Intelligent Fault Diagnosis by Fusing Domain Adversarial Training and Maximum Mean Discrepancy via Ensemble Learning

Yibin Li, Yan Song, Lei Jia, Shengyao Gao, Qiqiang Li, and Meikang Qiu, *Senior Member, IEEE*

***Abstract*—Nowadays, the industrial Internet of Things (IIoT) has been successfully utilized in smart manufacturing. The massive amount of data in IIoT promote the development of deep learning-based health monitoring for industrial equipment. Since monitoring data for mechanical fault diagnosis collected on different working conditions or equipment have domain mismatch, models trained with training data may not work in practical applications. Therefore, it is essential to study fault diagnosis methods with domain adaptation ability. In this article, we propose an intelligent fault diagnosis method based on an improved domain adaptation method. Specifically, two feature extractors concerning feature space distance and domain mismatch are trained using maximum mean discrepancy and domain adversarial training respectively to enhance feature representation. Since separate classifiers are trained for feature extractors, ensemble learning is further utilized to obtain final results. Experimental results indicate that the proposed method is effective and applicable in diagnosing faults with domain mismatch.**

***Index Terms*—Domain adaptation, domain adversarial training (DAT), ensemble learning, fault diagnosis, maximum mean discrepancy (MMD).**

I. Introduction

NOWADAYS the industrial Internet of Things (IIoT) has been successfully utilized in smart manufacturing. Many studies have been carried out for processing massive data from the manufacturing facilities in the workshop [1], [2], which is of great significance for promoting the development of data-driven prognostic and health management (PHM) systems. As an important part of PHM, fault diagnosis is essential in ensuring manufacturing security. While machine learning algorithms have difficulties in processing monitoring data with increased dimensions and dynamics, deep learning algorithms which can solve more complicated problems and produce high accurate results have been widely utilized in mechanical fault diagnosis.

Deep learning algorithms like deep belief network (DBN) and convolutional neural network (CNN) have outstanding performance in fault diagnosis and health monitoring. DBN is a kind of artificial network, which can be optimized layer by layer. Method proposed in [3] proposed to integrate sparse encoder and DBN for fault diagnosis. To explore the relationship of various inputs, Li *et al.* [4] proposed a backlash error prediction method based on DBN. Zhang *et al.* [5] *et al.* used DBN and multisensor data fusion to detect the degradation process of ball screw. The dataset included multiple degradation stages under several different working conditions by three accelerators. After Fourier transformation, the spectrum of three sensors was utilized as the input of DBN to train the network. In [6], with features extracted by an autoencoder, CNN was trained with annotated vibration data for fault diagnosis. Furthermore, to realize end-to-end fault diagnosis without manual operation, Zhang *et al.* [7] used 1-D sequential signal as input to CNN model, and proved that CNN has strong domain adaptation ability and generalization performance.

Most deep learning algorithms require massive training data, and assume that the training dataset and the data to be tested are related to the same distribution. However, due to the complexity of the on-site industrial environment, the working conditions (such as mechanical load and speed) or the equipment to be tested may change. As a result, the well-trained models for fault diagnosis may have low accuracy in practical applications because of the domain mismatch between the training dataset and the testing dataset. For example, in order to train a model for rolling bearing defect identification, data collected under no motor load condition can be used as the training samples, and the trained model may be used to classify the bearing defects under different motor load conditions in real applications. Although, the training data and the testing data share the same defects categories, the data distributions between them are different.

Manuscript received April 5, 2020; revised June 19, 2020 and June 23, 2020; accepted July 4, 2020. Date of publication July 8, 2020; date of current version January 4, 2021. This work was supported by the National Natural Science Foundation of China under Grant 41906164. Paper no. TII-20-1693. *(Corresponding authors: Yan Song; Lei Jia.)*

Yibin Li and Yan Song are with the Institute of Marine Science and Technology, Shandong University, Shandong 266237, China (e-mail: liyibing@sdu.edu.cn; ysong@sdu.edu.cn).

Lei Jia and Qiqiang Li are with the School of Control Sciences and Engineering, Shandong University, Jinan 250061, China (e-mail: jialei@sdu.edu.cn; qqli@sdu.edu.cn).

Shengyao Gao is with the China Naval Academy, Beijing 100161, China (e-mail: shengyaogao@163.com).

Meikang Qiu is with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: qiumeikang@yahoo.com).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2020.3008010

That is to say, the classification model trained by the training samples (source domain) does not fit the testing data (target domain) exactly. On the other hand, because the set of possible working conditions for different bearings is nearly infinite, creating extensive enough training data of representative faults is very challenging.

In this context, advanced deep learning algorithms like transfer learning [8] and domain adaptation [9] have emerged and developed. Methods introduced in [10]–[14] concern both source and target data to learn a feature extraction network for both domains through using corresponding loss functions like maximum mean discrepancy (MMD) [15] and correlation alignment (CORAL) [14]. Methods utilize architectures with shared weights to learn invariant features for both source and target domains, and train a classifier with annotated training dataset. The well-trained feature extractor and classifier will be used to predict labels for fault data in target domain.

While the abovementioned domain adaptation based methods consider invariant domain features, methods projecting data into the same feature space will lose some important feature representation if source domain and target domain have large difference. To this end, we present an intelligent fault diagnosis method based on an improved domain adaptation architecture in this article. Specifically, two feature extractors are used to project data into different feature space. One feature extractor learns features based on domain adversarial learning, whereas the other one learns through considering MMD as loss function. In short, corresponding loss functions are used for different feature extractors. Furthermore, since separate classifiers are utilized for the two feature extractors, respectively, ensemble learning is used to obtain the final results. Effectiveness of the proposed method is demonstrated on a widely used benchmark for fault diagnosis. It is shown that the proposed method is suitable for diagnosing faults with different working conditions, and performs satisfying when it is trained with artificial data and tested on real fault data. Moreover, given that the pseudolabels can facilitate domain adaptation [16], they are utilized for features learning and model performance improvement. Experimental results show that the proposed method outperforms the state-of-the-art methods.

The main contributions of this article are summarized as follows.

1) In order to obtain various features, the proposed method uses MMD and domain adversarial training (DAT) to project source domain data and target domain data toward similar feature space respectively.
2) Ensemble learning is introduced to obtain classification results since two classifiers are used for the different feature extractors.
3) An improved domain adaptation architecture is proposed by fusing MMD, domain adversarial learning and ensemble learning to obtain accurate fault diagnosis results.

