Mechanical Systems and Signal Processing 158 (2021) 107744

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Weighted domain adaptation networks for machinery fault diagnosis

Dongdong Wei [a], Te Han [b], Fulei Chu [c], Ming Jian Zuo [a,*]

[a] Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta T6G 2R4, Canada
[b] Department of Energy and Power Engineering, Tsinghua University, Beijing 100084, China
[c] Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 13 August 2020
Received in revised form 16 December 2020
Accepted 9 February 2021

2010 MSC:
62P30
68T10

Keywords:
Machine learning
Deep learning
Transfer learning
Domain adaptation
Fault diagnosis
Gearbox
Vibration signals

ABSTRACT

Intelligent fault diagnosis of machines has received much attention in this big data era. Most reported models are constructed under the assumption that the training and testing data are from the same distribution. However, data distribution will shift due to working condition changes, posing challenges on the performance of intelligent models. This study considers the case that out of many known working conditions with labeled historical data, the model is to be used under another unlabelled target working condition. A multiple source domain adaptation method is proposed to learn fault-discriminative but working-condition-invariant features from raw vibration signals. Different known working conditions will be assigned with different weights, on the basis of their distributional similarities to the target working condition. Two case studies are carried out to validate the effectiveness of the proposed method, respectively on rotating speed changes and load level changes.

1. Introduction

Fault diagnosis of rotating machines is of vital importance in modern industry. A reliable early fault diagnosis system is key to reduce maintenance costs and can help preventing major failures [1]. Nowadays, intelligent fault diagnosis has received much research interests as it can work for real-time diagnostic applications and its performance grows with data volume [2,3]. Three intelligent fault diagnostic frameworks are often studied: conventional, deep learning (DL) based, and transfer learning framework [4].

Conventional intelligent fault diagnosis mainly involves signal processing, feature extraction, feature selection, and fault classification [5,6]. This conventional framework heavily relies on expert knowledge as the above technologies need to be tailored for different machines and different working conditions (different combinations of load and rotating speed) [7,8]. DL based diagnosis framework incorporates signal processing, feature extraction, and fault classification into one step [6,9]. DL framework has been successfully applied for the diagnosis of various mechanical components, such as bearings

* Corresponding author.
E-mail addresses: do1@ualberta.ca (D. Wei), hant15@tsinghua.org.cn (T. Han), chufl@mail.tsinghua.edu.cn (F. Chu), ming.zuo@ualberta.ca (M.J. Zuo).

https://doi.org/10.1016/j.ymssp.2021.107744
0888-3270/

[10–12], gears [8,13], and rotors [14,15]. However, neither the conventional nor the DL based framework is intelligent enough [16] as they may not perform well under the environments of (1) lack of labeled data and (2) variable working conditions.

The above two frameworks are based on supervised learning, which will learn only from labeled data. In many fault diagnosis cases, labeled data are expensive or even prohibitive. In addition, the two frameworks have limitations in coping with the influence of working condition changes. That is, a trained fault classifier may not perform well under different rotating speeds or load levels [13,17,18]. The key is to extract fault-discriminative but working-condition-invariant features from raw data.

To this end, a recent developed technique, domain adaptation [19,17,20], can support a more intelligent transfer learning framework [21] for fault diagnosis. In a fault diagnosis context, a working condition can constitute a domain. The domains with labeled data available are called source domains while those with unlabeled data are called target domains. Domain adaptation is to train a model using both source-labeled data and target-unlabeled data, so that the trained model can perform well in the target working condition. In mechanical fault diagnosis, domain adaptation has been developed and applied to combat rotating speed changes [22,13,4], load level changes [23,24,13], and cross-machine diagnosis [25]. However, these existing studies are limited to single source domain adaptation. In practice, fault data may be collected from two or more working conditions, constituting a multiple source domain adaptation problem. Methods for multiple source domain adaptation have not been adequately investigated for machinery fault diagnosis.

For natural language processing and computer vision applications, many multiple source domain adaptation methods have been developed. The key is to efficiently learn and combine knowledge from multiple sources. Refs. [26–29] construct different source-specific predictors for each source domain and then combine, with possibly different weights, their predictions on the target data. The idea of learning from multiple sources is to be utilized in this study for machinery fault diagnosis.

For machinery fault diagnosis, the number of source working conditions can be arbitrarily large and constructing that many source-specific predictors is computationally inefficient. Ref. [30] train a single target domain predictor utilizing multiple source domains. This is a good idea for fault diagnosis which can largely reduce computational costs. However, when training that target predictor, Ref. [30] uses equal weighting on different sources. Ref. [31] demonstrated that non-uniform weighting on different source could give better results but did not discuss how to determine the weights. The weighting schemes in [27–29] for natural language processing and computer vision applications are all based on multiple source-specific predictors and cannot be applied to single predictor methods [30]. The necessity of weighing the source domains differently, we believe, depends on where the differences of the domains are from. In natural language processing studies, the differences are likely to come from novel samples of one domain to the others [32]. In such applications, treating each domain equally is a good choice. However, for fault diagnosis, we can have clear physical meanings of domains differences and we have prior knowledge that the samples within each domain should be similar. With the above considerations, we believe that it is important to propose a good domain weighting scheme for fault diagnosis applications.

Apart from its positive effects, domain adaptation has the risk of causing negative transfer [33], in which training with both the source and the target data leads to a worse performance than only with the source data. To alleviate such risk, Refs. [34,35] reported to use thresholds to filter out negative data during adaptation, Ref. [33] studied when to cease the adaptation. Essentially, we need to know where not to adapt. If the risk of negative transfer is high, switching back to traditional supervised training with source data only is needed.

In this study, we are going to develop a multiple source domain adaptation method with different weights applied to different source domains (working conditions) for machinery fault classification. Our motivations are (1) Multiple source domains characterized by different working conditions can help the diagnosis on a different target working condition and (2) The contributions of different sources should be different given that their working condition deviations from the target working condition are different. We follow Ref. [30] to construct a single target domain classifier for all the domains, but we assign non-uniform weights to different sources. Different schemes of weight assigning will be investigated. To avoid negative transfer for good classification accuracy, we propose to determine whether to use domain adaptation or traditional supervised learning first before commencing the training. Domain adaptation might not help when the source and the target working conditions are too close. Two case studies on two different experiment test rigs are performed, and both speed change and load level change will be studied. The effectiveness of our proposed multiple source domain adaptation will be demonstrated, and the soundness of our weighting scheme will be examined. In summary, the main novelty of this paper are:

1. We treat each working condition as a domain and learn from multiple source-labeled and one target-unlabeled domains;
2. We weigh different source domains differently to scale the contributions of different source working conditions;
3. We assess the necessity of domain adaptation first before training to avoid negative transfer.

In the following parts, Section 2 discusses the preliminary knowledge, Section 3 explain our proposed method and the reported methods to be compared with, Section 4 presents two case studies to demonstrate the proposed method, and then Section 5 concludes this paper.

2. Preliminaries

2.1. Domain adaptation

When the training and testing data are drawn from different distributions, traditional machine learning algorithms do not perform well. Domain adaptation become useful in this situation [21].

A *domain* consists of a data space $\mathcal{X}$ and a probability distribution $P(X)$ on its samples $X \in \mathcal{X}$. Domain adaptation means to adapt useful knowledge from a source-labeled domain S to a target-unlabeled domain T. Specifically, we are given a source-labeled dataset $(X_S, Y_S) = \{(x_S^1, y_S^1), (x_S^2, y_S^2), \dots, (x_S^m, y_S^m)\}$ and a target-unlabeled dataset $X_T = \{x_T^1, x_T^2, \dots, x_T^n\}$ for model training. The trained model is expected to have good classification or regression performance on unseen target domain samples.

