2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)

# DefectGAN: Weakly-Supervised Defect Detection using Generative Adversarial Network

Shuanlong Niu*, Hui Lin*, Tongzhi Niu, Bin Li† and Xinggang Wang

*Abstract*— **Traditional methods for defect detection applied in industry are complex, time-consuming, not robust and demanding for professional experience due to hand-crafted features extraction and pipeline design. Besides, current deep learning based methods for general object segmentation demand for a large number of region-level human annotations.**

**Instead, we present DefectGAN for defect detection in a weakly-supervised learning, which requires very a few human annotations. In practical application, images in training dataset are merely labeled with two categories: negative and positive. Despite being trained on image-level rather than region-level labels, DefectGAN has remarkable ability of localizing defect regions.**

**DefectGAN can have comparable and visually even better performance than SegNet, a supervised learning method on dataset CCSD-NL and DAGM 2007. The detected regions are more similar to the original defect regions visually and it has the potential of detecting unseen defects.**

## I. INTRODUCTION

Defect detection plays an essential role in product quality control during industrial manufacturing process. Not only can defect detection process ensure final products with high quality, but also can the detection process guide great improvement to production process. Current state-of-art methods for defect detection using machine vision make possible automated, precise and real-time defect detection, which are widely applied in kinds of products including chips [1] [2], steel [4] and fabrics [6]. We later revisit those methods before and in deep learning era.

### A. Traditional Methods before Deep Learning Era

Before deep learning era, traditional methods [3] [4] [5] [6] [7] [8] pay much attention to hand-crafted features extraction and pipeline design, which is not robust and effective. Traditional methods usually have several pipelines, including image pre-processing, object segmentation, feature extraction, defect recognition and classification. Parameters and thresholds during detection process are respectively optimized according to different types of defects. For instance, Zhang et al. [6] propose a combined detection system using Gabor filters to enhance image contrast and Pulse Coupled Neural Network (PCNN) to segment enhanced defect regions. However, with respect to error propagation through complex pipelines, this method can only detect simple defects of warp-knitted fabrics, not applicable to complex defects.

### B. Methods in Deep Learning Era

In deep learning era, convolution neural network (CNN) achieves remarkable performance on object detection and semantic segmentation tasks with some state-of-art models, such as SSD [9], YOLO [10], FCN [11] and U-Net [12]. Researches [14] [13] show that CNN-based methods can extract defect features automatically by training on dataset containing thousands of region-level labeled images. However, most of them suffer from serious pressure of image labeling. In industrial application, labelling images in region or pixel level is much expensive and collecting defect samples is usually difficult due to rather small quantity. To illustrate, inspired by SSD [9], MobileNet-SSD proposed by Li et al. [14] can achieve real-time detection of typical defects like burrs and abrasions, only 0.12s per image but such method is costly in human labor because of dataset preparation.

In order to save from serious pressure of region-level human annotations, some methods based on weakly-supervised learning are proposed. Some employ CAM [23], some use GAN [15] or its variants [16] [17] [18]. For instance, Lin et al. [1] proposed LEDNet combined with CAM, which learns from images labeled with four categories including three types of defects and positive chip. However, complex defect detection has not been validated. Zhao et al. [19] combine GAN and autoencoder to repair defect regions in test images and use LBP [20] to localize defect regions by comparing rebuilt images and original ones. This method only applies positive samples collected, avoiding difficult negative sample collection in industrial site. However, GAN is trained with several image pairs, so that negative images for each positive sample should be generated artificially. Then the generated image and the corresponding positive sample are labeled as a pair. It is difficult and time-consuming for generation algorithms to generate various defects as real as possible, especially to generate surface defects on commutator cylinder studied later.

We proposed another weakly-supervised method named DefectGAN, which learns from collected images merely labeled into two classes, negative and positive images. We

*Shuanlong Niu and Hui Lin contributed equally to this work.

