IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 23, 2020, accepted February 16, 2020, date of publication February 19, 2020, date of current version February 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975030

Deep Regression Neural Network for Industrial Surface Defect Detection

ZHIQUAN HE [1,2,3], **(Member, IEEE), AND QIFAN LIU** [1]

[1] Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China
[2] Guangdong Multimedia Information Service Engineering Technology Research Center, Shenzhen University, Shenzhen 518060, China
[3] Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518060, China

Corresponding author: Zhiquan He (zhiquan@szu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61971290.

ABSTRACT Industrial product surface defect detection is very important to guarantee high product quality and production efficiency. In this work, we propose a regression and classification based framework for generic industrial defect detection. Specifically, the framework consists of four modules: deep regression based detection model, pixel-level false positive reduction, connected component analysis and deep network for defect type classification. To train the detection model, we propose a high performance deep network structure and an algorithm to generate label data to capture the defect severity information from data annotation. We have tested the method on two public benchmark datasets, AigleRN and DAGM2007, and an in-house capacitor image dataset. The results have shown that our method can achieve the state-of-the-art performance in terms of detection accuracy and efficiency.

INDEX TERMS Deep convolutional neural networks, regression, surface defect detection.

I. INTRODUCTION

Surface defect inspection, which examines the surface of products, is an important step in quality control to guarantee the product quality and manufacturing efficiency. Defective items must be detected in time and removed from the production line, otherwise they will seriously affect the subsequent assembly line and lead to a decrease in the level of overall quality [1]. Defect detection by eye-checking has several problems such as low sampling rate, low accuracy and low efficiency, high labor intensity, high sensitivity to human experience, body and emotional status. Automated defect detection based on machine vision is an undergoing trend as it can significantly overcome these disadvantages. Machine vision based defect detection systems capture product surface images and apply image analysis to detect the defects. Fig. 1 shows an example of steel surface image with several types of defects. Surface defect detection is to find whether an image contains defects or not. Ideally, defect detection needs provide the pixel level location of the defects as well as the defect types, which can be used to improve the manufacturing quality and decide the actions to take on the defective product in subsequent pipeline.

FIGURE 1. Seven classes of typical steel defect. from left to right: black spot, bruise, hole, black line, seams, inclusion, scratches. images are taken from [2].

Various image processing methods have been proposed to detect defects in images such as thresholding based [3], segmentation based [4] and edge detection based methods [5], [6] using operators such as Sobel [7], Prewitt [8], and Canny [9]. However, this kind of pure image processing methods only works for a few simple cases. In the past decades, feature based methods along with traditional machine learning are widely used to detect defects, in which image processing techniques are used to extract features. Traditional machine learning methods include neural network [10], support vector machine (SVM) [11] etc.. For example, Jeon *et al.* use Gabor filter and carries out edge-pair detection, and the histogram-based and gradient texture features are input to an SVM to detect the scratches on slab surfaces [12], [13]. This category of methods relies on hand-crafted features from domain knowledge and the designer's subjective experience.

Deep neural networks (DNNs) have gained significant success and achieved the state-of-the-art performance in image

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

analysis and recognition [14]–[18] as deep neural networks are able to construct complex representations and automatically learn a compositional relationship between inputs and outputs, mapping input images to output labels. In surface defect detection, deep learning based methods are the undergoing trend recently, which roughly fall into three categories, 1) image classification based, 2) object detection based, and 3) pixel segmentation based methods. In order to provide high precision defect detection for real application purpose, we adopt the pixel segmentation strategy as it can provide pixel-level defect location. Many proposed deep learning based segmentation methods try to design a complicate network structure aiming to improve the performance of detection power and defect classification [19], [20]. However, such a complicate network needs a lot of data to train, and is difficult to debug and upgrade, which is not suitable for real industrial applications. Therefore, we adopt a design in which the detection and defect classification are separate modules.

Another important aspect is that defect severity information is very useful and often required in industrial defect detection and most of the segmentation based methods label the input image pixels as defective or non-defective and use the output confidence score to indicate the defect severity. Ideally, the confidence score should be linearly correlated with the defect severity, i.e. significant defects should have high confidence. However, this confidence score given by most deep neural networks are often not well-calibrated [21]. We believe the annotated defects are good ground-truth for defect severity prediction. Based on these major considerations, in this work, we propose a generic regression based four-stage framework for surface defect detection in which 1) the first stage is a regression based pixel segmentation module to assign a score to each pixel to indicate the defect severity, 2) the second stage is a filtering module to reduce the number of falsely reported defective pixels, 3) the third stage is a connected component analysis module to form defects from detected defective pixels, and 4) the fourth stage is defect type classification. We also optimize the segmentation deep network structure by reducing the number of layers and filters to meet the balance between detection accuracy and speed. Overall, our work has the following major contributions.

- We formulate the defect detection problem as regression based pixel segmentation and defect classification, which can provide both pixel-level defect severity and defect types at defect level.
- We propose a four-stage defect detection framework which is more generic and flexible for end-user's needs.
- We evaluate the method on two benchmark datasets and one in-house real capacitor image dataset.

The rest of this paper is organized as follows. Section II reviews the related works on this topic. Section III introduces our four-stage framework for defect detection. Experimental results are presented in Section IV. Finally, Section V concludes this paper and discusses the future work.

II. RELATED WORK

Many deep convolutional network based methods have been proposed for defect detection. According to the way of handling the defect detection problem, deep learning based methods can roughly divided into three categories, pure image classification methods, object detection based methods and pixel-level segmentation methods.

Pure classification methods divide the input image into overlapping blocks, and classify the block images into different classes. If a block contains a certain number of defect pixels or more, the block is labeled as defective. The size of the block depends on the input size of the deep learning model, typically are 256, 128 or 32. For example, in [22], the authors use MatConvNet [23] to classify the input pavement 256 × 256 images into defective and non-defective. In [24], Li and Zhao modify the structure of GoogleNet [25] to do the classification. Similar methods include [26]–[29]. Binary classification is quite simple. Multi-class classification is also used to classify image blocks. For example, in [30]–[32], a single CNN model is designed to classify defect images of resistance welding spot, panel glasses, wafer images respectively.