The following two assumptions are made for domain adaptation in Ref. [21]. 1) The data distributions of the source and the target domains should be different but similar, i.e. $P_S(X_S) \neq P_T(X_T)$ but $P_S(X_S) \approx P_T(X_T)$, so that the knowledge learned from the source can be adapted to the target. 2) The data space and label space $\mathcal{Y}$ should be the same across the source and the target domains, i.e. $\mathcal{X}_S = \mathcal{X}_T$ and $\mathcal{Y}_S = \mathcal{Y}_T$, so that neither new format nor new class of data will come in when testing. In this study, we follow these definitions and focus on scenarios meeting these assumptions.

2.2. Domain adversarial training

In domain adversarial training, a neural network can be viewed as consisting of three parts [20]: *feature extractor F*, *label classifier C*, and *domain discriminator D*. The feature extractor transforms an input data x into a feature vector f, i.e. $f = F(x; \theta_f)$. With f, the label classifier C calculates a vector of class scores $c = C(f; \theta_c)$ for each class while D produces a vector of domain scores $d = D(f; \theta_d)$ for a source or a target domain. That is, the label classifier predicts the label of an input and the domain discriminator tries to tell if the input is from the source or the target domain. The indices of the highest score are regarded as the class/domain predictions.

The key of domain adversarial training is to put C and D against each other as two players in a minimax game. The label classifier C plays to minimize its labeling error while D is set to maximize "*domain confusion*" [36]. A Nash equilibrium can be achieved if the feature extractor F is trained to produce label-discriminative yet domain-invariant features. More formally, considering the function $\widetilde{E}(X, Y; \theta_f, \theta_c, \theta_d) = L_C(X, Y; \theta_f, \theta_c) - L_D(X; \theta_f, \theta_d)$, it is to search for parameters $(\hat{\theta}_f, \hat{\theta}_c, \hat{\theta}_d) = \min_{\theta_f, \theta_c} \max_{\theta_d} \widetilde{E}$ that deliver a saddle point at maximum domain confusion L_D yet minimum labeling error L_C [20].

Further, by inserting a Gradient Reversal Layer (GRL) between F and D, this parameter searching can be implemented as a simple minimization via backpropagation [20]:

$$\min_{\theta_f, \theta_c, \theta_d} E = L_C(X, Y) + L_D(X). \tag{1}$$

The GRL simply copies f into D during the forward feeding, while reverses the sign of the gradient that backpropagates from it. This reversed gradient will move F towards the negative direction of minimizing the L_D term, so that the domains can be maximally confused. For the loss functions L_C and L_D, standard Cross Entropy Loss [37] will be used our study:

$$L_C(X, Y) = -\frac{1}{m} \sum_{(x,y) \in (X_S, Y_S)} \log \frac{\exp(c[y])}{\sum_i \exp(c[i])}, \tag{2}$$

$$L_D(X) = -\frac{1}{m} \sum_{x \in X_S} \log \frac{\exp(d[1])}{\sum_i \exp(d[i])} - \frac{1}{n} \sum_{x \in X_T} \log \frac{\exp(d[2])}{\sum_i \exp(d[i])}, \tag{3}$$

where (X_S, Y_S) is the source-labeled dataset with m samples, X_T is the target-unlabeled dataset with n samples, x denotes a data sample, y is its corresponding label, and $c[i]$ and $d[i]$ are respectively the ith element of the class score vector and the domain score vector which were explained in the first paragraph of Section 2.2.

2.3. Maximum mean discrepancy

Maximum Mean Discrepancy (MMD) measures the difference between two distributions on the basis of samples drawn from the two distributions [38]. Given samples $X = \{x_1, x_2, \dots, x_m\}$ and $Z = \{z_1, z_2, \dots, z_n\}$, using a kernel $\mathcal{K}(\cdot, \cdot)$, MMD can be estimated using Eq. (4):

$$MMD(X, Z) = \left[\frac{1}{m^2} \sum_{i,j=1}^{m} \mathcal{K}(x_i, x_j) - \frac{2}{mn} \sum_{i,j=1}^{m,n} \mathcal{K}(x_i, z_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} \mathcal{K}(z_i, z_j) \right]^{\frac{1}{2}}. \tag{4}$$

The MMD value is expected to be a small quantity if the distributions of X and Z are similar. The choice of kernel is critical to the power of this discrepancy measurement. A typical choice is the Radial Basis Function (RBF) kernel, i.e.

$\mathcal{K}(x,z) = \exp\left(-\|x-z\|^2/b\right)$, where b is the bandwidth of the RBF kernel. Ref. [39] further discussed that a linear combination of multiple kernels can render a good kernel. MMD using a combined kernel is often called Multi-Kernel MMD, or MK-MMD [40,23]. In addition, squared formulation of MMD, or MMD^2, is used and aliased with MMD in Refs. [40,23]. For simplicity, we will use MMD to denote the squared MK-MMD in the following parts of this paper.

Most MMD based domain adaptation methods will not apply MMD on the input data. Instead, MMDs of extracted features are usually measured [40,23,13,2]. This fits the goal of learning domain-invariant features. In practise, considering computational constraints and efficiency, the inputs of Eq. (4) are mostly mini batches (small partitions of a whole dataset).

3. The proposed method

As discussed in Section 1, we study domain adaptation to combat the impact of working condition changes on machinery fault diagnosis. To efficiently learn knowledge from multiple source working conditions, a good weighting scheme is needed. In addition, to avoid negative transfer, a criterion of determining whether to perform domain adaptation is to be adopted.

In this paper, a Weighted Domain Adaptation neural Network (WDAN) is proposed. Unlike Ref. [17] which merges multiple labeled working conditions as one source domain and build only one domain discriminator (see Section 2.2), we treat each working condition as a separate domain, and follow Ref. [30], which has been successfully applied to natural language processing and computer vision, to build multiple source-specific domain discriminators. On top of Ref. [30], we insert a weighting block so that different weights can be assigned on different source domains. The weights will be assigned based on the MMD measure explained in Section 2.3. The measured MMD values will also be used to determine whether to perform domain adaptation or traditional supervised learning, so that negative transfer can be avoided.

In the following part of this section, Section 3.1 presents the architecture of the proposed WDAN, and Section 3.2 explains the training procedure of WDAN. All the compared methods will be summarized in Section 3.3.

3.1. WDAN architecture

As shown in Fig. 1, WDAN has a feature extractor F, a classifier C, multiple domain discriminators Ds, and a domain weighting block. Each D has the same structure but is associated with a specific source domain, so that multiple source working conditions can be accommodated. Note that Ref. [17] has only one domain discriminator and Ref. [30] dose not have the domain weighting block which will be explained in Section 3.2.

The structures of F, C, and D are respectively shown in the upper, left lower, and right lower boxes of Fig. 2. The 3-d blocks annotated with Conv, FC, and BN in Fig. 2 are 1-d convolution layers, fully connected layers, and batch normalization layers, respectively. The three comma separated digits listed under a Conv layer in Fig. 2 are its number of input channels, number of output channels, and kernel size; the numbers pointed by an arrow under the FC layers are their output dimensions. All three types of layers have learnable parameters that will be updated during training. The empty V-shaped arrows are layers (or operations) without learnable parameters. As annotated at the top of Fig. 2, they are (max) pooling layers with a down sampling ratio of 0.25, ReLU activation layers, dropout layers with a drop rate of 0.5, a flatten layer to reshape high order tensors into the 1st order vectors, and a GRL that was explained in Section 2.2.

