This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

An Optimal Unsupervised Domain Adaptation Approach With Applications to Pipeline Fault Diagnosis: Balancing Invariance and Variance

***Abstract*—A practical yet challenging scenario in transfer learning is unsupervised domain adaptation (UDA), where knowledge is transferred from a labeled source domain to unlabeled target domains. The crucially important role of domain-variant characteristics is often neglected by most existing UDA methods, which can deteriorate adaptation performance and result in negative transfer. In this paper, an optimal unsupervised domain adaptation (OUDA) algorithm is proposed in order to address this issue, which balances the invariance of domain-sharing features and the variance of domain-specific features. In the proposed approach, a gradient adversarial adaptation (GAA) method is introduced to align the gradient directions of source and target features within the same category, thereby facilitating knowledge transfer. Additionally, a local manifold embedding (LME) technique is proposed to preserve the intrinsic geometric structure of the original feature space while implementing distribution alignment, providing distinguishable features for UDA. To stabilize the process of knowledge transfer, an evolutionary control strategy is developed to adaptively control the tradeoff between the GAA and LME by employing the particle swarm optimization algorithm. Extensive experiments are conducted on cross-domain natural gas pipeline fault diagnosis, and the results on nine cross-domain classification tasks indicate that our OUDA algorithm outperforms the existing state-of-the-art UDA methods. Moreover, the performance analysis in terms of accuracy, loss, and domain divergence demonstrates the superior stability of the proposed OUDA algorithm in dealing with unsupervised knowledge transfer.**

***Index Terms*—Unsupervised domain adaptation, invariance and variance, evolutionary computation, manifold learning.**

I. Introduction

Data-driven deep learning techniques have achieved impressive performance in numerous fields including image identification [13], [18], [28], semantic segmentation [24], [34], fault diagnosis [8], [43], [44], and object detection [33], [35]. Despite this remarkable success, existing studies still suffer from undesired performance degradation when deployed in real industrial scenarios owing to the difficulty in acquiring massive amounts of labeled data [32]. In such cases, leveraging the rich knowledge from the large-scale labeled dataset (i.e., source domain) to accomplish tasks on the small-scale unlabeled dataset (i.e., target domain) is a straightforward and effective way. Such a learning paradigm is referred to as transfer learning (TL). In general, it is meaningless to learn transferable knowledge since the data distributions of the source and target domains might be unidentical and even significantly different [31]. Accordingly, unsupervised domain adaptation (UDA) methods, as specialized forms of TL, have been developed to alleviate the domain shift issue.

The majority of existing UDA methods have been developed to combat distribution discrepancy, e.g. marginal distribution [31], conditional distribution [15], and joint distribution [16], by extracting domain-invariant but discriminative features. In this case, metric learning-based UDA, as one category of UDA method, aims to estimate domain discrepancy quantitatively by utilizing different statistical metrics (e.g. maximum mean discrepancy (MMD) [23], local MMD (LMMD) [14], and multi-kernel MMD (MK-MMD) [41]) and then optimize these statistical metrics to match the distributions across domains. For instance, in [31], a deep TL method has been proposed to minimize the feature discrepancy across domains by optimizing the MMD loss. In [41], a novel TL framework integrating prior knowledge has been introduced to improve adaptation performance by minimizing the MK-MMD. Moreover, in [14], feature transferability has been enhanced by using the LMMD through a deep adversarial subdomain adaptation network (DASAN).

Adversarial learning-based UDA is another type of UDA method, which generates domain-confused features through a two-player min-max game [5]. For instance, a shared feature generator and a double task-specific classifier have been introduced in [11] to learn class-separable and domain-general features via an adversarial adaptation network. Furthermore, in [10], a cycle-consistent adversarial adaptation network has been proposed to learn transferable features through an adversarial game between the domain discriminator and the feature extractor.

Despite significant progress in UDA, the two core challenges of knowledge transfer have not been fully overcome by existing approaches: 1) preserving domain-specific characteristics, and 2) balancing the invariance of domain-invariant features (DIFs) and the variance of domain-specific characteristics. Regarding the first challenge, the prerequisite operation for learning the DIFs is to project the source and target instances into a shared latent space through multi-layer nonlinear mapping, and then reduce the distribution discrepancy in this space to maximize the domain similarity. However, this process unconsciously breaks the geometrical structure of the original feature space, resulting in the loss of domain-specific characteristics, as shown in Fig. 1(b). As a consequence, the model trained on the DIFs cannot accurately identify all target data.

Preliminary attempts have recently been made by some studies to maintain the rich class-related information of the target data [9], [39], [48]. For instance, in [39], a co-training framework has been proposed for heterogeneous heuristic DA, where an extra target classifier has been trained to extract category representations. Furthermore, a domain generalization network has been proposed in [48] for fault diagnosis under unknown working conditions, whose main idea is to exploit domain invariance and retain domain specificity simultaneously. Moreover, a deep multi-representations adversarial learning method has been developed in [9] to explore and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

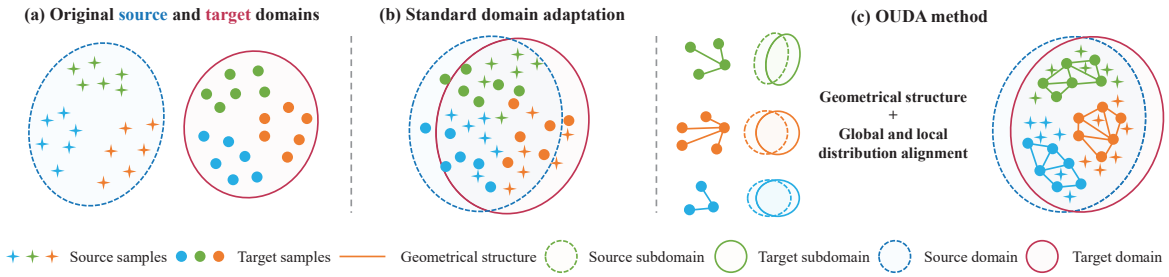

Fig. 1: Example of a failure case of standard DA (middle) and an illustration of our OUDA method (right). Standard DA learns the DIFs by combating the domain distance, which undermines the geometrical structure between target samples and degrades the adaptation performance. Our OUDA method explicitly explores the invariance of DIFs and the variance of domain-specific features, which helps to improve the feature transferability and discriminability.

mitigate the inconsistency between feature transferability and discriminability. Unlike the above studies, we employ the idea of manifold learning to construct stable geometric structures for target features in the latent space. In this way, the model can learn the DIFs without losing discriminative information, thus enhancing the effectiveness of knowledge transfer, as illustrated in Fig. 1(c).

Pertaining to the second challenge, the manifold learning might help to preserve discriminative features but, unfortunately, it overlooks the mutual correlation between invariance and variance, which may cause negative transfer. The invariance of the DIFs ensures that rich knowledge can be successfully transferred from the source domain to the target domain, while the variance of domain-specific features guarantees that the domain-shared classifier can be applied to the target domain with minimal generalization error. Under such circumstances, the excessive learning of domain-sharing features in the latent space breaks the original intrinsic structure, leading to the degraded performance of the model in the target domain. Conversely, the transferability can be negatively affected if the shared feature extractor performs well only in a specific domain. Therefore, the tradeoff between invariance and variance is of vital importance to the stability of knowledge transfer. The particle swarm optimization (PSO) algorithm, as a powerful evolutionary technique widely utilized in parameter optimization of deep learning, can be regarded as an effective solution to deal with such an optimization issue. Accordingly, this paper aims to deploy the PSO algorithm to balance invariance and variance.

With the aforementioned two challenges in mind, an optimal UDA (OUDA) method is proposed in this article, which aims to enhance the transferability of the DIFs while maintaining the discriminability of target domain representations. First, to learn transferable features, a gradient adversarial adaptation (GAA) approach is introduced in the OUDA, inspired by [6]. GAA aligns the feature gradient distributions between two domains, thus reducing domain discrepancies at both global and local levels. Note that pseudo labels are necessary to calculate the gradient of the target domain during the training process. Accordingly, a prototypical pseudo-label (PPL) method is proposed in this paper to assign high-confidence pseudo labels to the target data. Next, to preserve the geometrical structures among the target representations, a local manifold embedding (LME) method is presented to strengthen the correlation between the target instances and their neighbors. Finally, the relative contribution of invariance and variance is controlled by seamlessly embedding a standard PSO algorithm into the OUDA framework.