Defect detection can also be treated as object detection problem, which is an important task in computer vision. Its goal is to locate the object with a bounding box and decide the object type. Many deep CNN models have also been proposed to improve the accuracy and efficiency, such as Faster R-CNN [33], SSD [34], YOLO [35]. In [36], Suh and Cha use Faster R-CNN to detect the damages in civil infrastructure. Young-Jin *et al*. modify the Faster R-CNN using ZF-net [37] to speedup the feature extraction in [38]. Similarly, in [39], the authors modify Faster R-CNN for railway subgrade defect detection. Yiting *et al*. adopt a MobileNet-SSD framework in [40] to detect defects like breaches, dents, burrs and abrasions on the sealing surface of containers.

Defect detection based on pixel-level segmentation is more generic than the above pure classification based and object detection based method as it can provide pixel level location for defects as well as the defect types. Auto-encoder [41] and FCN (fully convolutional network) [42] are widely used for this purpose. In [19], Tao *et al*. propose a two-stage algorithm in which the first stage is using two cascaded auto-encoders to segment the defects, and the second stage is a CNN classifier to classify the defect regions cropped from the image using the region information provided by the first stage. Similarly, Qin *et al*. propose DeepCrack which uses an encoder-decoder architecture to segment pavement image into crack and background [43]. Zhiyang *et al*. combine a segmentation stage and a detection stage using two separate fully convolutional networks (FCN) [20]. The work of [1] also uses a FCN to segment the input image pixels.

As pointed out previously, these methods try to design a complicate structure to realize both defect detection and classification, and the defect severity problem is not well handled. In this work, we propose a four-stage regression based deep learning method for defect detection, which is

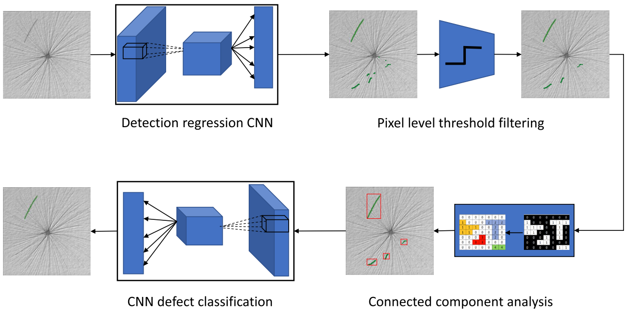
Detection regression CNN
Pixel level threshold filtering
CNN defect classification
Connected component analysis

FIGURE 2. Diagram of overall method. the first stage is a regression model to predict the defect severity using the annotated defect pixels as the training data. the second stage filters out some defect pixels to reduce the false positives. the third stage forms the defects from the pixels by connected component analysis and the last step is defect type classification.

easy to train and upgrade, and able to provide accurate defect severity information.

III. METHOD

A. REGRESSOIN BASED FOUR-STAGE DEFECT DETECTION

Fig. 2 shows the four major steps in the method.

1) DEFECT SEVERITY PREDICTION BY REGRESSION

In order to predict the defect severity at each pixel, the first step is a regression model using a deep convolutional neural network (CNN). The image block is divided into non-overlapping sub-blocks with size of 4 × 4 or 2 × 2. Each block is mapped to a number at the output within range of [0, 12]. The output size depends on the detection resolution. If the resolution is 4 × 4, which means a 4 × 4 input block is mapped to one output number. Suppose input image patch size is 256 × 256, the output size is $64 \times 64 = 4096$. Correspondingly, if the resolution is 2 × 2, then the output size will be $128 \times 128 = 16384$. And further if the resolution is 1 × 1, the output size will be 256 × 256. This will make the computation very intensive.

2) FALSE POSITIVE REDUCTION BY PIXEL FILTERING

The output of the first module might contains falsely reported defect pixels (false positives). The purpose of this step is to filter out those false positives. We observe that pixels with lower values are more likely to be false positives. Therefore, we simply adopt a step function, which means only pixels with values greater than the threshold are kept, i.e.

$$f(I_i) = \begin{cases} I_i & I_i >= T \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where I_i is the score of pixel i from the first module and T is a tuneable parameter. Rightly after the detection module is trained, the value of T can be determined by maximizing the mean IOU between the prediction and the ground truth.

3) FORMING DEFECTS BY CONNECTED COMPONENT ANALYSIS

In our method, we choose the decide the defect type by using a classification module on the whole defect. After filtering the false positive pixels, connected component analysis is applied to cluster the pixels into defect. The minimum distance between clusters is set to 4, which means if the distance between two clusters are less than 4 pixels, they will be joined into one.

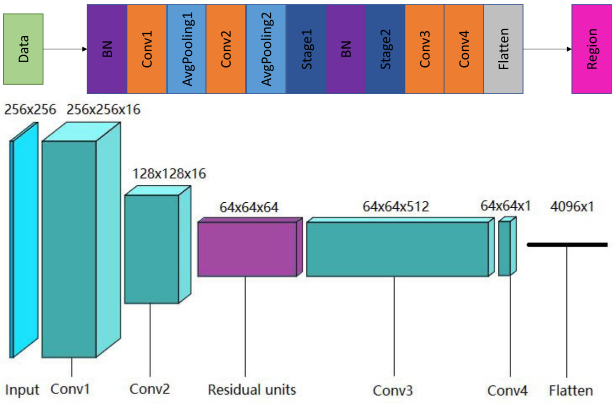

FIGURE 3. Network structure of detection model. in the upper figure, bn is batch normalization module, convs are convolution layers, avgpoolings are average pooling layer and flatten is the flatten layer and the region in the figure is a output label. the lower figure shows the feature maps at different layers, in which the purple residual units correspond to the stage1 and stage 2 as a whole.