From Fig. 2, we can see that the design of the feature extractor is a one-dimensional version of the very first reported convolutional neural network [41]. The kernel size of the Conv layers and the downsampling ratio of the pooling layers follow the original design. The use of ReLU activation [42] is standard for deep neural networks. We select the output dimension of FC1 from {500, 1000, 2000}, the output dimension of FC2 from {128, 256, 512}, the use of dropout operation, and the use of BN layer based on the average of testing accuracies on the source datasets of ST-1 in Table 2 (to be explained later) of source-only method (to be explained in Section 3.3). The classifier and domain discriminators are both standard two-layer perceptrons [37]. The size of FC4 equals the number of machine's fault classes while FC6 has only 2 neurons respectively for the source and the target. The output dimension of FC3 was pre-determined. We select the output dimension of FC5 from {16, 32, 64} based on the testing accuracy on the target dataset of ST-1 of equal-weighting method (to be explained in Section 3.3).

After the training is completed, only the F and C will be saved for testing while the Ds will be discarded. The saved F and C are expected to have good performance on the target domain. We use a held-out labeled dataset from the target domain to test the classification accuracy of the trained networks. All the input data samples are raw vibration signal segments, and their corresponding labels are machine's fault classes. Before being sent into WDAN, a normalization step is completed in which a data sample will be subtracted and divided respectively by the element-wise mean and the standard deviation of its original dataset. The held-out test set will use its own mean and standard deviation.

3.2. Training procedure

We train the WDAN with two objectives: (1). minimal classification error on each source domain; (2). maximum "domain confusion" between each source and the target. If there is only a single source domain, vanilla domain adversarial training (Eq. (1) in Section 2.2) can be applied. When there are multiple source domains, we need to combine their loss terms

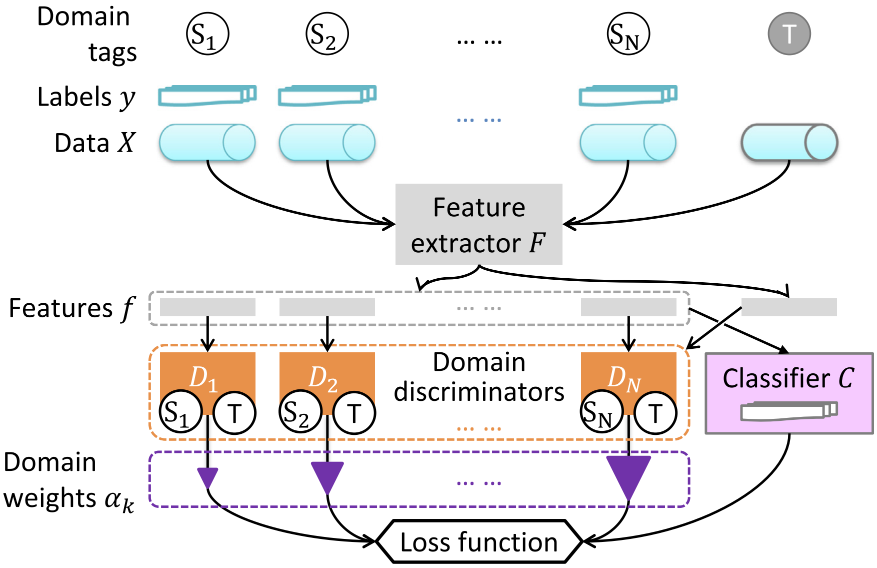

Fig. 1. Schematic diagram of WDAN.

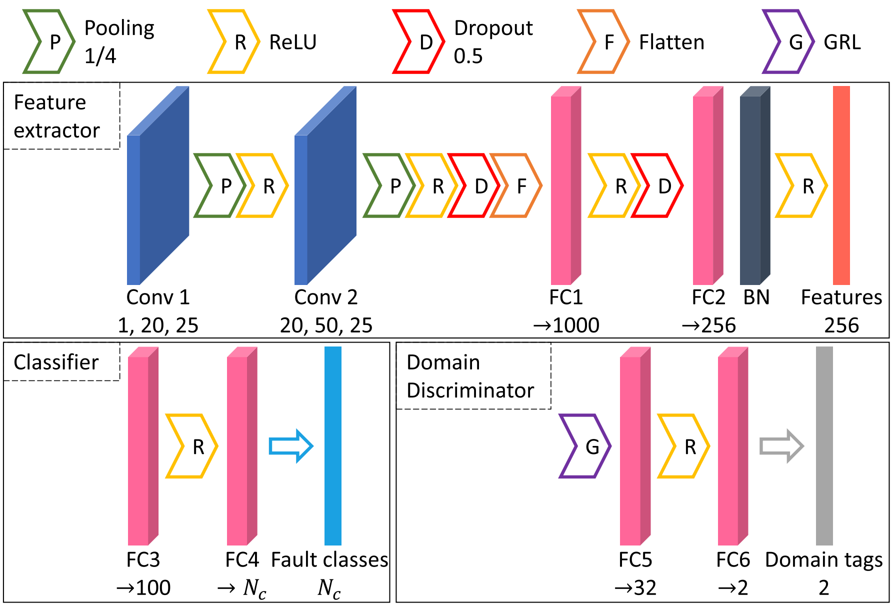

Fig. 2. Network structures used in this paper.

properly. A log-sum-exp operation is used in Ref. [30] but every source is equally weighted. We propose to assign different weights to different sources and formulate the training of WDAN as:

$$\min \ \log \sum_{k=1}^{N} \exp N\alpha_k (L_{Ck} + L_{D_k}), \tag{5}$$

where N is the number of source domains, $L_{Ck} = L_C(X_{S_k}, Y_{S_k})$ is the labeling error term on the kth source dataset, $L_{D_k} = L_{D_k}(\{X_{S_k}, X_T\})$ is the domain confusion term of the kth domain discriminator, and they are calculated using Eqs. (2) and (3) in Section 2.2, respectively, and α_k is the weight for domain k with two constraints $\sum_k \alpha_k = 1$ and $\alpha_k \geqslant 0$, which is proposed to replace the original uniform weighting of sources in Ref. [30].

Generally, higher weights should be assigned to sources more similar to the target. Refs. [27,28] measure distributional similarities between the sources and the target, and then divide the measured similarities by their sum. In such a way, all the sources were assigned with different and non-zero weights. In this paper, we propose to use the distribution metric MMD explained in Section 2.3 as the (negative) domain similarity measure. Then, we apply softmax function to the measured MMD values so that the two constraints listed at the end of the previous paragraph on weights can be met. Formally, for a source domain S_k, we assign

$$\alpha_k = \frac{\exp(-\beta v_k)}{\sum_i \exp(-\beta v_i)}, \tag{6}$$

where v_k is the measured MMD between the S_k and the target T. Following [40,23,13,2], the MMD are measured based on Eq. (4) with the extracted features f of mini-batch data as its input. Average MMD of all the mini-batches will be used. The coefficient $\beta \geqslant 0$ controls the "hardness" of weight assigning. Smaller β is softer as it gives close weight values for all the sources. When $\beta = 0$, our method is relaxed to Ref. [30] with equal weights; when $\beta \to \infty$, our weighting scheme becomes "hard max" as only the best source(s) will be assigned with non-zero weight(s). We believe that generally speaking, allowing a positive β which gives non-uniform weights to multiple sources is the best practice.

Apart from weight assigning, the measured MMD values will also be used to assess the risk of negative transfer. If the MMD values are all below a certain threshold, the source and the target domains are already well matched. In such a case, rather than risking testing performance with target-unlabeled data, we perform traditional supervised learning with source-labeled data only. That is, we merge all the source domains as one ($N = 1$) and deactivate the domain discriminators ($L_{D_k} = 0$). The pseudo code of our proposed WDAN training procedure is described in Algorithm 1.

