Positive-Sample-Based Surface Defect Detection Using Memory-Augmented Adversarial Autoencoders

Tongzhi Niu, Bin Li, Weifeng Li, Yuanhong Qiu, and Shuanlong Niu

Abstract—**In industrial quality inspection, large amounts of data on the desired product appearance are available at the time of training, while significantly few defective samples are available. In this study, we proposed a new Memory-augmented Adversarial Autoencoders to detect and localize defects in real-time using defect-free samples alone for model training. This research was conducted by reconstructing images using an adversarial autoencoder and detection results from the Fréchet Markov Distance. A threshold was determined based on the statistical characteristics of defect-free samples in training set. Innovatively, we introduced a memory module and redesigned the reconstruction loss function to avoid the situation where the reconstruction ability is too strong or too poor, which lead to missing detection of defects. Then we proposed Fréchet Markov Distance, which can accurately measure the distance between the distribution of test samples and positive samples. Moreover, the statistics-based threshold determination method is used to meet different industrial needs. The accuracy, robustness, and computational overhead of the proposed MAA were evaluated using three datasets obtained from the production line and two benchmark datasets. The results indicated the effectiveness and ability of the proposed method to adapt to the real-time nature of industrial production.**

Index Terms— **automatic optical inspection, Fréchet Markov distance, memory-augmented adversarial autoencoders, surface defect detection**

I. Introduction

DEFECT inspection is the main method to achieve quality management in the industrial production process. It can not only identify substandard products in time, but also improve the production process and increase production efficiency according to the test results. The existing detection methods are mainly through manual detection, relying on human detection experience and subjective judgment, resulting in low detection efficiency, high false-positive rate and high false-negative rate. The handcrafted feature representation-base approaches rely on manual extraction of features and setting thresholds. Consequently, features extraction capability is limited and the robustness is poor. Moreover, the approaches are expensive to maintain and unable to detect the defects that have not occurred in the production process.

In recent years, deep learning has made breakthroughs in the field of image processing. Meanwhile, convolutional neural networks (CNNs) possess excellent feature extraction abilities. In addition, they have achieved the most advanced effects on tasks, such as image classification, object detection, and semantic segmentation. However, CNNs may be classified as supervised learning methods, and thus require large amounts of labeled data, which is difficult to obtain in the industrial field.

The biggest challenge of deep learning in the field of industrial quality inspection is the data collection [1]. First, the overall distribution of samples presents extremely unbalanced characteristics, and the number of defect-free samples is considerably larger than that of defective samples. Therefore, obtaining a sufficient number of defective samples with complete features is the biggest challenge. Second, the ability of a supervised learning model depends on the quality of labels. To obtain high-quality labels, labelers with an engineering background need to be employed, which is a time-consuming and high-labor-cost process. Finally, the possibility of precise detection of the edge scenes that are not covered by the training data, such as the samples that have not appeared in the previous production processes, is another problem.

The existing defect detection methods mainly include the following three methods.

1) An image is cut into patches using sliding windows. Subsequently, the patches are classified based on various CNNs, such as Resnet, and Googlenet [2][3]. These methods exhibit excellent feature extraction capabilities, robustness, and accuracy; however, they require a sufficient number of defective samples with complete features.

2) Locating and outlining defects according to advanced semantic tasks, such as U-net, Yolo, and faster-RCNN [4][5]. These methods can obtain detailed information regarding the defects; however, they require high quality labels with pixel-level segmentation.

3) Training on the defect-free data, the module is expected to produce a larger number of reconstruction errors for the defective inputs than that for the defect-free ones, which is recognized as a criterion for defect detection [6][7][8]. These methods can not only meet the requirements of industrial inspection for positive/negative classifications but also avoid the data collection problems. Because only positive samples are required for training, no defective samples are needed, labeling cost is low, and unknown defects can be detected.

The positive-sample-based methods mainly focus on the following three defect detection aspects: constructing an improved positive sample distribution model; defining an reconstruction error to measure the distance between the test sample and distribution model of the positive samples; and determining the threshold.

As for the first aspect, the existing methods typically establish a positive sample distribution model through various pretext tasks, such as generation, reconstruction, and denoising. To the best of our knowledge, generative adversarial networks (GANs) and deep autoencoder (DAE) are two positive-sample-based methods, which have been successfully applied to surface defect detection; however, they have some limitations.