TABLE 1. Model configuration. the numbers are: kernel size / stride, number of filters.

Conv1	Avg pooling1	Conv2	Avg pooling2	Conv3	Conv4
5x5/1, 16	3x3/2	3x3/1, 16	3x3/2	3x3/1, 512	1x1/1, 1

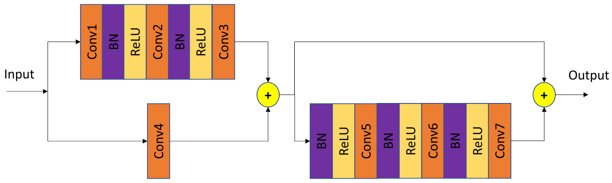

FIGURE 4. Structure of the residual unit. BN is batch normalization, ReLU is the ReLU activation function.

4) DEFECT TYPE CLASSIFICATION

Defect classification is applied using a deep CNN model to predict the type of the defects. Suppose the data contains N types of defects, the model will output $N+1$ classes, in which the extra class is the normal class or background. From this perspective, the classification module serves two purposes: decide the defect type and reduce the false positives at defect level. The loss function we use is cross entropy.

B. NETWORK STRUCTURE

1) DETECTION MODEL

The detection model is modified from Resnet18 [44] for regression purpose. The overall network structure is shown in Fig. 3. The structure mainly consists of convolution layer, average pooling layers and two similar residual units, followed by two convolutional layers and a flatten layer before the output. The flatten layer has 4096 number of elements, if the detection size is set to 4 × 4. The output layer is a linear regression which approximates the label by minimizing the L2 loss. Table 1 shows the detailed model configuration. Fig. 4 shows the structure of a residual unit, which mainly

TABLE 2. Stage 1 configuration.

conv1	conv2	conv3	conv4	conv5	conv6	conv7
1x1/1, 8	5x5/1, 8	5x5/1, 32	1x1/1, 32	1x1/1, 8	5x5/1, 8	5x5/1, 32

TABLE 3. Stage 2 configuration.

conv1	conv2	conv3	conv4	conv5	conv6	conv7
1x1/1, 16	5x5/1, 16	5x5/1, 64	1x1/1, 64	1x1/1, 16	5x5/1, 16	5x5/1, 64

consists of BN (batch normalization), convolution and ReLU layers. Table 2 and 3 shows the parameter configuration of these two stage units.

2) CLASSIFICATION MODEL

The defect type classification model is simply the Resnet101 [44]. Considering the fact that in practical defect detection, the defects are less likely to occur, the computation burden of this part is much less than the first module. Therefore, the module can be replaced with even more complicated structures, such as Resnet152 to ensure high classification accuracy.

C. END-TO-END MODEL TRAINING

1) TRAINING DATA GENERATION

In the method, the two CNN models need to trained. The first one is a regression model, and the second one is a common multi-class classification model. Before the training process, the defects in training images need to be annotated as accurate at pixel level as possible. If the data contains multiple types of defects, the defects are also marked using different colors.

a: DEFECT SEVERITY DATA FOR REGRESSION MODEL

In order to predict the defect severity information to end-users, our method chooses to explicitly generate defect severity data from the annotation, unlike the most of existing methods which rely the confidence score of the CNN model. We cut the input image into overlapping patches. For each image patch with significant area of defect, we analyze the image intensities of the defect pixels and the neighboring pixels around the defect. High contrast between the defect and its neighbors means more severe defect. The following algorithm is used to generate the defect severity data.

1) Divide the defect area into non-overlapping cells. If the detection resolution is 4 × 4, the cell is 4 × 4.
2) Inside each defect cell, calculate the minimum (m_D) and maximum (M_D) of pixel intensities. Do the same thing in each non-defect cell at the defect border, denoted by m_N and M_N respectively.
3) For each row of the defect cells, select the leftmost and rightmost non-defect cells as the reference, then for each defect cell, calculate
$$d_0 = min\{|m_D - m_N|, |M_D - M_N|\}$$
$$d_1 = |M_D - m_N|$$
Therefore, the score for this cell is $s = d_0 + d_1$.
4) Truncate all the scores of cells by a threshold λ, i.e.,
$$s = s + \lambda - max(s, \lambda) \tag{2}$$
where λ is a parameter.
5) Normalize all the cell scores to range of [0, 12]

2) TRAINING DATA FOR CLASSIFICATION MODEL

The classification model is trained after the detection model is finished. This model servers to distinguish among different defect types, as well as between defect and non-defect. For this purpose, the training data contains not only the marked defects, but also the false positives given by the detection model. As shown in Fig. 2, the image patch after connected component analysis contains 4 defects, 3 are false positives. All these 4 defects will be used to train the classification model.

It should be noted that the defects have various shapes and sizes. The most straightforward way is to resize the defect image patch to the size of the classification model. However, this will greatly deform the defect and misguide the model. In our method, we use a more systematic way to generate the data. We use a sliding window of a certain size to cut the image into overlapping blocks. And the blocks with significant area of defect are used to train the model. If the window is not large enough to hold the defect, we double the window size and then down-size the patches to the model input size. With this method, suppose defect d_i is cut into a list of blocks $(d_{i1}, d_{i2}, \ldots, d_{in})$, the corresponding class labels given by the model will be $(c_{i1}, c_{i2}, \ldots, c_{in})$. The overall defect class c_i is selected by a simple voting strategy which selects the most frequent label.

3) DATA AUGMENTATION

Defect detection in real industrial applications must consider various factors such as the illumination change, the motion of the objects. Data augmentation plays an important role to increase the detection stability and repeatability, which is often required by the detection devices. we adopt a list of such techniques, including image flipping, rotation with random angles, random noise, illumination and contrast change, to increase the data diversity. This will also help reduce the amount of the data labeling, which might be expensive and laborsome. Especially, to increase the prediction stability against the shift variations, we cut a number of blocks from the image around the defect so that the defect will appear at different places in the image patch. This will help reduce the prediction difference when the same product is tested on the device a number of times.