From the application perspective, pipeline fault diagnosis plays a crucial role in ensuring the high reliability and long-term stability of large-scale oil and gas transportation systems. For intelligent fault diagnosis problems, obtaining large-scale and well-annotated datasets is expensive, time-consuming, and even impractical in real industrial scenarios, and most existing diagnostic methods often suffer from poor accuracy performance and limited generalization ability. To address this challenging issue, we will use the proposed OUDA to learn geometry-aware, transferable, and distinguishable domain-sharing features, and extensive experiments are conducted on cross-domain natural gas pipeline fault diagnosis.

The core contributions of this paper are highlighted as follows.

1) A novel OUDA algorithm is proposed that simultaneously explores invariance and variance to improve feature transferability and discriminability.
2) An LME method is presented to preserve the domain-specific discriminative information by constructing stable geometrical structures between instances and their neighbors, thus enhancing the generalization performance of the source classifier in the target domain.
3) A PSO-based evolutionary control strategy is developed to automatically select reliable hyperparameters for balancing invariance and variance during the training process, which helps to enhance the stability of knowledge transfer.
4) Comprehensive experiments demonstrate that the proposed OUDA algorithm outperforms some existing state-of-the-art UDA algorithms.

The remaining parts of this article are arranged as follows. In Section II, related works regarding manifold learning are discussed. In Section III, a novel OUDA algorithm is described in detail. In Section IV, experimental results and relevant analysis are presented. Finally, conclusions of this article are drawn in Section V.

II. Related Work

A. Unsupervised Domain Adaptation

It is generally assumed in UDA that the feature distribution of the source domain is different from that of the target

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

domain, while learning tasks remain identical. The main purpose of UDA is to learn common characteristics across domains by mitigating domain gaps, so that models trained on the labeled source domain can generalize well on the unlabeled target domain [12]. Recent work on UDA can be categorized into two groups: metric learning-based UDA and adversarial learning-based UDA.

In metric learning-based UDA, the DIFs are learned by matching the statistical moments between two distributions. MMD is a widely utilized criterion that projects instances into a reproducing kernel Hilbert space (RKHS) and then measures the mean discrepancy. For example, a transfer component analysis method has been developed in [23] to learn the cross-domain transfer components in RKHS by using MMD. Moreover, a two-stage TL framework has been developed in [31] to minimize the hidden representational discrepancies through MMD. Since MMD is a first-order moment matching method, the correlation alignment (CORAL) metric has been introduced in [26] to align the first-order statistic (mean) and the second-order statistic (covariance) between two domains through a nonlinear transformation. After that, the CORAL problem has been generalized to infinite-dimensional feature spaces in [47], where the distributions across domains are matched by aligning the RKHS covariance matrices. Furthermore, the central moment discrepancy (CMD) inspired by the interpretation of MMD has been proposed in [42] to match the higher-order central moments of the feature distributions, which further reduces the computational complexity and improves the evaluation precision. In the metric learning-based UDA approach, selecting an appropriate criterion is critical for transferability improvement.

The idea behind adversarial learning-based UDA is borrowed from generative adversarial networks (GANs) to generate domain-confused features, known as domain adversarial adaptation [5], [30]. More specifically, the generator aims to extract common characteristics from the source and target domains to fool the domain discriminator, while the discriminator focuses on accurately predicting the domain labels of the input data. In the training process, the generator and discriminator compete with each other to learn domain-invariant representations. For example, a discriminative adversarial domain adaptation (DADA) method has been presented in [29] to improve the transferability by forming a mutually inhibitory relation between category and domain labels. Furthermore, a batch spectral penalization method has been presented in [1] to improve the discriminability of transferable features by penalizing the largest singular values. Moreover, an adversarial domain adaptation method has been presented in [38] to mix domains at the pixel and feature levels with well-designed soft domain labels.

B. Manifold Learning

Manifold learning aims to construct a compact and discriminative embedding space by mapping the original high-dimensional data to a low-dimensional manifold, which can reveal the intrinsic features of the high-dimensional data. Some studies have attempted to introduce manifold learning into UDA to learn transferable and discriminative features [17], [45], due to the advantages of manifold learning in ensuring geometric consistency before and after feature mapping. For example, in [36], a deep LogCORAL method has been proposed for visual UDA, where Riemannian distance has been used to replace the Euclidean distance for measuring the first-order and second-order discrepancies between two domains. Moreover, an optimal transport-based domain adaptation method has been presented in [17], where knowledge transfer has been implemented by minimizing the Wasserstein distance between source and target data on a Grassmannian manifold. Most of the existing manifold learning-based UDA methods rely on Grassmannian manifolds to minimize the differences in covariance matrices between two domains. However, our work focuses on constructing stable geometric structures between target instances and their nearest neighbors, on the basis of which the proposed algorithm can learn transferable features.

III. An Evolutionary Unsupervised Domain Adaptation Approach

A. Problem Definition and Model Overview

A domain $\mathbb{D}$ is defined as consisting of domain samples X and their corresponding category labels Y, where $X = \{x_1, x_2, \ldots, x_n\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$. The marginal probability distribution of X is denoted by $P(X)$. In the UDA scenario, we have a labeled source domain $\mathbb{D}_s = \{X_s, Y_s\}$, supported by n_s source samples $\{x_{s,1}, x_{s,2}, \ldots, x_{s,n_s}\}$ and source labels $\{y_{s,1}, y_{s,2}, \ldots, y_{s,n_s}\}$, and an unlabeled target domain $\mathbb{D}_t = \{X_t\}$, with n_t target samples $\{x_{t,1}, x_{t,2}, \ldots, x_{t,n_t}\}$. It should be noted that the distribution of the source and target domains is different, i.e., $P(X_s) \neq P(X_t)$. The goal of UDA is to minimize the distribution discrepancy between $\mathbb{D}_s$ and $\mathbb{D}_t$ so that the model trained on $\mathbb{D}_s$ can accurately predict the labels $\{y_{t,i}\}_{i=1}^{n_t}$ of the target samples.

Based on the above definitions, the OUDA model can be formulated as a feature extractor $F : X \to Z$ (Z denotes feature) that maps the samples to the feature space, a classifier $C : Z \to Y$ that outputs the predicted categories of features, and a domain discriminator $D : Z \to L$ (L denotes domain label) that distinguishes the target samples from the source samples. Specifically, the backbone network of the feature extractor F is a one-dimensional convolutional neural network (1D-CNN), as shown in Fig. 2. The classifier C consists of fully connected (FC) layers, rectified linear unit (ReLU) activation functions, and a LogSoftmax layer, while the domain discriminator D is achieved through the structure of FC→ReLU→FC→ReLU→FC→ReLU→FC→LogSoftmax.

The key idea of the OUDA algorithm is to achieve the learning of geometric-aware, transferable, and discriminable features. The OUDA algorithm comprises three fundamental steps: 1) learning of geometrical structures to enhance discriminability by maximizing geometric consistency between hidden representations and samples; 2) alignment of two-level distributions to boost transferability by simultaneously aligning the marginal and conditional distributions across domains; and 3) optimization of contribution coefficients to improve stability by adaptively adjusting the relative contribution between the invariance of the DIFs and the variance of domain-specific features. The framework of the OUDA algorithm is depicted in Fig. 2.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

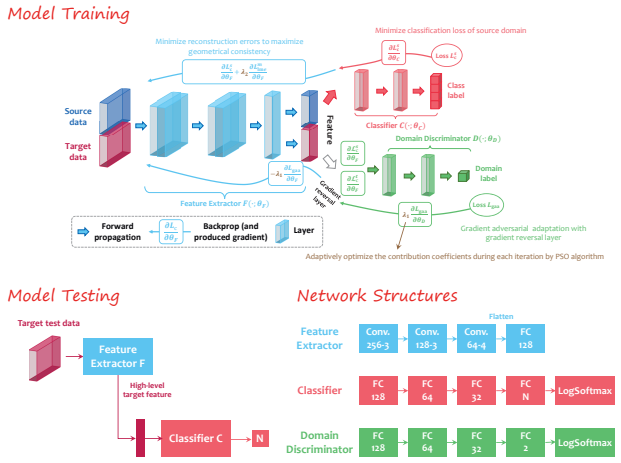

Fig. 2: An overview of the proposed OUDA method in terms of training optimization, testing, and overall network structure.

