Research on textile defects detection based on improved generative adversarial network

Jin Rui1,2 and Niu Qiang1

Abstract

Fabric defects seriously affect the textile industry in China. Given that traditional manual detection methods have low efficiency and poor accuracy, using automatic textile defect detection methods is urgently needed. A fabric defect detection method based on an improved generative adversarial network is thus developed to address the shortage of fabric defect samples. This method learns to reconstruct the fabric image in an unsupervised manner and locates the defect areas based on the differences between the original image and the reconstruction. Afterward, the defect-related features are extracted from these areas to further recognize specific fabric defects. The central loss constraint is introduced to improve the recognition performance of this method, and lightweight processing is applied to guarantee its real-time operation in embedded systems. The application of this method is then evaluated on the publicly available Tianchi dataset. Both quantitative and qualitative results show that the proposed method can accurately detect fabric defects.

Keywords
Fabric defect, generative adversarial network, center loss constrain, lightweight processing, auto-encoder

Introduction

Given that the presence of defects seriously affects the price of fabric,1 defect detection is critical to fabric production. Traditional defect detection methods usually involve manual detection and have the following disadvantages:

1. **Low efficiency**: a manual detection of defects only covers approximately 10 m/min, and such low speed reduces the automation level of production links and affects production efficiency.
2. **High labor intensity**: defect detection is a tedious manual work that requires inspectors to perform the same actions over an extended period.
3. **Low accuracy**: even the most skilled inspectors can find only about 70% of fabric defects, and small defects are challenging to recognize. When these inspectors are exhausted, their inaccurate and leakage inspection rates sharply increase, thereby introducing instability to the entire quality inspection link.

In response to these disadvantages, researchers have proposed the use of machine vision technology for the accurate and reliable detection of fabric defects. The current mainstream fabric defect detection methods include time-domain, frequency-domain, and deep learning methods.

1School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China
2Changzhou Vocational Institute of Textile and Garment, Changzhou, China

Corresponding author:
Niu Qiang, Changzhou Vocational Institute of Textile and Garment, Changzhou, Jiangsu 221116, China.
Email: niuq@cumt.edu.cn
The different grayscale values of pixels constitute the pattern and texture of fabric images. Time-domain methods, which mainly include statistics-based, texture-based, and gray co-occurrence matrix (GLCM) methods, detect defects by analyzing the grayscale distribution of fabric images. Statistics-based methods directly extract the grayscale histogram of fabric images as features and then classify these features by contrasting the grayscale histograms of normal/defective fabric images. However, these methods have poor applicability, and their accuracy is easily influenced by small defect areas and changes in illumination. To extract highly robust features, texture-based methods perform defect detection by extracting texture features from fabric images. Some commonly used texture features include the edge and local binary patterns. Texture filtering and statistical texture transformations are used to enhance the extracted texture features. However, categorizing defects based on these texture features leads to failure in recognizing specific defects. GLCM utilizes the spatial correlation properties of grayscale pixels to describe the patterns and textures of fabric images. These properties include contrast, inverse moments, entropy, and auto-correlation, all of which are sensitive to changes in fabric images and are easy to extract.

Frequency-domain methods are also widely used in fabric defect detection. Some commonly used frequency-domain methods include Fourier transform (FT), Gabor filtering (GF), and wavelet transform (WT). FT transforms an image from the space domain to the frequency domain using Fourier transform. Defect detection is then performed according to the characteristics of the defect-related high-frequency part. However, this method cannot easily position the defect areas. GF adds spatial representation to FT to achieve fabric defect localization. GF is also robust to changes in illumination, which may affect its defect detection performance. However, this method suffers from heavy computational overhead and cannot easily detect fabric defects in real-time. WT is similar to GF but has less computational overhead. However, WT achieves a higher defect detection success rate given the multi-scale property of wavelets.

With the recent development of deep learning methods, many researchers have applied deep neural networks to detect fabric defects. Wei et al. used Faster-RCNN to automatically detect textile defects and reported good detection results due to the robust feature learning ability of this method. Che et al. enhanced the applicability of the traditional Fast-RCNN to high-resolution input images and small region defects detection. However, Faster-RCNN is a two-stage object detection scheme that is insufficient for real-time applications. Jing et al. proposed an improved YOLOv3, which is a single-stage object detection method for detecting fabric defects in real time, and designed remarkable improvements based on the characteristics of fabric defects. Liu et al. proposed lightweight convolutional neural networks for detecting fabric defects, and their model can be run on edge computing platforms.

Some researchers have used sparse coding technology to achieve fabric defect detection. For example, Tong et al. proposed non-local sparse representation and constructed an over-complete dictionary for defect detection. Kang and Zhang developed a sparse-coding-based dictionary to learn defect patterns from fabric images. Unlike the method proposed by Tong et al., they designed an adaptive dictionary learning strategy to detect general fabric defects. Li et al. proposed a low-rank representation method for fabric defect detection and employed the matrix decomposition method to distinguish normal textile images from defective ones.

Although the above studies have examined fabric defect detection from different perspectives, several challenges remain, including the lack of defect samples and uneven sample distribution, which may affect the training of defect detection models, especially those that are based on supervised learning methods. Moreover, while deep learning methods have demonstrated exemplary performance in fabric defect detection, they continue to show certain deficiencies, such as their significant model computation and high dependence on training data. Achieving a balance between speed and accuracy remains a non-trivial problem.