IV. EXPERIMENTS AND RESULTS

A. IMPLEMENTATION DETAILS

To generate the data to train the detection model, we cut the input images into patches of 256 × 256 using a sliding

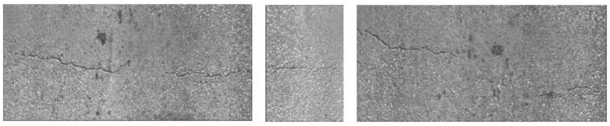

FIGURE 5. Sample defects from AigleRN dataset.

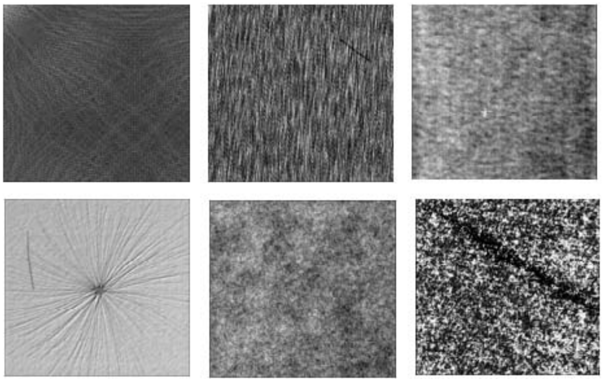

FIGURE 6. Sample defects from DAGM2007 dataset.

window from left to right, top to bottom with a overlap step of 64. The λ parameter in Eqn. (2) is 500. The input size of the classification model is 64 × 64. To speed up the training process, the model is previously pre-trained on ImageNet [45] and fine-tuned using the training data.

B. DATASETS

To compare the proposed method with existing works, we use three datasets, namely, two public datasets (AigleRN and DAGM 2007) and one in-house dataset (CapacitorDB) which contains capacitor images for defect detections. Each dataset will be divided into two parts, one for training and the rest for testing. All defect pixels are annotated using different colors to represent the corresponding type.

AigleRN dataset contains 38 pre-processed grayscale images on French road surface, including 991 × 462 and 331 × 462. In this test, we select 24 of them as the training set and 14 as the test set. Fig. 5 shows three example images and defects.

DAGM2007 dataset consists of 10 subsets, each corresponding to a class of defects. Each class has 1000 defect free images and 150 defective images with one defect on texture backgrounds. The image size is 512 × 512. We follow the work of [46] to select the first six classes to conduct the experiment. So, a total of 900 defective images are selected, where 720 images for training and the rest 180 for testing. Fig. 6 shows some example images.

CapacitorDB contains more than 3,200 training images and 639 test images with 673 defects. The dataset consists of 8 types of defects, such as scratch, bubble, damage, and broken covering resin. Fig. 7 shows two sample images.

C. PERFORMANCE METRICS FOR DEFECT DETECTION

Our proposed method provide pixel-level defect location as well as defect type classification at the defect level or object level. Defect detection can be evaluated at pixel level. Predicted pixels that are truly defective are true positives (TP), otherwise false positives (FP). False negatives (FN) are those marked defect pixels that are missed by the detector. Therefore, precision, recall and F1 are defined as follows

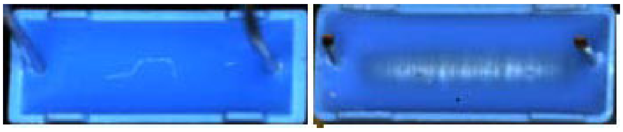

FIGURE 7. Samples defects from capacitor dataset.

$$\begin{aligned} Precision &= \frac{TP}{TP+FP} \\ Recall &= \frac{TP}{TP+FN} \\ F1 &= 2 * \frac{Precison * Recall}{Precison + Recall} \end{aligned}$$

The IOU (intersection over the union) value between the ground-truth defect and predicted one can also be calculated by counting the pixels.

Defect detection can also be evaluated at defect level or object level in terms of object detection accuracy. In this case, a defect is detected if the IOU between the ground truth and the prediction greater than a certain threshold. A true positive is a true defect that is detected with the correct type. In this setting, recall, precision can be accordingly calculated.

It should be noted that the AR column in Table 5 and 7 is defined by Eqn. (3) [47].

$$\begin{aligned} AR &= 2\int_{0.5}^{1} recall(o)\mathrm{d}o \\ &= \frac{2}{n}\sum_{i=1}^{n}\max\left(IOU\left(\mathrm{gt}_i\right) - 0.5, 0\right) \end{aligned} \tag{3}$$

where n is the number of defects.

D. RESULTS ON AIGLERN DATASET

Table 4 shows the comparison of pixel-level segmentation accuracy on AigleRN dataset in terms of precision, recall and F1 score. All the cracks in the images have the same class. Therefore, the classification module is used to distinguish normal and defective pixels. We can see that our method achieves the best performance with precision of 0.9313, recall of 0.9440 and F1 score of 0.9375. Fig. 8 shows the four test examples.

E. RESULTS ON DAGM2007 DATASET

Table 5 shows the segmentation performance on DAGM2007 dataset. In this test, we also show the precision, recall and F1 score, which is not provided from the literature. As we can see that our proposed method has the highest AR and Mean IOU scores. The AR score is 0.69, which is slightly better than that of Qiu *et al.* [46], and greatly higher than the

TABLE 4. Defect detection accuracy on AigleRN dataset.

