Received: 6 November 2021 | Revised: 26 January 2022 | Accepted: 7 March 2022

DOI: 10.1049/ipr2.12484

IET Image Processing

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

A weighted multi-source domain adaptation approach for surface defect detection

Bing Hu | Jianhui Wang

School of Information Science and Engineering, Northeastern University, Shenyang, China

Correspondence
Bing Hu, School of Information Science and Engineering, Northeastern University 3-11, Wenhua Road, Heping District, Shenyang, 110004,
P. R. China.
Email:hubing607@gmail.com

Abstract

This paper proposes a weighted multi-source domain adaptation (MSDA) method for industrial surface defect detection. The domain adaptation method is usually used to solve inaccurate results due to lacking target training samples. But in the scene of surface defects detection, this method is not adequate because the inspection samples often contain complex texture features. To get better performance, in this paper, we extend the single-source domain adaptation detection method to the multi-source domain. At the same time, we weighted different source domains samples during the adaptive training process, and prioritize the alignment of the target domain with the most similar source domain. The experimental results show that our proposed method performs well on the target dataset which improves existing methods' limitations for detecting surface defects with complex textures.

1 | INTRODUCTION

Automatic surface defect detection is an essential part of the industrial production process. Computer vision has gradually replaced manual inspectors with high precision, high efficiency, high-speed, non-contact, and other advantages as a critical technology in the manufacturing field. The global computer vision market was approximately US$7.2 billion in 2017 with a year-on-year growth of 6.8% [1].

In the traditional computer vision methods, histogram of gradient (HOG), local binary pattern (LBP), co-occurrence matrix, and scale-invariant feature transform (SIFT) are commonly used to extract features. Then those features can be processed by support vector machine (SVM), K-nearest neighbor (KNN), random forest or K-means (KM) to distinguish the defects [2–8]. However, these methods are limited to apply for specific objects.

Deep learning methods have also attracted more attention in computer vision. Compared with traditional methods, it does not rely on specially designed features and can be applied to different defect types. The supervised learning method based on convolutional neural networks (CNN) is the most popular deep learning method, which extracts multi-level image features to achieve accurate recognition. Typical scenarios include SSD [9], YOLO [10], Faster R-CNN [11], and Mask R-CNN [12]. These methods have a wide range of potential applications in terms of defect detection [13, 14]. However, these methods still have limitations in practical industrial applications. First, it is challenging to collect enough defect training data due to the low frequency of abnormal events. The number of standard samples occupies too many areas in the whole sample set. Unbalanced training data make the deep learning model challenging to converge in the training process and even cause over-fitting or under-fitting problems. Although some approaches proposed by researchers can address the issue to some extent [15, 16], it is still considerable when the training data in a specific class is inadequate. Second, the performance of the supervised learning model depends on the quality of the label. It takes a lot of time and expertise in label marking to obtain high-quality labels. If there is not enough trainable labelled data, the actual data distribution cannot be represented by samples, resulting in poor generalization of the trained model. Therefore, supervised learning methods are not a good and practical choice in the real production world. Semi-supervised and unsupervised learning methods for defect detection have better performance when the defective samples are scarce or unavailable. Some previous works in defect detection used autoencoders-base or GAN-base methods to achieve excellent performance. For instance, Ke et al. [17] used an autoencoder to train an adaptive template with normal samples and then calculated the difference between the input and the template to detect abnormalities on the mobile phone surface logos. Haselmann et al. [18] used a deep convolutional neural network for patch-wise completion of surface images to identify faults on fabric surfaces in textile manufacturing industries. Baur et al. [19] used pixel-wise reconstruction errors from deep spatial autoencoders to detect lesions in the brain. Ackay et al. [20] used GANomaly (an encoder-decoder-encoder model) to detect anomalies in X-ray scans to facilitate aviation and border security checking. Although these methods solve the lack of abnormal samples, their performance decreases when the sample's specification changes or with complex textures. The generator cannot accurately generate the template if the training and test samples are inconsistent. That is referred to as covariate shift [21] or dataset bias [22]. A practical method to solve the challenge is to use the domain adaptation method, which adapts the classifier to the new data by utilizing the domain characteristics of the labelled data in a related domain.

In our study, we proposed a defects detection method based on a domain adaptation network. The models trained with samples from other multi-source domains can detect the target samples better, especially for surface defect samples with a complex texture. The method used a weighted adversarial network to preferentially align the source domains more similar to the target samples. Compared with the traditional multi-source domain adaptation (MSDA) method that needs to align the various source domains, our proposed method assigns low weight to the source domain that is more different from the target domain samples in transfer learning. The purpose is to avoid the negative impact of salient differences in texture features on feature extraction. We evaluated our method on two public datasets, and the results showed that the process is superior to the previous defects detection methods

The other parts of this paper are structured as follows. Section 2 presents the related methods of surface defect detection and domain adaptation methods. Section 3 presents the details of our proposed method. Section 4 focus on the experimental results and discussions. Section 5 summarized our work and the prospects for future work.

2 | RELATED METHOD

2.1 | Surface defect detection

Previous research on surface defect detection, edge detection [23, 24], clustering [25], and other image processing methods are commonly used. However, applying these methods requires image pre-processing by adjusting parameters based on the experience and knowledge of the inspector to obtain smoother images and more distinct defect features. These methods are time-consuming, labour-intensive, and subjective.