Algorithm 1: Training procedure of WDAN

Input:

- Datasets $\{(X_{S_k}, Y_{S_k})\}_{k=1}^{N} = \left\{\left\{\left(x^i_{S_k}, y^i_{S_k}\right)\right\}_{i=1}^{m_k}\right\}_{k=1}^{N}$ and $X_T = \{x^i_T\}_{i=1}^{n}$,
- Initial WDAN $\{F, C, D\}$,
- Hardness parameter β,
- Mini-batch size p,
- MMD kernel $\mathcal{K}$,
- Transfer threshold τ.

Output: trained WDAN $\{F, C\}$

# calculate average MMDs: v_ks (initial $v_k = 0$)
for k from 1 to N **do**
  # randomly split datasets into mini-batches of p samples
  $\left\{\left(X^i_{S_k}, Y^i_{S_k}\right)\right\}_{i=1}^{\lfloor m_k/p \rfloor} \leftarrow \mathrm{split}(X_{S_k}, Y_{S_k})$
  $\left\{X^i_T\right\}_{i=1}^{\lfloor n/p \rfloor} \leftarrow \mathrm{split}(X_T)$
  for j from 1 to $\lfloor n/p \rfloor$ **do**
    if $j > \mathrm{size}\left\{\left(X^i_{S_k}, Y^i_{S_k}\right)\right\}$ **then**
      $\left\{\left(X^i_{S_k}, Y^i_{S_k}\right)\right\} \leftarrow \mathrm{duplicate}\left(\left\{\left(X^i_{S_k}, Y^i_{S_k}\right)\right\}\right)$ # use S_k repeatedly
    end if
    $v_k \leftarrow v_k + MMD\left(F\left(X^j_{S_k}\right), F\left(X^j_T\right); \mathcal{K}\right) / \lfloor n/p \rfloor$
  end for
end for
# apply threshold to avoid negative transfer
if $\forall v_k < \tau$ **then**
  $\min \sum_{k=1}^{N} L_{Ck}$ # execute source-only training
else
  for k from 1 to N **do**
    $\alpha_k \leftarrow \frac{\exp(-\beta v_k)}{\sum_i \exp(-\beta v_i)}$ # assign weights using Eq. (6)
  end for
  $\min \log \sum_{k=1}^{N} \exp N\alpha_k (L_{Ck} + L_{D_k})$ # execute adaptive training (Eq. (5))
end if

3.3. Compared methods

We compare our proposed WDAN with the following methods: **source-only** using Eq. (5) with $N = 1$ and $L_{D_k} = 0$; **best-single** which applies single source domain adaptation [20] on every single source domain ($N = 1$) and then report the best possible result; **merge-as-one** [17] and then apply single source domain adaptation with $N = 1$; and Ref. [30] that apply **equal-weighting** on sources by setting $\beta = 0$. We use **WDAN-**β to denote our proposed method with a "hardness" coefficient of β. Note that WDAN-0 is identical to Ref. [30]. We are going to explore the impact of different real positive values of β.

To put all the above methods on equal footing, all will use the same network structures in Fig. 2. We use standard mini-batch Stochastic Gradient Descend (SGD) with momentum [43] to solve their corresponding training objective functions. Note that Refs. [17,20] both used SGD with momentum. Refs. [44,45] also support that SGD generalizes better than adaptive gradient methods including Adam. Empirically, the momentum is set to be 0.9, the mini-batch size is set to be 100, and the best learning rate for each method will be grid-searched from {0.1, 0.01, 0.001}. For efficiency and ease of implementation, the mini-batch size for calculating the average MMD values is also set to be 100.

4. Experiments

We present two case studies on two different test rigs located at Tsinghua University (THU) and University of Alberta (UofA). Computational experiments are run on a computer with a single Intel i7-6700 CPU and a single Nvidia GTX-1060 GPU. All the tested methods are implemented using Pytorch.[1]

4.1. Case study I

4.1.1. THU planetary gearbox test rig

The HS-200 single-stage planetary gearbox[2], located at Tsinghua University, was used to conduct experiments and collect data in year 2019 by one of the co-authors. During the physical experiment, the gearbox was driven by a motor and the output rotating speed of the motor was set at 26 different constant levels, ranging from 15 Hz to 40 Hz with an interval of 1 Hz. The load level was fixed at 0. Two accelerometers were installed in x and y directions the of gearbox casing. By seeding artificial damages in the sun gear or one of the planetary gears, 9 different fault classes were created and tested. The 9 tested fault classes are described in Table 1. Fig. 3 shows the structure of the planetary gearbox and 4 example damaged gears.

4.1.2. Data description

We regard each rotating speed as a domain. Three different source-target adaptation tasks listed in Table 2 are studied. The physical implications of the three tasks are ST-1: adapt to a lower speed; ST-2: adapt to a speed in the middle; ST-3: adapt to a higher speed.

For each rotating speed and each fault class, the vibration data measured in the x-direction at a sampling frequency of 20 kHz, for a consecutive 256 s are used. The first 204.8s are used for training and the rest 51.2s are held out for testing. We slice the vibration signals into 2048 sized input samples for our neural networks. That is, there will be 2000 and 500 samples per domain per class respectively for training and testing.

4.1.3. Results and discussions

We apply all the five methods discussed in Section 3.3 and compare their performances (average over 10 repeat runs) on ST-1 to ST-3. For all the methods, the number of training epochs is 10. For the choice of β, we tested 4 different values, 5, 10, 25, and 100 on ST-1 and found that $\beta = 10$ provided the best accuracy on ST-1 (see Fig. 4). In this study, $\beta = 10$ is used as the default choice for other tasks. We follow Ref. [23] to calculate MMD values with 5 RBF kernels with bandwidths of 1, 2, 4, 8, and 16.

We first test the source-only method on the source domain data. The source test accuracies are 99.993%, 99.966%, and 99.993% respectively for ST-1, ST-2, and ST-3. These accuracies show that, when the testing and training data are from the same set of domains, our designed feature extractor and classifier (Section 3.1) can have very good performances. Then, we test all 5 methods on the target domains and show the target test accuracies in Table 3. The highest testing accuracy for each scenario is highlighted in boldface in Table 3.

From the left-hand side of Table 3, we can see that, when tested under a different working condition, the accuracies of source-only models drop 20.504%, 5.584%, and 30.457%, respectively for the three tasks, comparing to their source test accuracies. These numbers show how much the rotating speed gaps affect the traditional source-only learning method. Our proposed WDAN reduced the three previous listed accuracy drops to 8.891%, 0.808%, and 5.613%, by gaining 11.613%, 4.776%, 24.544% of accuracies comparing to source-only. It achieves the best performance among all the compared methods, which demonstrates the effectiveness of multiple domain discriminators and our proposed weighting scheme. Other three domain adaptation methods, i.e. merge-as-one, best-single, and equal-weighting, can also gain accuracies on top of the baseline

[1] https://pytorch.org/
[2] http://www.sh-wxjd.net/product/qdq/49.html

Table 1
Tested fault classes of the THU planetary gearbox.

Classes	Types	Levels
H	Health	–
SC1	Sun tooth crack	Crack depth: 1/8 dedendum
SC2	Sun tooth crack	Crack depth: 1/4 dedendum
SC3	Sun tooth crack	Crack depth: 1/2 dedendum
SB	Sun tooth broken	Break position: 1/3 tooth depth
PC1	Planet tooth crack	Crack depth: 1/8 dedendum
PC2	Planet tooth crack	Crack depth: 1/4 dedendum
PC3	Planet tooth crack	Crack depth: 1/2 dedendum
PB	Planet tooth broken	Break position: 1/3 tooth depth

Table 2
Adaptation tasks on THU dataset.