DAE [6] is a method based on reconstruction. The structure of a DAE can be divided into an encoder and a decoder. As depicted in Fig. 1 (a), the encoder down-samples the input to the latent space, and the decoder up-samples the latent space coding to obtain the output. By using the L1 or L2 norm as the loss function, the input and output are as close as possible. The

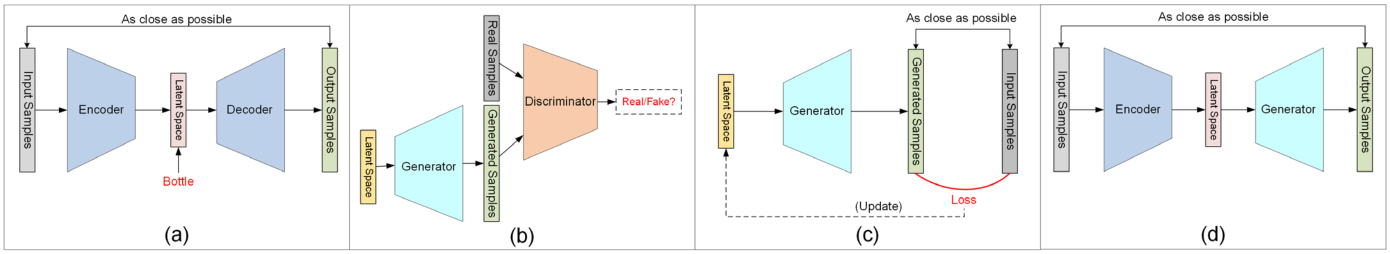


Fig. 1. Model structure of existing methods. (a) is the deep autoencoder, (b) is the training part of the generated adversarial networks, (c) is the testing part of the generated adversarial networks, (d) is the improved testing part of the generative adversarial networks.

basic principle of the autoencoder is to set the information bottleneck to filter information noise, such as defect information. When the information bottleneck is excessively small, the quality of the defect-free samples and defective sample reconstruction can easily deteriorate. In contrast, when the information bottleneck is significantly large or skip connections are introduced, both the defect-free samples and defective samples are reconstructed satisfactorily. A reconstruction ability that is excessively high or low may lead to missed detection of defective samples. We believe that the implementation of DAE may not be smooth owing to the lack of a sufficient number of defective training samples; hence, the reconstruction results are unpredictable. If some features of the defective share common latent coding with the defect-frees, the decoder will be too "strong" to decode defective coding as well. Furthermore, the loss function is designed to make the input equal to the output, which leads reconstruction ability is excessively high. This observation (Fig. 11 (d) (f)) has already been reported in the literature [9][10].

GANs [7] [31] can generate positive samples through the mutual game learning of the generative model and discriminant model during training, as depicted in Fig. 1 (b). When the adversarial training is completed, the generator learns the mapping from the latent space coding to the realistic (defect-free) images. However, GANs do not automatically yield the inverse mapping for free. Therefore, the existing methods iteratively calculate during testing to make the input samples and generated sample as close as possible, as depicted in Fig. 1 (c). However, these methods take 47 s to detect a single sample, as presented in Table IV, which is unacceptable in industrial scenarios. An improved method [8][33] train the generator and encoder separately, freezes the trained generator parameters, and uses the structure depicted in Fig. 1 (d) to update the encoder parameters. This method increases the test efficiency; however, because the encoder and generator are trained separately, it is easy to fall into the local optimum[10].

To address these challenges, we proposed a memory-augmented adversarial autoencoders (MAA). We made the following two improvements in the MAA. First, we introduced a memory module between the encoder and decoder. Given an input, the MAA does not directly feed its encoding into the decoder but uses it as a query to retrieve items in the memory. Second, we redesigned the reconstruction loss function by combining feature loss with adversarial loss. Thus, the input and output were no longer directly equal, although their features were the same. In addition, to improve the quality of the reconstructed image, an adversarial loss is introduced and coordinated training.

With regard to the second aspect, existing methods typically consider the value of the loss function as the anomaly score. The anomaly score of DAE is the L1 or L2 norm of the input and output, which is easily affected by image noise. In addition, geometric related research [11] on deep learning has proved that the distribution of images is manifold, and Euclidean distance has certain limitations. The anomaly scores of GANs often refer to the results of the discriminator [12] [36]. However, the discriminator is used to indicate whether a sample is real or fake, rather than positive or negative. Furthermore, the convergence result of GAN is to reach the Nash equilibrium [19], where the accuracy of the discriminator is approximately 50%.

Inspired by the Fréchet Inception Distance [13], we proposed the Fréchet Markov Distance (FMD). We first established a Markov random field model between the images and feature matrices, which was used to extract and characterize the local features of an image. The background of the image in defect detection task is complex and the area ratio is often large. In addition, the defects are often sparse and generally appear as local features. Determining whether there is a defect in a certain part of the image can be assumed to be related to the neighboring pixels alone. The patch of the image was considered as the site, and the probability of being determined as a defect was considered as the value of the phase space. The value of the phase space was only related to the neighboring sites. Subsequently, the Fréchet Distance (FD) [14] was used as measure between two multivariate normal distributions in the feature space. To consider global and local features simultaneously, we only considered the mean and covariance of the feature matrix. The Gaussian is the maximum entropy distribution for a given mean and covariance. Therefore, we assumed that the feature matrix followed a multidimensional Gaussian. The difference between the two Gaussians was measured using the Fréchet distance.

