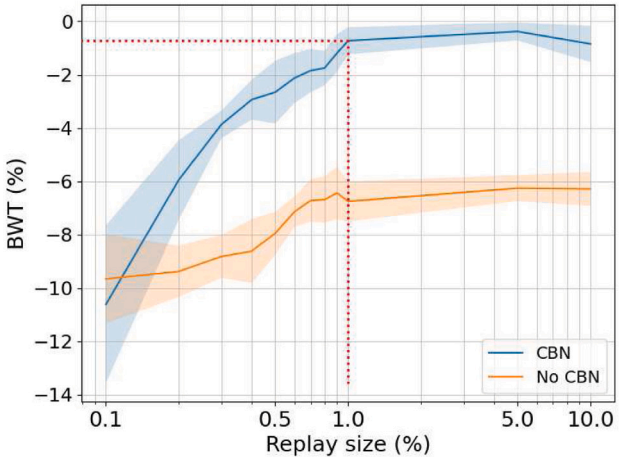
This superiority is further illustrated in Fig. 4(a), which plots the initial target accuracy as the model adapts to various target domains. The plot reveals that our method not only achieves significantly higher initial accuracy, indicating superior adaptation performance but also excels in knowledge retention, as demonstrated by the minimal performance drop compared to other baseline methods.

### 5.3. Model analysis

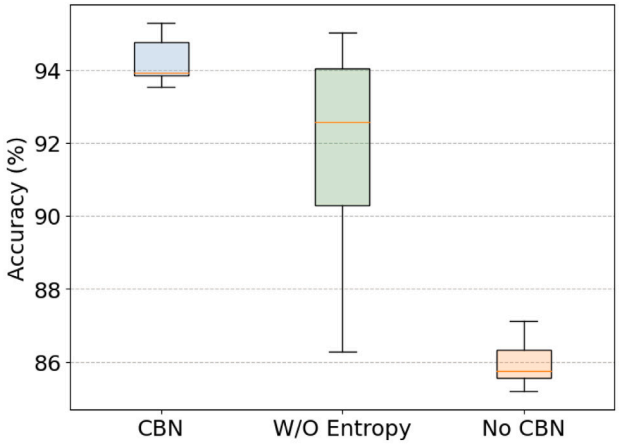
We conducted an extensive analysis to better understand how our model achieves its state-of-the-art performance.




(a) Accuracy of the models on the first target domain A2 as they are updated with new target domains




(b) Performance of EverAdapt with different replay sizes




(c) Stability analysis of EverAdapt

**Fig. 4.** Model analysis for Everadapt.

**Table 4**
Comparative performance of our approach and baseline methods on PU Artificial dataset across three distinct scenarios. Best results are denoted in bold while the second best are underlined.

| Methods | PU Artificial | | |
|---|---|---|---|
| | ACC | BWT | ADAPT |
| CADA-DE [30] | 80.94 ± 0.34 | −5.61 ± 0.42 | 84.67 ± 0.26 |
| IDANN [32] | 77.97 ± 0.92 | −7.01 ± 3.07 | 82.65 ± 2.24 |
| HDDA [31] | 82.18 ± 0.35 | −7.25 ± 1.52 | 87.01 ± 1.15 |
| SATLN [16] | 81.44 ± 0.17 | −14.92 ± 0.93 | 91.38 ± 0.64 |
| MMDA [33] | 82.04 ± 0.98 | −2.91 ± 2.57 | 83.98 ± 1.79 |
| *ConDA* [12] | 75.71 ± 10.26 | −7.12 ± 5.72 | 80.46 ± 12.02 |
| *CUA* [34] | 83.46 ± 1.65 | −2.90 ± 1.42 | 85.39 ± 1.96 |
| *DCTLN-DWA* [36] | 84.14 ± 0.92 | −2.39 ± 0.44 | 85.73 ± 1.05 |
| **EverAdapt** | **92.81 ± 0.39** | **−1.10 ± 0.29** | **93.55 ± 0.88** |

**Table 5**
Comparative performance of our approach and baseline methods on the PU Real dataset across three distinct scenarios. Best results are denoted in bold while the second best are underlined.

