IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 27, 2020, accepted February 13, 2020, date of publication March 2, 2020, date of current version March 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977821

A Weakly Supervised Surface Defect Detection Based on Convolutional Neural Network

LIANG XU[1], (Member, IEEE), SHUAI LV[1], YONG DENG[1], AND XIUXI LI[2]
[1]School of Automation, Guangdong University of Technology, Guangzhou 510006, China
[2]School of Chemistry and Chemical Engineering, South China University of Technology, Guangzhou 510640, China

Corresponding author: Liang Xu (celiangxu@gdut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 21376091, and in part by the Guangdong Provincial Key Lab of Green Chemical Product Technology under Grant GC201812.

ABSTRACT Surface defect detection is a critical task in product quality assurance for manufacturing lines. The deep learning-based methods recently developed for defect detection are typically trained using a supervised learning strategy and large defect sample sets. Conventional methods often require additional pixel-level labeling or bounding boxes to predict the location of defects. However, the number of required samples and the time-intensive annotation process limits the practical use of these algorithms. As such, this study proposes a weakly supervised detection framework in which a CNN model is trained to identify surface cracks in motor commutators. The model was trained using small subsets of defect samples (∼5–30) and does not require a pre-trained network. This approach consists of localization and decision networks that simultaneously predict both the location and probability of defects. A new loss function was also developed to identify abnormal regions in a sample with accessible image-level labels. A collaboration learning strategy was then applied to utilize the loss function and compensate for imbalances at the pixel level. Experimental results using a small number of image-level training labels from a real industrial dataset exhibited a 99.5% recognition accuracy, which is comparable to relevant methods using pixel-level labels.

INDEX TERMS Surface defect detection, quality control, weak supervision, convolutional neural network, localization network.

I. INTRODUCTION

Surface defect detection, an essential task in manufacturing production quality assurance, has historically been performed manually. However, this approach is highly inefficient, subjective, and time-consuming. As such, computer vision-based inspection techniques have been developed in recent years to assist with or even replace human intervention [1]–[3].

Conventional machine learning models rely on specific vision inspection tasks acquired by analyzing and extracting defect features manually. A decision is then made using either rule-based experience or learning-based classifiers. Support vector machines (SVMs), neural networks, and decision trees all utilize this type of approach, in which system performance depends heavily on the accurate representation of specific feature types. However, modern manufacturing technology requires product lines to be more robust, limiting the use of conventional machine learning models that require long development cycles to be adapted for different tasks.

The associate editor coordinating the review of this manuscript and approving it for publication was Shagufta Henna.

In recent years, deep learning (DL) methods have been successfully applied to a variety of image related tasks [4], [5].

These algorithms are more easily adapted to different products and they have produced accurate results when applied to surface quality control [6]. Unlike conventional machine learning, DL models can automatically learn features from low-level data, eliminating the need for manual intervention. However, the success of these algorithms depends heavily on the labeled images used to train the effective deep network, which must efficiently represent a broad range of features. Furthermore, the localizability and interpretability of defects must be considered.

Various approaches to representative feature collection have been proposed in the literature, to compensate for a lack of defect samples. This has included partitioning image patches to augment samples [7], re-using pre-trained models with transfer learning [8], [9], and using pixel-level labels to increase sample quantities [10] in a supervised model. For example, He *et al.* [11] developed a defect classification model for steel surfaces using a semi-supervised network. These methods have produced viable results but they also

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

have specific limitations [7]. For example, identifying defect regions by partition image patches can be time-consuming in the forward process [8], [9]. Transfer learning is effective but imposes restrictions on network structure design [10]. Pixel-level labels are highly precise but labor-intensive [11]. As such, we propose an effective method to predict the defect regions with image-level labels.

The interpretation of defects is an indispensable task in detection processes. It is typically conducted with visual localization and has been attempted using a variety of algorithms. Image patch partitioning is a highly accurate selection strategy, but is very difficult and time-consuming [7]. Object detection algorithms represent defective regions using a recognition model, which is trained by an additional bounding box label [12]. Semantic segmentation networks are trained by sample labeling in pixel-level and segment defect regions [10], [13]. However, both semantic segmentation and object detection are limited by the required labeling information.

This study proposes a novel framework for defect detection, one that consists of localization and decision networks. The model was trained using weakly labeled image-level samples, and considered defect detection to be an anomaly detection problem. Unlike conventional models, a large number of defect images are not required and labels are not annotated at the instance or pixel levels in an image. In addition, our framework is robust for different network structures as it does not include transfer learning. The primary contributions of this methodology are as follows:

1) A new multi-task framework is proposed for surface defect detection using weakly supervised learning(WSL), which consists of localization and decision networks. This approach avoids the need for large collections of defect-labeled samples.

2) A visual localization and classification description is developed for surface defects. The localization network is trained using image-level labels and outputs a heat map of potential defect locations. The output of the localization network was added to the decision network to improve classification performance.

3) A new WSL loss function is proposed for overcoming the limitations of small pixel-annotated samples, which is used to train the localization network with image-level labels. A collaboration learning strategy is applied in the training process to optimize the loss function and address pixel-level data imbalances, in which defective regions are differentiated from the background by the localization network.

4) After training with only approximately 5–30 defect images samples, the model achieves accurate defect detection without the use of a pre-trained network. Successful localization and decision results are demonstrated using a real-world industrial dataset.

The remainder of this paper is organized as follows. Section II examines previous research in automated machine vision inspection. Section III describes the proposed framework in detail, including a new WSL loss function and learning process. Section IV presents experimental results and compares the proposed model to conventional techniques. Section V presents our conclusions.

II. RELATED WORK

In conventional machine learning algorithms, the classification of individual pixels relies on the representation of manually identified features. Each pixel is classified as a defect or non-defect, based on features calculated from neighboring pixels. The latest research work [14] reports that a novel probabilistic salience framework is based on two saliency features (absolute intensity deviation and local intensity aggregation) and is proposed to shift the intensity of each pixel according to its saliency during its iterative process. Common features involve geometric and statistical descriptors (i.e., length, width, area, mean, and standard deviation), as well as localized wavelet decomposition [15]. Machine learning algorithms such as SVM [16], fuzzy logic [17], and random forest [2] have also been used, but have mostly been outperformed by deep-learning models based on computer vision techniques [18], [19].

The related task of automated surface inspection (ASI) has also been reported in previous studies, in which surface defects are generally described as local anomalies in homogeneous textures. ASI algorithms can be divided into four categories, depending on the properties of the surface texture [7]. These categories are structural models [20], [21], statistical methods [22], filter-based approaches [23], and model-based techniques [24].

Deep learning (DL) has recently become the most influential technology in computer vision and pattern recognition problems. Unprecedented achievements in image classification have been produced by convolutional neural networks (CNNs) such as AlexNet [25], VGG [26], ResNet [27], and DenseNet [28], which have outperformed conventional classification models like SVM. As such, DL was applied to surface defect detection after AlexNet was proposed [25]. For instance, Meier *et al.* [29] used a CNN for supervised steel defect classification. However, their work was limited to a shallow network in which ReLU and batch normalization operations were not included. Ferreira *et al.* [30] used a CNN to extract intrinsic stone patterns in small image patches during granite tile classification. Chen *et al.* proposed a novel vision-based method that used a deep CNN in the detection of fastener defects [31]. Martelli *et al.* used a DL-based spatio-temporal image analysis system to classify defects in metallic gearboxes [32]. However, these techniques are either based on image classification using DL models or depend heavily on annotated data acquired manually, which cannot predict defective regions using only image-level labeling.