The rest of the article is organized as follows. Section II reviews related work. Section III introduces preliminaries. Section IV details the proposed approach for fault diagnosis. Section V presents and analyzes the experimental results. Finally, Section VI concludes this article.

II. Related Work

Incorporating advanced deep learning method such as transfer learning and domain adaptation into IIoT applications provides performance improvement. Zhang *et al.* [17] applied neural networks for fault diagnosis based on transfer learning. They used massive source data to train a neural network, after which some parameters are transferred and trained by a small amount of target data. However, this method does not work when the labeled target data is not available. Unsupervised domain adaptation is proposed to address tasks with labeled training data on source domain and unlabeled data on target domain. Considering the feature space discrepancy between the two domains, methods such as MMD, CORAL, and Wasserstein distance are usually embedded in the loss function to extract domain-invariant features. In [13] and [18], MMD was integrated with transfer learning to train models with domain adaptation and less feature discrepancy. While MMD has high computation complexity when the number of samples is large, Wasserstein distance was used in [10] with domain adaptation for fault diagnosis. To learn invariant features furtherly, Guo *et al.* [11] proposed a method to train a feature extractor through maximizing domain classification error and MMD distance simultaneously.

Methods proposed in [10], [13], [18] minimized discrepancy between features from source domain and target domain, but they may lose discriminative features of the target domain. To this end, methods proposed in [14], [19] carried out cross-domain fault diagnosis through reducing distribution discrepancy and preserving discriminative information. In particular, considering that data in the target domain was not available, the method proposed in [12] provided reliable cross-domain diagnosis by artificially generating fake samples for domain adaptation.

The differences between our work and the relative methods published in [11], [13], and [18] are mainly shown as follows.

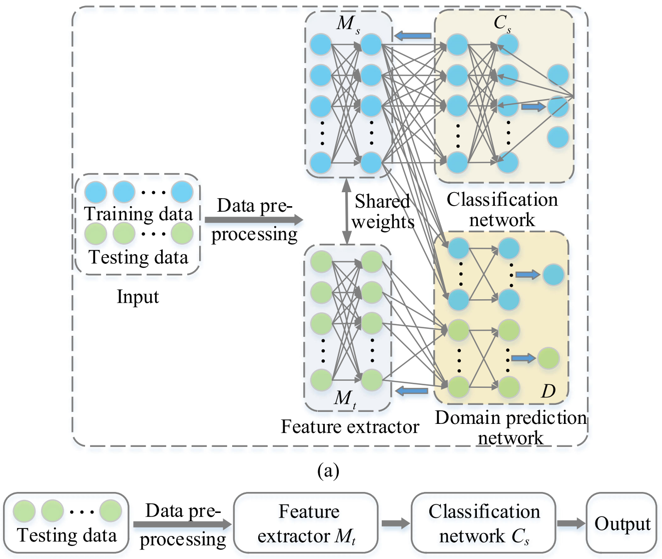
1) Our method uses two feature extractors to learn invariant features using MMD and DAT, respectively, which provides more invariant features than methods in [11], [13], and [18].
2) Our work employs one classifier for each feature extractor, which ensures the independence of the parameters learning process of two feature extractors.
3) Our work integrates invariant features learned using MMD and DAT, and ensemble of two classifiers for fault diagnosis.

III. Preliminaries

A. Problem Formulation

Our work is carried out according to the following assumptions.

1) The fault diagnosis tasks are the same for different domains, i.e., the labels of the source domain data and target domain data are shared.
2) The source and target domains are related to each other, but have different distributions due to different operating conditions.



(a)

(b)

Fig. 1. (a) Framework of DAT. (b) Testing data prediction using the well-trained model.

3) Labeled data from the source domain and unlabeled data from the target domain are available for training.
4) Unlabeled data from the target domain is available during testing.

Let X be the input space and $Y= \{1, 2 \cdots, N\}$ be the set of N health conditions. The n_s labeled samples from the source domain are $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$, and $D_t = \{(x_i^t)\}_{i=1}^{n_t}$ denotes n_t unlabeled samples from the target domain. D_s and D_t are sampled from joint distributions $P(X, Y)$ and $Q(X, Y)$ $(P \neq Q)$, respectively. This article aims to design a deep neural network $\mathbf{y} = f(\mathbf{x})$, which can reduce the cross-domain shifts in joint distributions and learn domain-invariant features and classifiers, and minimize the target risk $R_t(f) = \Pr_{(\mathbf{x},\mathbf{y})\sim Q}[f(\mathbf{x}) \neq \mathbf{y}]$ with source supervision.

B. Domain Adversarial Training

DAT [20] plays an important role in domain adaptation. The framework of DAT is shown in Fig. 1(a). Let the training data be $\mathbf{X}_s$, the training label be $\mathbf{Y}_s$, the testing data be $\mathbf{X}_t$, the feature extractor for the training data, and the testing data be M_s and M_t, respectively, the classification network be C_s, and the domain prediction network be D. During training process, the training data and the testing data are first input into the feature extractor network. Then, the feature $M_s(\mathbf{X}_s)$ for training data are used as input of the classification network. The following classification loss is minimized to train parameters in M_s and C_s using back propagation algorithm

$$\underset{M_s,C_s}{\text{loss}}(\mathbf{X}_s, \mathbf{Y}_s) = - \sum_{\mathbf{x}_s \in \mathbf{X}_s, y_s \in \mathbf{Y}_s} \sum_{k=1}^{N} t_{k,y_s} \cdot \log(C_s(M_s(\mathbf{x}_s))) \tag{1}$$

where N is the number of classes, and $t_{k,y_s} = 1$ if $k = y_s$, $C_s(M_s(\mathbf{x}_s))$ denotes the correct probability of classification. Then, the parameters of M_s are used to initialize M_t.