†Bin Li is the corresponding author,a faculty of the State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, HUST, Wuhan 430074, China li_bin_hust@163.com

Shuanlong Niu, Hui Lin, Tongzhi Niu are with the State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, HUST, Wuhan 430074, China slniu@hust.edu.cn, hui_lin@hust.edu.cn, tzniu@hust.edu.cn

Xinggang Wang is a faculty of Artificial Intelligence Institute, School of Electronic Information and Communications, HUST, Wuhan 430074, China xgwang@hust.edu.cn

978-1-7281-0356-3/19/$31.00 ©2019 IEEE

avoid trying to generate paired images. Compared to all aforementioned methods, our method has several advantages listed as follows in practical application:

- **Able to detect defect regions though by learning from image-level rather than region-level annotations.** DefectGAN learns to translate an image from domain *NG* (negative) to domain *P* (positive) by training images labeled with these two classes. Defect regions can be localized by simply comparing negative and positive images generated by DefectGAN.
- **Comparable and possibly better performance over supervised learning methods.** Although weakly supervised, our method performs comparably as SegNet method on dataset CCSD-NL with the accuracy of 81.05% and 82.33%. Moreover, our method visually performs better than SegNet method on DAGM 2007 in which defects are poorly labeled in region level. Experimental results of our method show that the shape and edge features of regions detected are similar to those of defect regions in original collected images. Instead, those of SegNet method are approaching poorly labeled regions.
- **Detect unseen defects.** DefectGAN is capable of detecting some types of defects which are not in training data. It means it has the potential to detect some unseen defects.

In the remainder of this paper, the details of our proposed method will be given in Sec. II, including how our method works in practical application and the technical fundamentals of generative adversarial networks. Then in Sec. III, the efficiency of our proposed method is validated over experiments on texture and commutator cylinder images as two case study.

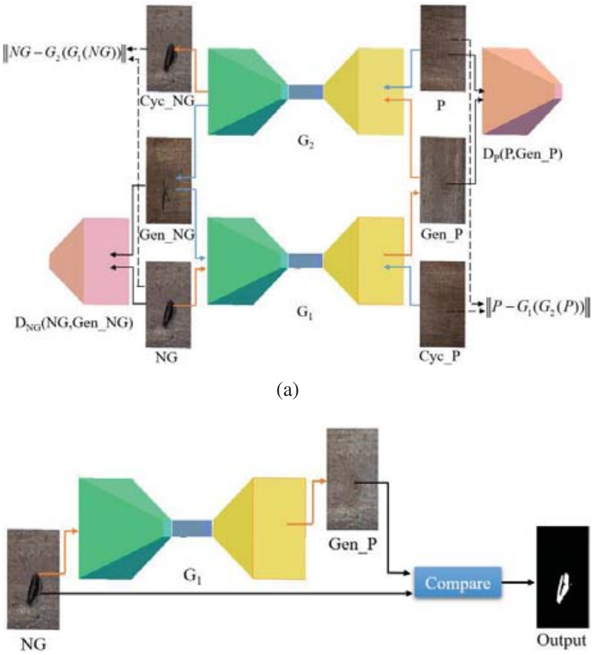
## II. Methodology

Fig.1 shows how our proposed method for defect detection works. In training stage, our proposed DefectGAN learns the mapping between negative images and positive image by training the dataset including positive and negative samples collected in industrial site. Then in testing stage, corresponding positive images are generated when negative images are fed into well-trained DefectGAN model followed by a simple post-process in which tested negative images and generated positive images are compared in gray scale to localize defect regions.

### A. Negative-to-Positive Translation

The key to our proposed method is to learn the mapping functions between negative samples and positive samples so that positive images are generated from negative images correspondingly.

In this work, we employ CycleGAN [18] to learn the mapping between these two domains *NG* (negative) and *P* (positive) given training samples $\{ng_i\}_{i=1}^{N}$ and $\{p_j\}_{j=1}^{M}$. As shown in Fig.1 (a), we introduce two generators which are a mapping from negative to positive: $G_1 : NG \rightarrow P$ and an inverse mapping: $G_2 : P \rightarrow NG$. Bedsides, we introduce two adversarial discriminators: (1) $D_{NG}$ which aims to distinguish $\{ng\}$ and $\{G_2(p)\}$ and (2) $D_P$ which aims to distinguish $\{p\}$ and $\{G_1(NG)\}$. After the adversarial training process, not only is the distribution of images in $G_1(NG)$ is indistinguishable from the distribution of *P*, but also can DefectGAN achieve $G_2(G_1(NG)) \approx NG$, named *cycle consistency*.