B. Geometrical Structure Learning

In Fig. 1(b), the common latent space mixes features from different categories, resulting in a degraded cross-domain model performance. This is due to the disruption of intrinsic geometrical structures among the original target samples during distribution alignment. Therefore, one research objective of this paper is to learn the domain characteristics that are aware of the geometry. In recent years, the locally linear embedding (LLE) [25], [40], [46] method has been extensively employed in deep learning because of its ability to preserve the local data structure.

The LME approach proposed in this paper is based on LLE, a manifold learning method that describes the geometrical relationship between each sample and its neighborhood points through local neighborhood linear reconstruction. The proposed approach aims to maintain the inherent relationships among the target samples by maximizing the geometrical consistency before and after feature mapping, thus improving feature discriminability. Specifically, the LME approach maps the target data from the original high-dimensional space to the common latent space while preserving discriminative information.

Let us now illustrate the procedure for implementing the LME method, which consists of four fundamental steps as follows.

1) The *first* step of the LME is to search for the nearest neighbor points of each sample $x_{t,i}$ by K-nearest neighbor (KNN) algorithm [2], which can be formulated as follows:

$$N_{t,i} = \mathrm{KNN}\left(x_{t,i}, K\right), \tag{1}$$

where K denotes the number of nearest neighbors, and $N_{t,i} = [x_{t,i,1}, \ldots, x_{t,i,K}]$ is the matrix of nearest neighbor points for sample $x_{t,i}$.

2) The *second* step is to obtain an optimal weight matrix by minimizing the reconstruction error between each sample and the weighted sum of its nearest neighbors, which is implemented as follows:

$$\min_{\omega} \mathcal{L}_{\mathrm{lme}}^{\mathrm{o}} = \sum_{i=1}^{n_t} \left\| x_{t,i} - \sum_{k=1}^{K} \omega_{i,k} x_{t,i,k} \right\|_2^2, \quad \text{s.t.} \sum_{k=1}^{K} \omega_{i,k} = 1, \tag{2}$$

where $\omega_{i,k} > 0$ represents the weight used for local linear reconstruction, and $\sum_{k=1}^{K} \omega_{i,k}$ is weight normalization.

3) The *third* step of the proposed LME is to map all samples from the high-dimensional space to the low-dimensional space through nonlinear manifold layers, as shown in Fig. 2.

4) The *fourth* step is to minimize the reconstruction error between each low-dimensional representation $F\left(x_{t,i}\right)$ and the weighted sum of its nearest neighbors $F\left(N_{t,i}\right)$. The purpose of this step is to utilize the weight matrix obtained from (2) to find an optimal mapping function that can preserve the geometric structure among the original samples. Therefore, this step can be described as follows:

$$\min_{\theta_F} \mathcal{L}_{\mathrm{lme}}^{\mathrm{m}} = \sum_{i=1}^{n_t} \left\| F\left(x_{t,i}\right) - \sum_{k=1}^{K} \omega_{i,k} F\left(x_{t,i,k}\right) \right\|_2^2, \tag{3}$$

where F is feature extractor constructed by stacking one-dimensional convolutional blocks.

Remark 1: The geometrical structure learning in this paper is based on two key ideas, both of which involve linear reconstruction. The first idea is that each sample in the original space can be reconstructed linearly using the weighted sum of its nearest neighbors in the local region. The second idea is that each sample can still be linearly represented by its nearest neighbors with the same weight matrix as the original space in a compact and discriminative manifold space. Therefore, the LME approach aims to maximize the geometric consistency of the target domain before and after feature mapping by finding the most suitable weight matrix and optimal mapping function. In this way, the geometric structures of the target representations can be preserved in the common latent space.

C. Two-Level Distribution Alignment

The main idea of adversarial adaptation is to learn the DIFs based on the GAN framework so that the classifier trained on a labeled source domain can be well generalized to an unlabeled target domain. Technically, adversarial adaptation introduces a minimax strategy to align the global distribution between the source and target domains. As shown in Fig. 2, unlike the standard GAN algorithm, high-level domain-invariant representations are not generated by the feature extractor F with parameter θ_F, but they are extracted from the source and target samples. Furthermore, the domain discriminator D with parameter θ_D is used to distinguish between high-level representations from the source and target domains. Therefore, the optimization objective of adversarial adaptation can be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

formulated as follows:

$$\min_{\theta_F}\max_{\theta_D}\mathcal{L}_{\text{adv}} = \frac{1}{n_s}\sum_{i=1}^{n_s}\log\left[D\left(F\left(x_{s,i}\right)\right)\right] + \frac{1}{n_t}\sum_{i=1}^{n_t}\log\left[1-D\left(F\left(x_{t,i}\right)\right)\right]. \quad (4)$$

Despite the significant success achieved in aligning feature distributions, adversarial adaptation methods still suffer from the issue of local domain shift. Specifically, these methods primarily reduce the global discrepancy without taking local class-aware information into account. Under such circumstances, although the cross-domain distributions are aligned, the internal structures of different categories are blended, leading to a significant degradation in feature transferability.

To address the above issue, a GAA approach is introduced that simultaneously aligns the global and local distributions between two domains, aiming to explore the invariance of domain-sharing features. The key idea of the GAA approach is to align the feature gradient directions of the source and target samples in the same category. Specifically, according to [6], the feature gradient direction of a sample generally points to its nearest decision region, which is a part of the highly complex decision boundary. From this perspective, samples within the same category should have similar gradient directions, while samples from different categories should have significantly different gradient directions. Therefore, adjusting the feature gradient directions helps to align the decision boundaries of the conditional distributions across domains, which further addresses the local domain shift issue.

Based on the above discussion, the GAA approach generates the DIFs by:

$$\min_{\theta_F}\max_{\theta_D}\mathcal{L}_{\text{gaa}} = \frac{1}{n_s}\sum_{i=1}^{n_s}\log\left[D\left(g\left(x_{s,i},F\right)\right)\right] + \frac{1}{n_t}\sum_{i=1}^{n_t}\log\left[1-D\left(g\left(x_{t,i},F\right)\right)\right], \quad (5)$$

where $g\left(\cdot\right)$ denotes the gradient vector calculated by

$$g\left(x_s,F\right) = \left[\frac{\partial\mathcal{L}_{\text{c}}^s}{\partial F\left(x_{s,1}\right)},\frac{\partial\mathcal{L}_{\text{c}}^s}{\partial F\left(x_{s,2}\right)},\ldots,\frac{\partial\mathcal{L}_{\text{c}}^s}{\partial F\left(x_{s,n_s}\right)}\right], \quad (6)$$

$$g\left(x_t,F\right) = \left[\frac{\partial\mathcal{L}_{\text{c}}^t}{\partial F\left(x_{t,1}\right)},\frac{\partial\mathcal{L}_{\text{c}}^t}{\partial F\left(x_{t,2}\right)},\ldots,\frac{\partial\mathcal{L}_{\text{c}}^t}{\partial F\left(x_{t,n_t}\right)}\right]. \quad (7)$$

Here, $\mathcal{L}_{\text{c}}^s$ and $\mathcal{L}_{\text{c}}^t$ denote losses of the source and target domains respectively, and their details are provided in Section III-D.