The domain prediction network D and feature extractors M_s, M_t are trained based on adversarial training for feature space alignment. First, the domain prediction network D aims to differentiate features of target domain from that of the source domain, and classify $M_s(\mathbf{X}_s)$ and $M_t(\mathbf{X}_t)$ into two classes. Let the label for features of the source domain data be 1, while that of the target domain data be 0. The loss function of the domain prediction network is defined as

$$\begin{aligned}&\underset{D,M_s,M_t}{\text{loss}}(\mathbf{X}_s, \mathbf{X}_t)\\ &= - \sum_{\mathbf{x}_s \in \mathbf{X}_s} \log(D(M_s(\mathbf{x}_s))) - \sum_{\mathbf{x}_t \in \mathbf{X}_t} \log(1 - D(M_t(\mathbf{x}_t))).\end{aligned} \tag{2}$$

During adversarial training, M_t is considered as a generator using target domain data as input. It is hoped that the data generated by $M_t(\mathbf{X}_t)$ can fool the domain prediction network. That is to say, the domain prediction network considers $M_t(\mathbf{X}_t)$ as the features of the source domain with label 1. Therefore, M_t is trained through minimizing the following loss function

$$\underset{D,M_t}{\text{loss}}(\mathbf{X}_t) = - \sum_{\mathbf{x}_t \in \mathbf{X}_t} \log(D(M_t(\mathbf{x}_t))). \tag{3}$$

After training domain adaptation network based on DAT, prediction of the testing data can be obtained using feature extractor M_t and the classification network C_s, as shown in Fig. 1(b).

In DAT, source domain data and target domain data are projected into the same subspace. Feature extractor outputs the invariant features for two domains. In most cases, DAT is effective and applicable to obtain adaptation models. However, if there is a large discrepancy between the two domains, it may lose some representative features when only one feature model is considered. Here, we introduce an effective structure to model different feature representation between the source and target data, and integrate it into ensemble learning based domain adaptation.

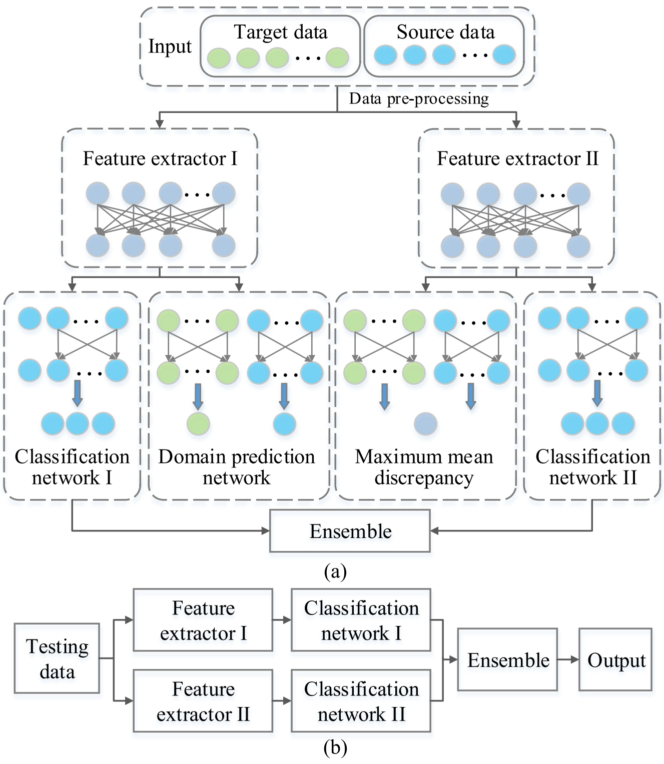
IV. Proposed Method

Framework of the proposed method is shown in Fig. 2. Details of the proposed method are introduced as follows.

A. Data Preprocessing

Data preprocessing involves data partition and data normalization. The sequence data can be collected under 48 kHz or 64 kHz. To obtain data suitable for deep learning method, the raw data should be partitioned into sequence of the same size w. After data partition, let the training dataset be $\mathbf{X}_s = \{\mathbf{x}_{s,1}^T, \mathbf{x}_{s,2}^T, \ldots, \mathbf{x}_{s,n_1}^T\}$, the testing data be $\mathbf{X}_t = \{\mathbf{x}_{t,1}^T, \mathbf{x}_{t,2}^T, \ldots, \mathbf{x}_{t,n_2}^T\}$, where n_1 and n_2 denote the number of samples in $\mathbf{X}_s$ and $\mathbf{X}_t$. Data normalization is realized as follows

$$\bar{\mathbf{x}}_{s,i} = \frac{\mathbf{x}_{s,i} - \text{mean}(\mathbf{x}_{s,i})}{\text{std}(\mathbf{x}_{s,i})}, (i = 1, 2, \ldots, n_1) \tag{4}$$



(a)

(b)

Fig. 2. (a) Framework of the proposed method. (b) Fault diagnosis using the well-trained model for the proposed method.

$$\bar{\mathbf{x}}_{t,j} = \frac{\mathbf{x}_{t,j} - \mathrm{mean}(\mathbf{x}_{t,j})}{\mathrm{std}(\mathbf{x}_{t,j})}, (j = 1, 2, \ldots, n_2) \tag{5}$$

where $\mathrm{mean}(\mathbf{x}_{s,i})$ denotes the mean value of $\mathbf{x}_{s,i}$, $\mathrm{std}(\mathbf{x}_{s,i})$ denotes the standard derivation of $\mathbf{x}_{s,i}$.

B. Training Feature Representation Based on DAT

In this procedure, as shown in Fig. 2(a), one feature extractor (feature extractor I), one classifier (classification network I), and a domain prediction network are combined and trained based on DAT. Let the training dataset be $\bar{\mathbf{X}}_s$, the training label be $\mathbf{Y}_s$, the testing dataset be $\bar{\mathbf{X}}_t$, the feature extractor I be M_I, the classification network I be C_I, and the domain prediction network be D. $M_I(\bar{\mathbf{X}}_s)$ and $M_I(\bar{\mathbf{X}}_t)$ are the feature representation for $\bar{\mathbf{X}}_s$ and $\bar{\mathbf{X}}_t$. To train the domain prediction network, with the source domain data $\bar{\mathbf{X}}_s$ labeled as 1 and the target domain data $\bar{\mathbf{X}}_t$ labeled as 0, the loss function of the domain prediction network is defined as

$$\begin{aligned}\underset{D,M_I}{\mathrm{loss}}(\mathbf{x}_i) = &-\frac{1}{n}\sum_{j=1}^{n}[y_j \log(D(M_I(\mathbf{x}_i))) \\ &+ (1-y_j)\log(1 - D(M_I(\mathbf{x}_i)))]\end{aligned} \tag{6}$$

where n is the number of samples in the batch, y_j is the domain label with 0 for the source data and 1 for the target data, and $\mathbf{x}_i$ denotes data from the source domain or the target domain.