Collecting and labeling large datasets for defect inspection is difficult in industrial environments and several techniques have been proposed to address this issue [7]–[11]. Ren *et al.* [7] developed a generic approach that requires partitioning of image patches. A classifier is then developed from these image patch features, which are transferred

from a pre-trained DL network. In the defect segmentation stage, the trained classifier iterates over all overlapping image blocks to output the classification probability for every pixel in an input image.

Transfer learning can also reduce training data requirements by transferring pre-trained model weights [8], [9]. Ferguson *et al.* [8] applied transfer learning to the CNN-based inspection of casting defects in X-ray images. This model was first trained using two large public image datasets and then optimized with a relatively small casting dataset. Transfer learning was also applied to the online training and classification of Mura defects, in which AlexNet was trained using the ImageNet LSVRC-2012 dataset and used as a feature extractor [9]. Learned features were then transferred from the pre-trained AlexNet to the network model. However, models pre-trained with non-industrial datasets have limited adaptability in complex industrial environments.

Tabernik *et al.* [10] proposed an anomaly detection algorithm based on semantic segmentation frames. This model was trained with pixel-level labels to effectively augment the capacity of the training dataset, but was only able to learn 25–30 defective training images. However, acquiring pixel-level labels is both time- and labor-intensive. He *et al.* [11] proposed a semi-surprised defect classification approach to compensate for a lack of labeled samples in supervised training. This approach used generative adversarial networks (GANs) to generate large quantities of unlabeled data. Both labeled and unlabeled samples have been used to train classifiers with different learning strategies, but have been unable to predict defect regions.

Defect localization, a necessary component of detection tasks, has been achieved using three different techniques in previous studies [6], [10], [12], [13]. Ren *et al.* [7] designed a patch classifier that was trained using features transferred from Decaf. The resulting classifier output the probability of an input image belonging to a given class. The primary issue with this technique is determining how to optimally select patch sizes. Defect localization has also been approached as a type of object detection task [12]. Jiangyun et al. used the You Only Look Once (YOLO) convolutional network, first introduced for object detection, to automatically extract multi-scale features from surface defects and inspect image regions for steal strips. However, this model required additional bounding boxes to be labeled in the training process. Defect localization has also been pursued as a semantic segmentation task. For example, Tabernik *et al.* [10] developed a segmentation network for performing pixel-based localization of surface defects. In this model, each pixel was considered an individual training sample to increase the effective size of the training data and to avoid overfitting. Lin *et al.* [33] used a class activation map (CAM) to localize defect regions in the construction of a CNN model for LED chip inspection. The predicted classification was mapped back to the previous convolution layer and weighted to generate the CAM (with higher weighted values indicating object location).

The present study represents the first weakly supervised multi-task approach for surface-defect detection of industrial products, specifically surface cracks in motor commutators. The proposed technique differs from conventional defect detection models based on image classification using DL [25]–[30], image patch partitioning [7], object detection [12], and semi-supervised learning [11]. In contrast, it is similar to the semantic segmentation developed by Tabernik *et al.*, in which a supervised learning strategy was used to train a segmentation network with pixel-level labels. However, our algorithm implements a weakly supervised learning strategy (an uncertain supervision) to train the localization network with accessible image-level labels. Unlike previous studies [11], the resulting model directly outputs defect localization. This approach is also similar to class activation mapping, since a heat map was used as a visual interpretation of classification results, which predicted object location [33]. However, unlike a CAM, our model outputs defect location probabilities using weakly supervised learning. The decision network also combines results from the location network, improving classification results.

III. PROPOSED FRAMEWORK

The automated localization and classification of surface defects is a complex task. As such, this study proposes a novel weakly supervised defect detection algorithm using a CNN. Specifically, the methodology focuses on surface crack detection in motor commutators (see FIGURE 1). In this study, weakly supervised learning was investigated in multi-task frames, consisting of both localization and decision networks. The decision network determines whether defects are present in an image and the localization network identifies corresponding defect regions.

A novel loss function is also proposed to train the localization network to recognize the defect regions using only image-level labeling. This model was trained in three stages, applied sequentially to the localization and decision networks, followed by a fine-tuning step. Every pixel was considered an individual sample when training the localization network, to augment effective sample capacity and to improve the pre-training model. The proposed architecture trains from scratch using image-level labels to predict both the probability and position of defects, thereby improving performance.

As shown in FIGURE 1, this framework consists of three parts: shared layers and two sub-networks, as well as a location network (LNet) and a decision network (DNet). The shared layer extracts ample representation features from the input image, to serve as inputs for the two sub-networks. The LNet then takes output features from the shared layer as inputs and outputs a single-channel image (called a heat map) to predict defect locations. Each pixel in the heat map exhibits a value between 0 and 1, which indicates the probability that a defect exists at the corresponding location in the original image. The DNet receives output characteristics from the shared layer and the LNet outputs a heat map that predicts the probability of defects being present at specific locations.

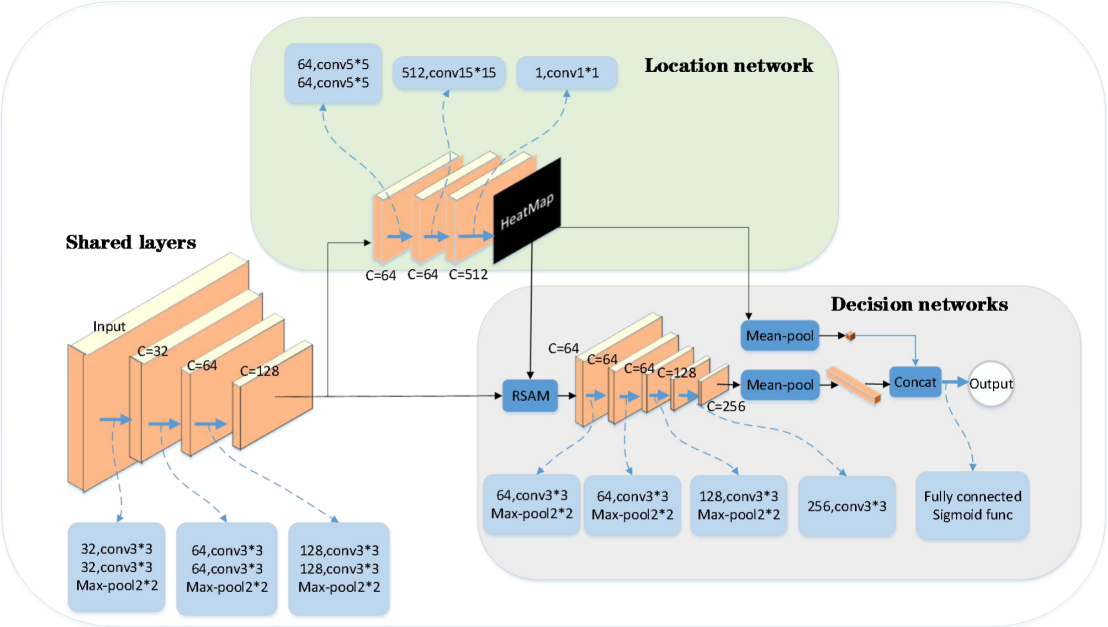

FIGURE 1. The proposed framework.