Deep learning methods, such as support vector machines, CNN, and random forest, have also been used to detect defects (anomalies). As is mentioned above, one of the main problems of supervised learning methods based on deep learning is that training data collection is usually expensive in terms of time and resources. And for some unsupervised learning methods, such as deep autoencoder (DAE) and generative adversarial network (GAN) [26], also have limitations when dealing with surface defect detection in complex textures. The implementation of GANs may not be reliable because the reconstruction results are unpredictable. Although DAE can obtain better reconstruction quality, we need to care if the defect and the texture have a similar potential feature. The too powerful reconstruction ability of the decoder will reconstruct the defective area, which makes the defective indistinguishable in the template contrast operation.

In this paper, we focus on defects detection for surfaces with complex textures. Unlike ordinary product surfaces, these products have particular patterns on the surface, which often interfere with identifying defects. Therefore, it is difficult to detect faults using traditional threshold segmentation methods. Template matching is a standard method for defect detection on complex textures [22]. However, the background features are not constant due to the irregular surface, uneven illumination, and other factors, making matching difficult. Therefore, none of these methods can meet our requirements. Many researchers have focused on the task of defect detection in complex textures. In [27], a transfer learning approach used the pre-trained networks and obtained promising results. In [28], a Bayes classifier that can be adapted to changing conditions is proposed. This classifier can achieve good performance by training with a small sample set. In [29], a particularly robust method named adjacent evaluation completed local binary patterns is proposed, improving the recognition rate of the hot-rolled steel strip surface defects.

2.2 | Domain adaptation

Transfer learning aims to achieve a classifier trained from a label-rich domain (i.e. source domain) to have good results in a label-scarce domain (i.e., target domain). Domain adaptation (DA) is a typical example of transfer learning methods. It tackles the problem that the source and target domains have the same feature space and category space, and only the feature distribution is inconsistent. Some previous methods achieve this purpose by minimizing explicit domain discrepancy metrics. Maximum mean-variance (MMD) is the most commonly used to reduce distribution offset [30, 31]. Such methods also include correlation alignment [32], Kullback-Leibler (KL) divergence [33], and $\mathcal{H}-$divergence [34]. Another widely used method in DA is generative adversarial networks. It uses a domain discriminator to confuse the target domain with the source domain to learn invariant features between different domains [35, 36, 37, 38].

Multi-source domain adaptation (MSDA) assumes that data is collected from multiple source domains with different distributions. Compared with single-source domain adaptation, this is a more realistic scenario. Ben-David et al. [34] express the target distribution as a weighted combination of multiple source distributions. The deep cocktail network (DCTN) [39] proposed a k-way domain discriminator and class classifier for

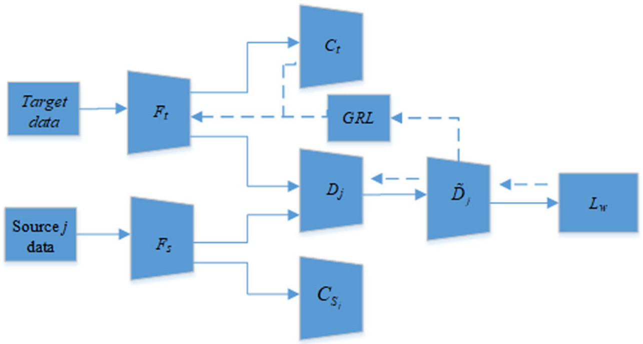

FIGURE 1 The overview of the proposed method. (F: feature extractor, C: classifier, D: domain discriminator, GRL: gradient reversal layer, Lw: Equation (7) weighted domain loss.)

digital classification and real-world object recognition. Peng et al. [40] proposed an approach with moment matching for MSDA which aims to transfer knowledge from multiple labelled source domains to an unlabelled target domain.

3 | PROPOSED METHOD

In MSDA, there are m source domains $S_1, S_2, \ldots, S_m$ and a target domain T. The domain $S_j = \{(x_i^{S_j}, y_i^{S_j})\}_{i=1}^{N_{S_j}}$ is characterized by N_{S_j} i.i.d. labelled samples, where $y_i^{S_j} \in \{1, 2, \ldots, K\}$ (K is the number of classes) and $x_i^{S_j}$ follows one of the source distributions X^{S_j}. Similarly, the target domain $T = \{x_i^T\}_{i=1}^{N_T}$ is represented by N_T i.i.d. unlabelled samples, where x_i^T follows target distributions X^T. The MSDA problem aims to train the model using samples of multiple source domains and target domains, minimizing the testing error of the target T.

In MSDA, samples from multiple source domains can provide richer feature information of the objects for the target domain. Based on more supporting data, the decision boundary of the features can be further refined. However, the different distribution of different source domains increases the difficulty of learning domain invariant features. In the task of defect detection, the size and proportion of texture features in samples are much larger than that of defect features, so they are inevitably represented as salient features during feature extraction. Therefore, aligning all domains without considering the correlation and consistency of different source domains and target domains is unreliable. To address this, our idea is to prioritize the alignment of the target domain with those source domains that are more difficult to separate samples from the target. Inspired by the work in [39] and [41], we use a weighted adaptation network to solve the issue.

The overview of the proposed method is shown in Figure 1, and it is based on the domain adversarial training framework. There are three subnets in the network, feature extractor, (multi-source) domain discriminator, and (multi-source) classifier. There are two unshared weights feature-extraction functions F_s and F_t in our network, which are employed by source domains and the target domain, respectively. We build m discriminators $D = \{D_{S_j}\}_{j=1}^m$ and m classifiers $C_S = \{C_{S_j}\}_{j=1}^m$. For each source domain S_j, the specific domain discriminator $D_{S_j} : F \rightarrow \{0, 1\}$ distinguishes the input feature that comes from the source domain S_j and the target domain T. Similarly, m classifiers accept features $F_s(x)$ or $F_t(x)$ and output the probability that the sample belongs to each class with the softmax function. The discriminator and classifier of the source domain S_j are independent of other sources.