Adversarial training is used to train M_I and D for optimal parameters. M_I is trained through minimizing the following loss function

$$\underset{D,M_I}{\mathrm{loss}}(\bar{\mathbf{X}}_t) = -\sum_{\bar{\mathbf{x}}_t \in \bar{\mathbf{X}}_t} \log(D(M_I(\bar{\mathbf{x}}_t))). \tag{7}$$

Then, the following classification loss function is used to train parameters in M_I and C_I:

$$\begin{aligned}\underset{M_I,C_I}{\mathrm{loss}}(\bar{\mathbf{X}}_s, \mathbf{Y}_s) = &-\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N} t_{k,y_{s,j}} \\ &\times \log\frac{\exp[(\mathbf{w}_{I,k})^T M_I(\bar{\mathbf{x}}_{s,j}) + b_I]}{\sum_{l=1}^{N}[\exp((\mathbf{w}_{I,l})^T M_I(\bar{\mathbf{x}}_{s,j}) + b_I)]}\end{aligned} \tag{8}$$

where N is the number of fault classes, $\mathbf{w}_{I,k}$ and b_I are the parameters in classifier C_I, and $t_{k,y_{s,j}} = 1$ if $k = y_{s,j}$, $M_I(\bar{\mathbf{x}}_{s,j})$ denotes the feature representation for source domain. After training, prediction of the testing data is obtained using feature extractor M_I and the classification network C_I.

C. Training Feature Representation Based on MMD

The other feature representation and classifier are trained based on MMD. As shown in Fig. 2(a), let the feature extractor II be M_{II}, and classifier II be C_{II}. Then, the feature representation for $\bar{\mathbf{X}}_s$ and $\bar{\mathbf{X}}_t$ can be denoted as $M_{II}(\bar{\mathbf{X}}_s)$ and $M_{II}(\bar{\mathbf{X}}_t)$, respectively. In this article, we use the RBF kernel-based MMD which can be written as

$$\begin{aligned}\underset{M_{II}}{\mathrm{loss}}(\bar{\mathbf{X}}_s, \bar{\mathbf{X}}_t) = &\sum_{i,i'}\frac{G(M_{II}(\bar{\mathbf{x}}_{s,i}), M_{II}(\bar{\mathbf{x}}_{s,i'}))}{n^2} \\ &- 2\sum_{i,j}\frac{G(M_{II}(\bar{\mathbf{x}}_{s,i}), M_{II}(\bar{\mathbf{x}}_{t,j}))}{n^2} \\ &+ \sum_{j,j'}\frac{G(M_{II}(\bar{\mathbf{x}}_{t,j}), M_{II}(\bar{\mathbf{x}}_{t,j'}))}{n^2}\end{aligned} \tag{9}$$

where $G(\cdot)$ denotes the RBF kernel which can be expressed as $G(x_1, x_2) = \exp(-\|x_1 - x_1\|^2/\tau)$. Since the parameters in M_{II} can be learned to adapt to the bandwidth τ, τ can be set to any constant. In our experiments, this parameter is set to 1.0.

Classifier C_{II} is trained with M_{II} simultaneously. The loss function used to train C_{II} is defined as

$$\begin{aligned}\underset{M_{II},C_{II}}{\mathrm{loss}}(\bar{\mathbf{X}}_s, \mathbf{Y}_s) = &-\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N} t_{k,y_{s,j}} \\ &\times \log\frac{\exp[(\mathbf{w}_{II,k})^T M_{II}(\bar{\mathbf{x}}_{s,j}) + b_{II}]}{\sum_{l=1}^{N}[\exp((\mathbf{w}_{II,l})^T M_{II}(\bar{\mathbf{x}}_{s,j}) + b_{II})]}\end{aligned} \tag{10}$$

where $\mathbf{w}_{II,k}$ and $b_I I$ are the parameters in classifier $C_I I$.

D. Ensemble Learning for Final Prediction

In our method, two classifiers are used, each of which is employed for one feature extractor. During the training processing, ensemble learning-based method is employed to get the final outputs from the two classifiers. In this article, we use the ensemble averaging to keep all networks around through

Algorithm 1: Proposed Method.

```
Input: source and target dataset: $(\mathbf{X}_s, \mathbf{Y}_s)$ and $\mathbf{X}_t$; the
  number of epochs: $L$; the batch size: $n$.
Steps:
  Preprocessing the source and target data and get
    $(\bar{\mathbf{X}}_s, \mathbf{Y}_s)$ and $\bar{\mathbf{X}}_t$.
  Network initialization of weights and biases;
  for $l = 1 : L$ do
    Sample $\{\bar{\mathbf{x}}_{s,i}, y_{s,i}\}_{i=1}^{n}$, a batch from source dataset
      $(\bar{\mathbf{X}}_s, \mathbf{Y}_s)$.
    Sample $\{\bar{\mathbf{x}}_{t,i}\}_{i=1}^{n}$, a batch from target dataset $\bar{\mathbf{X}}_t$.
    Update parameters in $M_I$ and $D$ by minimizing 6
      and 7 using source data $\{\bar{\mathbf{x}}_{s,i}\}_{i=1}^{n}$ labeled with '0' and
      target data with '1'.
    Update parameters in $M_{II}$ using $\{\bar{\mathbf{x}}_{s,i}\}_{i=1}^{n}$ and
      $\{\bar{\mathbf{x}}_{t,i}\}_{i=1}^{n}$ by minimizing 9.
    Update parameters in $M_I$ and $C_I$ using 8, $M_{II}$ and
      $C_{II}$ using 10 simultaneously.
  end for
Output: get predictions (pseudo labels) $\mathbf{Z}_t$ for $\bar{\mathbf{X}}_t$ using the
  trained model..
```

weighting the two classifiers in the training process adaptively. Let the prediction results for $\bar{\mathbf{x}}_s$ provided by C_I and C_{II} be $\mathbf{z}_{I,s}$ and $\mathbf{z}_{II,s}$, respectively. Therefore, the output $\mathbf{z}_s$ of ensemble learning for $\bar{\mathbf{x}}_s$ is

$$\mathbf{z}_s = \frac{\mathbf{z}_{I,s}+\mathbf{z}_{II,s}}{2}. \tag{11}$$

E. Retraining Model With Pseudolabeled Target Data

In the proposed domain adaptation framework, the unlabeled target data are utilized with their pseudolabels provided by the prediction results of trained model. Then, we use all the training data together with the pseudolabeled testing data to retrain the proposed domain adaptation network.