The output of the DNet also has a value between 0 and 1, indicating the probability of a defect existing somewhere in the image. The proposed network model can simultaneously predict whether defects are present in the input image, as well as the location of the defects. Each of these steps is discussed in detail below.

A. NETWORK STRUCTURE DESIGN

1) SHARED LAYERS

The shared layer is based on a VGG [26] architecture and consists of six convolutional layers and three maximum pooling layers. Batch normalization and non-linear ReLU layers were included after each convolutional layer, which used a 3×3 kernel with a step size of 1 to capture small targets (defects) in high-resolution images. The pooling layers used 2×2 kernels, which were more likely to capture image changes and provide local information differences. The step size in each maximum pooling layer was set to 2, thereby reducing the runtime but also the resolution. The number of channels in each convolutional layer was gradually increased to maintain the size of the parameter space. As shown in FIGURE 1, the shared layer output featured 128 channels, each of which was 1/64 the size of the input image due to the maximum pooling layers (see Table 1).

2) LNet

LNet input was the output feature in the shared layers. It consisted of four convolutional layers with 64 channels and a 5×5 kernel in the first two layers. The third layer included 512 channels and a 15×15 kernel. The larger convolutional kernel was used to increase the size of the network receptive field. The last convolutional layer included only a single channel with a 1×1 kernel, which can be regarded as a linear transformation. A batch normalization layer and a non-linear ReLU utilized a non-linear sigmoid activation function, which mapped the output range to 0 to 1. All convolutional operations in the LNet were performed in a single step and no pooling layer was included, leading to variable output sizes. Detailed LNet parameters are shown in Table 2. The output heat map is a single-channel image whose width and height are one-eighth the size of the original image, with each pixel corresponding to an 8×8 region. Values closer to 1 indicate higher defect probabilities. This LNet output was used as the input to the DNet.

TABLE 1. The network architecture and shared layer parameters.

Name	Layer	Filters	Kernel size	Stride
S1	Conv-BN-Relu	32	3×3	1
S2	Conv-BN-Relu	32	3×3	1
S3	Max-Pooling	-	2×2	2
S4	Conv-BN-Relu	64	3×3	1
S5	Conv-BN-Relu	64	3×3	1
S6	Max-Pooling	-	2×2	2
S7	Conv-BN-Relu	128	3×3	1
S8	Conv-BN-Relu	128	3×3	1
S9	Max-Pooling	-	2×2	2
S10	Conv-BN-Relu	128	3×3	1

TABLE 2. LNet architecture and parameters.

Name	Layer	Filters	Kernel size	Stride
L1	Conv-BN-Relu	64	5×5	1
L2	Conv-BN-Relu	64	5×5	1
L3	Conv-BN-Relu	512	15×15	2
L4	Conv-BN-Sigmoid	1	1×1	1

TABLE 3. DNet architecture and parameters.

Name	Layer	Filters	Kernel size	Stride
D1	RSAM	64	3×3	1
D2	Conv-BN-Relu	64	3×3	1
D3	Max-Pooling	-	2×2	2
D4	Conv-BN-Relu	64	3×3	1
D5	Max-Pooling	-	2×2	2
D6	Conv-BN-Relu	128	3×3	1
D7	Max-Pooling	-	2×2	2
D8	Conv-BN-Relu	256	3×3	1
D9	LSE	_	3×3	1
D10	Dense layer	266	2×2	2
D11	Dense layer	1	3×3	1

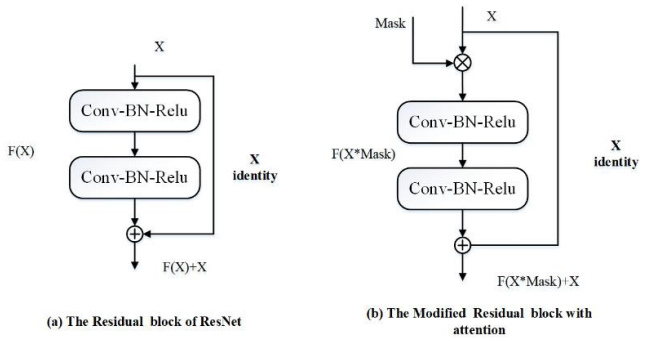

FIGURE 2. The residual spatial attention module.

3) DNet

A detailed list of structural parameters for the binary DNet are shown in Table 3. The front end is a residual spatial attention module (RSAM), used to improve DNet performance by weighting areas identified by the LNet (see FIGURE 2b). FIGURE 2a) shows the residual ResNet [27] module used to improve deep neural network performance by fitting the residuals. In contrast, the RSAM included attention mask input in the forward propagation step. In this process, each channel of the input feature is multiplied by the attention-weighted mask, causing the DNet to focus more attention on heavily weighted areas. The RSAM also adds feature tensors before and after the input mask in the next layer, to prevent a lack of information after attention weighting. As such, the output from shared layers was used as input features and the output of the LNet was used as the attention mask, causing the DNet to focus more on areas identified by the LNet. Four convolutional layers (with 64, 64, 128, and 256 channels) and three maximum pooling layers were stacked after the RSAM. Batch normalized and non-linear ReLU layers were included after each convolutional layer and each maximum pooling layer had a step size of 2, which reduced the height and width of the feature map by a factor of 2. Output features inthe shared layers were transformed into 256 channels via the RSAM, stacked convolutional layers, and the max pooling layer. The size of the final output feature was 1/64 of the original input feature. This DNet structure and corresponding parameters are shown in Table 3.

The back end of the DNet consisted of a global average pooling and two fully connected layers with 266 nodes and one node, respectively. The global average pooling eliminates spatial information in the feature map and produces fixed-length vectors, reducing the need to maintain a constant input image size. These 256-channel convolution features were transformed into vectors of length 256 using the global average pooling. The output LNet heat map was then transformed, using the global average pooling, into a vector of length 1. These two vectors were then connected to form a new 266-dimensional vector to be used as input to the fully connected layer, in which the last node is a neuron output of 1. A value between 0 and 1 represents the probability of a defect existing anywhere in the image, output through the sigmoid activation function.

B. WEAKLY SUPERVISED LOSS

While pixel labels for non-defective samples can be acquired directly in industrial settings, defect labels for defective samples must be added manually, which is cost-prohibitive and time-consuming. In this study, images containing defects are referred to as positive samples and non-defective images are referred to as negative samples.

In order to avoid the tedious process of rectangular frame labeling (or pixel labeling), we propose a new weakly supervised objective function $Loss_{weak}$, which includes $Loss_{positive}$ and $Loss_{negative}$ for training the LNet with only image-level labels. Different optimization targets in the objective function are used for positive and negative samples during training. $Loss_{positive}$ and $Loss_{negative}$ guide the network in learning how to identify defective and non-defective pixels in the sample, respectively. Table 4 introduces the variables and descriptions used in the paper.

TABLE 4. Variable representations and descriptions.