Adaptation task	Source domains	Target domain
ST-1	40 Hz, 34 Hz, 28 Hz	16 Hz
ST-2	16 Hz, 40 Hz	28 Hz
ST-3	16 Hz, 22 Hz, 28 Hz	40 Hz

Fig. 3. Structure of the HS-200 planetary gearbox and 4 example damaged gears.

source-only approach. With no negative transfer observed, domain adaptation should be applied for all the three tasks in this case study. The criterion for switching back to source-only will be investigated with the next case study in Section 4.2.

It is also observed that the weights assigned using MMDs well agree with the physical meanings of the working conditions. A source with smaller speed gap to the target will be assigned with higher weight. For the ST-2 where the two sources have the same speed gap to the target, the source of higher speed conditions will be assigned with higher weight. The ranking of sources reported by the best-single method also agrees with the ranking of weights assigned by MMDs. The best sources are both 28 Hz for ST-1 and ST-3, and the best-single source for ST-2 is 40 Hz. Although, all the best-single's testing accuracies are lower than the WDAN's. It can be said that adapting from multiple sources can be better than from only one.

Among the three tasks, the task of adapting to a middle speed (ST-2) is the easiest as its accuracy of source-only drops the least across domains. The adaptation gains of applying domain adaptation methods are also less significant comparing to the other two tasks. For example, adaptation gains of the best-single method are 10.47% and 20.27% respectively for ST-1 and ST-3, but only 0.54% for ST-2. That said, domain adaptation is more useful when the target speed condition is out of the range covered by the known speeds.

The influence of the "hardness" coefficient β is shown in Fig. 4. Task ST-1 with a target domain of 16 Hz is used for demonstration. In Fig. 4, α_1, α_2 and α_3 are the weights assigned to the three source domains 40 Hz, 34 Hz, and 28 Hz respectively.

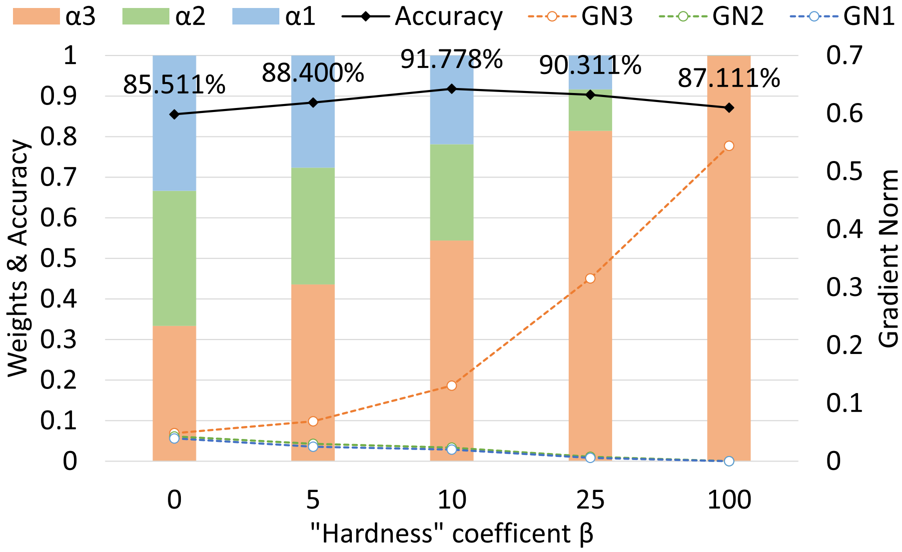

Fig. 4. Influence of β on ST-1. Rotating speeds of S_1: 40 Hz, S_2: 34 Hz, S_3: 28 Hz, T: 16 Hz.

Table 3
Target domain test accuracies and training time costs on THU dataset.

Method	ST-1 (%)	ST-2 (%)	ST-3 (%)	Training time (seconds)	
				ST-1 or ST-3	ST-2
source-only	79.489 ± 2.082	94.382 ± 3.276	69.836 ± 4.396	85.16 ± 0.35	57.83 ± 0.14
merge-as-one [17]	83.476 ± 4.942	97.816 ± 1.506	85.647 ± 11.18	58.94 ± 0.16	59.27 ± 0.25
best-single [20]	89.962 ± 3.365	94.924 ± 4.909	90.102 ± 7.154	171.06 ± 0.31	115.13 ± 0.38
eqaul-weigthing [30]	85.322 ± 4.006	99.042 ± 0.469	90.382 ± 6.279	107.94 ± 0.24	83.86 ± 0.33
WDAN-10 (proposed)	**91.102** ± 1.369	**99.158** ± 0.253	**94.380** ± 4.063	113.69 ± 0.27	88.22 ± 0.33

The solid line with markers shows the target test accuracies for each chosen β values. It shows equal-weighting ($\beta = 0$) gives the lowest accuracy and WDAN-10 is the best for ST-1. Gradient Norms (GNs) of the FC5 layers of D_1, D_2, and D_3 (see Section 3.1) are shown as the three dash lines. GN means the 2nd norm of the back-propagated gradients and here we use average GN across all learning steps. These gradients will be reverse by the GRL and backpropagate into the feature extractor F. The average GN from a D_k can describe how fast and how much the F changes in order to confuse its corresponded S_k and the T. We can see (GN1 and GN2 are overlapped) that GNs are positively correlated with the assigned weights. This proves that the sources assigned with higher weights are playing more important roles during learning.

For the case of $\beta = 100$, only the domain of 28 Hz is assigned with non-zero weight, making our WDAN approaches the best-single method. However, WDAN has more parameters to learn as it has three domain discriminators while best-single only has one. This makes the WDAN-100 more likely to overfit and achieve lower accuracy. A "soft" β will put WDAN in between pay equal attentions to every source (equal-weighting) and learn only from the best-single source.

To visualize the classification performance for each different category, Fig. 5 shows the confusion matrices of the source-only method and our proposed WDAN on ST-3. The notations of the fault classes are explained in Table 1. We can see that the source-only method performs poorly on classifying certain fault classes. For example, the accuracy is 0% for SC3 and 9% for PC3. This implies that, due to rotating speed difference, the target samples of a certain class may be wrongly aligned with the source samples of other classes. In contrast, domain adaptation (WDAN) can better align the source and target samples of each classes and produces much better classification performance. Its accuracies are 84% and 91% higher, for SC3 and PC3 respectively, than those of source-only.

The trade-offs behind adaptation gains include extra computational time and memory for training. As shown in the right-hand side of Table 3, for a 3-source-1-target task (ST-1 or ST-3), WDAN spends about 5.75 s to compute the weights. In addition, source-only needs about 85.16s for 10 epochs of training while WDAN takes about 113.69s. Memory cost is also higher for WDAN as more data and more learnable parameters will be used than it is in source-only. Nonetheless, all the five listed methods have the same speed and memory usage during testing stage.