The retraining processing works in the following way: We first acquire the pseudolabels $\mathbf{Z}_t$ for $\bar{\mathbf{X}}_t$ using the proposed model trained with $\{\bar{\mathbf{X}}_s, \mathbf{Y}_s\}$. Then, we can use $\{\bar{\mathbf{X}}_t, \mathbf{Z}_t\}$ and $\{\bar{\mathbf{X}}_s, \mathbf{Y}_s\}$ to retrain the proposed network. Since the pseudolabels are not consistent with the true labels, the pseudolabeled testing dataset is constrained by confidence weights during retraining. Specifically, M_I, M_{II}, D and RBF kernel-based MMD are jointly trained based on (6), (7), and (9). But C_I and C_{II} are trained as follows:

$$\begin{aligned}&\mathop{\text{loss}}_{M_I,C_I}(\bar{\mathbf{X}}_s,\mathbf{Y}_s,\bar{\mathbf{X}}_t,Z_t)\\&=-\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N}t_{k,y_{s,j}}\log\frac{\exp[(\mathbf{w}'_{I,k})^T M_I(\bar{\mathbf{x}}_{s,j})+b'_I]}{\sum_{l=1}^{N}[\exp((\mathbf{w}'_{I,l})^T M_I(\bar{\mathbf{x}}_{s,j})+b'_I)]}\\&\quad-W_I\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N}t_{k,z_t}\log\frac{\exp[(\mathbf{w}'_{I,k})^T M_I(\bar{\mathbf{x}}_{t,j})+b'_I]}{\sum_{l=1}^{N}[\exp((\mathbf{w}'_{I,l})^T M_I(\bar{\mathbf{x}}_{t,j})+b'_I)]}\end{aligned} \tag{12}$$

TABLE I
ARCHITECTURE OF THE PROPOSED METHOD

Name of networks	Layers	Parameters
Feature extractor I	Convolution 1D	Kernels/size/stride: 128/17/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 16
	Convolution 1D	Kernels/size/stride: 128/17/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 16
	Convolution 1D	Kernels/size/stride: 128/3/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 2
	Flatten	-
Classification network I	Fully connected layer	512
	BN	-
	Leaky ReLU	0.2
	Dropout	0.3
	Softmax layer	The number of classes
Feature extractor II	Convolution 1D	Kernels/size/stride: 128/17/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 16
	Convolution 1D	Kernels/size/stride: 128/17/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 16
	Convolution 1D	Kernels/size/stride: 128/3/1
	BN	-
	Leaky ReLU	0.2
	Max pooling 1D	Size: 2
	Flatten	-
	Dropout	0.3
	Fully connected layer	512
Classification network II	Batch normalization	-
	Leaky ReLU	0.2
	Dropout	0.3
	Softmax layer	The number of classes
Domain prediction network	Fully connected layer	1024
	Batch normalization	-
	Leaky ReLU	0.2
	Dropout	0.3
	Fully connected layer	256
	Batch normalization	-
	Leaky ReLU	0.2
	Dropout	0.3
	Softmax layer	2

$$\begin{aligned}&\mathop{\text{loss}}_{M_{II},C_{II}}(\bar{\mathbf{X}}_s,\mathbf{Y}_s,\bar{\mathbf{X}}_t,Z_t)\\&=-\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N}t_{k,y_{s,j}}\log\frac{\exp[(\mathbf{w}'_{II,k})^T M_I(\bar{\mathbf{x}}_{s,j})+b'_{II}]}{\sum_{l=1}^{N}[\exp((\mathbf{w}'_{II,l})^T M_I(\bar{\mathbf{x}}_{s,j})+b'_{II})]}\\&\quad-W_{II}\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{N}t_{k,z_t}\log\frac{\exp[(\mathbf{w}'_{II,k})^T M_I(\bar{\mathbf{x}}_{t,j})+b'_{II}]}{\sum_{l=1}^{N}[\exp((\mathbf{w}'_{II,l})^T M_I(\bar{\mathbf{x}}_{t,j})+b'_{II})]}\end{aligned} \tag{13}$$

where W_I and W_{II} are confidence weights for pseudolabels, which are chosen according to the optimal accuracy of the training processing.

F. Architectures

The procedures of the proposed method are summarized in Algorithm 1, and the optimal network architectures for the proposed are shown in Table I. The CNN architecture in feature extractors MI and MII is developed from LeNet proposed by

TABLE II
BEARINGS WITH DIFFERENT DAMAGES

Health	OR damage	IR damage
K001	KA01	KI01
K002	KA03	KI03
K003	KA05	KI05
K004	KA06	KI07
K005	KA07	KI08
-	KA09	KI14
-	KA04	KI16
-	KA15	KI17
-	KA16	KI18
-	KA22	KI21
-	KA30	-

TABLE III
DIFFERENT SETTINGS OF THE PADERBORN DATASET

NO.	Name of setting	Rotational speed (rpm)	Radial force (N)	Load torque (Nm)
A	N15_N07_F10	1500	1000	0.7
B	N15_M01_F10	1500	1000	0.1
C	N15_M07_F04	1500	400	0.7

TABLE IV
LIST OF DATA USED FOR TRAINING AND TESTING THE PROPOSED METHOD IN THE SECOND EXPERIMENT

Class	Conditions	Training	Testing
1	Health	K002	K001
2	OR Damage	KA04 KA01 KA05 KA07 KA30	KA15 KA16 KA22
3	IR Damage	KI14 KI01 KI05 KI07 KI21	KI16 KI17 KI18

LeCun *et al.* in [21]. Specifically, M_I and M_{II} utilize 1-D convolution layers and pooling layers in our work to solve the fault diagnosis problem. Moreover, batch normalization (BN) [22] and leaky rectified linear unit (Leaky ReLU) [23] are used after each convolution layer, and dropout [24] is used with them after each fully connected layer. The training and testing experiments are implemented using Chollet *et al.* [25] running on top of TensorFlow [26] on a GeForce RTX 2080 graphics card.

V. Experimental Results

A. Datasets

In this article, the Paderborn dataset [27] is used to evaluate the performance of the proposed method. Three kinds of damage samples are included in this dataset: inner ring (IR) damage, outer ring (OR) damage, and health condition. The artificial damages were introduced manually and caused by three different methods: electric discharge machining, drilling, and manual electric engraving. The real bearing damage samples were caused by accelerated lifetime tests using scientific test rigs. Two experiments are conducted using the Paderborn dataset, and the names and properties are shown in Tables II, III, and IV, respectively. Tables II and III indicate the bearing code and the settings for each code in experiments.