(a)

(b)

Fig. 1. The framework of our proposed method.(a) Training stage. (b) Test stage.

### B. Architecture

Tab.I and Tab.II respectively show the architectures of generator and discriminator. The generator networks contain several convolutional layers, several residual blocks and several transposed convolutional layers. We use 9 residual blocks for $256 \times 256$ training images. Each residual block includes two identical convolutional layers with $3 \times 3$ kernel, 1 stride and 256 filters. Instance normalization [25] and ReLU [26] are employed after each convolutional layer in generator.

For discriminator networks, one convolution is applied after the last layer to obtain 1-dimensional ouput. Leaky ReLU with a slope of 0.2 is employed instead of ReLU in discriminator to avoid vanishing gradient problem.

### C. Loss Function

In order to achieve two training goals mentioned above, we carefully design loss function. Typically, the total loss we adopt is composed of two types of loss: (1) *Adversarial loss* and (2) *Cycle consistency loss*.

*(1) Adversarial Loss*

Adversarial loss is computed for both mappings, $G_1$ and $G_2$. For the mapping from negative to positive $G_1 : NG \rightarrow P$ and its discriminator $D_P$, the adversarial loss is expressed as:

TABLE I
CONFIGURATION OF THE GENERATOR MODEL

| Type | Filters | Size | Stride |
|---|---|---|---|
| Convolutional | 64 | 7×7 | 1 |
| Convolutional | 128 | 3×3 | 2 |
| Convolutional | 256 | 3×3 | 2 |
| Residual block | | | |
| … | | | |
| Residual block | | | |
| Transposed convolutional | 128 | 3×3 | 2 |
| Transposed convolutional | 64 | 3×3 | 2 |
| Transposed convolutional | 3 | 7×7 | 1 |

TABLE II
CONFIGURATION OF THE DISCRIMINATOR MODEL

| Type | Filters | Size | Stride |
|---|---|---|---|
| Convolutional | 64 | 4×4 | 2 |
| Convolutional | 128 | 4×4 | 2 |
| Convolutional | 256 | 4×4 | 2 |
| Convolutional | 512 | 4×4 | 1 |
| Convolutional | 1 | 4×4 | 1 |

$$\begin{aligned} L_{GAN}(G_1,D_P,NG,P) = {} & E_{p\sim p_{data}(p)}[logD_P(P)] \\ & + E_{ng\sim p_{data}(ng)}[log(1-D_P(G_1(ng)))] \end{aligned} \tag{1}$$

where $E_{p\sim p_{data}(p)}[logD_P(P)]$ represents that the average value of the output result for discriminator $D_P$ by inputting defect-free images $p$ which belong to the defect-free domain $p_{data}(p)$ and $E_{ng\sim p_{data}(ng)}[log(1-D_P(G_1(ng)))]$ represents the average value of the result that 1 minus the output for discriminator $D_{NG}$ by inputting defect images $ng$ which belong to the defect domain $p_{data}(ng)$. The generator $G_1$ tries to generate images which are similar to images from domain P which forces $D_P(G_1(ng))$ to approach 1 and minimize $L_{GAN}(G_1,D_p,NG,P)$. At the same time, the discriminator $D_P$ tries to distinguish generated images $G_1(ng)$ and real image $p$ which forces $D_P(G_1(ng))$ to approach 0 and maximize $L_{GAN}(G_1,D_P,NG,P)$. Such optimization is described as $max_{D_P}min_{G_1}L_{GAN}(G_1,D_P,NG,P)$. For Another mapping $G_2 : P \rightarrow NG$ and its discriminator $D_{NG}$, the adversarial loss is similar with optimization described as $max_{D_{NG}}min_{G_2}L_{GAN}(G_2,D_{ng},P,NG)$.