Variable	Description
X	Input image
y(X)	Label for X
Φ	Model parameters
$p(y\|(X,\Phi))$	The predicted model for X
w,h	The height and width of X
χ	The pixel set for X
x_i	A pixel in X
$q(y\|(x_i,\Phi))$	The predicted model for x_i

The label for the input sample X is represented by y (X), where a value of 1 indicates the presence of defective pixels in the image (positive samples), 0 indicates the absence of defective pixels (negative samples), and Φ is the network learning weights. $p(y|(X, \Phi))$ represents the predicted probability that a defect is present in the input image. In binary image classification tasks, objective functions are commonly used as binary classification cross-entropy loss functions:

$$H[y(X), p(y(X, \Phi)] = y(X)log(p(y|X, \Phi)) + (1-y(X)) \log (1-p(y|X, \Phi)). \quad (1)$$

Here, χ represents the pixel set for the input sample X; h and w represent the height and width of the image, respectively; and $y(x_i)$ represents the label of pixel x_i in the image. Values of 1 and 0 indicate the pixel does and does not belong

to a defective area, respectively. The term $q(y|(x_i, \Phi))$ represents the predicted probability that pixel x_i belongs to the defective group. The cross entropy-based objective function in the foreground (defect) segmentation task is defined for positive samples as:

$$H_\chi\left[y(x),q(y\,|x,\,\Phi)\right] = -\frac{1}{h*w}\sum_{i=1}^{N}\Big[y(x_i)\log\left(q(y\,|x_i,\,\Phi)\right) + (1-y(x_i))\log\left(1-q\left(y\,|x_i,\,\Phi\right)\right)\Big]. \tag{2}$$

When the input image is a positive sample ($y(x) = 1$), pixels are divided into defective and non-defective categories. The expected entropy is minimized when all pixels in an input image are determined to be either defective or non-defective. Entropy increases as the LNet outputs a heat map with an increasingly uniform distribution of both defective and non-defective pixels. As such, we converted the positive sample foreground (defective pixel) segmentation problem into an unsupervised maximum entropy optimization problem, thereby avoiding the use of defective labels and including only image-level labels. The expected maximum entropy function $Loss_{positive}$ for all pixels in a positive sample is defined as:

$$\begin{aligned} &Loss_{positive}\\ &\quad= -H_\chi\left[q(y\,|x,\,\Phi)\right]*y\,(\mathrm{X})\\ &\quad= -H\left[\frac{1}{h*w}\sum_{i=1}^{N}\left[q(y\,|x_i,\,\Phi)\right]\right]*y(\mathrm{X})\\ &\quad= -H\left[Q(y\,|\mathrm{X},\,\Phi)\right]*y(\mathrm{X})\\ &\quad= \Big[Q(y\,|\mathrm{X},\,\Phi)\log\left(Q(y\,|\mathrm{X},\,\Phi)\right)\\ &\qquad+(1-Q(y\,|\mathrm{X},\,\Phi))(\log\left(1-Q(y\,|\mathrm{X},\,\Phi)\right)\Big]*y(\mathrm{X}). \end{aligned} \tag{3}$$

Here $Q(y\,|\mathrm{X},\,\Phi) = \frac{1}{h*w}\sum_{i=1}^{N} q(y\,|x_i,\,\Phi)$ and $y(\mathrm{X})$ is 1 when the input is a positive sample.

Cross-entropy loss was used to optimize negative samples but the two objective functions were not on the same scale. In this case, the larger objective function will dominate the direction of the gradient. Since non-defective samples satisfy $y(\mathrm{X}) = 0, p(x_i) = y(\mathrm{X}) = 0, \forall x_i \in \hat{\mathrm{E}}\chi$ (and the expectation value of all pixels is 0), the problem of optimizing all pixels can be converted into a problem of optimizing expectation values. The two objective functions are then on the same order of magnitude and the cross-entropy function for negative samples can be rewritten as:

$$\begin{aligned} Loss_{\mathrm{negative}} &= H_\chi\left[y(x),q(y\,|x,\,\Phi)\right]\\ &= \left(y\,(x)\log\left(\frac{1}{h*w}\sum_{i=1}^{N} q(y|x_i,\,\Phi)\right)\right.\\ &\quad+(1-y\,(x))*\log\\ &\quad\left.\left(1-\frac{1}{h*w}\sum_{i=1}^{N} q(y|x_i,\,\Phi)\right)\right)*(1-y\,(x))\\ &= -\log\left(1-Q(y\,|\mathrm{X},\,\Phi)\right)*(1-y(\mathrm{X}))^2, \end{aligned} \tag{4}$$

where $y(\mathrm{X}) = 0$ when the input is a negative sample.

The LNet optimization objective function can then be defined as:

$$\begin{aligned} Loss_{\mathrm{weak}} &= \alpha*Loss_{negative} + Loss_{positive}\\ &= \alpha*\left[-\log\left(1-Q(y\,|\mathrm{X},\,\Phi)^2\right)*(1-y(\mathrm{X}))\right]*y(\mathrm{X})\\ &\quad+Q(y\,|\mathrm{X},\,\Phi)\log\left(Q(y\,|\mathrm{X},\,\Phi)\right)\\ &\quad+(1-Q(y\,|\mathrm{X},\,\Phi))\left(\log\left(1-Q(y\,|\mathrm{X},\,\Phi)\right)\right), \end{aligned} \tag{5}$$

where α is a collaborative learning factor that controls the learning speed of both non-defective and defective samples ($\alpha > 0$).

C. LEARNING

We propose a three-stage collaborative learning strategy for training the LNet and a shared network using the weakly supervised loss function proposed in (6). Network weights are then fixed and the DNet is trained using the binary image classification cross-entropy loss function of (2). Finally, the entire network is fine-tuned using objective functions. This three-stage learning approach trains the network to identify defective and non-defective pixels using only image-level labels, thereby improving performance for small data sets.

Since positive and negative samples are unbalanced in the dataset, the distribution of non-defective and defective pixels is also unbalanced. This complicates network convergence during random sampling of training data. A novel sampling process was developed to compensate for image-level imbalances and to alleviate this issue. During the learning process, the batch size was set to one and defective and non-defective samples were alternately trained to ensure that images of the same category did not appear in adjacent positions. A training step that includes both a positive and negative sample is defined as an iteration. Meanwhile, a collaborative learning strategy was implemented during LNet training with weakly supervised loss, which effectively resolved imbalances at the pixel level.

1) LNet COLLABORATIVE LEARNING

The weak supervision loss function proposed in (6) was used to identify defective regions with image-level labels, in which $Loss_{negative}$ and $Loss_{positive}$ controlled the learning of non-defective and defective pixel features. Since the distribution of pixels in positive samples was unbalanced, the collaborative learning factor α in (6) controlled the learning speed for both sample types. As defective areas in positive samples were typically small, α was often set to a number greater than 1. During training, α was initialized with a constant (α_{init}) and updated after every n iterations during the total N iterations. The updated value at step i is given by:

$$\alpha_i = \alpha_{init}*\frac{cos\,(i/k*2\pi)+3}{4} \tag{6}$$

where k is the total number of updates: $k = round(N/n)$, $i \le k$.