Following the above discussions, this paper proposes a fabric defect detection method based on an improved generative adversarial network (GAN). Figure 1 illustrates the pipeline of the proposed method. As shown in the figure, a difference image is obtained by calculating the difference between the input image and its reconstructed result generated by the improved GAN. Then, post-processing is used to detect the ubiquitous defect, precisely, to position the approximate defect areas based on the difference image. Afterward, the input image is fed into the specific defect detection module to recognize the puncture hole, guiding the approximate defect areas. The red dashed line indicates that the specific defect detection module utilizes partial functions of the improved GAN, specifically, the encoder part in the generator. Details of different modules will be discussed later. The main contributions of this article are as follows:

1. To address the problem related to the lack of defect samples, this paper proposes an improved GAN to learn the reconstruction of fabric images in an unsupervised manner. Compared with traditional fabric defect detection methods that depend on large numbers of defect samples, the improved GAN takes a large number of normal fabric images as input. Hence, it is no longer dependent on plenty of defect samples of different categories. Afterward, the approximate defect areas are positioned based on the difference between the input and reconstructed images.
2. To further recognize specific defects, the approximate defect areas are used to guide the extraction of defect-related features from the generator of GAN. The center loss constraint is introduced to improve recognition performance.

Besides, the lightweight feature of the model is ensured through pruning operations and is optimized using TensorRT, which allows the model to run on NVIDIA Jetson TX2 in real-time. We argue that real-time performance is essential for fabric defect detection systems.

The proposed method is evaluated on the publicly available Tianchi AI dataset. Comparisons with other mainstream methods highlight the performance of the proposed method in detecting fabric defects. The rest of this paper is organized as follows. Section 2 introduces the related technologies. Section 3 presents the proposed method. Sections 4 and 5 present the evaluations and discussions, respectively.

**Related works**

**Fabric defect detection methods**

As we discussed before, the fabric defect detection methods mainly consist of the time-domain, frequency-domain, and deep learning methods. For a detailed review of modern fabric defect detection methods, we refer readers to the Li et al.24 For time-domain methods, early works always adopted grayscale histograms of fabric images.2 However, the grayscale histograms feature is not robust to disturbances like illumination changes. To improve the robustness of the fabric defect detection, many researchers designed various hand-crafted features3–9 based on local binary patterns, GLCM, and low-rank representation. For example, Raheja et al.8 combined GLCM and Gabor filter to detect fabric defects. Li et al.23 performed defect detection based on fabric low-rank representations.

Since fabric defects can be taken as the noises in fabric images, defect detection can also be performed on the frequency domain. Widely used frequency-domain approaches consist of Fourier transform (FT), Gabor filtering (GF), wavelet transform (WT), and their variants. To name a few, Yapi et al.25 designed an automatic fabric defect detection system using learning-based local textural distributions in the contourlet domain. Brad et al.26 detected fabric defects using the combination of Fourier transform and Gabor filtering. Yang et al.27 recognized different fabric defects based on wavelet transformation. Anandan and Sabeenian28 used the discrete curvelet transform to discover fabric defects.

Modern fabric defect detection methods always resort to deep neural networks to automatically extract defect-related features. These methods can be divided into two categories, including supervised and unsupervised defect detection. Supervised fabric defect detection methods always leverage state-of-the-art detectors (a detailed review can be found in the next section) to detect defect regions and recognize different defects directly. For example, the two-stage Faster-RCNN29 and the one-stage YOLO30 are widely used in fabric defect detection systems. Liu et al.31 utilized the single shot multi-box detector (SSD) to improve the defect detection, and the evaluation results demonstrate rationality and effectiveness. Ouyang et al.32 presented a CNN-based algorithm for on-loom fabric defect inspection. It introduces a dynamic activation layer that utilizes the defect probability information with a pairwise potential function to a CNN. However, these
supervised defect detection methods depend on many defective samples, which are hard to collect in practical conditions.

Unlike supervised defect detection methods, unsupervised fabric defect detection methods learn how to reconstruct a normal fabric image by feeding the model with plenty of normal fabric images to reduce the dependence on data. Theoretically, it cannot reconstruct a defective fabric image well, and hence defects can be found by calculating the difference between the defective input image and its reconstructed one. Typical unsupervised fabric defect detection methods are based on generators like GAN or auto-encoder. For example, Liu et al.35 trained a multistage GAN model to generate reasonable defective samples from normal ones. Le et al.34 utilized Wasserstein generative adversarial nets that combine transfer learning and multimodel ensembling technologies. Mei et al.35 introduced a multiscale convolutional denoising autoencoder network to synthesize fabric images further used to detect fabric defects. Our work is unsupervised, which mainly depends on an improved GAN to generate reasonable fabric samples.

**State-of-the-art object detectors**

Object detection refers to the identification of an object in an image along with its location and classification. As discussed in Section 2.1, many modern fabric defect detection methods rely on state-of-the-art object detectors, simultaneously locating the defect and recognizing its category. Hence, a brief survey of state-of-the-art object detectors based on deep learning is presented in this section.

The first deep neural network-based object detector is the Overfeat Network,36 which uses convolutional neural networks along with a sliding window. Later, Girshick et al.37 proposed Region-based Convolutional Neural Networks (RCNN), which improved nearly 30% over the previous state-of-the-art. RCNN performs object detection in a two-stage manner that firstly generates object proposals and then recognizes each proposal. Based on RCNN, Girshick further proposed Fast-RCNN,28 Faster-RCNN,29 and Mask-RCNN.39 These two-stage object detectors achieve state-of-the-art performance on benchmarking databases like COCO. However, their computational complexity restricts their applications on intelligent edge equipment.

To accelerate the detection process, researchers presented a series of one-stage object detectors. The two most well-known approaches are SDD and YOLO.30 These methods generate plenty of proposals simultaneously and classify each proposal to achieve object detection. Based on YOLO, many variants are proposed. Besides, some studies are devoted to proposing novel loss functions or improving network structures. For example, RetinaNet proposes a Focal loss to alleviate sample imbalance. Pyramid Pooling Network introduces the pyramid structure to solve the problem of scale difference.