We first pre-train the network to get each source domain classifier C_{S_j} and source feature extractor $F_s(x)$. In the pre-training phase, the classifier $C_{S_j}(F_s(x))$ loss can be described as follow:

$$\min_{F_s,C} \boldsymbol{\mathcal{L}}_{cls}\% \left(C, F_s\right) = -\sum_j^m \mathbb{E}_{(x,y)\sim\left(X^{S_j}, Y^{S_j}\right)} y \log C_{S_j}\left(F_s\left(x\right)\right) + \left(1-y\right)\log\left(1 - C_{S_j}\left(F_s\left(x\right)\right)\right) \quad (1)$$

Since each source domain is trained on a supervised model, it can obtain the best representations of the classifier and feature extractor. Then we use adversarial training to reduce the distance between target domains and source domain distributions. Intuitively, a transfer network of a source domain can provide better performance if the source domain distribution is closer to the target. Conversely, the distribution of the source domain farther away from the target will reduce the transfer performance. We use the domain discriminators to indicate the distribution distance between the target domain and each source domain.

We fixed F_s and C_S to $\bar{F}_s$ and $\bar{C}_S$ when the network converges, and then optimize the domain discriminator D and target feature extractor F_t, the objective is as follows:

$$\min_{F_t} \max_{D} \boldsymbol{\mathcal{L}}_{adv}\left(D, F_t\right) = \frac{1}{M}\sum_j^m \mathbb{E}_{x\sim X^{S_j}}\left[\log D_{S_j}\left(\bar{F}_s\left(x\right)\right)\right] + \mathbb{E}_{x\sim X^T}\left[\log\left(1 - D_{S_j}\left(F_t\left(x\right)\right)\right)\right] \quad (2)$$

It is worth noting that if the target feature extraction F_t and the source feature extraction F_s share weights ($F_t = F_s$) or both change during the adversary training, which will lead to oscillation. To solve this problem, reference [36] used domain confusion to replace the adversarial objective. In this paper, F_t and F_s have different parameters and F_s has been fixed. Therefore, we can use Equation (2) to only update F_t and D during the adversary training, similar to the original GAN. In the meantime, to avoid the disappearance of the gradient in the initial training, we use $\bar{F}_s$ to initialize F_t.

Simultaneously, when the domain discriminator D has converged to the optimal value of the current feature extractor, we use it to indicate the probability of samples from the source or target domain distribution. It is difficult to determine which domain the samples belong to if the score is close to 0.5. And these samples are more likely to come from the source domain closer to the target domain. To this end, we defined the confusion score as follow:

$$S_{S_j}\left(x^T, x^{s_j}\right) = 1 - \frac{1}{2}\left(\begin{array}{c} \frac{1}{N_T}\sum_{i}^{N_T}\left|D_{S_j}\left(F_t\left(x_i^T\right)\right) - 0.5\right| \\ +\frac{1}{N_{s_j}}\sum_{i}^{N_{s_j}}\left|D_{s_j}\left(\bar{F}_s\left(x_i^{s_j}\right)\right) - 0.5\right| \end{array}\right) \tag{3}$$

When $D_{S_j}(D_{S_j}(F_t(x)))$ is closer to 0.5, the S_{S_j} is larger, the samples of the source domain S_j may have more overlap with the target domain samples. Conversely, this function will be minor, and the source and target domain samples have little or no overlap. During domain transfer learning, the source domains which have more overlap should be assigned greater weights. The weight of S_j can be normalized as:

$$w_{s_j} = \frac{S_{s_j}(x)}{\sum_{j=1}^{m} S_{s_j}(x)} \tag{4}$$

The ideal objective function of the adversarial network with weights can be described as follow:

$$\min_{F_t}\max_{D}\mathcal{L}_w(D, F_t) = \frac{1}{M}\sum_{j}^{m}\mathbb{E}_{x\sim X^{s_j}}\left[w_{S_j}\log D_{S_j}\left(\bar{F}_S(x)\right)\right] + \mathbb{E}_{x\sim X^{\mathrm{T}}}\left[\log\left(1 - D_{S_j}(F_t(x))\right)\right] \tag{5}$$

But the weight w is defined as the correlation function of the domain discriminator D. Therefore, applying the weight w_{S_j} to the domain discriminator D_{S_j}, the result of the minimax game will not change. To solve this issue, we follow [41] to use the second domain discriminator $\tilde{D}_{S_j}$ after D_{S_j} for each source domain to update F_t in adversarial training. And the first domain discriminator D_{S_j} is only used to calculate the confusion score. Thus, we used Equation (6) to optimize D, and the objective of the adversarial network can be reformulated as Equation (7).

$$\min_{D}\mathcal{L}_D(D) = -\frac{1}{M}\sum_{j}^{m}\left(\begin{array}{c}\mathbb{E}_{x\sim X^{s_j}}\left[\log D_{S_j}\left(\bar{F}_S(x)\right)\right] \\ +\mathbb{E}_{x\sim X^{\mathrm{T}}}\left[\log\left(1 - D_{S_j}(F_t(x))\right)\right]\end{array}\right) \tag{6}$$

$$\min_{F_t}\max_{\tilde{D}}\mathcal{L}_w\left(\tilde{D}, F_t\right) = \frac{1}{M}\sum_{j}^{m}\mathbb{E}_{x\sim X^{s}}\left[w_{S_j}\log\tilde{D}_{S_j}\left(\bar{F}_S(x)\right)\right] + \mathbb{E}_{x\sim X^{\mathrm{T}}}\left[\log\left(1 - \tilde{D}_{S_j}(F_t(x))\right)\right] \tag{7}$$