As shown in FIGURE 3, the value of α first decreased and then increased, reaching a minimum value of $\alpha_{\mathrm{init}}/2$. In the

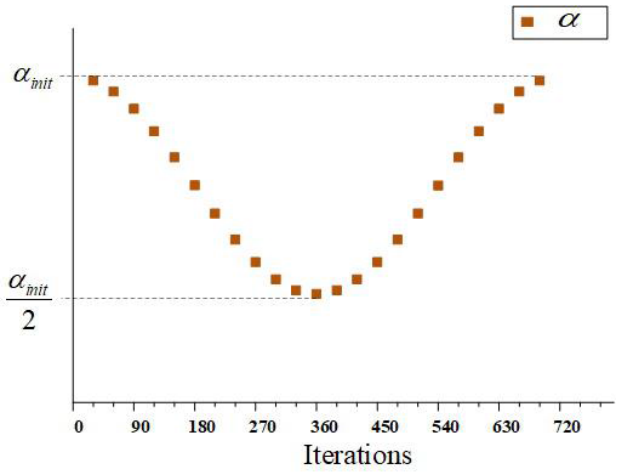

FIGURE 3. Variations in α during training the LNet.

early training stages, α is large and $Loss_{negative}$ dominates, which biases the network towards non-defective pixels in the learning process. In the middle stages, α is relatively small and the influence of $Loss_{positive}$ gradually increases as the network predicts defective pixels that differ from the back-background. In the latter stages, α gradually increases and the influence of $Loss_{negative}$ also increases to prevent the background from being incorrectly identified as a defective pixel.

Algorithm 1 LNet Collaborative Training

Input: dataset $D = \{x_i\}_{i=1}^{N}$, $y(x_i) \in [0, 1]$
Initialization: the number of steps N, the model Φ, the collaborative factor α, the α update cycle n, and the learning rate lr.
For step $= 1, 2, \ldots,$ N do:
  If step % n $= 0$:
    $i = step/n$; $k = round(N/n)$
    $\alpha \leftarrow$ Eq. (6) with i and k
  $\leftarrow$ sample a positive image from D
  Train the model to minimize Eq. (5) and update Φ with X
  $\leftarrow$ sample a negative image from D
  Train the model to minimize Eq. (5) and update Φ with X
End for

During the optimization process, $Loss_{positive}$ and $Loss_{negative}$ work collaboratively in the objective function to differentiate between defective and non-defective pixels. Weakly supervised training enables the LNet to locate defects and allows shared layers to acquire suitable initialization parameters to facilitate subsequent training. Equation (6) was used to train the shared network and LNet cooperatively, the specific implementation of which is shown in Algorithm 1.

2) DNet LEARNING AND NETWORK FINE-TUNING

Subsequent steps included training the DNet. Network weights for the shared layers and LNet were then held constant and the cross-entropy loss in (2) was used as the objective function to optimize the DNet. Once training was completed, the DNet could accurately predict the probability of defects in an image. Finally, Equations. (2) and (6) were combined to train the entire network using a lower learning rate. No network layers were frozen during this process in order to fine-tune the entire network.

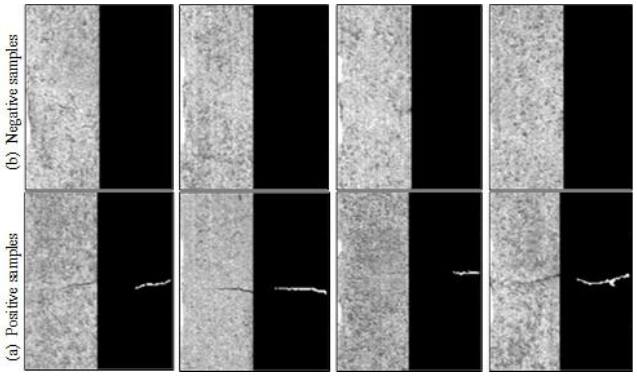

FIGURE 4. Sample from the KolektorSDD dataset.

IV. EXPERIMENTAL VALIDATION

A. DATASET DESCRIPTION

The KolektorSDD dataset (http://www.vicos.si/ Downloads/ KolektorSDD) consists of labeled surface crack images on plastics electronic commutators, as shown in FIGURE 4. This set contains 50 commutator samples, each with ∼8 surfaces, totaling 399 pictures. These data were acquired in an industrial environment to ensure high- quality images, with a resolution of 1408 × 512 pixels. Among of these, 52 exhibit clearly visible defects and serve as positive samples. The remaining 347 images are defect-free negative samples. The first row in FIGURE 4(a) contains defect-free negative samples and the second row contains positive samples with defects. Each image is labeled at the pixel level. The weakly supervised loss function proposed in this study does not require the use of pixel-level labeling, which is included only for a visual comparison with the positioning results.

B. PERFORMANCE METRICS

In this study, surface defect detection is treated as a binary image classification problem. The primary purpose is to classify images into two categories: positive (with defects) and negative (without defects). Since pixel-level positioning is not important for defect detection, the LNet only provides visualization results. As such, LNet positioning accuracy was not evaluated, only picture classification errors. LNet visualization results were output as interpretation results for network discrimination.

Three-fold cross validation was used to ensure that the same picture did not simultaneously appear in the training and testing sets. Average precision (AP) and the number of false call samples were used to assess classification accuracy. AP is determined using the calculated area under the precision-recall curve, making it a precise representation of comprehensive model performance for different thresholds, particularly when the dataset contains large quantities of negative (non-defective) samples. False positive (FP) represents negative samples that were misreported as positive (defective) samples and false negative (FN) represents positive samples that were misreported as negative (non-defective) samples. The TP, FP, TN, and FN values reported below were calculated under the best F-measure. The F-measure metric is defined as:

$$F = \frac{(\gamma^2 + 1) * P * R}{\gamma^2 * P + R}, \tag{7}$$

where P and R are respectively the precision and recall:

$$P = \frac{TP}{TP + FP}, \tag{8}$$

$$R = \frac{TP}{TP + FN} \tag{9}$$

In this study, γ was set to 1 to utilize the most common F1-measure.

C. IMPLEMENTATION

Since the weakly supervised objective functions were not strongly constrained to each pixel, the network occasionally classified the image boundary as anomalous when the convolution padding consisted of null values. Therefore, all convolution padding used a mirror mode in shared LNet layers and a stochastic gradient descent (SGD) optimizer was used in all the experiments. Training a pair of positive and negative samples constituted a single iteration.

The first step involved training the LNet and shared layers using the supervised loss proposed in (6). The learning rate was set to 0.1 and the collaborative learning factor α in the objective function was initialized to 8. This value varied periodically, according to (7), being updated after every 30 iterations. After 750 iterations, the defect location was determined by the output heat map.

The second step included training the DNet. After LNet training, the shared layers and LNet weights were frozen and the DNet weights were optimized. The initial learning rate was set to 0.01 and the total number of iterations was 1200. After 600 iterations, the learning rate was reduced to 0.001. After training, the DNet output the probability of defects existing in the input image.

Finally, Equations. (2) and (6) were used to jointly fine-tune the network, setting the learning rate to 0.001 and the number of iterations to 300. During this process, the collaborative learning factor α was fixed at its initial value of 8. Once joint training was completed, this framework achieved a simultaneous prediction of defect positions and existence probability. The following section evaluates DNet performance under different conditions. The network was constructed using the PyTorch machine learning library developed by Facebook' AI Research Lab. All simulations were conducted using a PC with an Intel Core i7-7820X CPU running Windows 10, two GTX1080Ti GPUs (total 22 GB), and 32.0 GB of RAM.

D. EVALUATION OF DNet PERFORMANCE FOR DIFFERENT CONFIGURATION GROUPS

The primary DNet parameter configurations included: (i) the number of unique defect samples, (ii) freezing/not freezing shared layers and LNet weights, and (iii) randomly rotating training images by 0°, 90°, 180°, and 270°. The proposed method was evaluated using experimental results for differing quantities of defective samples, and proved to be robust. Selectively frozen and non-frozen DNet weights assisted in evaluating the effects of LNet training on final decision results. Finally, the impact of random rotations on network discrimination was evaluated, as shown in FIGURE 5.