| Methods | PU Real | | |
|---|---|---|---|
| | ACC | BWT | ADAPT |
| CADA-DE [30] | 87.63 ± 1.35 | −10.15 ± 1.89 | 94.40 ± 1.12 |
| IDANN [32] | 91.62 ± 0.79 | −6.63 ± 1.52 | 96.04 ± 1.18 |
| HDDA [31] | 89.03 ± 1.56 | −10.75 ± 2.37 | 96.20 ± 1.21 |
| SATLN [16] | 94.14 ± 0.58 | −7.36 ± 0.80 | 99.05 ± 0.16 |
| MMDA [33] | 94.30 ± 0.56 | −3.72 ± 1.28 | 96.78 ± 0.77 |
| *ConDA* [12] | 95.73 ± 1.20 | −5.54 ± 1.85 | **99.42 ± 0.17** |
| *CUA* [34] | 96.69 ± 1.04 | −0.34 ± 0.80 | 96.92 ± 0.71 |
| *DCTLN-DWA* [36] | 93.77 ± 0.85 | −1.40 ± 0.99 | 94.70 ± 0.80 |
| **EverAdapt** | **99.05 ± 0.36** | **0.14 ± 0.31** | 98.96 ± 0.27 |

**Table 6**
Comparative performance of our approach and baseline methods on the UO dataset across three distinct scenarios. Best results are denoted in bold while the second best are underlined.

| Methods | UO | | |
|---|---|---|---|
| | ACC | BWT | ADAPT |
| CADA-DE [30] | 78.37 ± 3.03 | −4.55 ± 3.31 | 81.40 ± 3.49 |
| IDANN [32] | 84.68 ± 3.41 | −2.46 ± 4.96 | **86.32 ± 1.53** |
| HDDA [31] | 80.38 ± 2.86 | −5.26 ± 5.10 | 83.89 ± 4.05 |
| SATLN [16] | 81.98 ± 2.47 | −5.60 ± 6.60 | 85.71 ± 4.46 |
| MMDA [33] | 79.41 ± 5.48 | −1.26 ± 5.49 | 80.26 ± 3.91 |
| *ConDA* [12] | 62.98 ± 7.92 | −1.91 ± 6.88 | 64.26 ± 9.07 |
| *CUA* [34] | 78.97 ± 9.85 | −1.95 ± 2.88 | 80.27 ± 8.83 |
| *DCTLN-DWA* [36] | 83.18 ± 2.77 | 0.01 ± 3.32 | 83.17 ± 3.28 |
| **EverAdapt** | **85.61 ± 4.49** | **0.34 ± 0.99** | 85.38 ± 4.19 |

### *5.3.1. Ablation study*

An ablation study was conducted across three distinct scenarios to assess the efficacy of each component in the EverAdapt model, with results indicating consistent performance improvements in all scenarios. For each scenario, detailed findings are presented in Table 7. Initially, the class-conditional alignment exhibited adaptation capabilities but was inadequate in countering catastrophic forgetting. The addition of replay samples improved knowledge retention, enhancing overall BWT by 7.29% but slightly reduced adaptation performance by 1.57%. The integration of CBN significantly boosted both memory retention, with a 7.73% improvement in BWT, and adaptation performance, improving by 3.16%. This advancement not only mitigated the initial dip in adaptation performance but also surpassed the performance of the model with only class-conditional alignment by 1.59%. Here is the rotated version of your table in LaTeX format:

### *5.3.2. Replay samples efficiency*

This study investigates Continual Batch Normalization (CBN)'s role in addressing catastrophic forgetting, focusing on the use of minimal replay sample sizes. In scenario 3 of the PU Artificial dataset, we assessed the effectiveness of preserving merely 1% of data from each target domain. As illustrated in Fig. 4(b), our findings demonstrate CBN's substantial contribution to reinforcing replay sample utility. With just a 1% replay sample size, CBN notably enhances Backward Transfer (BWT) by nearly 7%, markedly reducing forgetting to 0.73%. Furthermore, augmenting the replay size to 10% while incorporating CBN yields only a slight BWT increment of 0.1%. This suggests that small replay samples, in conjunction with CBN, effectively combat the catastrophic forgetting challenge.