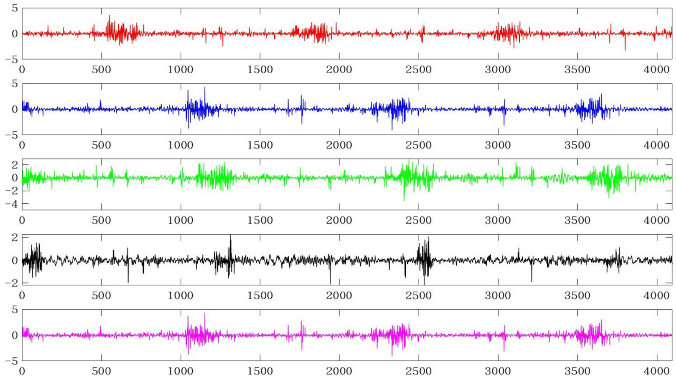


Fig. 3. Data illustration for N15_N07_F10 (red), N15_M01_F10 (blue), N15_M07_F04 (green), artificial fault (black), and real fault (fuchsine).

Two experiments are conducted. In the first experiment, the proposed method is trained with data of one working setting and tested with that of others, as shown in Tables II and III. In the second experiment, data used for training and testing the proposed method is listed in Table IV. In this article, the raw signals of Paderborn dataset are split into slices of the same length before input into the classification network. The split window size is 4096.

Fig. 3 illustrates four samples of the dataset.

B. Evaluation Metrics

The quantitative performance is measured using classification accuracy defined as follows:

$$\rho = \frac{c_w}{c} \tag{14}$$

where c_w indicates the number of samples with correct predictions, and c is the number of all testing samples.

C. Results and Analysis

1) Case Study 1: As for Paderborn dataset, the first experiment concerns the performance of the proposed method trained and tested with data obtained under different settings. Details of the data are shown in Tables II and III. In this experiment, the Adam optimizer [28] is used for 120 epochs, the initial learning rate is 0.0005 and will decrease to half of this value after 50 epochs. The batch size is 8.

Table V lists the classification accuracy of the proposed method and the compared methods. We can see from Table V that the proposed method performs much better the other methods, which proves the superiority of our method.

The time-consumption can be reflected in the prediction time of the fault diagnosis method. There are 32 332, 32 324, and 32 360 testing samples in N15_N07_F10, N15_N01_F10, and N15_N07_F04, respectively. Table V provides the prediction time for the six experiments. It can be seen from Table V that the prediction time for the model of the proposed method is much longer than DAT. The main reason is that the proposed method

TABLE V
CLASSIFICATION ACCURACY AND PREDICTION TIME OF OUR METHOD AND THE COMPARED METHODS ON PADERBORN DATASET LISTED IN TABLES II AND III *

Training dataset	Testing dataset	ρ				Prediction time (s)		
		Zhu et al. [29]	MMD	DAT	Ours	MMD	DAT	Ours
N15_N07_F10	N15_M01_F10	0.9427	0.9805	0.9819	**0.9826**	8.02	8.22	16.05
	N15_M07_F04	0.7950	0.8324	0.9555	**0.9680**	8.08	8.25	16.16
N15_M01_F10	N15_N07_F10	0.9697	0.9736	0.9881	**0.9901**	7.96	8.28	16.09
	N15_M07_F04	0.8067	0.8464	0.9091	**0.9177**	7.35	8.26	15.99
N15_M07_F04	N15_N07_F10	0.7067	0.8251	0.9055	**0.9192**	7.21	8.24	16.12
	N15_M01_F10	0.7023	0.8218	0.8806	**0.8921**	7.38	8.23	16.02

* The number in bold indicates the best result.

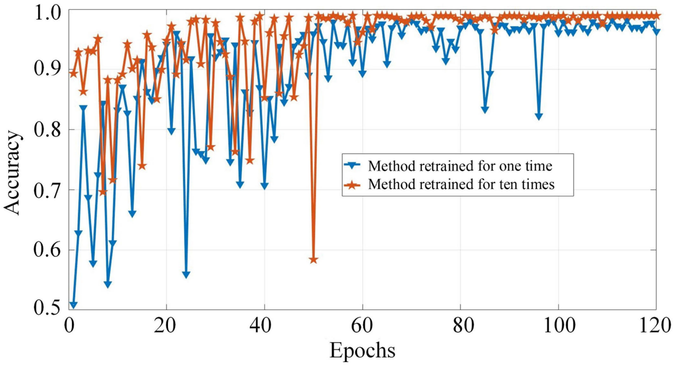


Fig. 4. Accuracy comparisons when the proposed method is retrained with N15_N07_F10 for one time and for ten times and tested with N15_M01_F10.

utilizes one more feature extractor and one more classifier than DAT for performance improvement.

Furthermore, the prediction result of the proposed method for testing dataset can be used as pseudolabel of testing dataset, based on which the model can be retrained. The optimal results are obtained when $W_I = 0.7$ and $W_{II} = 0.7$. Fig. 4 presents the accuracy comparisons when the proposed method retrained for one time and for ten times are applied to the testing dataset. Obviously, the model retrained for ten times is much better than that retrained for one time and the proposed method.

2) Case Study 2: The second experiment on Paderborn dataset concerns the accuracy of the proposed method when it is trained with artificial fault data and tested on real fault data. Details of the dataset are shown in Table IV. The Adam optimizer is also used in this experiment for 50 epochs, the initial learning rate is 0.0005 and will decrease to half of this value after 20 epochs. The batch size is 8.

In practical applications, it is much convenient to obtain plenty of artificial fault data than obtaining real fault data. Therefore, the research on fault diagnosis methods trained with artificial fault data and tested on real fault data is significant. Table VI shows the classification accuracy when the proposed method is compared with the state-of-the-art methods. We can see from Table VI that the proposed method has higher classification accuracy than other methods.

To further improve the performance of fault diagnosis models, the predictions of the proposed method are used as new pseudolabels for testing dataset. Then, the new pseudolabels are used to retrain the proposed method. The optimal results are

TABLE VI
PREDICTION ACCURACY AND TIME-CONSUMPTION OF OUR METHOD AND OTHER METHODS WITH ARTIFICIAL FAULT DATA FOR TRAINING AND REAL FAULT DATA FOR TESTING*

Methods	ρ	Prediction time (s)
Lessmeier et al. [27]	0.7500	-
MMD	0.7283	14.21
DAT	0.7764	9.52
Ours	0.7883	30.88
Model retrained for one time	0.7977	-
Model retrained for two times	0.8135	-
Model retrained for three times	0.8244	-
Model retrained for four times	0.8602	-
Model retrained for five times	0.8720	-
Model retrained for six times	0.8445	-
Model retrained for seven times	0.8790	-
Model retrained for eight times	0.8830	-
Model retrained for nine times	0.9004	-
Model retrained for ten times	**0.9086**	-

* The number in bold indicates the best result.