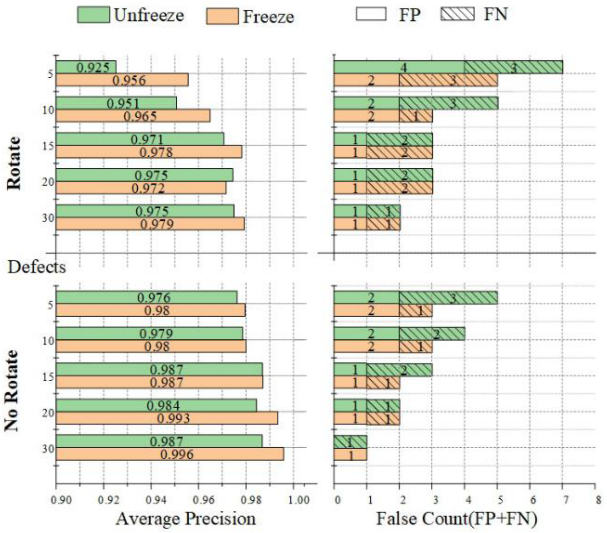

FIGURE 5. Classification results using the proposed model, applied to the KolektorSDD set. The average precision (AP) and number of error samples (FC) were determined with different parameters.

The figure displays detailed results using different color-labeled bar charts, where frozen and non-frozen weights are indicated as orange and green, respectively. The top and bottom groups show results with and without random rotations made during training, respectively. The left group shows the average accuracy under all experimental conditions and the right group shows the number of error samples. These results indicate that only one negative sample was detected as a positive sample (with frozen weights, no random rotations, and 30 positive samples). This corresponds to an optimal AP of 99.5%. The effect of a single learning configuration was evaluated by observing changes in the average performance for each specific setting parameter.

TABLE 5. Average performance using different training sets.

Training set	Average precision (%)	Improvement
Freeze/unfreeze	97.85 / 97.09	0.76
With rotation/without	96.45 / 98.49	−2.01

FIGURE 5 also suggests the proposed method is not sensitive to the number of defect samples, exhibiting strong robustness. Even with only five defective samples, the AP value reached 97.96%. The highest F-measure occurred when only three samples were misclassified, two of which were FPs and one was a FN. Random rotation did not result in significant performance increases (as expected), but had a detrimental effect on identification results. As shown in Table 5, rotating the image during training caused the average performance to decrease by 2.01%. FIGURE 5 indicates that frozen shared layers and LNet weights led to significantly better results. As seen in Table 5, for the case of frozen weights, the evaluation index AP increased by an average of 0.76%. This indicates that freezing weights can improve network performance for small sample sets.

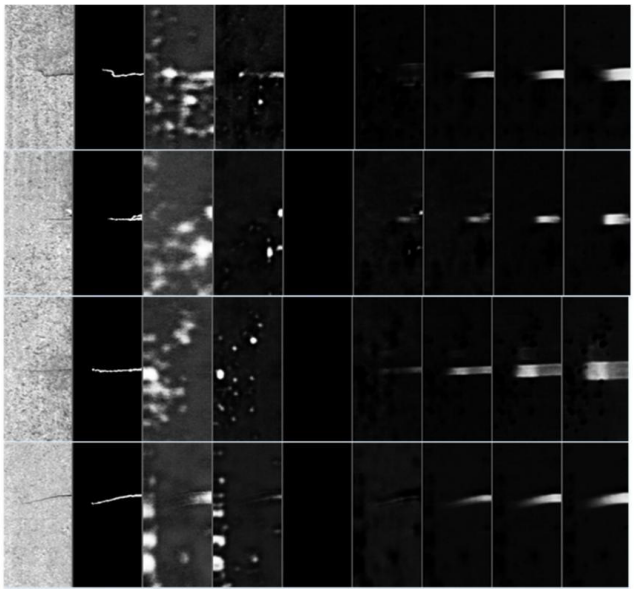

FIGURE 6. LNet output for various training iteration times. The images from left to right show data samples, pixel labels, and output results after 30, 120, 240, 360, 480, 600, and 720 iterations. Also shown are (a) the original image, (b) G.T., (c) output 1, (d) output 2, (e) output 3, (f) output 4, (g) output 5, (h) output 6, and (i) output 7.

E. VISUALIZATION OF THE COLLABORATIVE LEARNING PROCESS

The LNet identified defect locations by visualizing intermediate results during different periods of the collaborative learning process. The first and second columns in FIGURE 6 show sample input images and pixel labels, respectively. Columns 3–9 show sequential LNet results after 30, 120, 240, 360, 480, 600, and 720 iterations. Results in the third column indicate that prediction results were poor after 30 iterations, with a large portion of the background area predicted to be defective. In the early training stages, α is relatively large and $Loss_{positive}$ dominates the loss function, which biases the network toward non-defective pixels during the learning process. Columns 3–5 demonstrate gradual classification improvements as all pixels are non-defective after 240 iterations. After additional training, α becomes relatively small, the influence of the loss function on positive samples gradually increases, and the network begins to dis- tinguish defective pixels from the background in columns 5–7. In the latter training stages, α gradually increases and the effect of $Loss_{negative}$ increases to prevent the background from being falsely classified as defective. Columns 7–9 indicate that once training is completed, the LNet can accurately predict defect locations. However, due to the width of the CNN reception field, the region around the defect was also identified as defective during the optimization process. This caused some blurring on the edges of defective regions.

F. LNet CONTRIBUTIONS

DNet results were evaluated with and without LNet training, as an indicator of LNet performance. Network and shared layer weights were frozen during LNet training. The DNet was trained directly when the LNet was not trained and AP was used as an evaluation indicator in related experiments. In addition, FPs were included as a supplemental evaluation index with a recall rate of 100%. The loss risk for FPs is much smaller than that of FNs in industrial production environments, which is indicative of the number of manual rechecks required to achieve the desired accuracy.

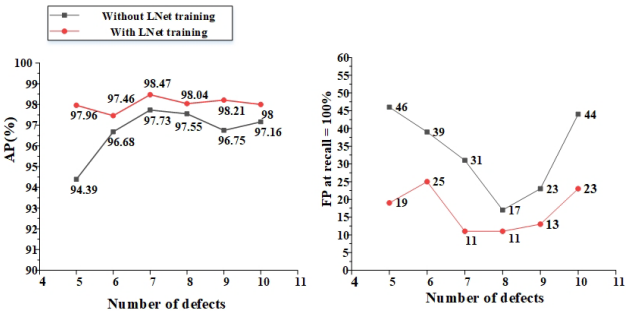

FIGURE 7. Results produced with and without LNet optimization. Output was evaluated using: (a) average precision (AP) and (b) the number of error samples at 100% recall.

The contribution of the LNet to the final results was evaluated when the defect sample set was relatively small. The number of positive (defective) samples in the training set was 5, 6, 7, 8,9, and 10 in the experimental configuration, the results of which are shown in FIGURE 7. The line chart on the left depicts changes in AP with and without LNet training (with trends represented by the red and black lines, respectively). As seen in FIGURE 7(a), the red curve is entirely above the black curve, indicating that AP was significantly improved by LNet training. With only 5–10 defective

TABLE 6. A comparison of conventional algorithms using the KolektorSDD dataset.