**Table 7**
Ablation study of EverAdapt. 1% was used as replay size. Best results are denoted in bold while the second best are underlined.

| PU Artificial | | | Scenario 1 | | |
|---|---|---|---|---|---|
| CC | Replay | CBN | ACC (%) | BWT (%) | ADAPT (%) |
| ✓ | | | 81.14 ± 0.22 | −15.65 ± 2.79 | 91.76 ± 1.92 |
| ✓ | ✓ | | 85.27 ± 1.23 | −9.12 ± 1.86 | 91.87 ± 2.08 |
| ✓ | ✓ | ✓ | **93.11 ± 2.31** | **−1.68 ± 0.64** | **94.69 ± 2.08** |
| | | | Scenario 2 | | |
| CC | Replay | CBN | ACC (%) | BWT (%) | ADAPT (%) |
| ✓ | | | 81.35 ± 0.23 | −18.84 ± 1.37 | **94.23 ± 0.95** |
| ✓ | ✓ | | 83.20 ± 0.32 | −9.19 ± 1.13 | 89.74 ± 0.82 |
| ✓ | ✓ | ✓ | **91.56 ± 0.41** | **−1.04 ± 0.36** | 92.88 ± 0.35 |
| | | | Scenario 3 | | |
| CC | Replay | CBN | ACC (%) | BWT (%) | ADAPT (%) |
| ✓ | | | 82.08 ± 0.36 | −14.06 ± 2.63 | 91.48 ± 1.84 |
| ✓ | ✓ | | 85.36 ± 0.98 | −8.37 ± 2.07 | 91.16 ± 2.00 |
| ✓ | ✓ | ✓ | **94.07 ± 0.98** | **−0.73 ± 0.51** | **94.67 ± 1.01** |

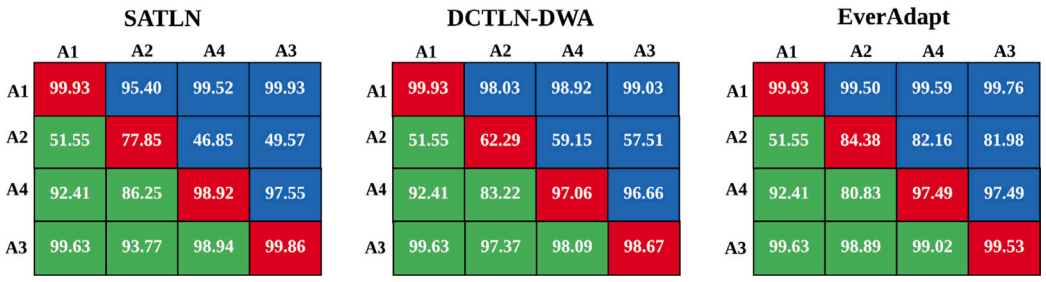



**Fig. 5.** R matrix comparison of EverAdapt with two baseline methods on the PU Artificial dataset (Scenario 3). black represents previously seen domains, green indicates unseen domains, and red marks the current domain of adaptation.

### *5.3.3. Stability study*

This section presents a stability study of the Continual Batch Normalization (CBN) module within the EverAdapt framework. Focusing on the PU Artificial dataset's scenario 3, we evaluated the significance of individual CBN components in stabilizing the model. Fig. 4(c) illustrates the performance comparisons between the full implementation of EverAdapt, a variant employing only source statistics normalization without entropy, and another variant excluding CBN entirely. The results affirm the full CBN model's superior performance, indicating the drawbacks of omitting certain components. Specifically, while normalizing target samples with source statistics improved median accuracy by 6.83%, it also introduced greater variability, evidenced by a fourfold increase in the range of performance outcomes. Integrating entropy, alongside source statistics normalization, significantly counteracted this variability. This emphasizes the critical roles of both entropy incorporation and source normalization in CBN, enhancing not only the model's performance but also its stability under dynamic environments.

### *5.3.4. Adaptation and forgetting analysis*

This section analyzes the R matrix shown in Fig. 5, which evaluates model performance during continual adaptation under scenario 3 on the PU Artificial dataset. Each row represents a model's performance on a specific domain before adaptation, while each column shows performance after adapting to another domain. For example, the value in row A4, column A2, indicates the model's performance on domain A2 after adapting to domain A4. The analysis highlights challenges such as catastrophic forgetting in SATLN, where performance on A2 drops after adapting to A4, and poor adaptation in DCTLN-DWA, where performance on the target domain remains low. In contrast, EverAdapt demonstrates strong adaptability while preventing catastrophic forgetting, outperforming the baseline methods.