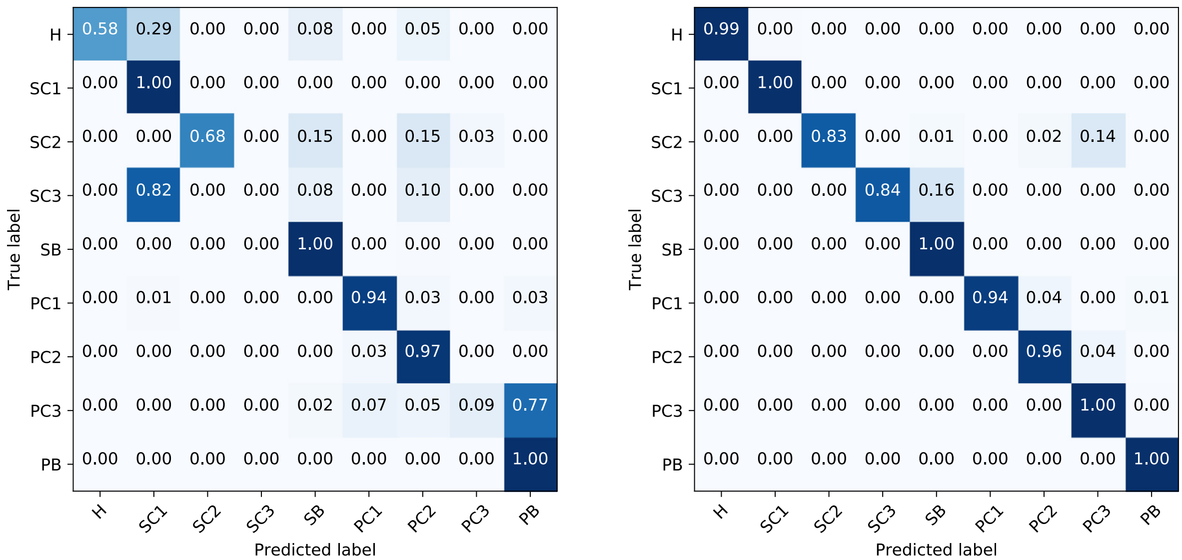

Fig. 5. Confusion matrices on ST-3. **Left**: source-only; **Right**: WDAN-10 (proposed).

4.2. Case study II

For the 2nd case study, a gearbox test rig at the University of Alberta [46,47] was used to collect the data in year 2018. The data is used directly in this case study. Six fault classes are considered including five different levels of gear tooth crack and one healthy state.

4.2.1. Data description

During the physical experiment, the rotating speed was set at 10 Hz and the machine was run at three different load levels: 3% (low), 8% (middle), and 13% (high) of the load motor's rated capacity. We regard the three load levels as domains and enumerate all three possible 2-source-1-target adaptation tasks. They are **ST-4**: the 8% and 13% loadings are sources while the 3% loading is the target; **ST-5**: the 3% and 13% loadings are sources while the 8% loading is the target; and **ST-6**: the 3% and 8% loadings are sources while the 13% loading is the target.

The vibration data provided by the accelerometer on its bearing cap (Sensor #2 in Ref. [46]), with a sampling frequency of 25.6 kHz, is to be used in this case study. For each load level and each fault class, four repeats of 30-s runs were conducted during the experiments. The first three runs are used for training and the last one is held out for testing. We slice the vibration signals into 2048 sized input samples for our neural networks. That is, there will be 1125 and 375 samples per domain per class respectively for training and testing.

4.2.2. Results and discussions

Following the first case study, we show the average performances over 10 repeat runs on ST-4 to ST-6. For WDAN, we keep $\beta = 10$ and the MMD kernels are also the same as in case study I. For all the methods, the number of training epochs is increased to 25 (from 10 in case study I). Under such setting, the number of training steps will be similar for the two case studies, given that the THU dataset is 2.667 times larger.

When tested by the source domain data, source-only models give accuracies of 96.667%, 95.482%, and 94.169% respectively for ST-4, ST-5, and ST-6. This is a solid performance for crack level classification. The target test accuracies are shown in the left-hand side of Table 4. The highest testing accuracy for each scenario is highlighted in boldface in Table 4. Same as in case study I, WDAN shows adaptation gains, over source-only, of 1.902% and 3.387% respectively on tasks ST-4 and ST-6, achieving the best accuracies among all the compared methods. However, negative transfer [48] is observed in this case study. Utilizing both the source and target data may result lower accuracy than source-only. Under such a case, our proposed MMD-based criterion comes into use.

Based on our observation, the measured MMDs in this case study is significantly lower than those in case study I. With the distributions of the sources and the target are already similar, performing domain adaptation become less beneficial. While the risk of false alignment of different categories becomes prominent. This explains why negative transfer occurs in this case study. Using our proposed MMD-based criterion, WDAN's negative transfer on ST-5 could be avoid. A critical issue is to select a proper threshold. If we use a threshold value of 0.07, source-only method will be executed only for ST-5, eliminating the negative transfer for our WDAN method. For other datasets, the optimal choice of the threshold value may be different.

We select ST-6 to study the influence of β in this case study and plot the weights, accuracies and GNs in Fig. 6. We can see that the best β is 100, among 0, 10, 100, and 1000. This is different to ST-1 where WDAN-10 produces the highest accuracy

Table 4
Target domain test accuracies and training time costs on UofA dataset.

Method	ST-4 (%)	ST-5 (%)	ST-6 (%)	Training time (seconds)
source-only	83.689 ± 4.643	**91.138** ± 2.460	83.929 ± 6.583	50.58 ± 0.21
merge-as-one [17]	81.831 ± 5.708	85.084 ± 9.803	78.031 ± 9.031	52.63 ± 0.22
best-single [20]	83.360 ± 6.127	88.196 ± 2.038	87.244 ± 3.104	103.82 ± 0.59
eqaul-weigthing [30]	83.996 ± 3.649	89.964 ± 2.867	86.476 ± 4.296	79.32 ± 0.26
WDAN-10 (proposed*)	**85.591** ± 2.696	90.382 ± 3.475	**87.316** ± 6.837	81.64 ± 0.23

* Applied regardless of the MMD-based threshold for avoiding negative transfer

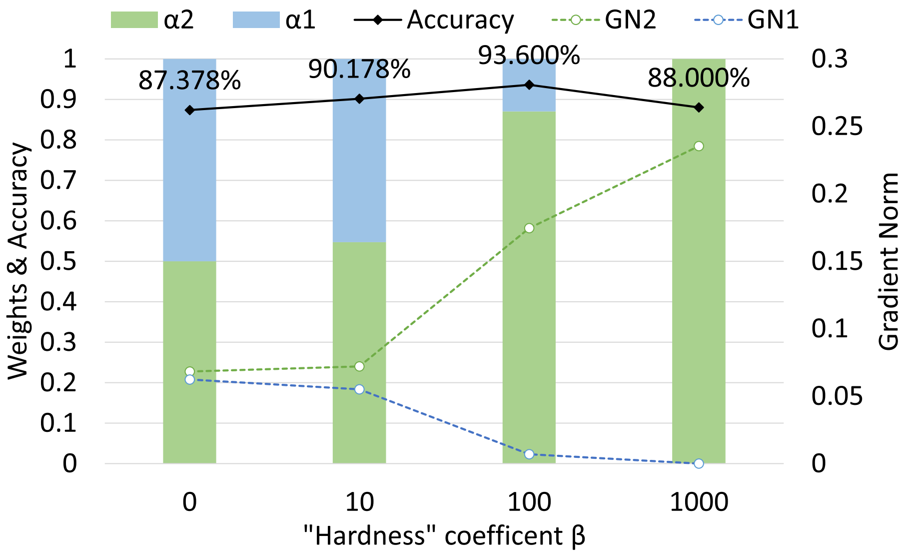

Fig. 6. Influence of β on ST-6. Load levels of S_1: 3%, S_2: 8%, T: 13%.

and WDAN-100 may assign zero weight values. For ST-6, the difference between domains in terms of their MMDs is smaller than it for ST-1. This explains why domain adaptation can gain less accuracies on the UofA dataset than on the THU dataset. Confusion matrices of the source-only method and WDAN on ST-4 are shown in Fig. 7. The class labels "H" stands for healthy and "C1" to "C5" denote crack level 1 to level 5. Comparing the two matrices, we can see that the efficacy of domain adaption