**Lightweight object detection methods**

It is essential to reduce model parameters to accelerate their calculations or adopt parallel computing strategies to deploy the proposed method on computationally limited platforms. A more efficient way is designing lightweight CNN-based backbones. Unlike VGG,33 ResNet,34 and DenseNet,35 there are lots of lightweight backbones, including MobileNet,36 SqueezeNet,37 ShuffleNet,38 and PeleeNet.39 Besides, several well-known strategies, for example, matrix decomposition, packet convolution, small convolutional kernels, model pruning, and model distillation, can also decrease algorithm complexity. In this work, we design a lightweight CNN-based generator, discriminator, and encoder. Details of our network will be introduced in Section 3.1.

Except for the lightweight object detection methods, studies on framework and hardware are also conducted to accelerate the modern deep learning-based methods. Specifically, TensorRT is widely used at the framework level to provide low latency and high throughput deployment inference for deep learning models.50 It supports different deep learning frameworks, including TensorFlow, Caffe, and Pytorch. Besides, the TensorRT-based NVIDIA graphics processing unit, such as the high-performance embedding device Jetson TX2,51 can be used to detect fabric defects with lightweight methods in real-time.

**Proposed method**

The proposed method is divided into two stages. The ubiquitous defects are detected at the first stage by reconstructing the input fabric images with an improved GAN. The coarse defect areas are positioned based on the difference between the input fabric images and their reconstructed versions. Specific defects recognition is performed at the second stage. The coarse defect areas are used to guide the extraction of defect-related features from the generator. The center loss constraint is introduced to improve recognition performance. Specifically, a center for deep features and their corresponding class centers. With such an additional constraint, the learned deep features are more discriminative to detect specific defects. The proposed model is deployed on the NVIDIA Jetson TX2 after lightweight processing based on TensorRT.

**Ubiquitous defect detection**

Traditional fabric defect detection suffers from various defects and an imbalanced data distribution. Therefore, training a detection model in a supervised manner presents a challenge. An improved GAN is designed to detect ubiquitous defects in an unsupervised manner. At the training stage, GAN takes normal fabric images as input, and the network reconstructs normal fabric images after training. In this case,
GAN cannot reconstruct both defect and normal images. Therefore, obvious differences can be observed between the input defect images and their reconstructed results at the test stage. On the basis of these differences, ubiquitous defects and coarse defect areas can be identified.

Figure 2 presents the structure of the improved GAN. A detailed introduction of vanilla GAN can be found in Creswell et al. In this work, we directly introduce our improved GAN used to reconstruct fabric images. As shown in the figure, the network contains the generator G, discriminator D, and encoder E, where the generator G is implemented using an auto-encoder. The encoder and decoder in G are denoted by $G_E$ and $G_D$, respectively. In this work, $G_E$, D, and E have the same lightweight structure that contains four convolutional layers. Each is followed by a max-pooling layer, a batch normalization layer, and a P-Relu layer. All convolutional layers consist of $3 \times 3$ convolutional kernels, and their channel numbers are 8, 16, 32, and 64, respectively. Notably, $G_E$ has an inverted structure of $G_D$ and replaces the convolutional layer with transposed convolutional layer. Details to reconstruct fabric images with the improved GAN are as follows:

Firstly, $G_E$ takes sample $x$ as input and outputs a latent variable $z$, which is subsequently fed into $G_D$ to output the reconstructed sample $\tilde{x}$ as follows:

$$\tilde{x} = G_D(z; W_{GD}) = G_D(G_E(x; W_{GE}))$$  \hspace{1cm} (1)

where $W_{GD}$ and $W_{GE}$ are the learnable weights of $G_D$ and $G_E$, respectively.

Afterward, the discriminator D takes the sample $x$ and reconstructed sample $\tilde{x}$ as inputs and then judges which of them is the real sample as follows:

$$y = D(x, \tilde{x}; W_D)$$  \hspace{1cm} (2)

where $y$ represents the output of discriminator D and $W_D$ represents its learnable parameter.

Unlike a vanilla GAN, an additional encoder $E$ is introduced to re-encode the reconstructed sample $\tilde{x}$ as $\tilde{z}$ as follows:

$$\tilde{z} = E(\tilde{x}; W_E)$$  \hspace{1cm} (3)

where $W_E$ represents the learnable parameter of the decoder E, which is used to compress the image $\tilde{x}$ that is reconstructed by the generator. Specifically, E down-scales to find its feature representation, which has the same size as that of $\tilde{z}$ for consistent comparison. In contrast to the prior auto-encoder-based approaches, in which the minimization of the latent vectors is achieved via the bottleneck features, the encoder E explicitly learns to minimize the distance with its parametrization. With the encoder E, the generator G can better encode the input image by simultaneously comparing the distribution differences in image and latent spaces.