Finally, the threshold is a critical parameter in distinguishing between the defective and defect-free samples. A commonly used method is to establish a verification set and determine the threshold based on the test results of the verification set. However, the overall distribution of the samples presents extremely unbalanced characteristics. In addition, the number of negative samples in the verification set is limited, ultimately leading to poor robustness of the threshold.

Based on the positive samples in the training set, we established a statistical model to determine the threshold. In industrial production, there is almost always a significant amount of defect-free data while defective data are relatively small negligible (1-5% for the worst modern automated production lines). As the amount of data used increases, the threshold generalization performance improves. Furthermore, qualified production lines generally have a better process capability index (CPK), which implies that the training set has a satisfactory consistency. Therefore, it is reasonable to assume that the number of samples is sufficient and only random errors exist. According to the

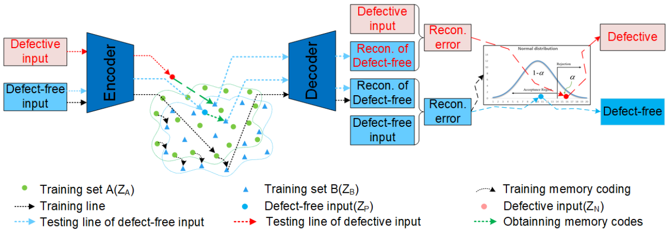


Fig. 2. Defect detection via the proposed MAA.

Pauta criterion, we established a statistical model of reconstruction error in defect-free samples. Finally, how to selected appropriate parameters to meet the needs of different scenarios is discussed.

The schematic illustration of the proposed method is shown in Fig. 2, which randomly dividing a training set with only defect-free samples into set A and set B. During training, the networks reconstruct the samples in set A into the samples in set B using MAA. Then, the reconstruction error of the defect-free samples in the training set is obtained by calculating FMD, and a statistical model is established to determine the threshold. During testing, given a defective input, MAA can transform the defective samples into defect-free samples in the memory module for reconstruction, resulting in an output significantly different from the defective input. Subsequently, MAA calculates the FMD between the input and output to obtain the reconstruction errors, and determine whether it conforms to the statistical distribution according to the threshold.

We compared seven of the most advanced methods (including Cognex software VIDI [42] and six state-of-art methods) using five datasets (three sets are from industrial sites and two sets are benchmark datasets).

In summary, we proposed a new Memory-augmented Adversarial Autoencoders to detect and localize defects for real-time with only defect-free samples for model training. The contributions of this paper are as following.

1) An MAA was proposed to avoid the situation where the reconstruction ability is too strong or too poor, which lead to missing detection of defects. We introduced an automatic memory mechanism between the encoder and decoder, and redesigned the reconstruction loss by combining feature loss and adversarial loss.
2) FMD was proposed to calculate the reconstruction error accurately. A Markov random field was established between the image and the feature matrix to obtain the local features of the image. Subsequently, the Fréchet distance was used as a measure between the two feature matrices of the input and output.
3) A statistic-based method for determining thresholds was proposed to adopt the actual needs of the industry. And we discussed how to select hyperparameters.

The remaining part of this paper is organized as follows. The studies related to this work are discussed in Section II. Then, in section III, we will describe the procedures of the proposed MAA model in detail. Then in order to analyze the model and compare the overall performance, experimental results are presented in section IV. Finally, we will make a conclusion of the paper in section V.

II. Related Works

Our proposed method is an anomaly detection method, which attempts to obtain the identification of the defective class among all objects by primarily learning from a training set containing only the objects of the defect-free class. DAE, GANs, Markov random field, and memory networks were applied to the proposed MAA model. This section briefly reviews the development of DAE, GANs, and memory networks. Then, latest research on the anomaly detection method is discussed.

A. Development of DAE, GANs, and Memory Networks

DAE [15] is proposed as a type of neural network structure, which includes an encoder and a decoder. An autoencoder is used to learn a representation (encoding) in an unsupervised manner for a set of data. Typically, an autoencoder is used for dimensionality reduction, by training the network to ignore signal “noise”. To achieve better representation, a spare autoencoder [16] constrains the sparsity of hidden units and a denoising autoencoder [17] changes the reconstruction criterion. Unlike classical autoencoders (spare, denoising, etc.), variational autoencoders are considered as generative models and use a variational approach for latent representation learning [18].

GANs are generative models invented by Goodfellow in 2014 [19]. The concept of the zero-sum game was introduced, where in a generator generates an image and a discriminator determines whether the generated image is real or fake. The generator resembles a decoder-like network that learns the distribution of the input data from a latent space. The discriminator network typically possesses a classical classification architecture, which implies that it can determine the validity after reading an input image. GANs have attracted widespread attention owing to their potential practical applications in the future. Deep convolutional GAN (DCGAN) was proposed by Radford [20], who introduced a fully convolutional generative network by removing the fully connected layers, and introducing convolutional layers and batch-normalization in the network. GANs with inference have been also used within image style transfer. For instance, this research [21] presents an approach for translating an image from domain to domain in the absence of paired examples.