Method	Positive samples	FN	FP	False count	FP (at recall = 100%)	AP (%)
ViDi	33	0	5	5	7	99.0
U-Net based segmentation decisions	33	4	5	9	108	96.1
DeepLab v3+ based segmentation decisions	33	4	5	9	68	98.0
Segmentation decision approach	33	0	1	1	3	99.9
Pre-trained ResNet	33	0	2	2	40	97.8
Our approach	33	0	1	1	5	99.5
Our approach	10	1	2	3	51	98.0

samples, the tortuosity of the black curve is not pronounced, while the red curve exhibits obvious drops as the number of defective samples decreases. This indicates that LNet training can improve AP robustness with respect to varying sample sizes. The red curve in FIGURE 7(b) is entirely below the black curve, indicating that LNet training effectively reduced the average number of FPs at 100% recall.

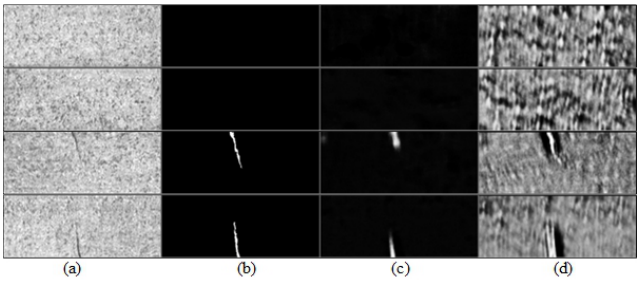

FIGURE 8. Series of LNet heat maps, including (a) initial images, (b) pixel labels, (c) output heat maps with LNet training, and (d) output heat maps without LNet training.

FIGURES 8(c) and 8(d) show output LNet heat maps with and without training, respectively. These heat maps were used as attention weights in the DNet's autonomous-learning RSAM, which focused on defective areas when the LNet was not trained. When the LNet was trained, the RSAM effectively guided the DNet to focus on areas where defects were located. A comparison of FIGURES 8(c) and 8(d) suggests that training the LNet increased differences between the foreground and background, highlighted defective pixels, and improved DNet performance.

G. A COMPARISON WITH CONVENTIONAL TECHNIQUES

Table 6 compares the proposed method with other conventional techniques. The results presented in this section involved the same experimental configuration and the same number of training and testing samples (33 defective images), each of which utilized triple cross-validation. Primary evaluation metrics included AP, false count, and FP at 100% recall. Compared with other methods, the proposed model exhibited competitive classification performance, achieving an accuracy of 99.5% for the KolektorSDD dataset. Tabernik et al. proposed a supervised two-stage network structure and used pixel labels to train the network [10]. The resulting AP reached 99.9% and produced only a single FN and FP. In addition, three FPs occurred at a 100% recall. The proposed model utilizes a weak supervision method with a three-stage learning strategy and requires only image-level labels. This approach offers comparable FN and FP rates, producing five FPs at a recall of 100%. Table 6 also provides experimental results using commercial software packages, such as the Cognex ViDi Suite (Cognex2018) [34], and two segmented networks (U-net [35] and DeepLabV3 + [36]).The ViDi model produced an AP of 99.0%, five false counts, and seven FPs at 100% recall. The U-Net produced an AP of 96.1%, nine false counts, and up to 108 FPs at 100% recall.

The DeepLab v3 + segmentation decision method produced an average accuracy of 98.0%, nine false counts, and 68 FPs at a recall of 100%. The proposed model outperformed DeepLabV3 +, Unet, and ViDi, as measured by the AP metric. A standard residual network ResNet [26] was also included in the comparison experiments. During training, the ResNet model used weights that were pre-trained with ImageNet, adjusting the number of nodes in the fully connected layer (2) for binary classification tasks. The ResNet achieved an AP of 97.8% and produced two FPs. There were 40 FNs at a recall of 100%. The proposed technique was also superior to a pre-trained ResNet18 across all indicators. Table 6 indicates that when the training set contained only 10 positive samples (with negative samples remaining unchanged), the average accuracy was lower than only ViDi and the segmentation-decision methods.

These experimental results suggest that the proposed method maintained good performance with fewer training samples. Weakly supervised learning and collaborative training effectively enhanced differences between defective and non-defective pixels, allowing the DNet to more effectively distinguish between samples.

H. ANALYSIS AND DISCUSSION

Experimental results from the KolektorSDD set indicate that our method achieved competitive performance using only image-level labeling, which is much simpler than pixel-level labeling. This benefit is the result of the proposed multi-task framework and three-stage learning process. Among these, the weak supervision approach and collaborative learning strategy enabled the LNet to identify defective areas that differed from the background, by enhancing the characteristics of the corresponding area. The included DNet used the output

LNet heat map as an attention mask, biasing the network toward anomalous regions in the input image.

A series of comparison experiments were conducted by freezing or not freezing the shared layers and LNet, the results of which showed that the positioning and decision tasks were highly correlated. In the 5–10 positive sample experiments, DNet performance improved after conducting weakly supervised LNet training. This indicated that differences between background and foreground features could be enhanced by the LNet, making this technique more effective.

Various optimal objectives were selected for positive and negative samples during training. The proposed weak (uncertainty) supervision method uses only image-level labels, which significantly reduces the required labeling time. The performance, however, is comparable to other supervised methods that do require instance or pixel labeling.

This study demonstrated that the adoption of a collaborative learning strategy can compensate for pixel-level imbalances. In the process of defect visualization, the area of interest can be gradually enlarged by varying the collaboration parameter α. This value is initially large, which biases the learning process toward non-defective pixels during the network learning process. As α decreases, the network gradually begins to predict defective pixels. In the latter stages, α gradually increases again to prevent the background from being mistaken for defective pixels. In this way, the network learns to distinguish pixel types and locate defects. A series of comparison experiments demonstrated that the proposed technique achieved an accuracy comparable to conventional models. It was also highly robust to variable positive sample quantities. Future work will focus on expanding the applicability of the algorithm, which is currently able to detect small anomalous features in large images. The edges of defects were also slightly blurred in the LNet output images, which could be improved by further development of the model.

V. CONCLUSION

A novel weakly supervised framework was proposed for surface defect detection using deep learning and demonstrated with surface cracks in motor commutators. A new weak supervision loss was developed to train the localization network using image-level labels. A collaborative learning strategy was implemented in the training process to assist the weak supervision algorithm and compensate for pixel-level data imbalances, in which defect position information can be directly output and explained. This network design enabled training with a small (approximately 5–30 defect images) number of samples, achieving excellent performance without pre-training. Experimental results demonstrated that our approach produced comparable accuracy for a weakly supervised mode, compared with fully supervised methods, using the KolektorSDD dataset. Future work will involve training the algorithm to identify defects in other materials, such as steel or glass.

ACKNOWLEDGMENT

The authors would like to thank LetPub for its linguistic assistance during the preparation of this manuscript.

The authors are grateful to all of the reviewers for suggestions and insights that improved the paper.

REFERENCES

[1] E. N. Malamas, E. G. M. Petrakis, M. Zervakis, L. Petit, and J.-D. Legat, "A survey on industrial vision systems, applications and tools," *Image Vis. Comput.*, vol. 21, no. 2, pp. 171–188, Feb. 2003.

[2] Y. Park and I. S. Kweon, "Ambiguous surface defect image classification of AMOLED displays in smartphones," *IEEE Trans Ind. Informat.*, vol. 12, no. 2, pp. 597–607, Apr. 2016.