Note that to calculate the gradients of $\mathcal{L}_{\text{c}}^t$, the labels of the target data are required. However, in the UDA setting, the real target labels are not available beforehand. Therefore, a PPL approach is designed to obtain robust pseudo labels of the target samples during the training stage. The prototypical networks [27] aim to construct an embedding space where the points of each category are centered around a single prototype representation. The PPL approach borrows this idea to assign pseudo labels for target samples. Technically, the prototype for each category is first calculated by:

$$\mathcal{P}_n^s = \frac{1}{|X_{s,n}|}\sum_{x\in X_{s,n}} f_{\phi^*}\left(F\left(x_s\right)\right), \quad (8)$$

where $X_{s,n}$ denotes the set of source samples belonging to category n, f_{ϕ^*} is embedding function with optimal parameter ϕ^*. Then, we calculate the similarity between the target sample and each source prototype by:

$$\text{Sim}\left(f_{\phi^*}\left(F(x_{t,i})\right),\mathcal{P}_n^s\right) = \frac{f_{\phi^*}\left(F(x_{t,i})\right)\cdot\mathcal{P}_n^s}{\|f_{\phi^*}\left(F(x_{t,i})\right)\|\,\|\mathcal{P}_n^s\|}, \quad (9)$$

where $\text{Sim}\left(\cdot\right)$ denotes the cosine similarity with "$\cdot$" representing the inner product and $\|\cdot\|$ being the Euclidean norm. Finally, we can acquire the pseudo label of the i-th target sample by:

$$\hat{y}_{t,i} = \arg\max_n \text{Sim}\left(f_{\phi^*}\left(F(x_{t,i})\right),\mathcal{P}_n^s\right) \quad (10)$$

for $n\in\{1,2,\ldots,N\}$ where N is the number of categories.

D. Classifier Learning

As shown in Fig. 2, the classifier is first trained to predict the categories of the source samples by exploiting the DIFs, and then tested on the unlabeled target domain. In the proposed OUDA algorithm, the popular cross-entropy loss is employed to improve the prediction accuracy, which can be formulated as follows:

$$\min_{\theta_F,\theta_C}\mathcal{L}_{\text{c}}^s = -\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{n=1}^{N}\mathbb{I}\left\{y_{s,i}\in n\right\}\log\left(C\left(F\left(x_{s,i,n}\right)\right)\right), \quad (11)$$

where $\mathbb{I}\left\{\cdot\right\}$ represents indicator function. $\mathbb{I}\left\{y_{s,i}\in n\right\}=1$ if data $x_{s,i}$ belongs to category n, otherwise $\mathbb{I}\left\{y_{s,i}\in n\right\}=0$. In addition, to implement the GAA approach, the loss between the predicted target labels and the pseudo labels is calculated by:

$$\mathcal{L}_{\text{c}}^t = -\frac{1}{n_t}\sum_{i=1}^{n_t}\sum_{n=1}^{N}\mathbb{I}\left\{\hat{y}_{t,i}\in n\right\}\log\left(C\left(F\left(x_{t,i,n}\right)\right)\right), \quad (12)$$

where $\hat{y}_t$ is the pseudo label of the target sample. It should be noted that the loss of the target domain is only used for feature gradient calculation rather than training the classifier.

E. Contribution Coefficient Optimization

Based on the aforementioned process, the geometric-aware features of the target domain are first learned by the proposed OUDA algorithm. Meanwhile, the marginal and class-conditional distributions between the source and target domains are aligned in a common latent space. The overall objective of the OUDA algorithm can be formulated as follows by incorporating (3), (5), and (11):

$$\mathcal{L}_{\text{OUDA}} = \mathcal{L}_{\text{c}}^s + \lambda_1\mathcal{L}_{\text{gaa}} + \lambda_2\mathcal{L}_{\text{lme}}^{\text{m}}, \quad (13)$$

where the contribution coefficients λ_1 and λ_2 are designed to balance the relative contribution between the invariance of the DIFs and the variance of domain-specific features.

The contribution coefficients, as discussed in Section I, are crucial for ensuring the stability and credibility of the model performance. Expert experience or grid search is generally used to select these coefficients, but both methods have limitations. The former is extremely laborious, while the latter requires significant computational resources and runtime.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

Moreover, due to limited search capabilities, it is challenging to find the optimal combination of parameters λ_1 and λ_2. In the past decade, the PSO algorithm has been widely used for hyperparameter optimization in deep learning due to its easy implementation and fast convergence [4], [20]. Therefore, to address these issues, we incorporate the PSO algorithm into the OUDA framework to adaptively optimize the contribution coefficients in each iteration.

In the PSO algorithm, particles are utilized to explore the optimal solution by constantly updating the velocity vector $v_i\left(t\right)=\left(v_{i1}\left(t\right),v_{i2}\left(t\right),\ldots,v_{iD}\left(t\right)\right)$ and the position vector $x_i\left(t\right)=\left(x_{i1}\left(t\right),x_{i2}\left(t\right),\ldots,x_{iD}\left(t\right)\right)$ in a D-dimensional problem space [3], [19]. Depending on the competition and cooperation among particles, the position of the i-th particle is adjusted towards two directions, where one direction is the personal best position p_{best} represented by $p_{\text{best},i}=(p_{i1},p_{i2},\ldots,p_{iD})$ and the other direction is the global best position g_{best} represented by $g_{\text{best}}=(g_1,g_2,\ldots,g_D)$. Formally, the velocity and position of the i-th particle in the (t+1)-th iteration are updated as follows:

$$\begin{cases} v_i(t+1) = \omega v_i(t) + c_1 r_1(p_{\text{best},i} - x_i(t)) \\ \qquad\qquad + c_2 r_2(g_{\text{best}} - x_i(t)), \\ x_i(t+1) = x_i(t) + v_i(t+1), \end{cases} \tag{14}$$

where t is the current iteration number of the i-th particle in the D-dimensional problem space, ω is the inertia weight, c_1 and c_2 are cognitive and social parameters, and r_1 and r_2 are the constants selected on the interval [0, 1].

In this paper, the joint optimization of the contribution coefficients λ_1 for domain invariance and λ_2 for domain variance is achieved by employing a standard PSO algorithm within the OUDA framework. The detailed process regarding the contribution coefficient optimization is provided as follows.

Step. 1: The parameters of the PSO algorithm, including the dimension $D=2$ of the problem space, the population size $N=20$, the maximum number $t_{\max}=5$ of iterations, and initial velocity $v_i(0)$ and position $x_i(0)$ of i-th particle, are initialized. Additionally, the constraint intervals of the two contribution coefficients are set to [0, 1] and [0, 1], respectively.

Step. 2: The inertia weight ω=0.75 and acceleration coefficients c_1=2.0 and c_2=2.0 are set.

Step. 3: The personal best position $p_{\text{best},i}$ and global best position g_{best} are calculated via the fitness function $\mathcal{L}_{\text{OUDA}}$.

Step. 4: The velocity and position are updated by (14) until convergence.

Step. 5: The best combination $[\lambda_1,\lambda_2]$ of contribution coefficients is obtained.

IV. Simulation Experiments

In this section, the proposed OUDA method is validated on two natural gas pipeline datasets comprising a negative pressure wave dataset (NPWD) and an acoustic wave dataset (AWD). Specifically, we first test the performance of the OUDA method on the UDA tasks of Case 1 and Case 2, and compare it with seven state-of-the-art UDA methods. The detailed descriptions of Case 1 and Case 2 will be presented in Section IV-A.

A. Data Description and Task Setting

1) NPWD: The NPWD is collected by a ZJ-CSGD-type pipeline fault simulation platform, as shown in Fig. 3. The pipeline length is 180.2 m, the flow rate is $10\text{m}^3/\text{h}$, and the sampling frequency is 1024 Hz. The NPWD contains four categories of pipeline data (large leakage, medium leakage, small leakage, and normal state) from three domains, including high pressure (HP), medium pressure (MP), and low pressure (LP).

2) AWD: The AWD is collected from acoustic sensors on the HD-II-type pipeline fault simulation platform, as shown in Fig. 4. The pipeline length is 160 m, the flow rate is $60\text{m}^3/\text{h}$, the sampling frequency is 5000 Hz, and the operating pressure is 0.5MPa. On the one hand, similar to the NPWD, the AWD can be grouped into four categories, i.e., large leakage, medium leakage, small leakage, and normal state. On the other hand, unlike the NPWD, the acoustic signals in the AWD are monitored at a fixed pressure condition.

(a) (b)

Fig. 3: (a) Console. (b) Pipeline platform.

(a) (b)

Fig. 4: (a) Console. (b) Pipeline platform.

3) Task Setting: The NPWD is a comprehensive pipeline dataset consisting of three domains (HP, MP, and LP) with a total of 19,200 samples in four categories. Therefore, six different UDA tasks are generated from the three domains: HP $\rightarrow$ MP, HP $\rightarrow$ LP, MP $\rightarrow$ HP, MP $\rightarrow$ LP, LP $\rightarrow$ HP, and LP $\rightarrow$ MP. In this paper, we refer to the knowledge transfer between different domains in the NPWD as Case 1.