Memory-augmented networks were first proposed by Google Brain in 2014 [22]. They used external memory to extend the capability of neural networks, in which content-based attention was used for addressing the memory. These networks are analogous to a Turing Machine or Von Neumann architecture; however, they are differentiable end-to-end, allowing them to be efficiently trained with gradient decent. The main idea to combine the successful learning strategies developed in the machine learning literature for inference with a memory component that can be read and written to, was proposed by Facebook AI Research [23]. Furthermore, memory-augmented networks have attracted significant interest for solving various problems, such as one-shot learning [24] and multi-modal data generation [25], [26].

B. Latest Research of Anomaly Detection Method

Anomaly detection is the identification of rare items, events, or observations, which raises suspicions by differing significantly from the majority of the data[27]. It exactly fits the characteristics of industrial images. Defective images correspond to rare items, events, or observations, while defect-free images

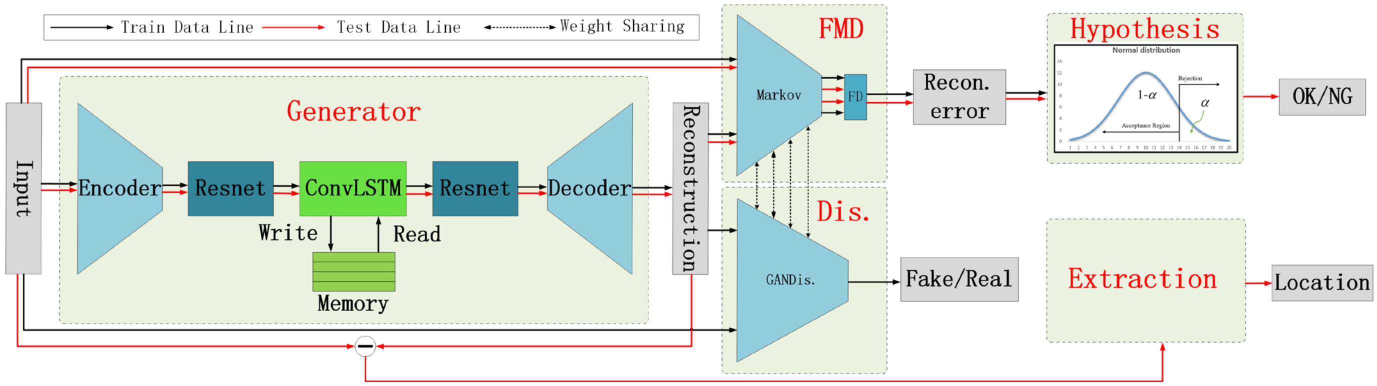


Fig. 3. Overview of the proposed structure for Memory-augmented Adversarial Autoencoders.

correspond to the majority of the data. Therefore, defect detection is the identification of a defective image from defect-free images.

In unsupervised anomaly detection, only normal samples can be used as the training set. Consequently, a natural choice to solve this problem would be a one-class classification method, such as the one-class support vector machine (OC-SVM) [28]. The objective of OC-SVM is to determine a maximum margin hyperplane in a feature space, which best separates the mapped data from the origin. Support vector data description (SVDD) [29] is a technique to identify the smallest hypersphere that encloses the majority of the data in a feature space. These methods often perform poorly when processing high-dimensional data, such as images.

The main approach use in anomaly detection is the use of a DAE based on the assumption that a model trained by defect-free data alone cannot correctly represent and reconstruct defects. For example, deep structured energy-based models (DSEBMs) [30] were proposed to address the anomaly detection problem by directly modeling the data distribution with deep architectures. Deep autoencoding Gaussian mixture models (DAGMMs) [10] utilize a DAE to generate a low-dimensional representation and reconstruction error for each input data point, which is further fed into a Gaussian mixture model. Although the reconstruction-based methods have achieved fruitful results, their performances are limited by the under-designed representation of the latent space.

Because of the successful application of GANs in modeling such complex and high-dimensional distributions, increasing number of researchers have attempted to implement anomaly detection using GAN-based methods. AnoGAN [31] was the first GAN-based anomaly detection method proposed for capturing imaging markers relevant to disease progression and treatment monitoring. Later, Efficient-GAN [32] and f-AnoGAN [33] were proposed to enable AnoGAN to run faster at test time. Furthermore, GANomaly [34] not only minimizes the distance of input and reconstructs like other methods but also constrains the latent vectors, resulting in improved performance. Moreover, CVAE-GAN [35] regularizes the latent space by fitting multiple Gaussian distributions and performing active negative training. ALOCC [36] changes the training strategy to focus on denoising, rather than reconstruction. They think that the separability of the enhanced inliers and distorted outliers by denoising is considerably better than deciding on the original samples. Although the GAN-based anomaly detection method is successful, our specific application in the field of defect detection requires corresponding improvements based on the image characteristics.