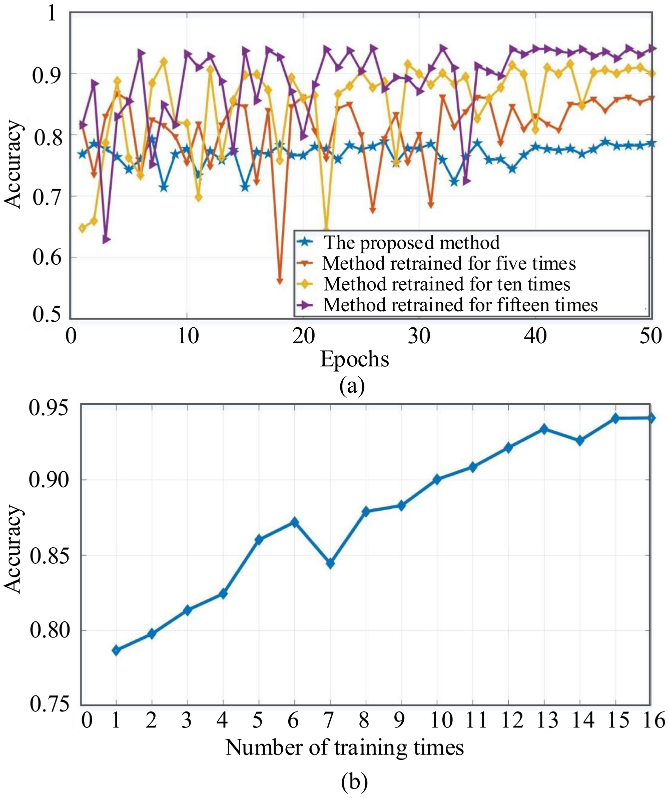
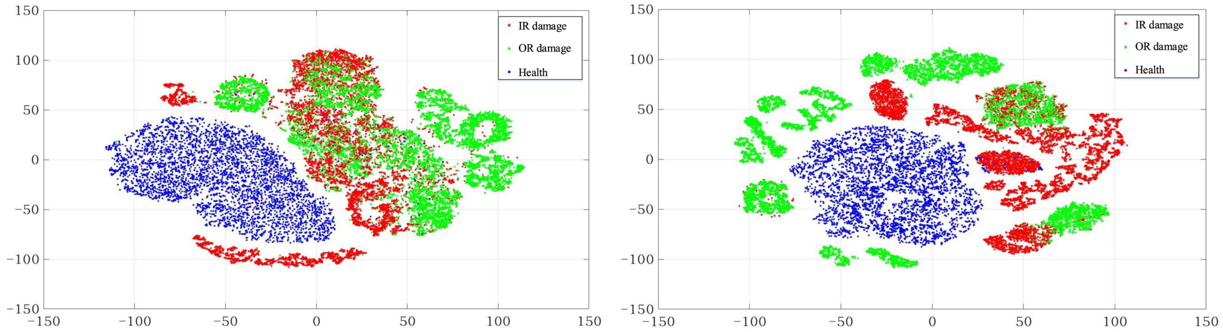


Fig. 5. Accuracy comparisons when the proposed method retrained from 1 time to 16 times with artificial fault data and tested with real fault data.



(a)

(b)

Fig. 6. (a) t-SNE maps data from the fully connected layer in classification network of DAT. (b) t-SNE maps fully connected layer output in classification network C_I of the proposed method.

obtained when $W_I = 0.7$ and $W_{II} = 0.7$. Table VI and Fig. 5 show the results when the proposed method is retrained for 10 times and 15 times, respectively. Obviously, the accuracy can be better if the proposed method is retrained iteratively with the new pseudolabeled testing data given by the last prediction models. Table VI also presents the time-consumption of DAT and the proposed method. The testing dataset includes 48 482 samples. The more complex network architecture of the proposed method makes it more time-consuming than DAT. Since retraining strategy only increases the training time, the prediction time for the models using and without using retraining strategy are the same.

Furthermore, Fig. 6(a) presents the t-distributed stochastic neighbor embedding (t-SNE) [30] map using data from the fully connected layer in classification network of DAT. The IR and OR damages in Fig. 6(a) cannot be separated into distinct clusters clearly. Most points of the two classes are mixed into the same cluster. Fig. 6(b) shows the t-SNE map for the output of the fully connected layer in C_I when the proposed method is retrained for ten times. The IR and OR damages are clustered more clearly than Fig. 6(a), which further demonstrates the improvement of feature representation ability of the proposed method.

3) Analysis: Based on the two case studies, we can observe that the proposed method has higher fault diagnosis accuracy than DAT and MMD-based domain adaptation, which proves the effectiveness of fusing different feature space and ensemble of two classifiers. On the other hand, experimental results reveal that: if the source and target domains are similar, such as domain data collected under different rotational speeds or load torque, our method can obtain fault diagnosis predictions with high accuracy; and if the source and target domains are with a large discrepancy, e.g., artificially induced damages as source domain and real damages as target domain, retraining the network with pseudolabeled target data will greatly improve the accuracy.

VI. Conclusion

In order to obtain an effective fault diagnosis model for data collected from different equipment or under different working conditions, in this article, we present an improved domain adaptation network to minimize the discrepancy of multidomain features, and simultaneously to learn representative features using DAT. Accordingly, two separate feature extractors and their corresponding classification networks were trained based on MMD and domain adversarial training. The proposed method further integrates ensemble learning for final prediction results. Two experiments involve fault data collected under different working conditions are carried out, results of which demonstrates the effectiveness of the proposed method.

Two works will be extended in future. First, we will optimize cross-domain fault diagnosis methods for more difficult scenarios, e.g., faults occurred on different equipment. Second, since the faulty data are much less than the health data, it is essential to develop methods for imbalanced data in practical applications.

References

[1] Y. Zuo, Y. Wu, G. Min, C.-Q. Huang, and K. Pei, “An intelligent anomaly detection scheme for micro-services architectures with temporal and spatial data analysis,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 548–561, Jun. 2020.

[2] C. Huang, G. Min, Y. Wu, Y. Ying, K. Pei, and Z. Xiang, “Time series anomaly detection for trustworthy services in cloud computing systems,” *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2017.2711039.