### *5.3.5. Computational analysis*

EverAdapt, powered by Continual Batch Normalization (CBN), is designed for computational efficiency, making it suitable for real-time industrial deployment. By operating directly on batch statistics, CBN minimizes overhead and integrates seamlessly into existing feature extraction pipelines. Table 8 compares the training time per epoch of CBN, the core component of EverAdapt, with baseline methods. While DANN and DeepCORAL are more computationally efficient ($1.68 \pm 0.01$ and $1.66 \pm 0.01$ s per epoch, respectively), they are not designed to handle continual adaptation tasks. Among the methods capable of continual adaptation, EverAdapt achieves the best balance of efficiency and performance, with a training time of $2.82 \pm 0.04$ s per epoch, outperforming DCTLN_DWA ($5.62 \pm 0.03$ s per epoch) and CUA ($4.17 \pm 0.02$ s per epoch).

**Table 8**
Comparison of training time per epoch for EverAdapt (CBN) and baseline methods.

| Method | EverAdapt (CBN) | DCTLN_DWA | CUA | ConDA | DANN | DeepCORAL |
|---|---|---|---|---|---|---|
| Performance | $2.82 \pm 0.04$ | $5.62 \pm 0.03$ | $4.17 \pm 0.02$ | $4.31 \pm 0.01$ | $1.68 \pm 0.01$ | $1.66 \pm 0.01$ |

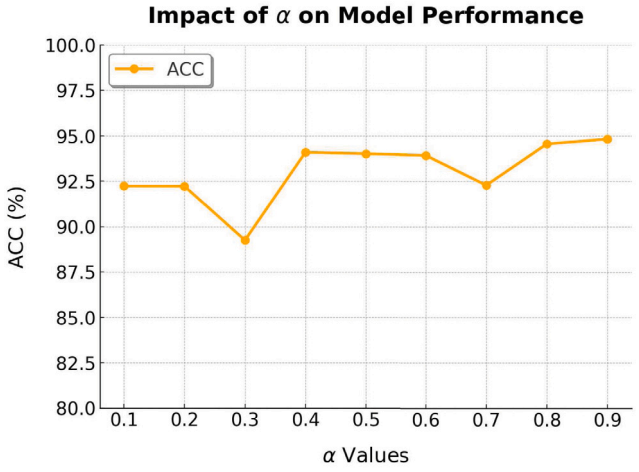



**Fig. 6.** Sensitivity analysis of EverAdapt's performance to the hyperparameter $\alpha$.

*5.3.6. Sensitivity analysis*

In this subsection, we analyze the sensitivity of EverAdapt's performance to the hyperparameter $\alpha$, which is used in the adaptive weighting strategy for balancing losses. To evaluate the robustness of our method, we varied $\alpha$ over a range of values from 0.1 to 0.9 and measured the corresponding performance. The results, presented in Fig. 6, indicate that EverAdapt achieves consistently high ACC values, ranging from 89.25% to 94.83%. While there is a slight dip in performance at $\alpha = 0.3$, the ACC values recover quickly at higher $\alpha$ values. This demonstrates the robustness of EverAdapt to the choice of $\alpha$, as the performance remains relatively stable across the evaluated range.

## 6. Conclusion

In this paper, we introduced EverAdapt, a novel approach for continual unsupervised domain adaptation in machine fault diagnosis. At the core of EverAdapt is the Continual Batch Normalization (CBN) technique, which preserves domain-specific batch statistics to mitigate catastrophic forgetting and reduce reliance on replay samples. This innovative module ensures robust adaptation across dynamic environments, addressing key challenges in continual domain adaptation. Leveraging CBN, EverAdapt achieves the best accuracy across three different datasets while also demonstrating state-of-the-art performance in mitigating forgetting, with a forgetting rate of −1.10% on the PU Artificial dataset. Additionally, it achieves positive transfer of 0.14% and 0.34% on the PU Real and UO datasets, respectively, showcasing its unparalleled effectiveness in both accuracy and forgetting mitigation. Despite its promise, EverAdapt has limitations. First, it relies on source data and a replay buffer, which may not suit privacy-sensitive settings, and struggles with extreme domain shifts that involve complex feature variations beyond what batch statistics can capture. Future work could explore source-free continual adaptation, improved handling of non-stationary data, and introducing intermediate domains to address high heterogeneity, further enhancing EverAdapt's utility in real-world applications.