Method	Precision	Recall	F1
CrackIT [48]	0.7684	0.7432	0.7656
MPS [49]	0.8666	0.9006	0.8833
CrackForest [50]	0.9028	0.8658	0.8839
U-Net [51]	0.9202	0.9321	0.9261
Ours	**0.9313**	**0.9440**	**0.9375**

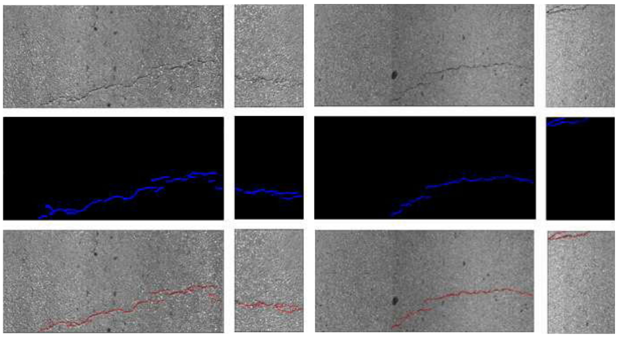

FIGURE 8. Test examples on AigleRN. from top to bottom are the original image, annotation and prediction respectively.

TABLE 5. Comparison of defect detection accuracy on DAGM2007.

Method	AR	MeanIOU	Presion	Recall	F1
FCN [42]	0.6062	0.6654	-	-	-
ViDi [52]	0.5604	0.6199	-	-	-
Qiu [46]	0.6843	0.7326	-	-	-
Ours	**0.6902**	**0.8450**	**0.925**	**0.908**	**0.915**

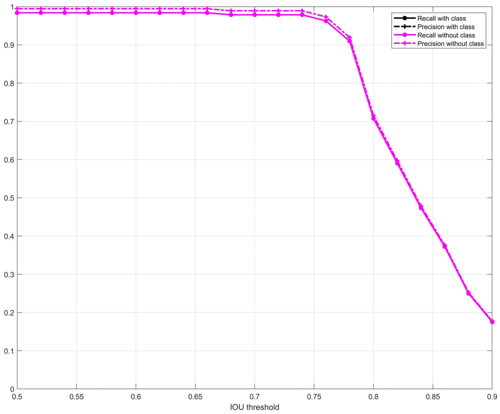

FIGURE 9. Recall and precision with respect to iou threshold on dagm2007 dataset. the curves without considering class prediction indicate the performance of detection module. the black curves are overlapped by the cyan ones.

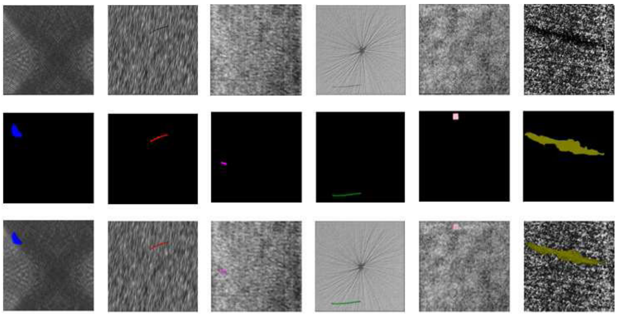

FIGURE 10. Test examples on DAGM2007. from top to bottom are the original, ground truth and prediction respectively.

TABLE 6. Defect classification performance on DAGM2007.

Ground Truth \Test	C1	C2	C3	C4	C5	C6
class1 (C1)	1	0	0	0	0	0
class2 (C2)	0	1	0	0	0	0
class3 (C3)	0	0	1	0	0	0
class4 (C4)	0	0	0	1	0	0
class5 (C5)	0	0	0	0	1	0
class6 (C6)	0	0	0	0	0	1

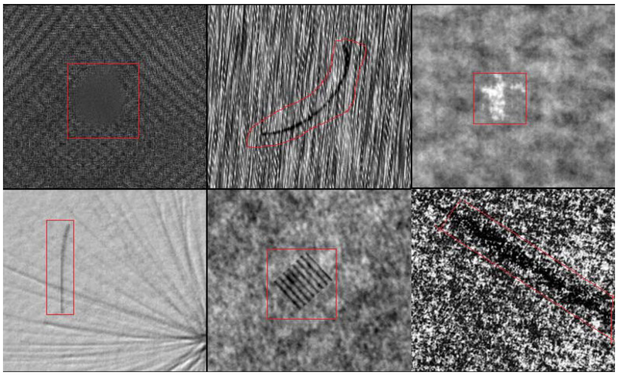

FIGURE 11. Six types of defects in DAGM2007. the defect types are easy to be classified.

performance of ViDi [52] and FCN [42]. Similarly, in terms of Mean IOU, ours is 0.8450, better than all other methods with a big margin.

The above result does not consider the correctness of defect type prediction. Fig. 9 shows the average recall and precision with respect to different IOU threshold on DAGM2007 dataset. A detected defect is true positive only if the IOU is greater than the threshold and its type prediction is correct. In the figure, the difference between recall/precision with and without class is whether the class prediction is included in the calculation. From the curves we can see that the black and cyan curves fully overlap, which means the class prediction is 100% correct. When the IOU threshold is set to 0.5, the recall and precision is around 0.98 and 0.99 respectively, which means the defect-level accuracy is very high. Fig. 10 shows several test examples in this dataset.

1) PERFORMANCE OF DEFECT CLASSIFICATION

Table 6 shows the defect classification performance in terms of confusion matrix. We can see that the type of all the defects in the DAGM2007 dataset are correctly predicted.

This is not surprising as the difference between different types of defects in this dataset are quite significant. Fig. 11 shows an comparison of different types of defects.

On the other hand, in the method, we use the classifier to reduce the false positives by distinguishing between normal

TABLE 7. Defect detection accuracy on DAGM2007.