The AWD contains a total of 4800 pipeline data in four categories. In this paper, the AWD is only utilized as a source domain. Thus, three UDA tasks can be constructed: AWD $\rightarrow$ HP, AWD $\rightarrow$ MP, and AWD $\rightarrow$ LP. We refer to the knowledge transfer from the AWD to the NPWD as Case 2.

B. Baseline Methods

In this part, we compare the OUDA method with some representative UDA methods, the details of which are presented as follows.

- *DTL:* Knowledge transfer by optimizing MMD [37].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

TABLE I: Precision, recall, and accuracy (%) of the proposed OUDA method in Case 1.

Tasks	Metric	DTL	DANN	JAN	DCTLN	DASAN	DRMEA	FGDA	OUDA(ours)
HP → MP	Precision	85.99±0.29	84.05±0.52	87.08±0.17	86.36±0.17	86.59±0.41	**92.67±0.25**	**92.32±0.29**	**98.19±0.16**
	Recall	84.12±0.37	83.47±0.50	85.78±0.17	85.21±0.16	86.47±0.44	**90.80±0.26**	**92.17±0.27**	**98.08±0.17**
	Accuracy	84.11±0.37	83.50±0.58	85.69±0.20	85.31±0.21	86.48±0.50	**90.82±0.38**	**92.17±0.30**	**98.10±0.17**
HP → LP	Precision	85.54±0.23	88.50±0.21	85.62±0.40	88.49±0.35	91.20±0.39	**95.88±0.20**	**94.88±0.28**	**98.42±0.20**
	Recall	82.12±0.10	78.99±0.26	84.26±0.32	87.34±0.45	90.72±0.33	**95.50±0.26**	**94.69±0.26**	**98.42±0.21**
	Accuracy	82.17±0.22	79.31±0.50	84.21±0.42	87.36±0.42	90.71±0.36	**95.47±0.24**	**94.71±0.29**	**98.41±0.20**
MP → HP	Precision	86.90±0.41	85.12±0.28	88.04±0.26	89.68±0.25	91.04±0.17	**97.30±0.18**	**94.47±0.32**	**97.37±0.20**
	Recall	85.30±0.27	84.15±0.31	86.09±0.27	89.02±0.22	90.76±0.25	**97.07±0.21**	**94.48±0.31**	**97.31±0.23**
	Accuracy	85.27±0.30	84.16±0.28	86.14±0.30	89.01±0.28	90.80±0.24	**97.09±0.17**	**94.49±0.37**	**97.31±0.22**
MP → LP	Precision	86.25±0.24	86.26±0.40	87.54±0.82	87.52±0.23	87.16±0.38	**92.12±0.22**	**95.14±0.21**	**98.62±0.15**
	Recall	84.04±0.34	79.11±0.39	86.06±0.52	87.32±0.24	86.85±0.33	**91.10±0.31**	**94.95±0.23**	**98.58±0.17**
	Accuracy	84.04±0.34	79.16±0.64	86.08±0.38	87.31±0.34	86.88±0.31	**91.09±0.28**	**94.96±0.22**	**98.58±0.17**
LP → HP	Precision	84.57±0.31	89.20±0.18	85.95±0.37	83.27±0.24	**93.84±0.33**	90.88±0.43	**97.78±0.12**	**97.73±0.17**
	Recall	81.06±0.34	83.03±0.21	84.47±0.45	81.72±0.20	**93.51±0.34**	89.33±0.49	**97.66±0.12**	**97.58±0.19**
	Accuracy	81.11±0.30	82.85±0.34	84.48±0.41	81.74±0.34	**93.53±0.36**	89.31±0.55	**97.67±0.14**	**97.59±0.19**
LP → MP	Precision	84.25±0.33	83.65±0.38	85.50±0.25	89.88±0.35	**93.65±0.37**	**94.50±0.31**	93.02±0.45	**96.49±0.13**
	Recall	82.36±0.31	82.16±0.45	84.30±0.30	89.01±0.39	**92.75±0.36**	**93.62±0.24**	92.53±0.43	**96.40±0.15**
	Accuracy	82.36±0.33	81.38±0.52	84.30±0.35	88.98±0.38	**92.75±0.40**	**93.60±0.41**	92.51±0.50	**96.42±0.16**

- *DANN:* Adaptation with adversarial learning [5].
- *JAN:* Adaptation employing JMMD metric and adversarial training strategy [16].
- *DCTLN:* Knowledge transfer by maximizing domain recognition error and minimizing probability distribution distance [7].
- *DASAN:* Adaptation by simultaneously alleviating domain shift at the category and domain levels [14].
- *DRMEA:* Adaptation with a Riemannian manifold embedding and alignment framework [21].
- *FGDA:* Adaptation employing feature gradient distribution alignment [6].

C. Experiment Configurations

Following the standard UDA paradigm, we train a cross-domain classification model with a labeled source domain and an unlabeled target domain. We employ 1D-CNN as the backbone to construct components for all the compared algorithms. Taking the OUDA method as an example, 1D-CNN is applied as a feature extractor. Next, we build the classifier and domain discriminator in the OUDA method by utilizing task-specific fully connected layers, as illustrated in Fig. 2. In the experiments, the training data is resized to 1×1024 and normalized to the range of [0, 1].

In network optimization, the back propagation algorithm is applied to update the model parameters, and mini-batch stochastic gradient descent is employed as an optimizer to minimize the model loss, where the initial learning rate, momentum, and weight decay are set to 0.01, 0.9, and 1e-3, respectively. In this paper, all balance hyper-parameters in the OUDA method are adaptively updated by the PSO algorithm, whose parameters are illustrated in Section III-E. Furthermore, the batch size is set to 64 for all domains, and the maximum number of iterations is set to 5000. Moreover, the proposed OUDA method and other compared UDA approaches are implemented with PyTorch using NVIDIA GEFORCE RTX 3090 GPU, Intel(R) Core(TM) i9-10900k, 3.70-GHz CPU.

D. Experimental Results of Case 1

To avoid the negative effect of randomness, we repeat all methods ten times in this experiment and report the mean accuracy and standard deviation. Table I shows the results of the OUDA method and compared methods on the six tasks in Case 1. Red, green, and purple indicate the best result, the second-best result, and the third-best result, respectively. Based on the observation of Table I, we can draw the following conclusions.

- First, the methods that take advantage of fine-grained categorical information perform better than the basic methods. Concretely, JAN, DASAN, DRMEA, FGDA, and the OUDA significantly improve Case 1 by an accuracy of more than 3%, which implies that aligning the conditional distributions across domains is able to effectively reduce domain discrepancy, thus improving the target classification accuracy of the diagnostic model trained on the source domain.
- Second, the discriminability of the DIFs tends to deteriorate when the model directly uses the pseudo labels predicted by the source classifier without considering the domain shift. Specifically, the performance of JAN and DASAN is not satisfactory in Case 1 due to the error and instability brought by the pseudo labels from the source classifier.
- Third, preserving the intrinsic geometric structure of the target domain while learning the DIFs can considerably enhance the performance of the source classifier in the target domain. Specifically, DRMEA and our OUDA achieve much better performance than the basic methods, and even better than some advanced methods that adopt sub-domain alignment strategies, such as JAN, DCTLN, and DASAN.
- Fourth, the proposed OUDA method outperforms all compared methods and achieves the best performance on five tasks. For instance, compared to the sub-optimal DRMEA algorithm, the OUDA method achieves the best experimental results with accuracy improvements of 7.28%, 2.94%, 0.22%, 7.49%, and 2.82% on tasks HP

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

TABLE II: Precision, recall, and accuracy (%) of the proposed OUDA method in Case 2.