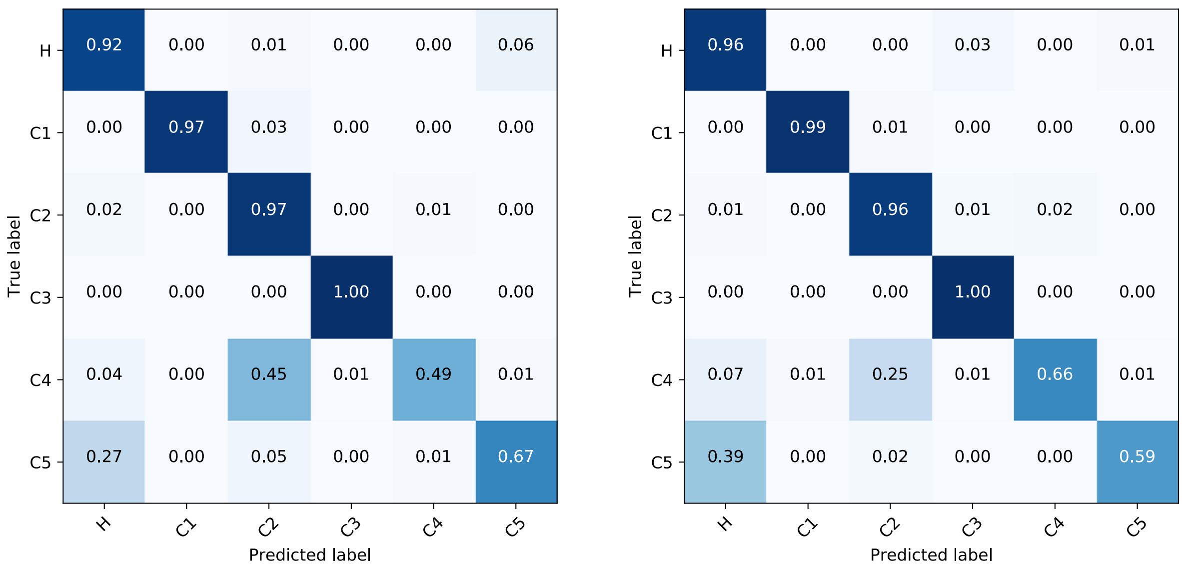

Fig. 7. Confusion matrices on ST-4. **Left**: source-only; **Right**: WDAN-10 (proposed).

(WDAN) is divided for different classes. WDAN performs better than source-only on "H", "C1", and "C4" while loses accuracy on "C2" and "C5". This indicates that WDAN may correctly or falsely align the source and target samples for different classes. In ST-4, the benefit of applying WDAN outweighs the risk of false alignment as a higher overall accuracy can be obtained.

As shown in the right-hand side of Table 4, for the three tasks in this case study, WDAN spends about 2.32 s more than equal-weighting does to compute the weights. In addition, source-only needs about 50.58s for 25 epochs of training while WDAN takes about 81.64s.

5. Summary and conclusion

The main contributions of this work are summarized as follows: 1) A multiple source domain adaptation method is presented for mechanical fault diagnosis tasks. Compared with other related domain adaptation methods, the proposed is more suitable and powerful for industrial applications; 2) Different weights on different source domains are assigned during model training. A balance weighting between uniform weighting and emphasizing on a single source is optimal. The assigned weights by MMD are demonstrated to be in accordance with physical meanings (speed and load level) of the domains; 3) MMD values can also help us avoid negative transfer.

Beyond combating working condition changes, a wider scope of applications can be considered, such as adaptation across different machines. To successfully avoid negative transfer, we need a good method to set proper threshold for our proposed MMD-based criterion. On the probability metric, this study limits to MMD, while other metric such as KL-divergence and K-S statistic, can be considered. Better and task-specific neural network structures may be searched to further boost fault classification accuracy.

CRediT authorship contribution statement

Dongdong Wei: Conceptualization, Methodology, Software, Investigation, Visualization, Writing - original draft. **Te Han:** Resources, Data curation, Writing - review & editing. **Fulei Chu:** Funding acquisition, Writing - review & editing. **Ming Jian Zuo:** Supervision, Resources, Funding acquisition, Writing - review & editing.

Declaration of Competing Interest

The authors 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

Financial support from Future Energy Systems under Canada First Research Excellent Fund (FES-T11-P01, FES-T14-P02, FES-T14-T01), Natural Sciences and Engineering Research Council of Canada (Grant #RGPIN-2015-04897), and China Scholarship Council (Grant #201806070147) is acknowledged. Comments and suggestions from reviewers and the Editor are very much appreciated.

References

[1] E.P. Carden, P. Fanning, Vibration Based Condition Monitoring: A Review: Structural Health Monitoringdoi:10.1177/1475921704047500.
[2] B. Yang, Y. Lei, F. Jia, S. Xing, An intelligent fault diagnosis approach based on transfer learning from laboratory bearings to locomotive bearings, Mechanical Systems and Signal Processing 122 (2019) 692–706, https://doi.org/10.1016/j.ymssp.2018.12.051.
[3] 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, Mechanical Systems and Signal Processing 138 (2020), https://doi.org/10.1016/j.ymssp.2019.106587 106587.
[4] T. Han, C. Liu, W. Yang, D. Jiang, Deep transfer network with joint distribution adaptation: A new intelligent fault diagnosis framework for industry application, ISA Transactions 97 (2020) 269–281, https://doi.org/10.1016/j.isatra.2019.08.012.
[5] R. Liu, B. Yang, E. Zio, X. Chen, Artificial intelligence for fault diagnosis of rotating machinery: A review, Mechanical Systems and Signal Processing 108 (2018) 33–47, https://doi.org/10.1016/j.ymssp.2018.02.016.
[6] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, R.X. Gao, Deep learning and its applications to machine health monitoring, Mechanical Systems and Signal Processing 115 (2019) 213–237, https://doi.org/10.1016/j.ymssp.2018.05.050.
[7] Y. Chen, X. Liang, M.J. Zuo, Time series modeling of vibration signals from a gearbox under varying speed and load condition, IEEE International Conference on Prognostics and Health Management (ICPHM) 2018 (2018) 1–7, https://doi.org/10.1109/ICPHM.2018.8449003.
[8] M. Rao, M.J. Zuo, A new strategy for rotating machinery fault diagnosis under varying speed conditions based on deep neural networks and order tracking, in: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, pp. 1214–1218, https://doi.org/10.1109/ICMLA.2018.00197.
[9] S. Khan, T. Yairi, A review on the application of deep learning in system health management, Mechanical Systems and Signal Processing 107 (2018) 241–265, https://doi.org/10.1016/j.ymssp.2017.11.024.
[10] F. Jia, Y. Lei, J. Lin, X. Zhou, N. Lu, Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data, Mechanical Systems and Signal Processing 72–73 (2016) 303–315, https://doi.org/10.1016/j.ymssp.2015.10.025.
[11] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, S. Van Hoecke, Convolutional Neural Network Based Fault Detection for Rotating Machinery, Journal of Sound and Vibration 377 (2016) 331–345, https://doi.org/10.1016/j.jsv.2016.05.027.
[12] L. Li, M. Zhang, K. Wang, A Fault Diagnostic Scheme Based on Capsule Network for Rolling Bearing under Different Rotational Speeds, Sensors (Basel, Switzerland) 20 (7). doi:10.3390/s20071841.