III. Proposed Approach

A. Overview of Proposed MAA

The proposed MAA model comprises five major components – a generator (for achieving reconstruction), a discriminator (for obtaining high quality reconstruction samples), FMD (for obtaining accuracy reconstruction error), hypothesis testing (for obtaining the threshold with training samples), and extraction (for localizing defects). As depicted in Fig. 3, during training, the encoder first obtains the encoding of the input. Subsequently, deeper features of the encoding are extracted through the first Resnet blocks. By utilizing ConvLSTM [37], the memory preserves historic information and learns the group characteristics of the defect-free samples. Then, the second Resnet block and decoder obtains the memory-coding. After obtaining the reconstruction of the input, both the reconstruction and input are introduced in the discriminator network together. On the one hand, the Markov encoder is responsible for dealing with the feature loss, where it is minimized. On the other hand, the adversarial loss is optimized in the GANs discriminator. Given a testing sample, the generator transforms it into a defect-free sample. As a result, the reconstruction tends to be close to a defect-free sample, resulting in small reconstruction errors in defect-free samples and large reconstruction errors in defective samples, which serves as the criterion for defect detection.

Subsequently, the reconstruction errors are calculated using FMD. A Markov random field is established between the image and feature matrix to obtain the local features of the image. Then, Fréchet distance [14] is used as a measure between two multivariate normal distributions in the feature space.

Furthermore, a statistical model of the reconstruction errors of the training set samples is established based on the Pauta criterion. The parameters are set by considering the highest defective samples recall rate as the target, and the threshold is calculated.

Finally, to identify the type of defect and improve the production process, further information is often required. Therefore, we compare the input with the reconstruction to locate the defect.

B. Generator Networks

We introduced an automatic memory mechanism in the autoencoder and redesigned the loss function and training strategy.

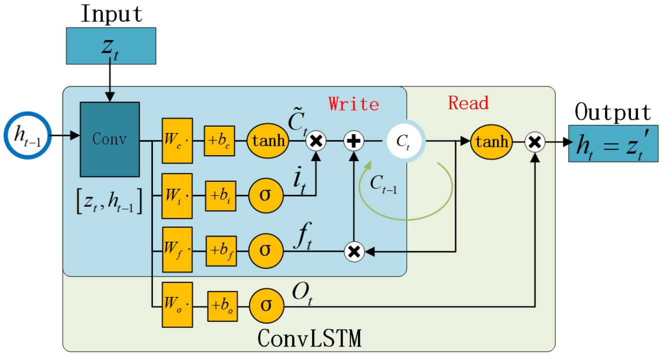


Fig. 4. Memory Block Architecture. See table I for specific parameters.

The generative network is made up of the encoder, decoder, memory module and Resnet block.

1) Encoder and Decoder

The encoder is used to represent the input in an informative latent domain. In our model, the encoded representation is recoded by the memory module and used as an input to the decoder. The decoder is trained to reconstruct the samples using the memory-coding.

We designed the encoder and decoder to be symmetrical. In the first layer of convolution of the encoder, a large convolution kernel is used for a large receptive field. A large amount of information leads to better, characteristics; however, a larger number of parameters are also required. Therefore, we refer to inceptionV3 [38], using a 1×7 convolution kernel and a 7×1 convolution kernel to reduce the number of parameters required and improve the accuracy. Dimension reduction uses 3×3 convolution kernels. Similarly, the decoder uses 3×3 convolution kernels and the last layer uses a large convolution kernel similar to the encoder.

2) Memory Module

We introduced the concept of a memory module. By using the encoded representation as a query, the memory-coding is obtained through the secondary encoding of the memory module, which is then input to the decoder. The proposed memory module consists of two major components: a memory for storing the training set sample features, and a ConvLSTM for writing and reading the memory according to a certain mechanism.

Practically, the characteristics of defect-free samples are group-specific, context-sensitive, and relatively single, and their appearances are basically the same. Conversely, the defects of defective samples show various features. In an actual production process, factors, such as tool wear and environmental changes, may cause context-related changes in a sample. Therefore, we adopted the method of RNN and designed our memory module with reference to ConvLSTM. As depicted in Fig. 4, the memory module contains a hidden-state (short-term memory), cell-state (long-term memory), forget gate, input gate, output gate, and convolutional layer. The model is represented in equations (1)-(6). In the writing phase, the input gate processes the current encoding (as shown in (1)) and updates it to the next cell-state with a certain probability (as shown in (2)). The forget gate forgets the current cell-state with a certain probability (as shown in (3)). It then obtains a new cell-state (as shown in (4)). In the reading phase, the current hidden-state and input code gains the next hidden-state (as shown in (5), (6)). The memory-coding and next hidden-state share the same value.