[3] Z. Chen and W. Li, “Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network,” *IEEE Trans. Instrum. Meas.*, vol. 66, no. 7, pp. 1693–1702, Jul. 2017.

[4] Z. Li, Y. Wang, and K. Wang, “A data-driven method based on deep belief networks for backlash error prediction in machining centers,” *J. Intell. Manuf.*, pp. 1–13, 2017, doi: 10.1007/s10845-017-1380-9.

[5] L. Zhang, H. Gao, J. Wen, S. Li, and Q. Liu, “A deep learning-based recognition method for degradation monitoring of ball screw with multisensor data fusion,” *Microelectronics Rel.*, vol. 75, pp. 215–222, 2017.

[6] H. Hu, B. Tang, X. Gong, W. Wei, and H. Wang, “Intelligent fault diagnosis of the high-speed train with big data based on deep neural networks,” *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2106–2116, Aug. 2017.

[7] W. Zhang, C. Li, G. Peng, Y. Chen, and Z. Zhang, “A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load,” *Mech. Syst. Signal Process.*, vol. 100, pp. 439–453, 2018.

[8] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[9] H. Daumé and D. Marcu, “Domain adaptation for statistical classifiers,” *J. Artif. Intell. Res.*, vol. 26, pp. 101–126, 2006.

[10] C. Cheng, B. Zhou, G. Ma, D. Wu, and Y. Yuan, “Wasserstein distance based deep adversarial transfer learning for intelligent fault diagnosis,” 2019, *arXiv:1903.06753*. [Online]. Available: http://arxiv.org/abs/1903.06753

[11] 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.

[12] X. Li, W. Zhang, and Q. Ding, "Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 7, pp. 5525–5534, Jul. 2019.

[13] X. Li, W. Zhang, Q. Ding, and J.-Q. Sun, "Multi-layer domain adaptation method for rolling bearing fault diagnosis," *Signal Process.*, vol. 157, pp. 180–197, 2019.

[14] X. Wang, H. He, and L. Li, "A hierarchical deep domain adaptation approach for fault diagnosis of power plant thermal system," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5139–5148, Sep. 2019.

[15] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22 14, pp. 49–57, 2006.

[16] O. Sener, H. O. Song, A. Saxena, and S. Savarese, "Learning transferrable representations for unsupervised domain adaptation," in *Proc. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2110–2118.

[17] R. Zhang, H. Tao, L. Wu, and Y. Guan, "Transfer learning with neural networks for bearing fault diagnosis in changing working conditions," *IEEE Access*, vol. 5, pp. 14 347–14 357, 2017.

[18] L. Wen, L. Gao, and X. Li, "A new deep transfer learning based on sparse auto-encoder for fault diagnosis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 1, pp. 136–144, Jan. 2019.

[19] W. Lu, B. Liang, Y. Cheng, D. Meng, J. Yang, and T. Zhang, "Deep model based domain adaptation for fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2296–2305, Mar. 2017.

[20] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7167–7176.

[21] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, pp. 541–551, 1989.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/pdf/1502.03167

[23] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. Int. Conf. Mach. Learn.*, 2013, vol. 30, no. 1, Art. no. 3.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] F. Chollet *et al.*, "Keras: Deep learning library for theano and tensorflow," 2015. [Online]. Available: https://keras. io/k

[26] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Des. Implementation*, 2016, pp. 265–283.

[27] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification," in *Proc. Euro. Conf. Prognostics Health Manage. Soc.*, 2016, pp. 5–08.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2014.

[29] Z. Zhu, G. Peng, Y. Chen, and H. Gao, "A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis," *Neurocomputing*, vol. 323, pp. 62–75, 2019.

[30] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. Nov, pp. 2579–2605, 2008.

Yibin Li received the Ph.D. degree in electrical and electronic engineering from the Department of School of Electronic, Electrical and Systems Engineering, Loughborough University, Loughborough, U.K., in 2009.

He is currently a Professor and Ph.D. Supervisor in control science and engineeringwith the Institute of Marine Science and Technology, Shandong University, Qingdao, China. His research interests include mechanical system monitoring, fault diagnosis, remaining usage life prognosis, and machine learning.

Yan Song received the B.S., M.S. and Ph.D. degrees in computer science and technology from Ocean University of China, Qingdao, China, in 2012, 2015, and 2018, respectively.

She is currently a Research Assistant with Institute of Marine Science and Technology, Shandong University, Qingdao. Her current research interests include machine learning, deep learning, fault diagnosis and side scan sonar image segmentation.

Lei Jia received the B.S. degree in mechatronics from Shandong Polytechnic University, Jinan, China, in 1982, the M.S. degree in automation from Shandong University, Jinan, in 1988, and the Ph.D. degree in automation from Zhejiang University, Hangzhou, in 1993.

In 1993, he joined Shandong Polytechnic University. In 1999, he joined Shandong University as a Professor of the School of Control Science and Engineering. His current research interests mainly include intelligent detection technology, advanced ocean sensing technology, structural health monitoring, and seawater desalination.

Shengyao Gao received the Ph.D. degree in ordnance science and technology from the Naval University of Engineering, Wuhan, China, in 2009.

He is currently working with the Naval Research Academy, Washington, DC, USA. His current research interests include vibration and noise testing of mechanical equipment, fault diagnosis, remaining useful life prediction, and control technology.

Qiqiang Li received the Ph.D. degree in industrial automation from the Institute of Industrial Process Control, Zhejiang University, Hangzhou, China, in 1998.

He is a Professor with the School of Control Science and Engineering, Shandong University, Jinan. His research area focuses on modeling, algorithm, and simulation of complex systems optimization. His current research interests are concerned with prediction, sizing and economic operation optimization of photovoltaic systems and smart grids, energy efficiency of process industry and commercial buildings.

Meikang Qiu (Senior Member, IEEE) received the B.E. and M.E. degrees in engineering from Shanghai Jiao Tong University, Shanghai, China, in 1992 and 1998, respectively, the M.S. degree in computer science in 2003 and the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, TX, USA, in 2007.

He is currently an Adjunct Professor with Columbia University, New York, NY, and a Distinguished Professor in Computer Science with Shenzhen University, Shenzhen. He has authored or coauthored 15 books, 400 peer-reviewed journal/conference papers, and three registered patents. His current research interests include cybersecurity, machine learning, Big Data, cloud computing, heterogeneous systems, and embedded systems.