*(2) Cycle Consistency Loss*

For generators $G_1$ and $G_2$, we hope that after feeding $G_1(ng)$ generated by generator $G_1$ into another generator $G_2$, the output $G_2(G_1(ng))$ is similar with $ng$, i.e., $ng \rightarrow G_1(ng) \rightarrow G_2(G_1(ng)) \approx ng$. Similarly, another cycle should achieve the similarity: $p \rightarrow G_2(p) \rightarrow G_1(G_2(p)) \approx p$. Particularly, $L_1$ norm is employed to measure the cycle consistency loss, expressed as:

$$\begin{aligned} L_{cyc}(G_1,G_2) = {} & E_{ng\sim p_{data}(ng)}[\|G_2(G_1(ng))-ng\|_1] \\ & + E_{p\sim p_{data}(p)}[\|G_1(G_2(p))-p\|_1] \end{aligned} \tag{2}$$

*(3) The Total Loss*

The total loss includes adversarial loss of two GANs and cycle consistency loss:

$$\begin{aligned} L_{GAN}(G_1,G_2,NG,P) = {} & L_{GAN}(G_1,D_p,NG,P) \\ & + L_{GAN}(G_2,D_{ng},P,NG) \\ & + \lambda L_{cyc}(G_1,G_2) \end{aligned} \tag{3}$$

where $\lambda$ controls the importance of two types of loss. The composition of loss function in DefectGAN architecture is shown in Tab.I. Our final object is to optimize generators with right parameters: $G_1^*,G_2^*$ achieving $\max_{D_{NG},D_P}\min_{G_1,G_2} L(G_1,G_2,D_{NG},D_P)$.

## D. Defect Detection

It is worthy to note that through cycle consistency loss, the common features of negative images and positive images are saved [18]. After thorough training, the mapping from negative to positive, i.e. the generator $G_1$ is used to generate positive images in test stage. Some examples of negative images tested and corresponding positive images generated by DefectGAN ($G_1$) are shown in Fig.3, Fig.4 and Fig.5. After comparing between paired images, we demonstrate that visually obvious differences only exit in defect regions and almost no difference exit in other regions with respect to gray, texture, and margin features. Here, we output a new image of the same size by simply computing the difference of each pixel in gray level. The smaller the value is, less relevant to defect. Hence, defect regions are highlighted in output image. After simple denoising and dilation processing, defects are localized as shown in Fig.3, Fig.4, and Fig.5.

# III. EXPERIMENTS

## A. Implementation Details

In this paper, we verify the effectiveness of DefectGAN on defect detection of two kinds of products, including workpiece and texture. The former is from dataset CCSD-NL and the latter is from dataset DAGM 2007 [21]. Then we compare our method with two methods, SegNet [22] and CAM [23] respectively based on supervised and weakly-supervised learning.

For all experiments, we set $\lambda$ =10 in (3). DefectGAN is trained using Adam solver [24] with a batch size of 1. We train all networks at the initial learning rate of 0.0002. We keep the same learning rate for the first 100 epochs and decay into zero linearly for the second also the last 100 epochs. Each dataset is trained for totally 200 epochs. All experiments are carried out with $256*256$ images on the server set up by ourselves. Our server is configured with a GPU, GTX1080Ti of 11G memory and a CPU, Intel(R) Core (TM) i7-7700K CPU @ 4.20GHz. DefectGAN runs smoothly on GPU, since it includes a large number of matrix-vector multiplication.

TABLE III
THE DIVISION OF DATASET1 PICKED FROM CCSD-NL

| | positive | negative |
|---|---|---|
| Training set | 350 | 156 |
| Test set | 350 | 100 |
| Sum | 700 | 256 |

### B. Dataset

- **CCSD-NL**[1]
  Commutator cylinder images are used in this paper as a case study of workpiece images. We introduce dataset CCSD-NL( Commutator Cylinder Surface Defect dataset without Labels). CCSD-NL contains 1800 positive images and 900 negative images with several types of defects such as wound, brush and tin-reside shown in Fig.4. Those defects visually show various appearances, in contrast positive images share the common appearance. we pick totally 956 images of CCSD-NL randomly as dataset1. The division of dataset1 is shown in Tab.III.
- **DAGM 2007**
  DAGM 2007 is a synthetic benchmark for defect detection of texture surface images. DAGM 2007 contains ten types of defects which all are artificially generated to simulate real world problem. Fig.3 shows some examples of DAGM 2007. We pick totally 2600 images randomly as dataset2. The division of dataset2 is shown in Tab.IV. We use this dataset for defect detection of texture images to evaluate and compare performance.