Finally, to train the improved GAN, a joint loss function is introduced to optimize the network in a weighted manner. The loss function contains the following parts:

1. Adversarial loss: The adversarial loss characterizes the loss between the generator and discriminator during the game process. This loss is calculated as follows using L2 loss:

$$L_{adv} = \| f(x) - f(G(x)) \|_2$$  \hspace{1cm} (4)
2. Content loss: Content loss reflects the difference between the reconstructed image $\hat{x}$ and the real image $x$ and is calculated as follows using L1 loss:

$$L_{con} = \|x - G(x)\|_1$$  \hspace{1cm} (5)

3. Encoding loss: The adversarial loss and content loss in the traditional GAN can enforce the generator to generate realistic and contextually sound samples. To better produce normal samples while restricting defective ones, an encoding loss is used to minimize the distance between the bottleneck features of the input data and the encoded features of the generated samples. Therefore, the generator learns how to encode features of the generated image for normal samples. It will fail to minimize the distance between the input data and encoded features of the generated images for defective samples. The encoding loss is calculated as follows using L2 loss:

$$L_{enc} = \|G_E(x) - E(G(x))\|_2$$  \hspace{1cm} (6)

The joint loss function is defined as follows in a weighted manner:

$$L = W_{adv}L_{adv} + W_{con}L_{con} + W_{enc}L_{enc}$$  \hspace{1cm} (7)

where $W_{adv}$, $W_{con}$, and $W_{enc}$ are the weighted parameters used to adjust the effect of an individual loss on the overall objective function. These parameters are set to 0.2, 0.7, and 0.1, respectively, through cross-validation on a public dataset.

After reconstructing fabric images, post-processing is used to detect ubiquitous defects and position coarse defect areas based on the input $x$ and reconstructed $\hat{x}$. As shown in Figure 3, the difference image $\Delta x$ can be calculated by subtracting $\hat{x}$ from $x$. Median and morphology filtering are applied on $\Delta x$ to filter out the small noise areas. Afterward, connected component analysis is performed on the filtered $\Delta x$. If the connected area exceeds a given threshold, then $x$ is considered a defect sample. All connected areas are merged to constitute the set of coarse defect areas, which is denoted by $R$.

### Specific defect recognition

In specific defect recognition, extracting defect-related features is considered non-trivial if the defect areas are too small. Therefore, the information obtained at the ubiquitous defect detection stage is fully utilized. Specifically, the coarse defect areas guide the extraction of defect-related features from $G_E$ given that $G_E$ has learned the fabric patterns at the first stage. To improve recognition performance, the center loss constraints are introduced to enhance the discriminative ability of the learned features.

Figure 4 presents the flowchart of specific defect recognition. Given that the fabric image has ubiquitous defects at the first stage, the coarse defect areas are enlarged 1.5 times to generate fabric defect mask $M$, which contains probable defect areas and spatial context information. After the structure of $G_E$ and the size of the original image are fixed, the sizes of the feature maps generated by $G_E$ at different depths are determined. These feature maps contain visual features of fabric and we want to extract defect-related features from them. Therefore, we use the fabric defect mask $M$ as a guide to extract defect-related features from $F_i$ ($i = 1, 2, \ldots, L$). To achieve this, $M$ is re-sized to generate the mask sequence $\{M_i\}$, in which $M_i$ retains the same size as $F_i$. Then, $M_i$ can be used to constrain $F_i$ as follows:

$$F_i = F_i \odot M_i, \hspace{0.5cm} i = 1, \ldots, L$$  \hspace{1cm} (8)

where $\odot$ denotes element-wise multiplication, $L$ is the number of convolutional layers in $G_E$, and $F$ represents the extracted defect-related features.
A three-layer MLP (neuron numbers are 64, 32, and 16, respectively) is used to further process $F$ and outputs $F'$, whose dimension is reduced from the channel number of $F$ to the number of defect categories. To train the specific defect classifier, $F'$ is fed into a SoftMax activation function, and the following cross-entropy loss is used for classification:

$$L_S = \sum_{i=1}^{K} \tilde{y}_i \log s_i$$  \hspace{1cm} (9)

where $K$ represents the number of defect categories, $\tilde{y}_i$ represents the one-hot encoding of the ground truth label, and $s_i$ represents the output of the SoftMax activation function.

To enhance the discriminability of the learned deep features, the following center loss is introduced as a constraint:

$$L_C = \frac{1}{2} \sum_{i=1}^{N} \left\| F'_i - c_{y_i} \right\|_2^2$$  \hspace{1cm} (10)

where $F'_i$ represents the learned deep features output by the MLP as illustrated in Figure 4. The $c_{y_i}$ denotes the $y_i$-th category center of the deep features. Ideally, the $c_{y_i}$ should be updated as the deep features changed.

The joint loss function is then calculated as follows in a weighted manner:

$$L = w_1 L_S + w_2 L_C$$  \hspace{1cm} (11)

where weights $w_1$ and $w_2$ are set to 0.9 and 0.1, respectively, based on the cross-validation results on a public dataset.

**Model lightweight processing**

The proposed method is lightweight processed to meet the real-time requirements in embedded devices. The lightweight processing is achieved by pruning the proposed model. The pruning process can be divided into two steps. First, the decoder $G_D$, in the generator, the encoder $E$, and the discriminator $D$ are removed because these modules are only used in the training stage. After training, only the encoder $G_E$ is used to reconstruct the input fabric image. Afterward, the channel numbers of convolution layers in the remaining modules are reduced to decrease the computational overhead. Specifically, we employ a commonly used network optimization tool, optuna, to automatically shrink the channel numbers and fine-tune the hyperparameters to prevent sharp performance degradation.

The Pytorch-based network is initially converted into the ONNX format for the lightweight models. Afterward, TensorRT is used to run the ONNX format model. TensorRt mainly comprises the build and deployment modules. The build module is mainly responsible for the transformation of the model, during which the module automatically completes inter-layer optimization and precision calibration. Meanwhile, the deployment module deserializes the plan files obtained in the previous step to create a running engine that enables network forward propagation. The model is eventually deployed on NVIDIA Jetson TX2.
Experimental results analysis

Dataset

The experimental data are obtained from the fabric defect detection dataset of Xuelang Tianchi AI Challenge, which contains 3331 textile images with defective positioning labels. Among these images, 2163 and 1168 are considered normal and defective, respectively. A total of 22 defects are detected, including knots, thin spinning, puncture hole, stains, and jumps. Given the data imbalance, the defect image samples are reintegrated into 10 categories, including puncture hole, knots, rubbing hole, brushed hole, thin spinning, hanging warp, lacking warp, jumps, stains, and others. These defects are common and not very complex. Notably, the proposed method can be used to detect other complex defects theoretically if enough training samples can be collected in practical conditions. The protocol proposed in18 is adopted to ensure fair comparisons. Specifically, 70% of the dataset is used as a training set, whereas the remaining 30% is used as the test set. The samples and labels of the training data are shown in Figure 5.