In this paper, since the feature extractor F_t is independent of F_s, it is essential to constraint F_t to preserve the data structure of target samples. Using the Pseudo-labels of the target samples to constrain the feature extractor is a common method in domain adaptation [39,42]. We use the confusion score to find the source domain closest to the target domain is currently $j^* = \operatorname{argmax}\{S_{S_j}\}_{j=1}^{m}$. Then use the classifier of j^* to predict the target samples pseudo-labels when their confidence is higher than the present threshold γ. It can be expressed as:

$$\tilde{y}^T = \bar{C}_{s_{j^*}}\left(F_t\left(x^T\right)\right) \tag{8}$$

In MSDA, the target domain classifier can be presented as the weighted combination of source classifiers. When the sample feature of the source is closer to the target, the source classifier may predict more accurately on the target samples. Therefore, we use w_{S_j} as the weight of each source classifier, then the target classifier can be formulated as:

$$C_t\left(x^T\right) = \sum_{j}^{m} w_{S_j}\bar{C}_{S_j}\left(F_t\left(x^T\right)\right) \tag{9}$$

Then the classification loss of the target samples with pseudo-labels can be expressed as:

$$\mathcal{L}_{\mathrm{tar}}(F_t) = \frac{1}{N_t}\sum_{i=1}^{N_t}\mathcal{L}_C\left(C_t\left(F_t\left(x_i^T\right)\right), \tilde{y}_i^T\right) \tag{10}$$

Where $\mathcal{L}_C(\cdot)$ is the softmax loss function (or logistic loss function in image-level defect detection task).

Hence, the overall objectives of the adversarial network can be written as:

$$\min_{F_t}\max_{\tilde{D}}\mathcal{L}_{\mathrm{tar}}(F_t) + \lambda\mathcal{L}_w\left(\tilde{D}, F_t\right) \tag{11}$$

Where λ is the hyperparameter, and we follow [43] to set it up.

After pre-trained with all source domains data, we use Equations (6) and (11) to update D, $\tilde{D}$ and F_t in the minimax game until the network converges. The fixed CSj and Ft are not be updated in this learning process.

4 | EXPERIMENTAL SECTION

This section first compares our proposed method with existing domain adaptation methods on the Digits-five dataset. Next, we use datasets DAGM 2007 [44] to evaluate the performance of our proposed method in surface defect detection.

All experiments are implemented with the Pytorch platform on a PC with Intel i7- 8700K 3.70 GHz CPU, 32GB RAM, and one Nvidia 1080 GPU. In all experiments, we use ResNet-50 [45] without the last fully connected layer as our proposed network feature extractors. And the two domain classifiers are with the same architecture, which is three fully connected layers. We use Adam [46] to accelerate network convergence

TABLE 1 Classification accuracy on digits recognition

Models	mm,sv,sy,up→ mt	mt,sv,sy,up→ mm	mt,mm,sy,up→ sv	mt,mm,sv,up→ sy	mt,mm,,sv,sy → up	Avg
Source Only	**93.2 ± 0.42**	**70.2 ± 0.50**	**71.5 ± 0.55**	**82.8 ± 0.49**	**91.8 ± 0.54**	**81.9 ± 0.50**
DANN	96.7 ± 0.72	71.8 ± 0.81	68.3 ± 0.65	87.1 ± 0.58	91.3 ± 0.77	83.0 ± 0.71
DAN	94.2 ± 0.71	68.1 ± 0.67	68.8 ± 0.76	88.3 ± 0.62	93.5 ± 0.71	82.6 ± 0.69
M^{3}SDA	98.8 ± 0.39	76.5 ± 0.73	87.3 ± 0.58	91.5 ± 0.60	96.2 ± 0.72	90.1 ± 0.60
DCTN	97.2 ± 0.72	70.1 ± 1.24	79.8 ± 0.71	84.5 ± 0.78	93.8 ± 0.44	85.1 ± 0.78
Our method	99.2 ± 0.10	85.3 ± 0.42	88.6 ± 0.47	95.0 ± 0.27	99.1 ± 0.23	93.4 ± 0.30

Note: MT, MM, SV, SY, UP are abbreviations for MNIST, MNIST-M, SVHN, SYNTHETIC DIGITS, USPS, respectively.

and optimize it with learning $\ell\gamma = 2e^{-4}$ and momentums $\beta 1 = 0.5, \beta 2 = 0.999$.

It is worth noting that although in the experiments we used gray image as a standard input for training and testing to speed up the computation, our method also works with colour samples. Benefiting from convolution-based ResNet, when using RGB image as input, it is only needs to adjust the depth of convolution kernels corresponding to it.

4.1 | Experiments on digit recognition

The Digits-five dataset is widely used in the performance evaluation of MDA. The dataset consists of samples from five different sources, namely MNIST [47], MNIST-M [48], SVHN [49], USPS and Synthetic Digits [48]. Following [39], for MNIST, MINST-M, SVHN, and Synthetic Digits, we sample 25,000 images for training and 9000 for testing in each dataset. And choose the entire 9298 images in USP as a domain.

We compared our method with four state-of-the-art domain adaptation methods: Deep adaptation network (DAN) [50], Domain adversarial neural network (DANN) [35], Deep cocktail network (DCTN) [39] and moment matching for multiSource (M3SDA) [40]. For Source Only and single-source method experiments, we follow the source combine setting in [40]. All source domains data are combined into a single source. For a fair comparison, all the deep learning models are used ResNet-50 as the backbone. We run each experiment five times to take the average and deviation.

The results are shown in Table 1. Our proposed method achieves a 91.6% average accuracy, outperforming other baselines by a large margin.