TABLE I
MEMORY-AUGMENTED ADVERSARIAL AUTOENCODERS ARCHITECTURE

Operation	Kernel	Strides	Padding	Features Maps/Units	BN?	Non-Linearity
Encoder						Input:x=[B,3,256,256]
Conv	1×7	1×1	0×3	3,16	√	ReLU
Conv	7×1	1×1	3×0	16,32		
Conv	3×3	2×2	1	32,64	√	ReLU
Conv	3×3	2×2	1	64,128	√	ReLU
Conv	3×3	2×2	1	128,256	√	ReLU
Resnetblock1, Resnetblock2, Resnetblock3						
Auto-Memory Input:x=[B,256,32,32],h=[B,256,32,32],c=[B,256,32,32]						
Conv	3×3	1×1	1	512,1024	×	Linear
Split				1024,4×256	×	Linear
Resnetblock4, Resnetblock5, Resnetblock6						
Decoder						Input:x=[B,256,32,32]
T.cv	3×3	2×2	1	256,128	√	ReLU
T.cv	3×3	2×2	1	128,64	√	ReLU
T.cv	3×3	2×2	1	64,32	√	ReLU
Conv	7×1	1×1	3×0	32,16		
Conv	1×7	1×1	0×3	16,3	√	Tanh
Discriminator						Input:x=[B,3,256,256]
Conv	4×4	2×2	1	3,64	×	LeakyReLU
Conv	4×4	2×2	1	64,128	√	LeakyReLU
Conv	4×4	2×2	1	128,256	√	LeakyReLU
Conv	4×4	1×1	1	256,512	√	LeakyReLU
Conv	4×4	1×1	1	512,1	×	Linear
Conv	4×4	1×1	1	1,1	×	Linear
Others:						
Optimizer				Adam(α=10e-5, β=0.5)		
Batch Size(B)				32		
Latent Dimension				64×64		
LeakyReLU Slope				0.1		
Weight & Bias initialization				Isotropic Gaussian(μ=0,σ=0.02),constant(0)		

[Conv] Convolution, [T.cv] Transposed Convolution, [x] Input Tensor, [BN] Batch Normalization, [h] Hidden-state, [c] Cell-state, [B] Batch size

Input gate: input hidden feature information. $\tilde{C}_t$ is the input hidden feature information. i_t is the probability of controlling whether to input the hidden feature information.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, z_t] + b_C) \tag{1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, z_t]) + b_i) \tag{2}$$

Forget gate: f_t is the probability of forgetting the existing memory cell-state.

$$f_t = \sigma(W_f \cdot [h_{t-1}, z_t]) + b_f) \tag{3}$$

Memory cell status update: the forget gate and input gate work together to update the state of memory cells.

$$C_t = f_t \times c_{t-1} + i_t \times \tilde{C}_t \tag{4}$$

Output gate: output the next state. o_t is the probability of controlling whether to output the current memory cell state. h_t is the output.

$$o_t = \sigma(W_o[h_{t-1}, z_t] + b_o) \tag{5}$$

$$h_t = o_t \times \tanh(C_t) \tag{6}$$

In equations (1)-(6), the coding obtained by the encoder is represented by $z_i,\ldots,z_t$, memory-coding is represented by

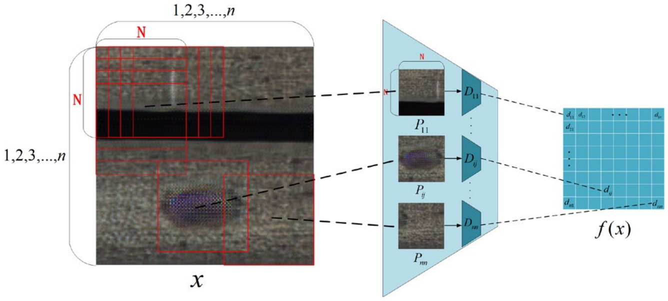


Fig. 5. Diagram of the Markov discriminator. See table I for specific parameters.

$z_i', \ldots, z_t'$, cell-state is represented by $C_i, \ldots, C_t$, and hidden-state is represented by $h_i, \ldots, h_t$. Furthermore, the gates i_t, f_t, and o_t are all 3D tensors, σ is the sigmoid function, tanh is the tanh function. In addition, W_c, W_i, W_f, and W_o are 2D convolution kernels. The architecture is illustrated in Fig. 3 with its composition further elaborated in Table I.

3) Resnet Block

We added the Resnet block between the encoder, decoder and memory module. To the best of our knowledge, the depth of the feature representation is critical in visual recognition tasks and Resnet can increase the network depth and avoid degradation problems.

C. Discriminator Networks

The discriminator is used for the generator to form a GAN [11]. By reaching the Nash equilibrium through the zero-sum game, the reconstructed samples of the generator are close to the real samples. The design of the network structure is depicted in Table I.

D.FMD

The FMD includes a Markov encoder sharing weights with the discriminator and FD. The design of the Markov encoder is illustrated in Fig. 5. The input is represented by x, and the local feature matrix is represented by $f(x)$. The red sliding window splits the input into a patch of size $N \times N$, which is represented by $P_{ij}, \ldots, P_{nm}$, and the value of the corresponding feature matrix is represented by $d_{ij}, \ldots, d_{nm}$. Furthermore, P_{ij} is the position, and d_{ij} is the phase space. The value for each position of an image is given assigned in a phase space according to the CNN map. The value of each position is related to the pixels alone.