[3] L. XU, H. B. XU, X. X. Li, and M. Pan, "A defect inspection for explosive cartridge using an improved visual attention and image weighted eigenvalue," *IEEE Trans. Instrum. Meas.*, to be published, doi: 10.1109/TIM.2019.2912237.

[4] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc IEEE Conf. Comput. Vis. Pattern Recognit*, Jun. 2012, pp. 3642–3649.

[5] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1713–1721, doi: 10.1109/CVPR.2015.7298780.

[6] X. Xie and M. Mirmehdi, "TEXEMS: Texture exemplars for defect detection on random textured surfaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 8, pp. 1454–1464, Aug. 2007.

[7] R. Ren, T. Hung, and K. C. Tan, "A generic deep-learning-based approach for automated surface inspection," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 929–940, Mar. 2018.

[8] M Ferguson, R. Ak, Y.-T. T. Lee, K. H. Law, "Detection and segmentation of manufacturing defects with convolutional neural networks and transfer learning," 2018, *arXiv:1808.02518*. [Online]. Available: https://arxiv.org/abs/1808.02518

[9] H. Yang, S. Mei, K. Song, B. Tao, and Z. Yin, "Transfer-Learning-Based online mura defect classification," *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 1, pp. 116–123, Feb. 2018.

[10] D. Tabernik, S. Sela, J. Skvarc, and D. Skocaj, "Segmentation-based deep-learning approach for surface-defect detection," *J. Intell. Manuf.*, vol. 31, pp. 759–776, May 2010, doi: 10.1007/s10845-019-01476-x.

[11] Y. He, K. Song, H. Dong, and Y. Yan, "Semi-supervised defect classification of steel surface based on multi-training and generative adversarial network," *Opt. Lasers Eng.*, vol. 122, pp. 294–302, Nov. 2019.

[12] J. Li, Z. Su, J. Geng, and Y. Yin, "Real-time detection of steel strip surface defects based on improved YOLO detection network," *IFAC-PapersOnLine*, vol. 51, no. 21, pp. 76–81, 2018.

[13] D. Racki, D. Tomazevic, and D. Skocaj, "A compact convolutional neural network for textured surface anomaly detection," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1331–1339.

[14] Y. Yan, S. Kaneko, and H. Asano, "Accumulated and aggregated shifting of intensity for defect detection on micro 3D textured surfaces," *Pattern Recognit.*, vol. 98, Feb. 2020, Art. no. 107057.

[15] X. Li, S. K. Tso, X.-P. Guan, and Q. Huang, "Improving automatic detection of defects in castings by applying wavelet technique," *IEEE Trans. Ind. Electron.*, vol. 53, no. 6, pp. 127–1934, Dec. 2006.

[16] M. Quintana, J. Torres, and J. M. Menendez, "A simplified computer vision system for road surface inspection and maintenance," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 3, pp. 608–619, Mar. 2016.

[17] L. Xu, X. He, X. Li, and M. Pan, "A machine-vision inspection system for conveying attitudes of columnar objects in packing processes," *Measurement*, vol. 87, pp. 255–273, Jun. 2016.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[20] J. Kittler, R. Marik, M. Mirmehdi, M. Petrou, and J. Song, "Detection of defects in color texture surfaces," in *Proc. IAPR Workshop Mach Vis Appl (MVA)*, 1994, pp. 558–567.

[21] A. Rebhi, S. Abid, and F. Fnaiech, "Fabric defect detection using local homogeneity and morphological image processing," in *Proc. Int. Image Process., Appl. Syst. (IPAS)*, 2016, pp. 1–5, doi: 10.1109/IPAS.2016.7880062.
[22] M. Niskanen, O. Silvn, and H. Kauppinen, "Color and texture-based wood inspection with non-supervised clustering," in *Proc. Scandin. Conf. Image Anal. (SCIA)*, 2001, pp. 336–342.
[23] F. Ade, N. Lins, and M. Unser, "Comparison of various filter sets for defect detection in textiles," in *Proc. 14th Int. Conf. Pattern Recognit. (ICPR)*, vol. 1, 2014, pp. 428–431.
[24] X. Xie, "A review of recent advances in surface defect detection using texture analysis techniques," *ELCVIA Electron. Lett. Comput. Vis. Image Anal.*, vol. 7, no. 3, pp. 1–25, Jun. 2008.
[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*. [Online]. Available: http://arxiv.org/abs/1512.03385
[28] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2016, *arXiv:1608.06993*. [Online]. Available: http://arxiv.org/abs/1608.06993
[29] J. Masci, U. Meier, D. Ciresan, J. Schmidhuber, and G. Fricout, "Steel defect classification with max-pooling convolutional neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2012, pp. 1–6, doi: 10.1109/IJCNN.2012.6252468.
[30] A. Ferreira and G. Giraldi, "Convolutional neural network approaches to granite tiles classification," *Expert Syst. Appl.*, vol. 84, pp. 1–11, Oct. 2017.
[31] J. Chen, Z. Liu, H. Wang, A. Nunez, and Z. Han, "Automatic defect detection of fasteners on the catenary support device using deep convolutional neural network," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 2, pp. 257–269, Feb. 2018.
[32] S. Martelli, L. Mazzei, C. Canali, P. Guardiani, S. Giunta, A. Ghiazza, I. Mondino, F. Cannella, V. Murino, and A. D. Bue, "Deep endoscope: Intelligent duct inspection for the avionic industry," *IEEE Trans Ind. Informat.*, vol. 14, no. 4, pp. 1701–1711, Apr. 2018.
[33] H. Lin, B. Li, X. Wang, Y. Shu, and S. Niu, "Automated defect inspection of LED chip using deep convolutional neural network," *J. Intell. Manuf.*, vol. 30, no. 6, pp. 2525–2534, Mar. 2018.
[34] Cognex. (2018). *VISIONPRO VIDI: Deep Learning-Based Software for Industrial Image Analysis*. [Online]. Available: https://www.cognex.com/products/machine-vision/vision-software/visionpro-vidi
[35] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018, *arXiv:1802.02611*. [Online]. Available: http://arxiv.org/abs/1802.02611
[36] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI*, 2015, pp. 234–241, doi: 10.1007/978-3-319-24574-4_28.

LIANG XU (Member, IEEE) received the master's degree in control theory and control engineering from the Lanzhou University of Technology, in 2002, and the Ph.D. degree in computer science and technology from the South China University of Technology, in 2007. He is currently an Associate Professor with the School of Automation, Guangdong University of Technology. His current research interests include machine vision, wireless sensor networks, and deep learning.

SHUAI LV received the B.Sc. degree in automation from the North China University of Water Resources and Electric Power, in 2017. He is currently pursuing the master's degree in control engineering with the School of Automation, Guangdong University of Technology. His research interests include computer vision, deep learning, and machine learning.

YONG DENG received the B.E. degree in electrical engineering and automation from the Hunan University of Technology, in 2018. He is currently pursuing the master's degree in control theory and control engineering with the School of Automation, Guangdong University of Technology. His research interests include machine learning, generative adversarial networks, and deep learning.

XIUXI LI received the master's and Ph.D. degrees in chemical engineering from the South China University of Technology, in 1998 and 2008, respectively. He is currently a Professor with the School of Chemical and Chemical Engineering, South China University of Technology. His current research interests include chemical process safety, product quality control, and process optimal.