TABLE IV
THE DIVISION OF DATASET2 PICKED FROM DAGM 2007

| | Type1 | | Type2 | |
|---|---|---|---|---|
| | positive | negative | | defect |
| Training set | 575 | 83 | 575 | 84 |
| Test set | 575 | 67 | 575 | 66 |
| Sum | 1150 | 150 | 1150 | 150 |

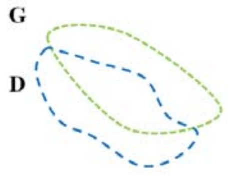
### C. Evaluation Metrics

- **mIoU**
  IoU (Intersection over Union) is a commonly used metric for object detection and semantic segmentation evaluation. Mean IU [11] is adapted based on pixel accuracy and IoU to evaluate the accuracy of semantic segmentation. In this work, we adopt IoU to evaluate detection accuracy on one test image and mIoU of the whole test set to evaluate detection accuracy of models. As shown in Fig.2, IoU of $D$ and $G$ and mIoU are defined as:

$$IoU(D,G) = \frac{A(D\cap G)}{A(D\cup G)} \tag{4}$$

$$mIoU = \frac{\sum_i IoU_i}{N} \tag{5}$$

where $A(D\cap G)$ and $A(D\cup G)$ respectively denote the areas of intersection and union of region (instead of rectangle) $D$ and $G$. $N$ denotes the number of test images, $IoU_i$ denotes the IoU value of image $i$.



(a)

Fig. 2. Compute IoU of Ground truth region (G) and Detect region (D).

- $t_{test}$
  Moreover, $t_{test}$ is another essential metric to evaluate the time cost of models. Test time for image $i$ is denoted as $t_i$, then we compute the average test time of the whole test set as $t_{test}$:

$$t_{test} = \frac{\sum_i t_i}{N} \tag{6}$$

In next section, models are evaluated and compared quantificationally using mIoU and $t_{test}$ introduced here. The smaller they are, the better performance on defect detection.

### D. Results and Analysis

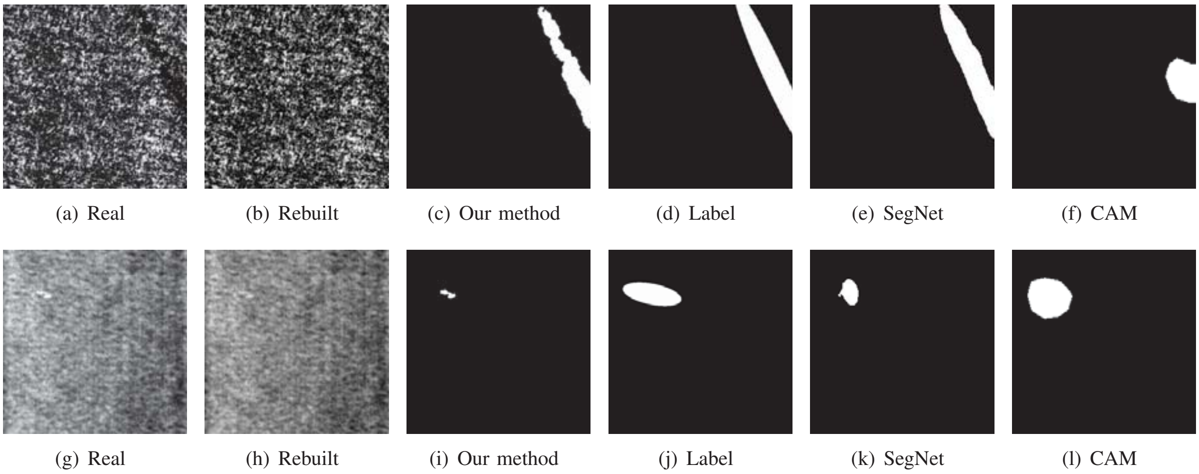
TABLE V
COMPARISON OF TWO METHODS IN QUANTITATIVE RESULTS.