After obtaining the feature matrix, where $f(x_i)$ represents the feature matrix of input and $f(x_r)$ represents the feature matrix of reconstruction, we employed FD to calculate reconstruction error, as follows:

$$FMD = \left|\mu_i - \mu_r\right|^2 + Tr(\Sigma_i + \Sigma_r - 2(\Sigma_i \Sigma_r)^{1/2}) \qquad (7)$$

where μ_i, μ_r and Σ_i, Σ_r are the respective means and covariance matrices of $f(x_i)$ and $f(x_r)$, and the positive square root is taken. The above formula holds in particular when $f(x_i)$ and $f(x_r)$ are normal distributions on R^n [14].

Algorithm 1: Minibatch stochastic gradient descent training of memory-augmented adversarial autoencoders. The number of steps to apply to the discriminator, k is a hyperparameter. We used k = 1, the least expensive option, in our experiments.

```
For number of training iterations do
    For k steps do
        Sample minibatch of $m$ samples $\{x_a^{(1)},\ldots,x_a^{(m)}\}$ from data of train
set A distribution $p_{data}(x_A)$ .
        Sample minibatch of $m$ samples $\{x_b^{(1)},\ldots,x_b^{(m)}\}$ from data of train
set B distribution $p_{data}(x_B)$ .
        Update the discriminator by ascending its stochastic gradient:
            $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D(x_b^{(i)}) + \log(1 - D(G(x_a^{(i)})))\right]$
    End for
        Sample minibatch of $m$ samples $\{x_a^{(1)},\ldots,x_a^{(m)}\}$ from data of train
set A distribution $p_{data}(x_A)$ .
        Update the generator by ascending its stochastic gradient:
            $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \left[\log(1 - D(G(x_a^{(i)}))) + \left|f(x_b^{(i)}) - f(G(x_a^{(i)}))\right|_1\right]$
End for
The gradient-based updates can use any standard gradient-based learning rule
[39]. We used momentum in our experiments.
```

E. Training Mechanism

First, the samples are resized to 256×256 pixels and the pixel values are normalized by scaling between 0 and 1. Subsequently, the samples are classified into defective and defect-free samples, and seven out of ten of the defect-free samples are considered as the training set. The remaining samples are used as the testing set. Finally, we randomly divide the training set into training set A and training set B.

As shown in Algorithm 1, samples of the training set A are represented by x_a, and samples of the training set B are represented by x_b . We input sample x_a into the generator to obtain reconstructed samples $G(x_a)$. Then, samples x_b and $G(x_a)$ are input into the Markov encoder to acquire $f(x_b)$ and $f(G(x_a))$, and input into the GAN discriminator to obtain $D(x_b)$ and $D(G(x_a))$.

Initially, we propose a local feature consistency loss as follows:

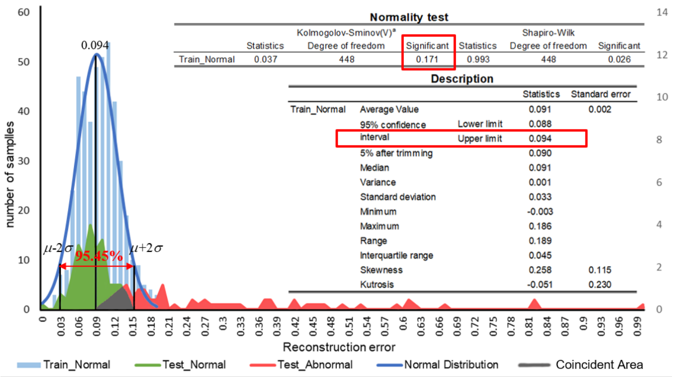
$$L_{con} = \left|f(x_b) - f(G(x_a))\right|_1 \qquad (8)$$

In comparison with the original loss function, there are two advantages.

1) The local features extracted by the Markov encoder, are used in our model, whereas other autoencoders typically determine the reconstruction error through direct pixel subtraction.

2) Instead of minimizing the reconstruction loss of the input and reconstruction, we minimize the reconstruction loss of a sample in B and that of a sample in A. Because A and B are both defect-free samples, the local features can be considered to be identical.