4.2 | Experiments on DAGM

The DAGM 2007 dataset covers many types of manufacturing material surfaces in the industry. The samples are shown in Figure 2. It comprises 8050 training images and 8050 testing images with a size of 512 × 512 and 8-bit grayscale PNG format. There are ten classes of artificially generated surfaces with specific textures in DAGM. The dataset provides 2112 groundtruth images to identify the defect region. In each experiment, we set one of the classes as the target domain and the rest as source domains.

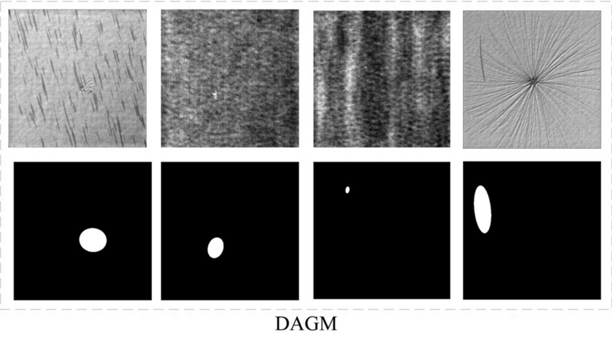

FIGURE 2 Examples of sample images from DAGM

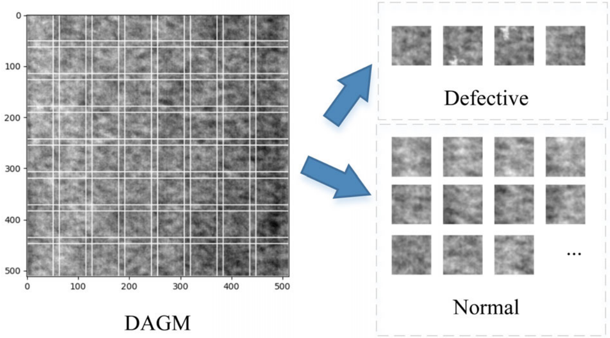

FIGURE 3 The example of segmentation sample images from DAGM

TABLE 2 Comparison with related work on the DAGM dataset (for MAP, ROC AUC, F1-measure)

Methods	Acc	mAP	AUC	F1
Faster R-CNN (source combine)	0.89	0.66	0.87	0.68
Faster R-CNN (source only)	0.80	0.43	0.64	0.56
AnoGAN	0.71	0.51	0.57	0.56
AE (SSIM)	0.76	0.54	0.70	0.60
Our method	0.91	0.78	0.92	0.82

As shown in Figure 3, we cropped each image used for training and testing into 64 patches with a size of 77 × 77 (the

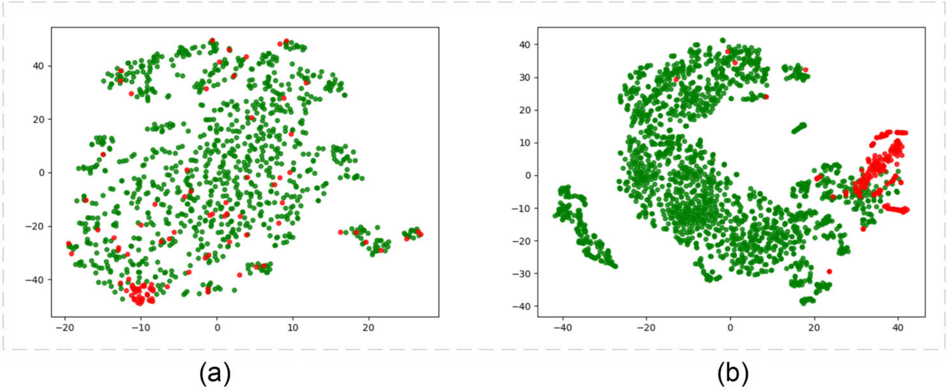

FIGURE 4 T-SNE visualization of the features mapped from the well-trained network, (a) source only (b) our method (normal: green; defective: red)

first row and the first column patches size is 64 × 64) before inputting our network. Each neighbour patch has a 20% overlap to avoid only the defects edge in the patch. According to the smallest sum of abnormal pixels contained in the corresponding ground truth picture, we divide the patches into abnormal (positive) and normal (negative). The threshold is calculated based on half of the most petite side length of the defect in the dataset. At last, pixel values of all patches are normalized into a range of [−1,1] to avoid excessive deviations in the calculation. We need such pre-processing due to reasons: (1) Industrial cameras usually have a considerable high resolution in industrial surface defect detection. Cropping can reduce computation and solve insufficient training data problems. (2) Image-level annotation is more efficient than pixel-level annotation in the network training stage. And in the inference stage, we can locate the defect position based on image prediction. (3) The scaling operation does not affect network performance due to the CNN as the feature extractor.

It is not enough to perform a single accuracy score on the unbalanced data set in defect detection. Therefore, we have adopted several comprehensive indicators, such as ACC, Precision, and F-Score. These indicators are shown in Equations (12), (13), and (15).

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

$$F_1 = \frac{2\times \text{ Precision } \times \text{ Recall}}{\text{Precision } + \text{ Recall}} \tag{15}$$

The prediction results are divided into true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP represents the correct prediction of normal pictures, TN represents the correct prediction of defective pictures, FP represents the wrong prediction of normal pictures, and FN represents the wrong prediction of defective pictures. AUC represents the two-dimensional measurement area under the receiver operating characteristic (ROC) curve, which is used for the performance evaluation of a classification model.

We compare our proposed method with three state-of-the-art defect detection algorithms. These methods include one supervised learning method and two unsupervised learning methods. Faster-RCNN as a supervised method has an excellent performance in the defect detection task [18, 51]. This method can perform well by fine-tuning the pre-trained cross-domain model when the target domain samples are unbalanced. Fine-tuning can be seen as a simple transfer learning method.