## CRediT authorship contribution statement

**Edward:** Writing – original draft, Visualization, Validation, Data curation. **Mohamed Ragab:** Software, Methodology, Investigation, Data curation, Conceptualization. **Min Wu:** Writing – review & editing, Supervision, Project administration. **Yuecong Xu:** Writing – review & editing, Writing – original draft, Visualization, Methodology, Conceptualization. **Zhenghua Chen:** Supervision, Project administration. **Abdulla Alseiari:** Writing – review & editing. **Xiaoli Li:** Supervision, Project administration.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Mohamed Ragab reports financial support was provided by AI Singapore. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

This work is supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under its AI-Singapore Grant (Grant No. AI-SG2-RP2021-2027).

## Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ymssp.2025.112317.

## Data availability

Data will be made available on request.

## References

[1] Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, A.K. Nandi, Applications of machine learning to machine fault diagnosis: A review and roadmap, Mech. Syst. Signal Process. 138 (2020) 106587.
[2] Z. Zhu, Y. Lei, G. Qi, Y. Chai, N. Mazur, Y. An, X. Huang, A review of the application of deep learning in intelligent fault diagnosis of rotating machinery, Measurement 206 (2023) 112346.
[3] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. (2010) http://dx.doi.org/10.1109/tkde.2009.191.
[4] B. Yang, Y. Lei, F. Jia, S. Xing, An intelligent fault diagnosis approach based on transfer learning from laboratory bearings to locomotive bearings, Mech. Syst. Signal Process. 122 (2019) 692–706.
[5] Y. Xiao, X. Zhou, H. Zhou, J. Wang, Multi-label deep transfer learning method for coupling fault diagnosis, Mech. Syst. Signal Process. 212 (2024) 111327.
[6] J. Jiao, H. Li, T. Zhang, J. Lin, Source-free adaptation diagnosis for rotating machinery, IEEE Trans. Ind. Inform. 19 (9) (2022) 9586–9595.
[7] Q. Li, B. Tang, L. Deng, P. Zhu, Source-free domain adaptation framework for fault diagnosis of rotation machinery under data privacy, Reliab. Eng. Syst. Saf. 238 (2023) 109468.
[8] J. Jiao, H. Li, Inter-to intradomain: A progressive adaptation method for machine fault diagnosis, IEEE Trans. Ind. Inform. (2023).
[9] X. Zheng, Z. He, J. Nie, P. Li, Z. Dong, M. Gao, A progressive multi-source domain adaptation method for bearing fault diagnosis, Appl. Acoust. 216 (2024) 109797.
[10] G.I. Parisi, R. Kemker, J.L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Netw. 113 (2019) 54–71.
[11] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A.A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proc. Natl. Acad. Sci. 114 (13) (2017) 3521–3526, http://dx.doi.org/10.1073/pnas.1611835114, arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.1611835114. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.
[12] A.M.N. Taufique, C.S. Jahan, A. Savakis, Continual unsupervised domain adaptation in data-constrained environments, IEEE Trans. Artif. Intell. 5 (1) (2023) 167–178.
[13] S. Tang, P. Su, D. Chen, W. Ouyang, Gradient regularized contrastive learning for continual domain adaptation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 2665–2673.
[14] W. Lu, B. Liang, Y. Cheng, D. Meng, J. Yang, T. Zhang, Deep model based domain adaptation for fault diagnosis, IEEE Trans. Ind. Electron. 64 (3) (2016) 2296–2305.
[15] X. Li, W. Zhang, N.-X. Xu, Q. Ding, Deep learning-based machinery fault diagnostics with domain adaptation across sensors at different places, IEEE Trans. Ind. Electron. 67 (8) (2019) 6785–6794.
[16] Z. Wang, X. He, B. Yang, N. Li, Subdomain adaptation transfer learning network for fault diagnosis of roller bearings, IEEE Trans. Ind. Electron. 69 (8) (2021) 8430–8439.