Tasks	Metric	DTL	DANN	JAN	DCTLN	DASAN	DRMEA	FGDA	OUDA
AWD → HP	Precision	72.68±0.57	80.06±0.23	82.80±0.13	86.66±0.33	89.44±0.55	**91.13±0.39**	**92.04±0.26**	**93.92±0.29**
	Recall	72.11±0.52	78.14±0.25	81.26±0.13	83.97±0.40	88.62±0.51	**89.13±0.29**	**89.15±0.16**	**93.68±0.24**
	Accuracy	72.11±0.57	78.10±0.22	81.23±0.18	84.13±0.43	88.60±0.46	**89.12±0.43**	**89.14±0.28**	**93.66±0.32**
AWD → MP	Precision	81.77±0.41	84.71±0.29	84.56±0.33	84.08±0.30	**89.62±0.40**	**87.93±0.25**	87.70±0.25	**93.30±0.27**
	Recall	76.50±0.36	75.64±0.33	83.59±0.38	83.64±0.29	**89.08±0.43**	86.89±0.30	**87.52±0.27**	**92.43±0.26**
	Accuracy	76.53±0.42	75.95±0.47	83.53±0.47	83.62±0.36	**89.13±0.38**	86.78±0.41	**87.54±0.38**	**92.44±0.29**
AWD → LP	Precision	75.61±0.37	82.64±0.38	83.09±0.27	87.17±0.41	88.18±0.34	**91.03±0.20**	**88.39±0.25**	**94.89±0.21**
	Recall	75.48±0.35	76.01±0.54	82.54±0.29	84.89±0.35	**87.12±0.38**	**88.99±0.18**	86.04±0.16	**93.77±0.28**
	Accuracy	75.49±0.35	76.10±0.51	82.59±0.35	84.82±0.49	**87.06±0.41**	**88.93±0.29**	86.11±0.25	**93.80±0.29**

→ MP, HP → LP, MP → HP, MP → LP, and LP → MP, respectively. While on task LP → HP, the proposed OUDA method loses to FGDA by a narrow margin of 0.08%. This is because the difference between the LP and HP domains is much larger than any of the two domains in Case 1 (domain difference is irreversible). Accordingly, the feature extractor makes more effort to learn the DIFs in this task than in other tasks, which leads to the degradation of discriminability.

To sum up, the OUDA method surpasses all the compared methods and achieves the highest classification accuracies in Case 1. The superior performance of the OUDA method can be attributed to the fact that the proposed strategies adequately learn the DIFs and reasonably reserve the domain-variant structural information.

To verify the superiority of the OUDA method over compared algorithms, we carry out the statistical significance test on the mean accuracies in Table I. When the p-value < 0.05, it implies that the OUDA method performs significantly better than a compared algorithm. As can be observed in Fig. 5, the OUDA method is significantly different from the compared methods, which indicates that our method greatly enhances the classification accuracy.

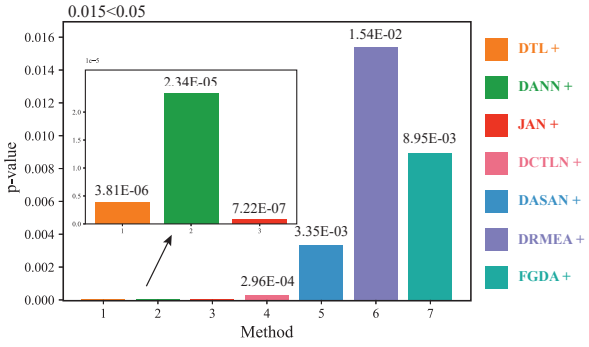

Fig. 5: Results of paired t-test for Case 1.

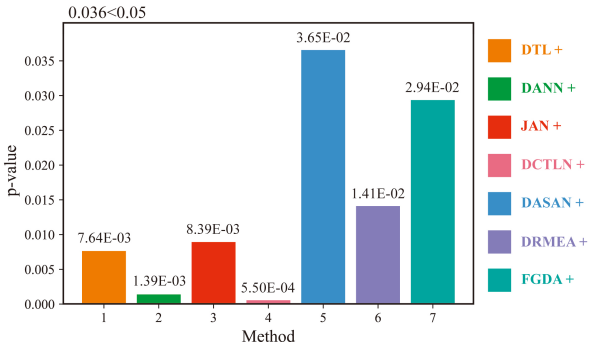

Fig. 6: Results of paired t-test for Case 2.

E. Experimental Results of Case 2

Table II presents the experimental results of the OUDA method and all compared methods on three tasks in Case 2. The shape and amplitude of the acoustic signal are significantly different from that of the negative pressure signal, causing difficulty in aligning each local distribution across domains. Nevertheless, we can observe from the table that our OUDA method still achieves the best performance and obtains the highest accuracies on all tasks. Specifically, the OUDA method achieves the highest accuracies of 93.66%, 92.44%, and 93.80% on AWD → HP, AWD → MP, and AWD → LP, surpassing the state-of-the-art gradient adversarial method by 4.52%, 4.9%, and 7.69%. Moreover, compared to the method using manifold learning, the OUDA method achieves the highest accuracies on all three tasks in Case 2, while DRMEA only obtains the second-best accuracy of 88.93% on one out of three tasks. To sum up, the proposed OUDA method can achieve superior and stable performance in transferring knowledge from a labeled AWD to an unlabeled NPWD. This is because the proposed OUDA can: 1) generate reliable pseudo labels for the target data, which facilitates positive knowledge transfer across domains; 2) learn the DIFs while retaining the geometrical structure of target distribution, which simultaneously enhances the transferability and discriminability of features; and 3) adaptively adjust the relative proportions of invariance and variance, thus improving the stability and credibility of the trained cross-domain classification model.

Fig. 6 shows the p-values of paired t-test. From the experimental results, it can be observed that all values are close to zero, which means that the OUDA method significantly outperforms all compared algorithms.

F. Performance Analysis

To comprehensively evaluate the proposed OUDA algorithm, we analyze its performance in terms of loss, accuracy, domain discrepancy, and feature visualization. Moreover, this parts conducts an ablation study to verify the contribution of each component in our method.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

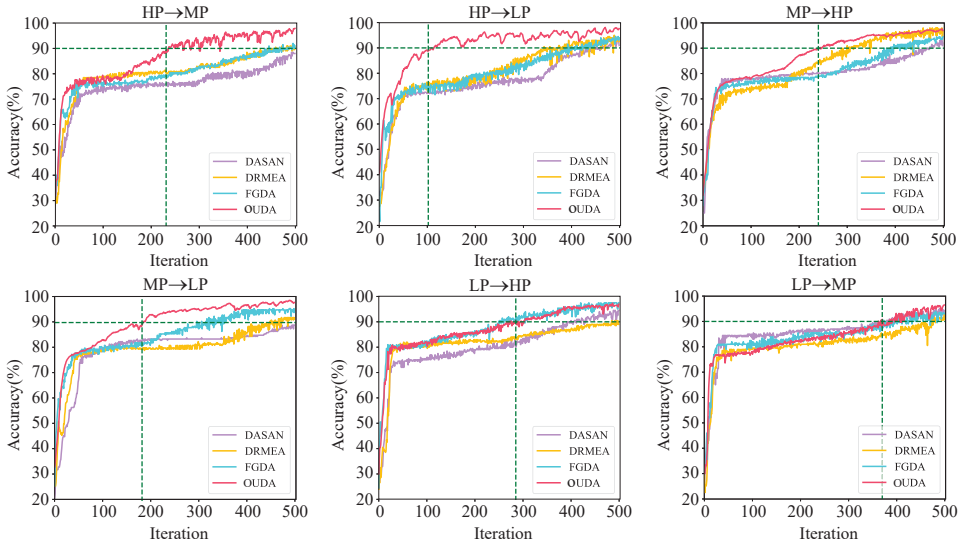

Fig. 7: Model performance in terms of accuracy convergence.

1) Accuracy: Fig. 7 presents the changing trend of classification accuracy with the increasing number of iterations on six tasks of Case 1. Different methods are represented by different colored curves. The horizontal dotted line denotes an accuracy of 90%. The vertical dotted line represents the minimum number of iterations required to achieve 90% classification accuracy. It is clear from Fig. 7 that the OUDA method always reaches 90% accuracy at the fastest rate. Meanwhile, our method performs better than DASAN, DRMEA, and FGDA, especially in tasks HP → MP, HP → LP, and MP → LP.

2) Loss: To verify the stability of the OUDA method, we depict the loss variation of each task in Fig. 8 and give the convergence analysis. In the initial stage of training, the loss values of all methods are around 1.4. Although there are some fluctuations during the training process, the loss of the OUDA method converges to the lowest point at the fastest velocity on each task.