The experiment results on DAGM are shown in Table 2. We obtained the results by binary classification of cropped images in the test dataset. The results show that the model performance which uses source combined setting for pre-training is better than using single-source domains. The feature of defects is more generalized when referencing multi-source domain data. However, because different texture features in different domain samples interfere with the extraction of defect features, equal treatment of this interference will not obtain accurate decision boundaries. Therefore, the method has insufficient defect detection capabilities under complex textures.

In the two unsupervised learning methods, GANomaly [21] and SSIM Autoencoder [52], Only the target domain normal samples are used for training. As shown in Table 2, the two methods have a common shortcoming: the performance varies in different domains. It is due to these two methods depending on the quality of the reconstruction image. When the target sample is a relatively stable structural texture feature, the methods perform well. Once this structural feature is destroyed, the model cannot reconstruct the texture feature from the training data. As described in the previous section, this is not applicable in detecting surface defects of industrial products.

We visualize data distribution to two-dimensional features as Figure 4. Red points indicate defect data, and green points indicate normal data. We can see that the feature boundary between normal and defective samples is more apparent after using our method for domain adaptation.

5 | CONCLUSION

In this paper, we propose a multi-domain adaptation method for detecting surface defects of industrial products. The method uses a reweight adversarial domain adaption. More weight is assigned to the target-related source domain in the adaptation process and achieves a better adaption performance. The method can well solve the issue of sparse or unbalanced target data in surface defect detection. And it can deal with the interference of detection defects with complex textures. It is concluded from the experiments that our proposed method has more advantages than previous domain adaptive methods on Digits-five datasets and has satisfactory results in defect detection, especially for complex textures. We will continue developing our approach and applying it to various material surface defect detection tasks in future work.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in 'MNIST' at https://doi.org/10.1109/5.726791, reference number [42], 'MNIST-M','USPS','Synthetic Digits' at https://doi.org/10.1109/10.1007/978-3-319-58347-1_10, reference number [43], 'SVHN' reference number [44],'DAGM 2007' at https://doi.org/10.1109/1057-7149(2008)17:9 < 1700:WSLOAC > 2.0.TX;2-H, reference number [40].

FUNDING INFORMATION

None.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

1. Sun X., Gu J., Tang S., Li J.: Research progress of visual inspection technology of steel products—a review. Appl. Sci. 8(11), 2195 (2018)
2. Aghdam S.R., Amid E., Imani M.F.: A fast method of steel surface defect detection using decision trees applied to LBP based features. In: 2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA), Singapore, 18–20 July 2012
3. Kim C.-W., Koivo A.J.: Hierarchical classification of surface defects on dusty wood boards. Pattern Recogn. Lett. 15(7), 713–721 (1994) /07/01/1994
4. Tsanakas J.A., Chrysostomou D., Botsaris P.N., Gasteratos A.: Fault diagnosis of photovoltaic modules through image processing and Canny edge detection on field thermographic measurements. Int. J. Sustainable Energy 34(6), 351–372 (2015) /07/03 2015
5. Mak K.L., Peng P., Yiu K.F.C.: Fabric defect detection using morphological filters. Image Vision Comput. 27(10), 1585–1592 (2009)
6. Heydarzadeh M., Nourani M.: A two-stage fault detection and isolation platform for industrial systems using residual evaluation. Ieee Trans. Instrum. Meas. 65(10), 2424–2432 (2016)
7. Bai X., Fang Y., Lin W., Wang L., Ju B.: Saliency-based defect detection in industrial images by using phase spectrum. IEEE Trans. Ind. Inf. 10(4), 2135–2145 (2014)
8. Wang H., Zhang J., Tian Y., Chen H., Sun H., Liu K.: A simple guidance template-based defect detection method for strip steel surfaces. IEEE Trans. Ind. Inf. 15(5), 2798–2809 (2019)
9. Liu W., et al.: SSD: Single Shot MultiBox Detector. p. arXiv:1512.02325Accessed on: December 01, 2015 [Online]. Available: https://ui.adsabs.harvard.edu/abs/2015arXiv151202325L
10. Redmon J., Divvala S., Girshick R., Farhadi A.: You only look once: unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016
11. Ren S., He K., Girshick R., Sun J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett. (eds.) Advances in Neural Information Processing Systems vol. 28, pp. 91–99. Curran Associates (2015)
12. He K., Gkioxari G., Dollár P., Girshick R.: Mask R-CNN. p. arXiv:1703.06870 Accessed on: March 01, 2017 [Online]. Available: https://ui.adsabs.harvard.edu/abs/2017arXiv170306870H
13. Lin H., Li B., Wang X., Shu Y., Niu S.: Automated defect inspection of LED chip using deep convolutional neural network. J. Intell. Manuf. 30(6), 2525–2534 (2019)
14. Hu B., Wang J.: Detection of PCB surface defects with improved faster-rcnn and feature pyramid network. IEEE Access 8, 108335–108345 (2020)
15. Lin T.-Y., Goyal P., Girshick R., He K., Dollár P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017
16. Kervadec H., Bouchtiba J., Desrosiers C., Granger E., Dolz J., Ayed I.B.: Boundary loss for highly unbalanced segmentation. In: International Conference on Medical Imaging with Deep Learning, Zurich, Switzerland, 6–8 July 2019
17. Ke M., Lin C., Huang Q.: Anomaly detection of Logo images in the mobile phone using convolutional autoencoder. In: 2017 4th International Conference on Systems and Informatics (ICSAI), Hangzhou, China, 11–13 November 2017
18. Haselmann M., Gruber D.P., Tabatabai P.: Anomaly detection using deep learning based image completion. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018
19. Christoph B., Benedikt W., Shadi A., Nassir N.: Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. arXiv preprint arXiv:1804.04488, 2018. URL. Available: http://arxiv.org/abs/1804.04488
20. Akcay S., Atapour-Abarghouei A., Breckon T.P.: GANomaly: Semi-supervised anomaly detection via adversarial training. p. arXiv:1805.06725 Accessed on: May 01, 2018 [Online]. Available: https://ui.adsabs.harvard.edu/abs/2018arXiv180506725A
21. Shimodaira H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. J. Stat. Plann. Infer. 90(2, 227–244 (2000)
22. Zadrozny B.: Learning and evaluating classifiers under sample selection bias. Paper presented at the Proceedings of the twenty-first international conference on machine learning, Banff, Alberta, Canada, 4 July 2004 https://doi.org/10.1145/1015330.1015425
23. Hocenski Z., Vasilic S., Hocenski V.: Improved canny edge detector in ceramic tiles defect detection. In: IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics, Paris, France, 6–10 November 2006
24. Abdel-Qader I., Abudayyeh O., Michael E.K.: Analysis of edge-detection techniques for crack identification in bridges. J. Comput. Civil Eng. 17(4), 255–263 (2003)
25. Wu S., Wu Y., Cao D., Zheng C.: A fast button surface defect detection method based on Siamese network with imbalanced samples. Multimedia Tools Appl. 78(24), 34627–34648 (2019)
26. Goodfellow I., Pouget-Abadie J., Mirza M., et al. Generative adversarial nets. In NIPS, Montreal, Canada, 8–13 December 2014
27. He Y., Song K., Meng Q., Yan Y.: An end-to-end steel surface defect detection approach via fusing multiple hierarchical features. IEEE Trans. Instrum. Meas. 69(4), 1493–1504 (2020)
28. Xiao M., Jiang M., Li G., Xie L., Yi L.: An evolutionary classifier for steel surface defects with small sample set. EURASIP J. Image Video Process. 2017(1), 48 (2017)
29. Song K., Yan Y.: A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. Appl. Surf. Sci. 285, 858–864 (2013)
30. Long M., Zhu H., Wang J., Jordan M.I.: Deep transfer learning with joint adaptation networks. Paper presented at the Proceedings of the 34th