Method	AR	MeanIOU	Presion	Recall	F1
FCN [42]	0.6062	0.6654	-	-	-
ViDi [52]	0.5604	0.6199	-	-	-
Qiu [46]	0.6843	0.7326	-	-	-
Ours (full)	0.6902	0.8450	0.925	0.908	0.915
Ours (no classifier)	**0.6366**	**0.8172**	**0.891**	**0.908**	**0.897**

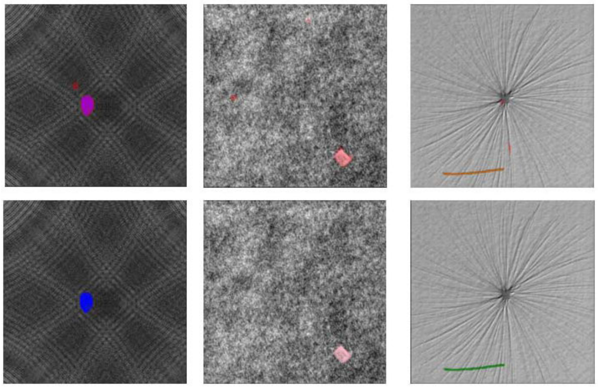

FIGURE 12. Effects of reducing false positives using a classifier on the DAGM2007 dataset. First row: prediction without classification. Second row: prediction with classification.

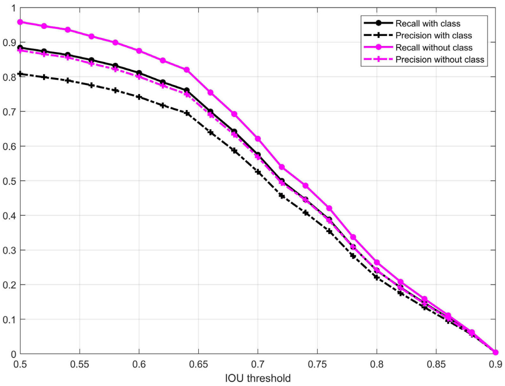

FIGURE 13. Recall and precision with respect to IOU threshold on capacitor dataset. the curves without considering class prediction indicate the performance of detection module.

image patches and defective ones. So we show the segmentation performance again in Table 7 when the classifier is turned off. From the table, we can see that when the classifier is off, AR, MeanIOU and Precision drops a lot, which means the classification module effectively reduces the number of false positive pixels. On the other hand, the recall score does not change (0.908), which means the classifier does not mistakenly classify the defect pixels to non-defective pixels. Fig. 12 gives an illustration, where the classification module in the second row removes some false positive predictions.

F. RESULTS ON CAPACITOR DATASET

We also test the proposed method on capacitor image dataset from real industrial defect detection. Fig. 13 shows the result. When the IOU threshold is set to 0.5, the recall and precision with classification is 0.88 and 0.80 respectively and recall and precision without classification is 0.96 and 0.87 respectively. The defect classification significantly influences the recall and precision. The overall defect type classification accuracy is 0.92. We look into the classification result and find that the accuracy of class 8 is only 0.57 out of 28 cases, in which 16 are correctly predicted, 5 are predicted as class 4, 4 as class 7 and 3 as class 2. We look into the datasets and find that some defects of different types look very similar. Fig. 14 shows three examples that are very similar to a defect of class 8, which will confuse the classifier to make a wrong prediction.

Fig. 15 shows two test examples, each with three images of the original one, annotation and prediction.

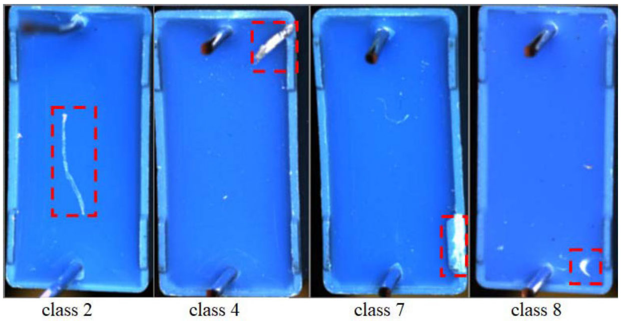

FIGURE 14. Similar defect shapes from different classes. from left to right, the classes are class 2,4,7, and 8 respectively.

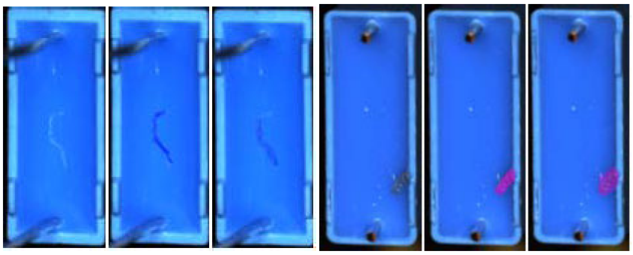

FIGURE 15. Two test examples from capacitor dataset. in each group, the leftmost is the original image, the middle one the annotation and the right one is the prediction result.

G. THRESHOLD PARAMETER DETERMINATION

In our method, the threshold parameter T in Eqn. 1 needs to be tuned to optimize the output of the regression model. The threshold value is set to 5.4 for AlgleRN dataset by searching the best value by maximizing the mean IOU. Fig. 16 shows the result. Similarly, Fig. 17 is for dataset DAGM2007, in which the corresponding threshold value is 4.1.

H. TIME PERFORMANCE OF DEEP LEARNING MODELS

For real industrial defect detection, the time performance is very critical for production efficiency. In our method, the regression detection model is the most time consuming part as it has to predict a score for each pixel in the input image, and the classification model only needs to work on the

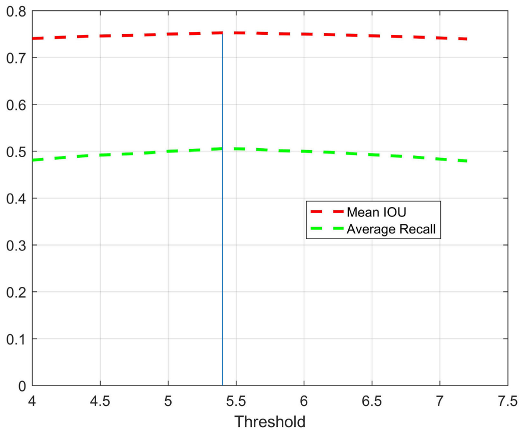

FIGURE 16. Searching for the best value of the filter threshold on AigleRN dataset.