Ubiquitous defect detection

Figure 6 shows typical fabric defects in the test set. All images are transformed from RGB to grayscale to reduce computational complexity. The images from left to right and from up to down show the puncture hole, rubbing hole, knots, and thin spinning, respectively.

Figure 7 presents the reconstructed results of samples shown in Figure 6 using the improved GAN. At the training stage, the generator only learns knowledge from normal fabric images and is therefore unable to effectively reconstruct the defect regions.

Figure 8 presents the differences between the test samples and their reconstructed results. Significant differences can be observed in the possible defect regions given that the generator cannot efficiently reconstruct the defective regions. As shown in the figure, the difference image contains the possible defect areas, and a binarization operation needs to be applied to detect ubiquitous defects and determine the coarse defect areas.

Figure 9 presents the binarization results of the difference images shown in Figure 8. After a series of operations, such as filtering denoising and connected domain analysis, ubiquitous defects are detected by comparing the ratio (foreground white pixels/total pixels) with a given threshold. If the ratio exceeds the threshold, the fabric image is considered a defective sample. The coarse defect areas are positioned based on the foreground white pixels.

Thresholds determination

As described in Section 4.2, ubiquitous defects are detected based on the binarization results of the difference images, which are calculated between the input and reconstructed fabric images. Moreover, the ubiquitous defects are detected by comparing the ratio with a given threshold. Ideally, the reconstructed images should contain no defect for further defect detection processing. However, we must highlight that all reconstructed images generated by the proposed GAN may contain noise and ghosts, even when reconstructed from normal fabric images. Therefore, the binarization threshold T and the given threshold P are essential in ubiquitous defect detection. What makes the thresholds determination thornier is that T and P have a mutual influence. Therefore, a detailed analysis is conducted to determine these thresholds.
Figure 6. Typical fabric defects in the test set.

Figure 7. Reconstructed results of the samples shown in Figure 6.
Figure 8. Differences between the test samples and their reconstructed results.

Figure 9. Binarization results of the difference images shown in Figure 8.
Small deviations are observed between the normal fabric images and their reconstructed results. Therefore, determining $T$ and $P$ not only ensures the detection of defective areas but also avoids judging the normal fabric images as defective. All fabric images in this work are divided into normal and defect samples. The well-trained GAN reconstructs all normal and defect samples, and the recognition rates of different $T/P$ settings are calculated to determine the $T/P$ values.

Table 1 shows the rates at which the normal fabric images are detected as normal when using different $T/P$ settings. The recognition rate is used as the evaluation index, which is computed as the ratio of correctly identified samples to all test samples. Recognition rates above 0.78 (empirical threshold) are highlighted in faint yellow. These corresponding $T/P$ settings are assumed to successfully recognize the normal fabric images.

Table 2 shows the rates at which defect fabric images are detected as defects when using different $T/P$ settings. Recognition rates above 0.60 (empirical threshold) are highlighted in faint yellow. N/A indicates that the recognition rate is less than 0.5. Given that $T/P$ values should identify the defective areas and avoid misclassification of normal images as defects, overlapping faint yellow regions are highlighted in dark yellow. In these regions, the proposed method can meet the abovementioned requirements.

Following the above discussions, the highest recognition rate is achieved when the $T/P$ values are set to 70 and 0.03. Therefore, these $T/P$ settings are used as thresholds in the subsequent analyses.

### Quantitative evaluations

After determining the $T/P$ settings, the proposed method is quantitatively compared with several mainstream fabric defect detection methods on a public dataset. These methods include the AlexNet-based OurNet,\textsuperscript{54} CNN-based defect detection method,\textsuperscript{19} and non-local sparse representation-based defect detection method.\textsuperscript{20} All these methods use the same data settings to ensure fair comparisons. Table 3 presents the recognition rates for each method.

As shown in the table, OurNet has a poor defect detection performance and cannot efficiently handle small region defects because it only used an AlexNet as the backbone. Hence, it is non-trivial to recognize small region defects by directly extracting defective features from the whole image. Jing et al.\textsuperscript{19} proposed a lightweight CNN designed according to the characteristics of fabric defects, thereby improving its defect detection performance. They proposed a lightweight variant of YOLO, named YOLO-LFD, to detect fabric defects. Multi-scale features were extracted to improve the detection ability of the proposed method for