[13] W. Lu, B. Liang, Y. Cheng, D. Meng, J. Yang, T. Zhang, Deep model based domain adaptation for fault diagnosis, IEEE Transactions on Industrial Electronics 64 (3) (2017) 2296–2305, https://doi.org/10.1109/TIE.2016.2627020.
[14] S. Ma, F. Chu, Ensemble deep learning-based fault diagnosis of rotor bearing systems, Computers in Industry 105 (2019) 143–152, https://doi.org/10.1016/j.compind.2018.12.012.
[15] D. Wei, K. Wang, S. Heyns, M.J. Zuo, Convolutional neural networks for fault diagnosis using rotating speed normalized vibration, in: A. Fernandez Del Rincon, F. Viadero Rueda, F. Chaari, R. Zimroz, M. Haddar (Eds.), Advances in Condition Monitoring of Machinery in Non-Stationary Operations, Applied Condition Monitoring, Springer International Publishing, Cham, 2019, pp. 67–76. doi:10.1007/978-3-030-11220-2_8.
[16] S. Legg, M. Hutter, A collection of definitions of intelligence, in: Proceedings of the 2007 Conference on Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms: Proceedings of the AGI Workshop 2006, IOS Press, NLD, 2007, pp. 17–24.
[17] T. Han, C. Liu, W. Yang, D. Jiang, A novel adversarial learning framework in deep convolutional neural network for intelligent diagnosis of mechanical faults, Knowledge-Based Systems 165 (2019) 474–487, https://doi.org/10.1016/j.knosys.2018.12.019.
[18] W. Zhang, C. Li, G. Peng, Y. Chen, Z. Zhang, A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load, Mechanical Systems and Signal Processing 100 (2018) 439–453, https://doi.org/10.1016/j.ymssp.2017.06.022.
[19] M. Long, H. Zhu, J. Wang, M.I. Jordan, Unsupervised domain adaptation with residual transfer networks, in: D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29, Curran Associates Inc, 2016, pp. 136–144.
[20] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: International Conference on Machine Learning, 2015, pp. 1180–1189.
[21] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1345–1359, https://doi.org/10.1109/TKDE.2009.191.
[22] B. Zhang, W. Li, X.-L. Li, S.-K. Ng, Intelligent fault diagnosis under varying working conditions based on domain adaptive convolutional neural networks, IEEE Access 6 (2018) 66367–66384, https://doi.org/10.1109/ACCESS.2018.2878491.
[23] X. Li, W. Zhang, Q. Ding, J.-Q. Sun, Multi-Layer domain adaptation method for rolling bearing fault diagnosis, Signal Processing 157 (2019) 180–197, https://doi.org/10.1016/j.sigpro.2018.12.005.
[24] X. Li, W. Zhang, Q. Ding, Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks, IEEE Transactions on Industrial Electronics 66 (7) (2019) 5525–5534, https://doi.org/10.1109/TIE.2018.2868023.
[25] L. Guo, Y. Lei, S. Xing, T. Yan, N. Li, Deep convolutional transfer learning network: a new method for intelligent fault diagnosis of machines with unlabeled data, IEEE Transactions on Industrial Electronics 66 (9) (2019) 7316–7325, https://doi.org/10.1109/TIE.2018.2877090.
[26] Y. Mansour, M. Mohri, A. Rostamizadeh, Domain adaptation with multiple sources, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems 21, Curran Associates Inc, 2009, pp. 1041–1048.
[27] R. Chattopadhyay, Q. Sun, W. Fan, I. Davidson, S. Panchanathan, J. Ye, Multisource domain adaptation and its application to early detection of fatigue, ACM Transactions on Knowledge Discovery from Data 6 (4). doi:10.1145/2382577.2382582.
[28] R. Xu, Z. Chen, W. Zuo, J. Yan, L. Lin, Deep cocktail network: multi-source unsupervised domain adaptation with category shift, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3964–3973.
[29] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, B. Wang, Moment matching for multi-source domain adaptation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1406–1415.
[30] H. Zhao, S. Zhang, G. Wu, J.M.F. Moura, J.P. Costeira, G.J. Gordon, Adversarial multiple source domain adaptation, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31, Curran Associates Inc, 2018, pp. 8559–8570.
[31] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J. Wortman, Learning bounds for domain adaptation, in: J.C. Platt, D. Koller, Y. Singer, S.T. Roweis (Eds.), Advances in Neural Information Processing Systems 20, Curran Associates Inc, 2008, pp. 129–136.
[32] J. Jiang, C. Zhai, Instance weighting for domain adaptation in NLP, in: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Association for Computational Linguistics, Prague, Czech Republic, 2007, pp. 264–271.
[33] L. Gui, R. Xu, Q. Lu, J. Du, Y. Zhou, Negative transfer detection in transductive transfer learning, International Journal of Machine Learning and Cybernetics 9 (2) (2018) 185–197, https://doi.org/10.1007/s13042-016-0634-8.
[34] H. Rhee, N.I. Cho, Efficient and robust pseudo-labeling for unsupervised domain adaptation, in: 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2019, pp. 980–985, https://doi.org/10.1109/APSIPAASC47483.2019.9023239.
[35] Y. Gao, A.J. Ma, Y. Gao, J. Wang, Y. Pan, Adversarial open set domain adaptation via progressive selection of transferable target samples, Neurocomputing 410 (2020) 174–184, https://doi.org/10.1016/j.neucom.2020.05.032.
[36] E. Tzeng, J. Hoffman, T. Darrell, K. Saenko, Simultaneous deep transfer across domains and tasks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4068–4076.
[37] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[38] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, A.J. Smola, A Kernel method for the two-sample-problem, in: B. Schölkopf, J.C. Platt, T. Hoffman (Eds.), Advances in Neural Information Processing Systems 19, MIT Press, 2007, pp. 513–520.
[39] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, B.K. Sriperumbudur, Optimal kernel choice for large-scale two-sample tests, in: F. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates Inc, 2012, pp. 1205–1213.
[40] M. Long, Y. Cao, J. Wang, M. Jordan, Learning Transferable Features with Deep Adaptation Networks, in: International Conference on Machine Learning, 2015, pp. 97–105.
[41] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324, https://doi.org/10.1109/5.726791.
[42] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 315–323.
[43] I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning, in: International Conference on Machine Learning, 2013, pp. 1139–1147.
[44] A.C. Wilson, R. Roelofs, M. Stern, N. Srebro, B. Recht, The marginal value of adaptive gradient methods in machine learning, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, vol. 30, Curran Associates Inc, 2017, pp. 4148–4158. URL: https://proceedings.neurips.cc/paper/2017/file/81b3833e2504647f9d794f7d7b9bf341-Paper.pdf.
[45] P. Zhou, J. Feng, C. Ma, C. Xiong, S.C.H. Hoi, W. E, Towards Theoretically Understanding Why Sgd Generalizes Better Than Adam in Deep Learning, https://papers.nips.cc/paper/2020/hash/f3f27a324736617f20abbf2ffd806f6d-Abstract.html (2020).
[46] Y. Chen, X. Liang, M.J. Zuo, An improved singular value decomposition-based method for gear tooth crack detection and severity assessment, Journal of Sound and Vibration 468 (2020), https://doi.org/10.1016/j.jsv.2019.115068 115068.
[47] M. Rao, Q. Li, D. Wei, M.J. Zuo, A deep bi-directional long short-term memory model for automatic rotating speed extraction from raw vibration signals, Measurement 158 (2020), https://doi.org/10.1016/j.measurement.2020.107719 107719.
[48] Z. Pei, Z. Cao, M. Long, J. Wang, Multi-adversarial domain adaptation, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.