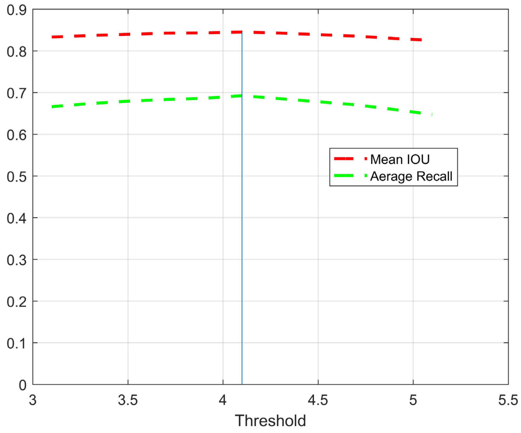

FIGURE 17. Searching for the best value of the filter threshold on DAGM2007.

TABLE 8. Time performance (ms) of the detection model. the second column, Gray, 4 × 4 means input is one channel gray images and the output resolution is 4 × 4. Column 3 and 4 are similar.

Batch size	Gray, 4 × 4	Color, 4 × 4	Color, 2 × 2
8	8	11	23
16	14	20	45
32	26	42	84
64	50	80	164

defect areas. Table 8 shows the time in unit of millisecond with respect to batch size under different model configurations. For example, when the input is color images and the output resolution is 4 × 4, one image block (256 × 256) takes less than 1.5ms and when the output resolution is 2 × 2, an image block takes around 2.5ms. The test configuration is: Windows 10, x64, CPU i7-8700K@3.70GHz, Memory 24GB and GPU is GTX 1080Ti 11GB. With this, we can finish testing a smart phone cover glass within 2.5 seconds, where there are 7 images with size 8192 × 12800 for each glass.

V. CONCLUSION

In this paper, we propose a generic framework for industrial defect detection based on CNN, which consists of four major stages. The first one is a regression based detection model which can predict the defect severity information, in stead of using the model confidence score. The second one is pixel filtering module aiming to reduce the false positives at pixel level. The third one is connect component analysis to form defects from pixels. And the final module is a classification model which serves to classify the defect types and reduce the false positives again at defect level. We propose an algorithm to generate defect severity information from annotation data to train the regression model. The test results on three datasets have shown that our method can not only achieve high detection accuracy, but also high speed performance. In our future work, we will plan to further improve the network structure to achieve even more accurate defect detection, faster test speed, and higher prediction stability against input corruptions or variations.

REFERENCES

[1] Z. Xiao, Y. Leng, L. Geng, and J. Xi, “Defect detection and classification of galvanized stamping parts based on fully convolution neural network,” *Proc. SPIE*, vol. 10615, Apr. 2018, p. 106150K.

[2] L. Yi, G. Li, and M. Jiang, “An End-to-End steel strip surface defects recognition system based on convolutional neural networks,” *Steel Res. Int.*, vol. 88, no. 2, Apr. 2016, Art. no. 1600068.

[3] H.-F. Ng, “Automatic thresholding for defect detection,” *Pattern Recognit. Lett.*, vol. 27, no. 14, pp. 1644–1649, Oct. 2006.

[4] H. Oliveira and P. L. Correia, “Automatic road crack segmentation using entropy and image dynamic thresholding,” in *Proc. 17th Eur. Signal Process. Conf.*, Aug. 2009, pp. 622–626.

[5] C. Jia and Y. Wang, “Edge detection of crack defect based on wavelet multi-scale multiplication,” *Comput. Eng. Appl.*, vol. 47, no. 15, pp. 219–221, 2011.

[6] X. Yang, D. Qi, and X. Li, “Multi-scale edge detection of wood defect images based on the dyadic wavelet transform,” in *Proc. Int. Conf. Mach. Vis. Hum.-Mach. Interface*, Apr. 2010, pp. 120–123.

[7] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, “Design of an image edge detection filter using the sobel operator,” *IEEE J. Solid-State Circuits*, vol. 23, no. 2, pp. 358–367, Apr. 1988.

[8] W. Dong and Z. Shisheng, “Color image recognition method based on the prewitt operator,” in *Proc. Int. Conf. Comput. Sci. Softw. Eng.*, vol. 6, Dec. 2008, pp. 170–173.

[9] L. Er-sen, Z. Shu-long, Z. Bao-shan, Z. Yong, X. Chao-gui, and S. Li-hua, “An adaptive edge-detection method based on the canny operator,” in *Proc. Int. Conf. Environ. Sci. Inf. Appl. Technol.*, vol. 1, Jul. 2009, pp. 465–469.

[10] R. Hecht-Nielsen, “Theory of the backpropagation neural network,” *Neural Netw.*, vol. 1, no. 1, p. 445, Jan. 1988.

[11] A. Ukil, “Support vector machine,” *Comput. Sci.*, vol. 1, no. 4, pp. 1–28, 2002.

[12] Y. J. Jeon, D. C. Choi, J. P. Yun, and W. K. Sang, “Detection of scratch defects on slab surface,” in *Proc. 11th Int. Conf. Control, Automat. Syst.*, Oct. 2011, vol. 86, no. 1, pp. 1274–1278.

[13] N. Neogi, D. K. Mohanta, and P. K. Dutta, “Review of vision-based steel surface inspection systems,” *EURASIP J. Image Video Process.*, vol. 2014, no. 1, p. 50, Nov. 2014.

[14] H. Xing, G. Zhang, and M. Shang, “Deep learning,” *Int. J. Semantic Comput.*, vol. 10, no. 03, pp. 417–439, 2016.

[15] W. Cao, Q. Lin, Z. He, and Z. He, “Hybrid representation learning for cross-modal retrieval,” *Neurocomputing*, vol. 345, pp. 45–57, Jun. 2019.