---

**Table 1. Rates of detecting normal fabric images as normal when using different $T/P$ settings.**

| $P/T$ | 30 (%) | 35 (%) | 40 (%) | 45 (%) | 50 (%) | 55 (%) | 60 (%) | 65 (%) | 70 (%) | 75 (%) |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 0.03  | 3.4    | 8.8    | 20.2   | 32.9   | 46.0   | 55.8   | 64.1   | 71.7   | 78.1   | 83.5   |
| 0.05  | 4.7    | 13.1   | 26.2   | 40.9   | 53.9   | 61.9   | 70.2   | 78.4   | 83.3   | 87.2   |
| 0.07  | 5.6    | 16.4   | 31.5   | 46.3   | 57.6   | 67.4   | 75.8   | 82.1   | 86.1   | 89.9   |
| 0.09  | 7.0    | 18.8   | 36.5   | 50.5   | 61.2   | 71.5   | 79.3   | 84.6   | 88.5   | 92.0   |
| 0.11  | 8.7    | 22.0   | 39.9   | 54.8   | 65.4   | 74.9   | 82.6   | 87.4   | 90.3   | 93.3   |
| 0.13  | 10.3   | 24.6   | 43.4   | 58.2   | 68.6   | 77.4   | 84.8   | 88.8   | 91.4   | 94.1   |
| 0.15  | 11.7   | 26.4   | 46.6   | 60.7   | 71.1   | 79.4   | 86.5   | 89.8   | 92.9   | 94.5   |
| 0.17  | 13.0   | 29.5   | 50.1   | 63.1   | 73.5   | 83.2   | 87.9   | 90.6   | 93.7   | 94.8   |
| 0.19  | 14.9   | 32.5   | 51.7   | 65.2   | 75.0   | 84.4   | 88.6   | 91.2   | 94.1   | 95.3   |
| 0.21  | 16.3   | 35.1   | 54.2   | 66.7   | 77.3   | 85.5   | 89.3   | 91.7   | 94.5   | 95.6   |
| 0.23  | 17.0   | 37.5   | 55.9   | 68.5   | 78.5   | 86.3   | 89.8   | 92.2   | 95.1   | 95.9   |
| 0.25  | 18.6   | 39.5   | 58.6   | 71.0   | 80.3   | 87.2   | 90.3   | 92.7   | 95.6   | 96.2   |
| 0.27  | 19.4   | 41.3   | 59.7   | 72.2   | 81.4   | 88.0   | 90.6   | 93.5   | 95.8   | 96.8   |
| 0.29  | 21.2   | 44.3   | 62.5   | 74.0   | 82.9   | 89.3   | 91.4   | 94.9   | 96.0   | 97.0   |
| 0.31  | 21.7   | 45.0   | 62.7   | 74.3   | 83.3   | 89.7   | 91.5   | 94.9   | 96.1   | 97.0   |
| 0.33  | 23.2   | 46.6   | 64.6   | 75.6   | 84.4   | 90.1   | 91.8   | 95.5   | 96.6   | 97.3   |
| 0.35  | 24.2   | 47.9   | 65.2   | 76.1   | 84.9   | 90.4   | 92.6   | 95.9   | 96.8   | 97.3   |
| 0.37  | 25.3   | 49.9   | 66.8   | 77.6   | 85.7   | 90.6   | 92.8   | 96.2   | 97.0   | 97.5   |
| 0.39  | 26.9   | 51.5   | 68.5   | 78.4   | 86.6   | 90.9   | 93.2   | 96.4   | 97.2   | 97.8   |
| 0.41  | 27.7   | 52.3   | 69.6   | 78.8   | 86.8   | 91.1   | 94.2   | 96.5   | 97.3   | 97.8   |
| 0.43  | 28.3   | 53.3   | 70.5   | 79.8   | 87.7   | 91.4   | 94.4   | 96.6   | 97.3   | 97.8   |
| 0.45  | 29.7   | 54.3   | 70.6   | 80.8   | 88.1   | 91.4   | 94.8   | 96.6   | 97.3   | 97.8   |
different size defects. However, its supervised training strategy restricts its detection performance. Liu et al.\textsuperscript{20} used non-local sparse representation to characterize the fabric patterns and detect the defects. Despite its ability to describe large and middle region defects, it is struggling to characterize the patterns of small region defects. Its performance is between OurNet and YOLO-LFD. Compared with the above-mentioned methods, our method learns the fabric patterns with an improved GAN, uses the coarse defect areas to guide the extraction of defect-related features, and introduces the center loss constraint to improve the discriminability of the learned features. From these improvements, the proposed method achieves the best defect detection performance on the public dataset among the compared methods.

Lightweight processing is applied to improve the detection speed of the proposed method. The total defect detection process contains three parts, including reconstructing the fabric image with the improved GAN, localizing the defect regions from the difference between the input fabric image and the reconstructed image, and recognizing the defect category by classifying the defect-related features. These parts can be worked in a cascaded manner after training the improved GAN and classifier, respectively. Specifically, the total defect detection process spends approximately 80 ms in defect detection on the TX2 processor without lightweight processing. However, the whole process only spends 36 ms in defect detection with lightweight processing. Such improvement in detection speed meets the real-time running requirements of edge intelligent devices.

**Qualitative evaluations**

Qualitative evaluations are performed on a public dataset, and OurNet is used for comparison. Figure 10 shows the fabric defect detection results, where the inaccurate predictions are highlighted in red. The proposed method successfully recognizes different defect types, whereas OurNet fails to identify the hanging warp and jump defects as shown in the third line of Figure 10. Moreover, OurNet mistakes the thin spinning and stain defects for the lacking warp and hanging warp defects, respectively. The proposed method is obviously more powerful than