International Conference on Machine Learning, Proceedings of Machine Learning Research, Sydney, Australia, 6–11 August 2017) http://proceedings.mlr.press/v70/long17a.html

31. Ghifary M., Bastiaan Kleijn W., Zhang M.: Domain adaptive neural networks for object recognition. p. arXiv:1409.6041 (2014) Accessed on: September 1 [Online]. Available: https://ui.adsabs.harvard.edu/abs/2014arXiv1409.6041G
32. Sun B., Feng J., Saenko K.: Return of frustratingly easy domain adaptation. In: Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, Arizina 12–17 February 2016
33. Zhuang F., Cheng X., Luo P., Pan S.J., He Q.: Supervised representation learning: Transfer learning with deep autoencoders. In: Proceedings of the 24th International Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015
34. Ben-David S., Blitzer J., Crammer K., Kulesza A., Pereira F., Vaughan J.W.: A theory of learning from different domains. Mach. Learn. 79(1), 151–175 (2010) 2010/05/01
35. Ganin Y., Lempitsky V.: Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning, Proceedings of Machine Learning Research, Lille, France, 6–11 July 2015. [Online]. Available: http://proceedings.mlr.press/v37/ganin15.html
36. Tzeng E., Hoffman J., Saenko K., Darrell T.: Adversarial discriminative domain adaptation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017
37. Bousmalis K., Silberman N., Dohan D., Erhan D., Krishnan D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017
38. Sankaranarayanan S., Balaji Y., Castillo C.D., Chellappa R.: Generate to adapt: Aligning domains using generative adversarial networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June (2018)
39. Xu R., Chen Z., Zuo W., Yan J., Lin L.: Deep cocktail network: multi-source unsupervised domain adaptation with category shift. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018
40. Peng X., Bai Q., Xia X., Huang Z., Saenko K., Wang B.: Moment matching for multi-source domain adaptation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 27 October–2 November 2019
41. Zhang J., Ding Z., Li W., Ogunbona P.: Importance weighted adversarial nets for partial domain adaptation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018
42. Zhang W., Xu D., Ouyang W., Li W.: Self-paced collaborative and adversarial network for unsupervised domain adaptation. IEEE Trans. Pattern Anal. Mach. Intell. 43(6), 2047–2061 (2021)
43. Yang L., Balaji Y., Lim S.-N., Shrivastava A.: Curriculum manager for source selection in multi-source domain adaptation. in computer vision – ECCV 2020. Cham, Springer International Publishing, 608–624, (2020)
44. Jager M., Knoll C., Hamprecht F.A.: Weakly supervised learning of a classifier for unusual event detection. IEEE Trans. Image Process. 17(9), 1700–1708 (2008)
45. He K., Zhang X., Ren S., Sun J.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016
46. Kingma D.P., Ba J.: Adam: A method for stochastic optimization. arXiv.org (2017)
47. Lecun Y., Bottou L., Bengio Y., Haffner P.: Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 2278–2324 (1998)
48. Ganin Y., et al.: Domain-adversarial training of neural networks. in domain adaptation in computer vision applications. In: C. Gabriela, (ed.) Advances in Computer Vision and Pattern Recognition: Cham, Springer (2017)
49. Netzer Y., Wang T., Coates A., Bissacco A., Wu B., Ng A.Y.: Reading digits in natural images with unsupervised feature learning. in NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, 16–17 December 2011
50. Mingsheng L., Yue C., Jianmin W., Michael J.: Learning transferable features with deep adaptation networks. In: Proceedings of the 32 nd International Conference on Machine Learning, Lille, France, 6–11 July 2015
51. Sun X., Gu J., Huang R., Zou R., Giron Palomares B.: Surface defects recognition of wheel hub based on improved faster R-CNN. Electronics 8(5), 481, (2019)
52. Bergmann P., Löwe S., Fauser M., Sattlegger D., Steger C.: Improving unsupervised defect segmentation by applying structural similarity to autoencoders. In: Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Prague, Czech Republic, 25–27 February 2019