| Methods | CCSD-NL | | DAGM 2007 | |
|---|---|---|---|---|
| | $t_{test}(ms)$ | $IoU(\%)$ | $t_{test}(ms)$ | $IoU(\%)^*$ |
| DefectGAN | 41.10 | 81.05 | 40.80 | – |
| SegNet [22] | 39.09 | 82.33 | 33.65 | – |
| CAM [23] | 269.60 | 52.03 | 277.90 | – |

* We do not compute mIoU on dataset DAGM 2007, which is meaningless because its region-level annotations are quite deviated from the real defects in shape and edge features.

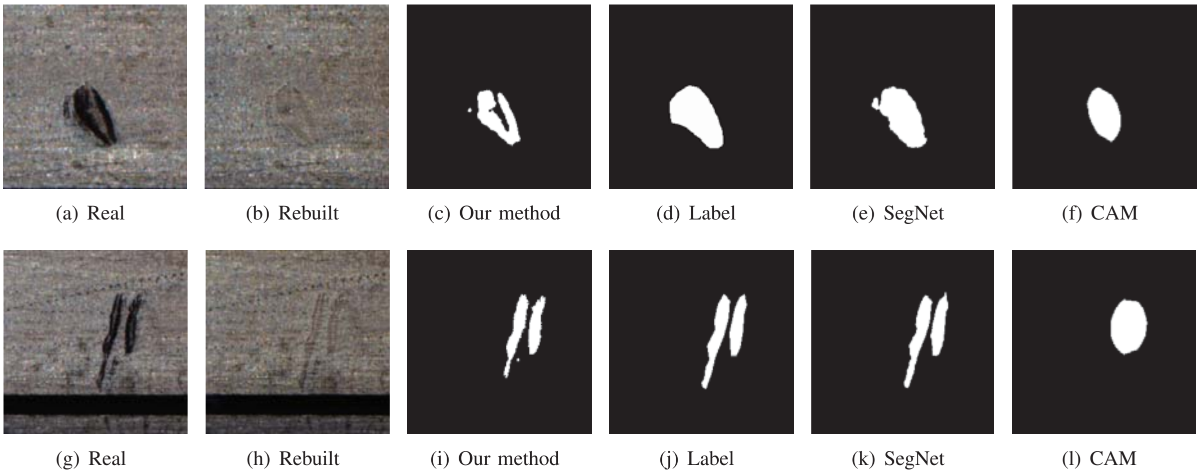
In this section, we compare our method with SegNet [22] and CAM [23] method on dataset CCSD-NL and DAGM 2007 as two case study. Qualitative results are shown in Fig.3, Fig.4 and Fig.5. All quantitative results are list in Tab.V. Here, we do not computer mIoU of compared methods on dataset DAGM 2007 but on dataset CCSD-NL. IoU on DAGM 2007 is meaningless, because the region-level annotations are quite deviated from the real defects in shape and edge features, as shown the fourth row of Fig.3. We mainly focus more on qualitative analysis than

[1] To get hands on dataset CCSD-NL, please contact the corresponding author by email: li_bin_hust@163.com.


(a) Real (b) Rebuilt (c) Our method (d) Label (e) SegNet (f) CAM

(g) Real (h) Rebuilt (i) Our method (j) Label (k) SegNet (l) CAM

Fig. 3. Some examples of defect detection of texture images. Real images in the first row, rebuilt images by DefectGAN in the second row, results of our method in the third row, labels in the fourth row, results of SegNet [22] in the fifth row, results of CAM [23] in the last row.


(a) Real (b) Rebuilt (c) Our method (d) Label (e) SegNet (f) CAM

(g) Real (h) Rebuilt (i) Our method (j) Label (k) SegNet (l) CAM

Fig. 4. Some examples of defect detection of commutator cylinder images. Real images in the first row, rebuilt images by DefectGAN in the second row, results of our method in the third row, labels in the fourth row, results of SegNet [22] in the fifth row, results of CAM [23] in the last row.