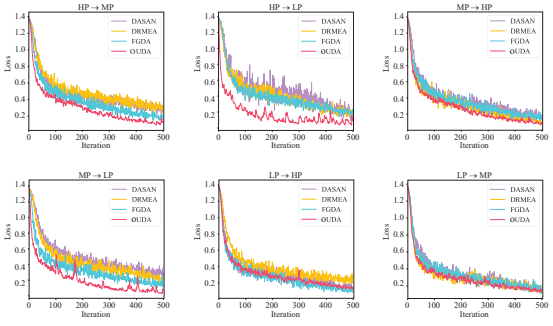

Fig. 8: Model performance in terms of loss convergence.

3) Domain Divergence: In this section, we employ MMD as a metric for measuring domain divergence. As illustrated in Table III, the proposed OUDA method is superior to the compared UDA methods in addressing the domain shift issue.

4) Feature Visualization: Our experiment utilizes t-distributed stochastic neighbor embedding (t-SNE) [22] to verify the effectiveness of the OUDA method in learning transferable and discriminative features. Each instance is represented by a number corresponding to the predicted label, and different domains are shown in different colors. From Fig. 9, it can be observed that instances belonging to the same category are clustered in a small region, which suggests that the OUDA method can learn the DIFs while preserving discriminable representations of the target category.

TABLE III: Results of model performance in narrowing domain divergence.

Tasks	DASAN	DRMEA	FGDA	OUDA
HP → MP	0.0203	0.0213	0.0392	**0.0166**
HP → LP	0.0340	0.0222	0.0326	**0.0149**
MP → HP	0.0669	0.0348	0.0575	**0.0301**
MP → LP	0.0303	0.0228	0.0326	**0.0129**
LP → HP	0.0676	0.0243	**0.0184**	0.0294
LP → MP	0.0444	0.0283	0.0754	**0.0202**

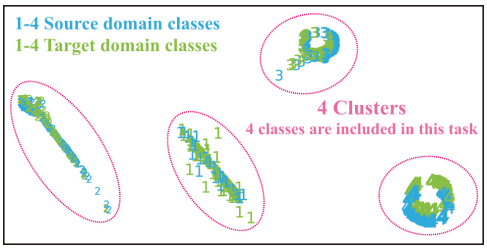

Fig. 9: Model performance in terms of feature visualization.

5) Ablation Study: To verify the contribution of each component in the OUDA algorithm, we carry out incremental experiments. As illustrated in Section III, the OUDA algorithm consists of three components: 1) LME, including reconstruction error of latent space shown in (3); 2) GAA, the domain alignment term in (5); and 3) PSO, the evolutionary control strategy illustrated in Section III-E. Therefore, we take the classification model without using any DA method as the base framework and add GAA, LME, and PSO in order. Compared

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/TII.2024.3385533, IEEE Transactions on Industrial Informatics

methods are listed as follows: Base, Base+LME, Base+GAA, Base+GAA+LME, and OUDA.

The ablation study is implemented on datasets NPWD and AWD, and the corresponding results are reported in Table IV. As can be seen in Table IV, the performance of Base is significantly degraded by the domain shift issue. Furthermore, the performance of Basic+LME and Basic+GAA is not satisfactory since the learned features lack transferability and discriminability. Although Base+GAA+LME achieves slightly better diagnostic accuracies than the above three algorithms, its performance still falls far short of expectations due to the imbalance between invariance and variance. Note that the OUDA method gains the highest accuracies in all four tasks. To summarize, the best cross-domain diagnostic performance can only be achieved when all three components in the proposed OUDA algorithm work together. In other words, each component plays an significant and indispensable role in the OUDA algorithm.

TABLE IV: Ablation study on four tasks.

GAA	*LME*	*PSO*	HP → MP	HP → LP	LP → HP	LP → MP	Avg.
–	–		73.08	73.63	76.14	72.01	73.71
–	✓		79.43	76.35	76.98	77.62	77.60
✓	–		84.95	84.72	84.57	85.64	84.97
✓	✓	–	90.84	89.57	88.32	92.82	90.39
✓	✓	✓	98.10	98.41	97.59	96.42	97.63

[1] "✓" indicates that the proposed component is adopted.
[2] "–" denotes that an alternative method is employed.

G. Contribution Coefficient Exploration

The optimal contribution coefficients of the proposed OUDA algorithm on tasks HP → MP, HP → LP, MP → HP, MP → LP, LP → HP, LP → MP, AWD → HP, AWD → MP, and AWD → LP are shown in Table V. It can be seen from Table V that the contribution coefficients in each combination are significantly different, which demonstrates the effectiveness of the proposed OUDA algorithm in exploring the optimal parameter. Furthermore, the experimental results in the two cases demonstrate the superiority of the evolutionary control strategy-based OUDA algorithm for cross-domain fault diagnosis.

TABLE V: The optimal contribution coefficients selected by the evolutionary control strategy.

Tasks	OUDA	
	$[\lambda_1,\lambda_2]$	Fitness values
HP → MP	[0.96,0.57]	0.14
HP → LP	[0.56,0.93]	0.10
MP → HP	[0.31,0.50]	0.09
MP → LP	[0.28,0.88]	0.09
LP → HP	[0.38,0.87]	0.10
LP → MP	[0.58,0.27]	0.10
AWD → HP	[0.77,0.50]	4.40E-07
AWD → MP	[0.47,0.17]	5.25E-08
AWD → LP	[0.99,0.45]	4.17E-08

V. Conclusion

In this paper, an OUDA algorithm has been proposed which aims to learn geometry-aware, transferable, and distinguishable domain-sharing features. To achieve this, the algorithm involves three fundamental steps: 1) geometrical structure learning, which maximizes the geometrical consistency between the original space and the common space to depict the intrinsic geometric structure of the target domain; 2) two-level distribution alignment, which minimizes the discrepancies between cross-domain marginal and conditional distributions to handle global and local domain shifts; and 3) contribution coefficient optimization, which controls the relative contribution between invariance and variance using the PSO algorithm. Extensive experiments have been conducted on cross-domain natural gas pipeline fault diagnosis, and the results of nine classification tasks have demonstrated that the proposed OUDA algorithm outperforms state-of-the-art UDA methods. Additionally, the performance analysis in terms of loss, accuracy, and domain discrepancy has also verified the stability and credibility of the proposed OUDA method in addressing the domain shift issue. In the future, we will consider improving the proposed OUDA algorithm in the following two aspects: 1) considering that embedding the PSO into the OUDA framework increases the computational complexity to a certain extent, a simpler and more efficient optimization strategy will be introduced in the future; and 2) giving physical interpretability with respect to the variance and invariance.

References

[1] X. Chen, S. Wang, M. Long, and J. Wang, Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation, In: *Proceedings of the 36th International Conference on Machine Learning*, PMLR, vol. 97, 2019, pp. 1081–1090.
[2] T. Cover and P. Hart, Nearest neighbor pattern classification, *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.
[3] S. Dong, M. Liu, and Z.-G. Wu, A survey on hidden Markov jump systems: asynchronous control and filtering, *International Journal of Systems Science*, vol. 54, no. 6, pp. 1360–1376, 2023.
[4] J. Fang, W. Liu, L. Chen, S. Lauria, A. Miron, and X. Liu, A survey of algorithms, applications and trends for particle swarm optimization, *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 24–50, 2023.
[5] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, and V. Lempitsky, Domain-adversarial training of neural networks, *Journal of Machine Learning Research*, vol. 17, pp. 1–35, 2016.
[6] Z. Gao, S. Zhang, K. Huang, Q. Wang, and C. Zhong, Gradient distribution alignment certificates better adversarial domain adaptation, In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 8937–8946.
[7] 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 Transactions on Industrial Electronics*, vol. 66, no. 9, pp. 7316–7325, 2019.
[8] Y. Hou, Y. Zhang, J. Lu, N. Hou, and D. Yang, Application of improved multi-strategy MPA-VMD in pipeline leakage detection, *Systems Science & Control Engineering*, vol. 11, no. 1, art. no. 2177771, 2023.
[9] J. Huang, N. Xiao, and L. Zhang, Balancing transferability and discriminability for unsupervised domain adaptation, *IEEE Transactions on Neural Networks and Learning Systems*, in press, DOI: 10.1109/TNNLS.2022.3201623.
[10] J. Jiao, J. Lin, M. Zhao, K. Liang, and C. Ding, Cycle-consistent adversarial adaptation network and its application to machine fault diagnosis, *Neural Networks*, vol. 145, pp. 331–341, 2022.
[11] J. Jiao, M. Zhao, and J. Lin, Unsupervised adversarial adaptation network for intelligent fault diagnosis, *IEEE Transactions on Industrial Electronics*, vol. 67, no. 11, pp. 9904–9913, 2020.
[12] H. Li, P. Wu, N. Zeng, Y. Liu, and F. E. Alsaadi, A survey on parameter identification, state estimation and data analytics for lateral flow immunoassay: from systems science perspective, *International Journal of Systems Science*, vol. 53, no. 16, pp. 3556–3576, 2022.
[13] X. Li, M. Li, P. Yan, G. Li, Y. Jiang, H. Luo. and S. Yin, Deep learning attention mechanism in medical image analysis: Basics and beyonds, *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 93–116, Mar. 2023.