How to cite this article: Hu, B., Wang, J.: A weighted multi-source domain adaptation approach for surface defect detection. IET Image Process. 16, 2210–2218 (2022). https://doi.org/10.1049/ipr2.12484

APPENDIX A

A.1 | Ablation study

Compared with the original MSDA method using adversarial training, there are two improvements to our proposed approach. One is that we utilize different weights w_{S_j} for different source domains in multi-source domain adversarial training. The other is to use pseudo-labels to update the target feature extractor. We designed a set of ablation experiments on DAGM to verify the importance of each part. In the first experiment, we set the same w_{S_j} for all source domains. The target classifier was also composed of a combination of source classifiers fairly. Table 3 shows that the model is unavailable when the target domain is aligned to each source domain. In fact, during the training process, the network can hardly converge which proves that the negative transfer will reduce model performance when the target domain is aligned to a wrong source domain. And when the target data is shifted, applying pseudo-labels to constrain the feature extractor can efficiently avoid confusion in the classification system and thus improve model performance.

B.1 | Learning

In this paper, we choose to use the GRL for solving the minimax game between F_t and $\tilde{D}$. It works by inserting a gradient reversal layer (GRL) to multiply the gradient of F_t by -1 to learn F_t and $\tilde{D}$ simultaneously. The algorithm flow has been summarized in Algorithm 1. Another solution would be iteratively training the two objectives. For our method, This adversarial learning algorithm is represented by the Algorithm 2.

C.1 | Network Architecture

In DAMG experiments, our proposed network framework consists of three components: feature extractor, domain discriminator, and category classifier. The source domain feature extractor F_s and target domain feature extractor F_t have the same network architecture, and both of them use Resnet-50 without the last fc (fully connected) layers as the backbone, which consists of 1 convolutions layer (conv1) and 4 ResidualBlock (conv2, conv3, conv4, conv5). The details of Resnet-50 can be found in [45].

ALGORITHM 1 Learning algorithm for our proposed method

```
Input: $N$ source labelled datasets
    $\{X^{S_j}, Y^{S_j}\}_{j=1}^{N}$; target unlabelled dataset $X^T$; confidence threshold $\gamma$.
Output: target feature extractor $F_t$; target category classifier $C_t$; domain
    discriminators $\{\tilde{D}_{S_j}\}_{j=1}^{N}$
Pre-train $\{C_{S_j}\}_{j=1}^{N}$, $F_s$ and initiated $F_t = F_s$
while not converged, do
    Sample mini-batch from $\{X^{S_j}\}_{j=1}^{N}$ and $X^T$
    Update domain discriminator $\{D_{S_j}\}_{j=1}^{N}$ by Eq.6
    Calculate domain weight $\{w_{S_j}\}_{j=1}^{N}$ from $\{D_{S_j}\}_{j=1}^{N}$ by Eq.3 and Eq.4
    Estimate confidence for $x^T$ by Equation 2 with confusion scores
    offered by Equation 3. Samples $x^T \subset X^T$ with confidence larger
    than $\gamma$ get annotations $\tilde{y}^T$
    Initiated $C_t$ by Equation 9
    Update $F_t$ and $\{\tilde{D}_{S_j}\}_{j=1}^{N}$ by Equation 11
end while
return $F_t$; $C_t$; $\{\tilde{D}_{S_j}\}_{j=1}^{N}$.
```

The domain discriminator is composed of two fc layers where we present the fc layers as (input, output). The two fc layers are: fc1 (2048, 1024), fc2 (1024, 1). The classifier is a single fc layer, i.e.fc3 (2048, 1).

ALGORITHM 2 Learning algorithm for our proposed method

```
Input: $N$ source labelled datasets
    $\{X^{S_j}, Y^{S_j}\}_{j=1}^{N}$; target unlabelled dataset $X^T$; confidence threshold
    $\gamma$; adversarial iteration threshold $\beta$.
Output: target feature extractor $F_t$; target category classifier $C_t$; domain
    discriminators $\{\tilde{D}_{S_j}\}_{j=1}^{N}$
Pre-train $\{C_{S_j}\}_{j=1}^{N}$, $F_s$ and initiated $F_t = F_s$
while not converged, do
    for $1 : \beta$ do
        Sample mini-batch from $\{X^{S_j}\}_{j=1}^{N}$ and $X^T$
        Update domain discriminator $\{D_{S_j}\}_{j=1}^{N}$ by Equation 6;
        Calculate domain weight $\{w_{S_j}\}_{j=1}^{N}$ from $\{D_{S_j}\}_{j=1}^{N}$ by Equation 3 and
        Equation 4;
        Update the second discriminator $\{\tilde{D}_{S_j}\}_{j=1}^{N}$ and $F_t$ by Equation 7,
        sequentially
    end for
    Estimate confidence for $x^T$ by Equation 2 with confusion scores
    offered by Equation 3. Samples $x^T \subset X^T$ with confidence larger than
    $\gamma$ get annotations $\tilde{y}^T$
    Initiated $C_t$ by Equation 9
    Update $F_t$ by Equation 10
end while
return $F_t$; $C_t$; $\{\tilde{D}_{S_j}\}_{j=1}^{N}$.
```