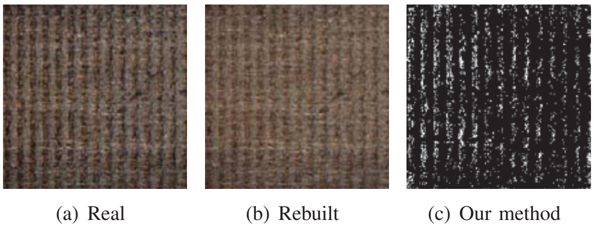

(a) Real (b) Rebuilt (c) Our method

Fig. 5. Defect detection of commutator cylinder images with defect named thread-line.

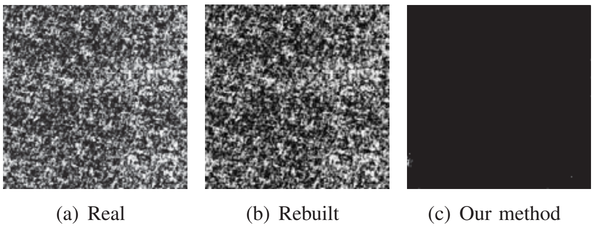

(a) Real (b) Rebuilt (c) Our method

Fig. 6. Defect detection of commutator cylinder images without defect.

quantitative comparison of detection performance to validate our unsupervised learning method.

**(1) Despite of weakly-supervised learning, our method achieves comparable accuracy as supervised learning methods.**

Although DefectGAN is trained with unpaired defect and non-negative images without any region-level annotations, on dataset CCSD-NL our method achieves the IOU of 81.05%, which is comparable as Segnet method trained with region-level annotations which achieves 82.33% as shown in Tab.V. Achieving almost the same accuracy, our method saves much time and money in data preparation. Besides, our method outperforms CAM-based method considerably by almost 30 points.

**(2) The proposed method tends to extract features of**

**the defect regions rather than labeled regions.** Comparing the fourth and the fifth row of Fig.3, we find that SegNet can only predict defect region close to labeled region with the similar outline. Defect regions in dataset DAGM 2007 are roughly labeled in pixel level, so that defect detection on this dataset is much rough. CAM method may not detect all defect regions and the detection is rough. In contrast, for our method, the outline of detected region is more similar to defect region in the original image rather than labeled region, as shown in the first and third row of Fig.3. Thus, we infer that our method relatively shows better performance on texture defect extraction. The same for commutator cylinder images.

**(3) Our method has the potential for detecting unseen defects.** As shown in Fig.5, our method successfully detects thread-line which is not in training set. We infer that DefectGAN has the ability to detect new defects, which greatly outperforms SegNet method which merely detects learned defects. Such results further validate the two advantages of our method inferred above.

**(4) DefectGAN is an effective defect cleaner.** As shown in Fig.3 (a) and (b), there is no visually obvious defect regions in texture images rebuilt by DefectGAN, but positive regions of original and rebuilt images share almost the common features, just like a defect cleaner. We infer that for texture images, DefectGAN successfully achieves the mapping from negative to positive and simultaneously keeps features of positive regions. DefectGAN works the same for commutator cylinder images, as shown in the first and second row of Fig.4. As for non-negative images, our method output images whose gray value in each pixel is almost zero because there is no defect regions and then no difference between input and output, as shown the detection results of non-negative images in Fig.6

## IV. CONCLUSIONS

We apply generative adversarial network in an essential industrial application, defect detection. Unlike SegNet method trained with region-level human annotations, DefectGAN localizes defect regions precisely by training with image-level labels. Images collected in real industrial site are labeled into two categories, negative and positive.

Although weakly supervised, it achieves comparable and even better performance in defect detection. On the one hand, defect regions detected by the proposed method are close to real defect regions rather than labeled regions, which outperforms supervised learning methods which might be constrained by poor region-level annotations. On the other hand, it can detect new types of defects which have not been seen in training set.

## REFERENCES

[1] Lin H , Li B , Wang X , et al. Automated defect inspection of LED chip using deep convolutional neural network[J]. Journal of Intelligent Manufacturing, 2018.