---

### Table 2. Rates of detecting fabric defect images as defects when using different T/P settings.

| P/T  | 30 (%) | 35 (%) | 40 (%) | 45 (%) | 50 (%) | 55 (%) | 60 (%) | 65 (%) | 70 (%) | 75 (%) |
|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 0.03 | 96.8   | 93.2   | 89.1   | 83.7   | 79.3   | 75.7   | 71.8   | 67.9   | 65.9   | 60.3   |
| 0.05 | 95.3   | 90.9   | 84.7   | 80.2   | 75.4   | 71.0   | 67.4   | 63.0   | 59.6   | 56.7   |
| 0.07 | 95.2   | 90.5   | 84.4   | 79.0   | 75.1   | 70.3   | 66.5   | 62.1   | 58.5   | 55.6   |
| 0.09 | 93.2   | 88.1   | 81.6   | 76.6   | 72.2   | 66.9   | 63.0   | 58.9   | 55.2   | 51.4   |
| 0.11 | 92.1   | 86.6   | 80.8   | 75.7   | 70.7   | 65.4   | 61.4   | 57.2   | 53.4   | N/A    |
| 0.13 | 90.9   | 84.6   | 78.7   | 74.2   | 68.9   | 63.5   | 59.1   | 55.0   | 51.4   | N/A    |
| 0.15 | 90.3   | 83.1   | 77.5   | 72.2   | 67.2   | 61.4   | 56.9   | 53.8   | 50.3   | N/A    |
| 0.17 | 89.4   | 82.3   | 76.4   | 71.0   | 65.9   | 60.8   | 56.2   | 52.2   | N/A    | N/A    |
| 0.19 | 88.7   | 81.9   | 75.8   | 70.3   | 64.7   | 59.7   | 55.8   | 51.1   | N/A    | N/A    |
| 0.21 | 87.8   | 80.5   | 74.2   | 69.3   | 63.2   | 58.3   | 54.3   | 50.2   | N/A    | N/A    |
| 0.23 | 86.9   | 80.1   | 73.4   | 67.7   | 62.1   | 57.3   | 52.6   | N/A    | N/A    | N/A    |
| 0.25 | 86.4   | 79.5   | 72.7   | 67.2   | 61.7   | 56.7   | 51.7   | N/A    | N/A    | N/A    |
| 0.27 | 85.7   | 78.9   | 70.9   | 66.0   | 60.3   | 55.2   | 50.4   | N/A    | N/A    | N/A    |
| 0.29 | 85.2   | 78.1   | 70.6   | 64.8   | 59.5   | 54.3   | N/A    | N/A    | N/A    | N/A    |
| 0.31 | 85.0   | 77.1   | 69.8   | 63.9   | 59.0   | 54.0   | N/A    | N/A    | N/A    | N/A    |
| 0.33 | 84.4   | 76.8   | 69.1   | 62.9   | 58.5   | 52.9   | N/A    | N/A    | N/A    | N/A    |
| 0.35 | 83.8   | 76.0   | 68.2   | 62.1   | 57.3   | 51.9   | N/A    | N/A    | N/A    | N/A    |
| 0.37 | 82.8   | 75.7   | 67.6   | 61.1   | 56.7   | 51.3   | N/A    | N/A    | N/A    | N/A    |
| 0.39 | 81.9   | 74.9   | 67.1   | 61.0   | 55.8   | N/A    | N/A    | N/A    | N/A    | N/A    |
| 0.41 | 81.1   | 74.0   | 66.3   | 60.5   | 54.4   | N/A    | N/A    | N/A    | N/A    | N/A    |
| 0.43 | 80.4   | 72.4   | 65.3   | 59.4   | 53.5   | N/A    | N/A    | N/A    | N/A    | N/A    |
| 0.45 | 79.8   | 71.6   | 64.7   | 59.1   | 53.2   | N/A    | N/A    | N/A    | N/A    | N/A    |

### Table 3. Recognition rates for different methods.

| Method                              | Recognition rate (%) |
|-------------------------------------|----------------------|
| OurNet\textsuperscript{54}          | 66.2                 |
| YOLO-LFD\textsuperscript{19}        | 70.6                 |
| Non-local sparse representation\textsuperscript{20} | 68.4                 |
| Ours                                | 74.9                 |
Figure 10. Qualitative analysis of defect detection on a public dataset.

Figure 11. Defect detection results on self-collected data.
OurNet in recognizing defects in small regions. Such superiority can be ascribed to its use of coarse defect areas to guide the extraction of defect-related features. Moreover, the center loss constraint can improve the discriminability of the learned features. Therefore, the proposed method effectively addresses the challenges in detecting small defects.

Besides, we evaluated the proposed method on several self-collected fabric defective images to prove the generalization performance of the proposed method. Similar to the qualitative evaluations on benchmarking database, OurNet is adopted for comparison. As shown in Figure 11, the proposed method successfully detects all fabric defects, including stains, hanging warp, and knots. However, OurNet cannot detect the stains and knots defects as shown in the second line because it is not suitable to detect small region defects. The results of Figure 11 verify that the proposed method can achieve satisfactory performance in practical conditions.

Summary and discussion

An automatic fabric defect detection algorithm based on an improved GAN was proposed in this work. Considering the various fabric defects and imbalanced data distribution, we introduced the GAN to detect ubiquitous defects in an unsupervised manner. The GAN learned to reconstruct fabric images from normal samples in the training stage. Therefore, it could not reconstruct fabric defects because it learned nothing about them in the training stage. In contrast to the vanilla GAN, an additional encoder was introduced to re-encode the reconstructed sample. The proposed encoding loss improves the learning process of the latent vectors. Moreover, ubiquitous defect detection was achieved by calculating the difference between the input fabric images and their reconstructed results generated by the well-trained GAN. The difference images were further used to generate the coarse defect masks, which in turn were used to guide the extraction of defect-related features from the feature maps generated by the encoder in the generator (GE). An MLP processed the defect-related features, and the output of MLP was fed into the Softmax to recognize the category of specific defect. The center loss constraint was then used to improve the discriminability of the learned features. Evaluations on the publicly available dataset of Xuelang Tianchi AI Challenge verified the feasibility and accuracy of the proposed method.

This method was proven useful in achieving automatic quality detection in textile production and in improving textile production efficiency. In our future work, we aim to accurately locate the defect areas and further improve the defect detection accuracy of the proposed method.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was funded by the National Natural Science Foundation of China under grant number 51674265.