[17] J. Zhu, N. Chen, C. Shen, A new multiple source domain adaptation fault diagnosis method between different rotating machines, IEEE Trans. Ind. Inform. 17 (7) (2020) 4788–4797.
[18] A. Bobu, E. Tzeng, J. Hoffman, T. Darrell, Adapting to continuously shifting domains, 2018, URL: https://openreview.net/forum?id=BJsBjPJvf.
[19] S. Rakshit, A. Mohanty, R. Chavhan, B. Banerjee, G. Roig, S. Chaudhuri, FRIDA - generative feature replay for incremental domain adaptation, Comput. Vis. Image Underst. 217 (2021) 103367, URL: https://api.semanticscholar.org/CorpusID:245537652.
[20] S. Yang, Y. Wang, J. van de Weijer, L. Herranz, S. Jui, Generalized source-free domain adaptation, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 8958–8967, URL: https://api.semanticscholar.org/CorpusID:236881316.
[21] A. Mallya, S. Lazebnik, PackNet: Adding multiple tasks to a single network by iterative pruning, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017, pp. 7765–7773, URL: https://api.semanticscholar.org/CorpusID:35249701.
[22] S. Niu, J. Wu, Y. Zhang, Y. Chen, S.D. Zheng, P. Zhao, M. Tan, Efficient test-time model adaptation without forgetting, in: International Conference on Machine Learning, 2022, URL: https://api.semanticscholar.org/CorpusID:247996873.
[23] Y. Zhu, F. Zhuang, J. Wang, G. Ke, J. Chen, J. Bian, H. Xiong, Q. He, Deep subdomain adaptation network for image classification, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2020) 1713–1722.
[24] X. Wu, Q. Zhou, Z. Yang, C. Zhao, L.J. Latecki, et al., Entropy minimization vs. diversity maximization for domain adaptation, 2020, arXiv preprint arXiv:2002.01690.
[25] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, C.H. Lampert, Icarl: Incremental classifier and representation learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2001–2010.
[26] Z. Zhao, T. Li, J. Wu, C. Sun, S. Wang, R. Yan, X. Chen, Deep learning algorithms for rotating machinery intelligent diagnosis: An open source benchmark study, ISA Trans. 107 (2020) 224–255.
[27] C. Lessmeier, J.K. Kimotho, D. Zimmer, 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: PHM Society European Conference, Vol. 3, 2016.
[28] H. Huang, N. Baddour, Bearing vibration data collected under time-varying rotational speed conditions, Data Brief 21 (2018) 1745–1749.
[29] M. Ragab, E. Eldele, W.L. Tan, C.-S. Foo, Z. Chen, M. Wu, C.-K. Kwoh, X. Li, Adatime: A benchmarking suite for domain adaptation on time series data, ACM Trans. Knowl. Discov. Data 17 (8) (2023) 1–18.
[30] X. Yu, Z. Zhao, X. Zhang, C. Sun, B. Gong, R. Yan, X. Chen, Conditional adversarial domain adaptation with discrimination embedding for locomotive fault diagnosis, IEEE Trans. Instrum. Meas. 70 (2020) 1–12.
[31] X. Wang, H. He, L. Li, A hierarchical deep domain adaptation approach for fault diagnosis of power plant thermal system, IEEE Trans. Ind. Inform. 15 (9) (2019) 5139–5148.

[32] D. Zhang, L. Zhang, A multi-feature fusion-based domain adversarial neural network for fault diagnosis of rotating machinery, Measurement 200 (2022) 111576.
[33] M. Azamfar, X. Li, J. Lee, Deep learning-based domain adaptation method for fault diagnosis in semiconductor manufacturing, IEEE Trans. Semicond. Manuf. 33 (3) (2020) 445–453.
[34] A. Bobu, E. Tzeng, J. Hoffman, T. Darrell, Adapting to continuously shifting domains, 2018.
[35] J. Liang, D. Hu, J. Feng, Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation, in: International Conference on Machine Learning, PMLR, 2020, pp. 6028–6039.
[36] J. Li, R. Huang, Z. Chen, G. He, K.C. Gryllias, W. Li, Deep continual transfer learning with dynamic weight aggregation for fault diagnosis of industrial streaming data under varying working conditions, Adv. Eng. Inform. 55 (2023) 101883.