[16] W. Cao, J. Yuan, Z. He, Z. Zhang, and Z. He, “Fast deep neural networks with knowledge guided training and predicted regions of interests for real-time video object detection,” *IEEE Access*, vol. 6, pp. 8990–8999, 2018.

[17] D. Meng, L. Zhang, G. Cao, W. Cao, G. Zhang, and B. Hu, “Liver fibrosis classification based on transfer learning and FCNet for ultrasound images,” *IEEE Access*, vol. 5, pp. 5804–5810, 2017.

[18] D. Meng, G. Cao, Y. Duan, M. Zhu, L. Tu, D. Xu, and J. Xu, "Tongue images classification based on constrained high dispersal network," *Evidence-Based Complementary Alternative Med.*, vol. 2017, Mar. 2017, Art. no. 7452427.

[19] X. Tao, D. Zhang, W. Ma, X. Liu, and D. Xu, "Automatic metallic surface defect detection and recognition with convolutional neural networks," *Appl. Sci.*, vol. 8, no. 9, p. 1575, Sep. 2018.

[20] Z. Yu, X. Wu, and X. Gu, "Fully convolutional networks for surface defect inspection in industrial environment," in *Proc. Int. Conf. Comput. Vis. Syst.* Cham, Switzerland: Springer, 2017, pp. 417–426.

[21] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1321–1330.

[22] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, Mar. 2017.

[23] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for MATLAB," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 689–692.

[24] S. Li and X. Zhao, "Convolutional neural networks-based crack detection for real concrete surface," in *Sensors Smart Struct. technol. for Civil, Mech., Aerosp. Syst. 2018*, vol. 10598, Mar. 2018, Art. no. 105983V.

[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[26] L. Shang, Q. Yang, J. Wang, S. Li, and W. Lei, "Detection of rail surface defects based on CNN image recognition and classification," in *Proc. 20th Int. Conf. Adv. Commun. Technol.(ICACT)*, Feb. 2018, pp. 45–51.

[27] L. Pauly, H. Peel, S. Luo, D. Hogg, and R. Fuentes, "Deeper networks for pavement crack detection," in *Proc. 34th Int. Symp. Autom. Robot. Construct. (ISARC)*, Jul. 2017, pp. 479–485.

[28] D. Denhof, B. Staar, M. Lütjen, and M. Freitag, "Automatic optical surface inspection of wind turbine rotor blades using convolutional neural networks," *Procedia CIRP*, vol. 81, pp. 1166–1170, Jan. 2019.

[29] U. Markowska-Kaczmar and M. Skiba, "Is convolutional network competitive for vision method in the furniture dowel quality control?" in *Proc. Innov. Intell. Syst. Appl. (INISTA)*, Jul. 2018, pp. 1–8.

[30] Z. Guo, S. Ye, Y. Wang, and C. Lin, "Resistance welding spot defect detection with convolutional neural networks," in *Proc. Int. Conf. Comput. Vis. Syst* Cham, Switzerland: Springer, 2017, pp. 169–174.

[31] R. Ye, C.-S. Pan, M. Chang, and Q. Yu, "Intelligent defect classification system based on deep learning," *Adv. Mech. Eng.*, vol. 10, no. 3, Mar. 2018, Art. no. 168781401876668.

[32] T. Nakazawa and D. V. Kulkarni, "Wafer map defect pattern classification and image retrieval using convolutional neural network," *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 2, pp. 309–314, May 2018.

[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[34] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[36] G. Suh and Y.-J. Cha, "Deep faster R-CNN-based automated detection and localization of multiple types of damage," *Proc. SPIE*, vol. 10598, Mar. 2018, Art. no. 105980T.

[37] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[38] Y.-J. Cha, W. Choi, G. Suh, S. Mahmoudkhani, and O. Büyüköztürk, "Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 33, no. 9, pp. 731–747, Nov. 2017.

[39] X. Xu, Y. Lei, and F. Yang, "Railway subgrade defect automatic recognition method based on improved faster R-CNN," *Sci. Program.*, vol. 2018, pp. 1–12, Jun. 2018.

[40] Y. Li, H. Huang, Q. Xie, L. Yao, and Q. Chen, "Research on a surface defect detection algorithm based on MobileNet-SSD," *Appl. Sci.*, vol. 8, no. 9, p. 1678, Sep. 2018.

[41] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.

[42] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.

[43] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "DeepCrack: Learning hierarchical convolutional features for crack detection," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1498–1512, Mar. 2019.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[45] 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.

[46] L. Qiu, X. Wu, and Z. Yu, "A high-efficiency fully convolutional networks for pixel-wise surface defect detection," *IEEE Access*, vol. 7, pp. 15884–15893, 2019.

[47] J. Hosang, R. Benenson, P. Dollar, and B. Schiele, "What makes for effective detection proposals?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 814–830, Apr. 2016.

[48] H. Oliveira and P. L. Correia, "CrackIT—An image processing toolbox for crack detection and characterization," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 798–802.

[49] R. Amhaz, S. Chambon, J. Idier, and V. Baltazart, "Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2718–2729, Oct. 2016.

[50] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3434–3445, Dec. 2016.

[51] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2015, pp. 234–241.

[52] Cognex. *Vidi of Cognex (2017).* Accessed: Sep. 2017. [Online]. Available: http://www.cognex.cn/productstemplate.aspx?id=19178&langtype=2052

ZHIQUAN HE (Member, IEEE) received the M.S. degree from the Institute of Electronics, Chinese Academy of Sciences, in 2001, and the Ph.D. degree from the Department of Computer Science, University of Missouri, Columbia, in 2014. He is currently working as an Assistant Professor with the College of Information Engineering, Shenzhen University, China. His research areas are image processing, computer vision, and machine learning.

QIFAN LIU is currently pursuing the M.Eng. degree in communication and information engineering with Shenzhen University, Shenzhen, China. His research interests are image processing and machine learning.