ORCID iD

Rui Jin https://orcid.org/0000-0001-9743-3133

References

1. Srinivasan K, Dastoor PH, Radhakrishnaiah P, et al. FDAS: a knowledge-based framework for analysis of defects in woven textile structures. J Text Inst 1992; 83(3): 431–448.
2. Liu Z, Zhao Q, Li C, et al. Fabric defect detection algorithm using local statistic features and global saliency analysis. J Textile Res 2014; 35(11): 62–67.
3. Jia X. Fabric defect detection based on open source computer vision library OpenCV. In: 2010 2nd international conference on signal processing systems, 2010, vol. 1, pp.V1-342–V1-345. New York: IEEE.
4. Hao M, Junfeng J, Zebin S, et al. Patterned fabric defect detection based on LBP and HOG feature. J Electron Meas Instrum 2018; 32(4): 95–102.
5. Jing J, Zhang H, Wang J, et al. Fabric defect detection using Gabor filters and defect classification based on LBP and Tamura method. J Text Inst 2013; 104(1): 18–27.
6. Hamdi AA, Sayed MS, Foud M, et al. Unsupervised patterned fabric defect detection using texture filtering and K-means clustering. In: 2018 international conference on innovative trends in computer engineering (ITCE), 2018, pp.130–144. New York: IEEE.
7. Alper Selver M, Aşar V and Özdemir H. Textural fabric defect detection using statistical texture transformations and gradient search. J Text Inst 2014; 105(9): 998–1007.
8. Raheja JL, Kumar S and Chaudhary A. Fabric defect detection based on GLCM and Gabor filter: a comparison. Optik 2013; 124(23): 6469–6474.
9. Konda Reddy RO, Eswara Reddy B and Keshava Reddy E. Classifying similarity and defect fabric textures based on GLCM and binary pattern schemes. Int J Inf Eng Electron Bus 2013; 5(5): 25–33.
10. Sakhare K, Kulkarni A, Kumbhakarn M, et al. Spectral and spatial domain approach for fabric defect detection and classification. In: 2015 international conference on industrial instrumentation and control (ICIC), 2015, pp.640–644. New York: IEEE.
11. Malek AS, Drear JY, Bigue L, et al. Optimization of automated online fabric inspection by fast Fourier transform (FFT) and cross-correlation. Text Res J 2013; 83(3): 256–268.
Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. Adv Neural Inf Process Syst 2015; 28: 91–99.

Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp.779–788. New York: IEEE.

Liu Z, Liu S, Li C, et al. Fabric defects detection based on SSD. In: Proceedings of the 2nd international conference on graphics and signal processing: ICGSP’18, Sydney, Australia, October 2018. New York: Association for Computing Machinery.

Ouyang W, Xu B, Hou J, et al. Fabric defect detection using activation layer embedded convolutional neural network. IEEE Access 2019; 7: 70130–70140.

Liu J, Wang C, Su H, et al. Multistage GAN for fabric defect detection. IEEE Trans Image Process 2020; 29: 3388–3400.

Le X, Mei J, Zhang H, et al. A learning-based approach for surface defect detection using small image datasets. Neurocomputing 2020; 408: 112–120.

Mei S, Wang Y and Wen G. Automatic fabric defect detection with a multi-scale convolutional denoising autoencoder network model. Sensors 2018; 18(4): 1064.

Viola P and Jones R. Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society conference on computer vision and pattern recognition (CVPR 2001), 2001, vol. 1, pp.1-511–I-518. New York: IEEE.

Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp.580–587.

Girshick R. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, 2015, pp.1440–1448. New York: IEEE.

He K, Gkioxari G, Dollár P, et al. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, 2017, pp.2961–2969. New York: IEEE.

Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector. In: European conference on computer vision, 2016, pp.21–37. Cham: Springer.

Lin TY, Goyal P, Girshick R, et al. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. 2017, pp.2980–2988. New York: IEEE.

Jin P, Rathod V and Zhu X. Pooling pyramid network for object detection. 2018. ArXiv preprint arXiv: 1807.03284.

Simonyan K and Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014. arXiv preprint arXiv: 1409.1556.

He K, Zhang X, Ren S, et al Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, pp.770–778. New York: IEEE.

Huang G, Liu Z, Van Der Maaten L, et al Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, pp.4700–4708. New York: IEEE.

Howard AG, Zhu M, Chen B, et al Mobile nets: efficient convolutional neural networks for mobile vision applications. 2017. ArXiv preprint arXiv: 1704.04861.
47. Iandola FN, Han S, Moskewicz MW, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size 2016. ArXiv preprint arXiv: 1602.07360.
48. Zhang X, Zhou X, Lin M, et al. Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp.6848–6856. New York: IEEE.
49. Wang RJ, Li X and Ling CX Pelee: a real-time object detection system on mobile devices. 2018. ArXiv preprint arXiv: 1804.06882.
50. Ren H, Manivannan N, Lee GC, et al. Improving OCT B-scan of interest inference performance using TensorRT based neural network optimization. Invest Ophthalmol Vis Sci 2020; 61(7): 1635–1635.
51. Yang B, Cao J, Liu X, et al. Edge computing-based real-time passenger counting using a compact convolutional neural network. Neural Comput Appl 2020; 32(9): 4919–4931.
52. Creswell A, White T, Dumoulin V, et al. Generative adversarial networks: an overview. IEEE Signal Process Mag 2018; 35(1): 53–65.
53. Wen Y, Zhang K, Li Z, et al. A discriminative feature learning approach for deep face recognition. In: European conference on computer vision, 2016, 499–515. Cham: Springer.
54. Wu Z, Zhuo Y, Li J, et al. A fast monochromatic fabric defect fast detection method based on convolutional neural network. J Comput Aided Des Comput Graph 2018; 30(12): 2262–2270.