[14] Y. Liu, Y. Wang, T. W. S. Chow, and B. Li, Deep adversarial subdomain adaptation network for intelligent fault diagnosis, *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6038–6046, 2022.

[15] M. Long, Z. Cao, J. Wang, and M. I. Jordan, Conditional adversarial domain adaptation, In: *Advances in Neural Information Processing Systems 31 (NeurIPS)*, 2018.

[16] M. Long, H. Zhu, J. Wang, and M. I. Jordan, Deep transfer learning with joint adaptation networks, In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017, pp. 2208–2217.

[17] T. Long, Y. Sun, J. Gao, Y. Hu, and B. Yin, Domain adaptation as optimal transport on Grassmann manifolds, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7196–7209, 2023.

[18] P. Lu, B. Song, and L. Xu, Human face recognition based on convolutional neural network and augmented dataset, *Systems Science & Control Engineering*, vol. 9, no. s2, pp. 29–37, 2021.

[19] Z. Lu and G. Guo, Control and communication scheduling co-design for networked control systems: a survey, *International Journal of Systems Science*, vol. 54, no. 1, pp. 189–203, 2023.

[20] X. Luo, Y. Yuan, S. Chen, N. Zeng, and Z. Wang, Position-transitional particle swarm optimization-incorporated latent factor analysis, *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pp. 3958–3970, 2022.

[21] Y. W. Luo, C. X. Ren, P. Ge, K. K. Huang, and Y. F. Yu, Unsupervised domain adaptation via discriminative manifold embedding and alignment, In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 34. no. 4, 2020, pp. 5029–5036.

[22] L. V. D. Maaten and G. Hinton, Visualizing data using t-SNE, *Journal of Machine Learning Research*, vol. 9, no. 11, pp. 2579–2605, 2008.

[23] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, Domain adaptation via transfer component analysis, *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2011.

[24] D. F. Qiu, H. Yang, X. F. Deng, and Y. H. Liu, Superpixel segmentation based on image density, *Systems Science & Control Engineering*, vol. 11, no. 1, art. no. 2185915, 2023.

[25] S. T. Roweis and L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[26] B. Sun and K. Saenko, Deep coral: Correlation alignment for deep domain adaptation, In: *Proceedings of European Conference on Computer Vision*, Berlin, Germany: Springer, 2016, pp. 443–450.

[27] J. Snell, K. Swersky, and R. Zemel, Prototypical networks for few-shot learning, In: *Advances in Neural Information Processing Systems (NIPS)*, Long Beach, USA, 2017.

[28] M. Szankin and A. Kwasniewska, Can AI see bias in X-ray images? *International Journal of Network Dynamics and Intelligence*, vol. 1, no. 1, pp. 48–64, 2022.

[29] H. Tang and K. Jia, Discriminative adversarial domain adaptation, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, 2020, pp. 5940–5947.

[30] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, Adversarial discriminative domain adaptation, In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7167–7176.

[31] C. Wang, Z. Wang, W. Liu, Y. Shen, and H. Dong, A novel deep offline-to-online transfer learning framework for pipeline leakage detection with small samples, *IEEE Transactions on Instrumentation and Measurement*, vol. 72, art. no. 3503913, 2022.

[32] C. Wang, Z. Wang, L. Ma, H. Dong, and W. Sheng, A novel contrastive adversarial network for minor-class data augmentation: Applications to pipeline fault diagnosis, *Knowledge-Based Systems*, vol. 271, art. no. 110516, 2023.

[33] J. Wang, Y. Zhuang, and Y. Liu, FSS-Net: A fast search structure for 3D point clouds in deep learning, *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 2, art. no. 100005, Jun. 2023.

[34] W. Wang, G. Sun, and L. V. Gool, Looking beyond single images for weakly supervised semantic segmentation learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 3, pp. 1635–1649, 2024.

[35] W. Wang, J. Zhang, W. Zhai, Y. Cao, and D. Tao, Robust object detection via adversarial novel style exploration, *IEEE Transactions on Image Processing*, vol. 31, pp. 1949–1962, 2022.

[36] Y. Wang, W. Li, D. Dai, and L. V. Gool, Deep domain adaptation by geodesic distance minimization, In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2651–2657.

[37] L. Wen, L. Gao, and X. Li, A new deep transfer learning based on sparse auto-encoder for fault diagnosis, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 136–144, 2019.

[38] M. Xu, J. Zhang, B. Ni, T. Li, C. Wang, Q. Tian, and W. Zhang, Adversarial domain adaptation with domain mixup, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, 2020, pp. 6502–6509.

[39] C. Yang, B. Xue, K. C. Tan, and M. Zhang, A co-training framework for heterogeneous heuristic domain adaptation, *IEEE Transactions on Neural Networks and Learning Systems*, in press, DOI: 10.1109/TNNLS.2022.3212924.

[40] D. Yang, J. Lu, H. Dong, and Z. Hu, Pipeline signal feature extraction method based on multi-feature entropy fusion and local linear embedding, *Systems Science & Control Engineering*, vol. 10, no. 1, pp. 407–416, 2022.

[41] T. Yin, N. Lu, G. Guo, Y. Lei, S. Wang, and X. Guan, Knowledge and data dual-driven transfer network for industrial robot fault diagnosis, *Mechanical Systems and Signal Processing*, vol. 182, no. 1, art. no. 109597, 2023.

[42] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. S. Platz, Central moment discrepancy (cmd) for domain-invariant representation learning, *arXiv preprint arXiv:1702.08811*, 2019.

[43] J. Zhang, C. Huang, M.-Y. Chow, X. Li, J. Tian, H. Luo, and S. Yin, A data-model interactive remaining useful life prediction approach of lithium-ion batteries based on PF-BiGRU-TSAM, *IEEE Transactions on Industrial Informatics*, vol. 20, no. 2, pp. 1144–1154, 2024.

[44] J. Zhang, K. Zhang, Y. An, H. Luo, and S. Yin, An integrated multitasking intelligent bearing fault diagnosis scheme based on representation learning under imbalanced sample condition, *IEEE Transactions on Neural Networks and Learning Systems*, in press, DOI: 10.1109/TNNLS.2022.3232147.

[45] Y. Zhang and B. D. Davison, Deep spherical manifold gaussian kernel for unsupervised domain adaptation, In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021, pp. 4443–4452.

[46] Y. Zhang, L. Zou, Y. Liu, D. Ding, and J. Hu, A brief survey on nonlinear control using adaptive dynamic programming under engineering-oriented complexities, *International Journal of Systems Science*, vol. 54, no. 8, pp. 1855–1872, 2023.

[47] Z. Zhang, M. Wang, Y. Huang, and A. Nehorai, Aligning infinite-dimensional covariance matrices in reproducing kernel Hilbert spaces for domain adaptation, In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018, pp. 3437–3445.

[48] C. Zhao and W. Shen, A domain generalization network combing invariance and specificity towards real-time intelligent fault diagnosis, *Mechanical Systems and Signal Processing*, vol. 173, art. no. 108990, 2022.