In practical application, our method can be used as image segmentation for image processing in next step, such as defect classification. Results can be further classified using deep neural networks or other machine learning methods.

[2] Zhong F , He S , Li B . Blob analyzation-based template matching algorithm for LED chip localization[J]. The International Journal of Advanced Manufacturing Technology, 2015:1-9.

[3] Kuo, C. F. J., Hsu, C. T. M., Liu, Z. X., Wu, H. C. (2014). Automatic inspection system of LED chip using two-stages backpropagation neural network. Journal of Intelligent Manufacturing, 25(6), 1235-1243.

[4] Zhao Y J , Yan Y H , Song K C . Vision-based automatic detection of steel surface defects in the cold rolling process: considering the influence of industrial liquids and surface textures[J]. International Journal of Advanced Manufacturing Technology, 2016:1-14.

[5] Chuanxia J , Jian G , Yinhui A . Automatic Surface Defect Detection for Mobile Phone Screen Glass Based on Machine Vision[J]. Applied Soft Computing, 2016, 52.

[6] Ngan H , Pang G , Yung N . Automated fabric defect detection-A review[J]. Image  Vision Computing, 2011, 29(7):442-458.

[7] Tsai D M , Chen M C , Li W C , et al. A fast regularity measure for surface defect detection[J]. Machine Vision and Applications, 2012, 23(5):869-886.

[8] Tolba A S , Raafat H M . Multiscale image quality measures for defect detection in thin films[J]. The International Journal of Advanced Manufacturing Technology, 2015, 79(1-4):113-122.

[9] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In In Proceedings of European Conference on Computer Vision (ECCV), pages 21-37. Springer, 2016.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779â788, 2016.

[11] Long J , Shelhamer E , Darrell T . Fully Convolutional Networks for Semantic Segmentation[J]. IEEE Transactions on Pattern Analysis Machine Intelligence, 2014, 39(4):640-651.

[12] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation[C]// International Conference on Medical Image Computing  Computer-assisted Intervention. 2015.

[13] Yu Z , Wu X , Gu X . Fully Convolutional Networks for Surface Defect Inspection in Industrial Environment[J]. 2017.

[14] Li Y , Huang H , Xie Q , et al. Research on a Surface Defect Detection Algorithm Based on MobileNet-SSD[J]. Applied Sciences, 2018, 8(9).

[15] Goodfellow I J , Pouget-Abadie J , Mirza M , et al. Generative Adversarial Nets[C]// International Conference on Neural Information Processing Systems. MIT Press, 2014.

[16] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.

[17] Arjovsky M, Chintala S, Bottou L. Wasserstein GAN[J]. 2017.

[18] Zhu J Y, Park T, Isola P, et al. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks[C]// IEEE International Conference on Computer Vision. 2017.

[19] Zhao Z , Li B , Dong R , et al. A Surface Defect Detection Method Based on Positive Samples[C]// Pacific Rim International Conference on Artificial Intelligence. Springer, Cham, 2018.

[20] Ojala T, Harwood I. A Comparative Study of Texture Measures with Classification Based on Feature Distributions[J]. Pattern Recognition, 1996, 29(1):51-59.

[21] HCI: Weakly Supervised Learning for Industrial Optical Inspection. https://hci.iwr.uniheidelberg.de/node/3616. Accessed 13 Nov 2017.

[22] Badrinarayanan V , Kendall A , Cipolla R . SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation[J]. 2015.

[23] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.(2016). Learning deep features for discriminative localization, The IEEE Conference on Computer Vision and Pattern Recognition, 2921-2929.

[24] Kingma D , Ba J . Adam: A Method for Stochastic Optimization[J]. Computer Science, 2014.

[25] Ulyanov D , Vedaldi A , Lempitsky V . Instance Normalization: The Missing Ingredient for Fast Stylization[J]. 2016.

[26] Krizhevsky, A., Sutskever, I., Hinton, G. E.(2012). Imagenet classification with deep Convolutional Neural Network, Krizhevsky A, Sutskever I, International Conference on Neural Information Processing Systems, 1097-1105.