To make the reconstructed samples as realistic as possible and establish an improved generation model of defect-free samples, we refer to the most advanced generation model, GAN, and design the following loss function:



Normality test

	Kolmogolov-Sminov(V)ª			Shapiro-Wilk		
	Statistics	Degree of freedom	Significant	Statistics	Degree of freedom	Significant
Train_Normal	0.037	448	0.171	0.993	448	0.026

Description

			Statistics	Standard error
Train_Normal	Average Value		0.091	0.002
	95% confidence interval	Lower limit	0.088	
		Upper limit	0.094	
	5% after trimming		0.090	
	Median		0.091	
	Variance		0.001	
	Standard deviation		0.033	
	Minimum		-0.003	
	Maximum		0.186	
	Range		0.189	
	Interquartile range		0.045	
	Skewness		0.258	0.115
	Kutrosis		-0.051	0.230

Fig. 6. Example of statistical model decision threshold.

$$L_{adv} = \underset{x\sim p_x}{\mathrm{E}}\left[\log D(x_b)\right] + \underset{x\sim p_x}{\mathrm{E}}\left[\log(1-D(G(x_a))\right] \quad (9)$$

We obtain the final loss function:

$$L = L_{con} + \lambda L_{adv} \quad (10)$$

where $\lambda > 0$ is a trade-off hyper parameter that controls the relative importance of the two terms.

F. Threshold Determines by Statistical Model

To establish a statistical model of positive samples, we make the following three assumptions. First, because the training set and test set are randomly divided, it can assumed that their samples have identical distribution. Second, there are significantly more defect-free samples than defective samples. Thus, it can be assumed that there are sufficient number of samples in the training set. Third, an acceptable process capability coefficient corresponds to an acceptable consistency. Therefore, it can be assumed that there exist random errors between the reconstruction errors of the training set. The first hypothesis guarantees that the training set can be used to predict the test set. If the latter two assumptions are true, the reconstruction error of the training set samples conforms to the normal distribution. According to the Pauta criterion, we build the following model.

We Assume that $\xi = \left\{\vartheta(x_k) \mid k = 1, 2, ..., N\right\}$ refers to the reconstruction residual set of images of defect-free samples in the training set, where N is the number of the samples, and $\vartheta(x)$ is the reconstruction error of x. The threshold can be defined as $T = \mu + \gamma \cdot \sigma$, where μ, σ are the mean and standard deviation of set ξ. The parameter γ can be adjusted according to the discriminate sensitivity. We consider one of the data sets as an example, as depicted in Fig. 6. In the normal test, the significance value is 0.171, which is greater than 0.05. Apparently, it obeys the normal distribution. At 95% confidence level, the upper limit of the mean is 0.094, and the standard deviation is 0.033. When $\gamma = 2$, the recall rate of the defect-free samples is 100%, and when $\gamma = 0$, the recall rate of the defective samples is 100%. In the case of an industrial defect detection, the recall rate of defective data is required to be high, therefore, $\gamma = 0$.

G.Extraction Location InformationBy inputting a sample, we can obtain the corresponding defect-free sample, determine the local variance of the residual map, and obtain the defect location. The process can be divided into the following

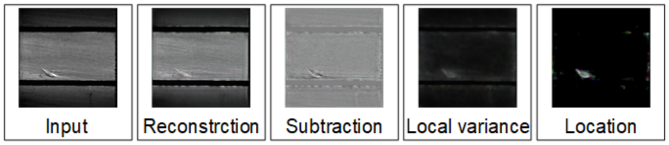


Fig. 7. Example of extraction location information

three steps, as depicted in Fig. 7.

First, the reconstruction and input are directly subtracted to obtain the subtraction.

Second, because the texture, color, gray and other features of the normal area are reconstructed well, the characteristics of the normal area in the subtraction are balanced and uniform, whereas the characteristics of the abnormal area are unbalanced and non-uniform. The local variance of the subtraction is salved, it can be found that the abnormal area has been clearly distinguished.

Finally, to enhance the display effect, we obtain the location through image self-multiplication and the threshold method.

IV. Evaluation Of MAA Model

The proposed MAA model, was evaluated using five datasets, as depicted in Fig. 8, namely, three datasets obtained from the motor commutator production site, (MCSD-g, MCSD-c, and MCSD-h) and two benchmark datasets, (RSDDs [40], and KSDD [41]). Through this empirical evaluation, we demonstrated the capability of the proposed MAA to detect and localize defects in real-time. In addition, the performance of the introduced MAA model was equivalent to that of the state-of-the-art anomaly detection methods proposed in the literature. MAA was implemented in Python3.5 with the Pytorch0.4 framework to enhance the deep learning and GPU utilization capabilities. In addition, MAA was trained on a high-performance server, 40-core CPU 2.4 GHz with 256 GB memory and dual NVIDIA Tesla P100 of 32 GB GPU units. The evaluation of MAA was conducted on a typical personal computer configuration, 6-core CPU 3.6 GHz with 16 GB memory and GPU of NVIDIA GeForce GTX 1060 of 6 GB GPU, to ensure that the proposed MAA model could be realistically deployed in an industrial setting.

A. Datasets

As depicted in Fig. 9, the motor commutator surface defect dataset (MCSD) was shot on a production line using a fixed camera under a fixed light source. We collected images of the groove, cylinder, and hook of the motor commutator. The groove dataset (MCSD-g) included 1458 defect-free samples, 373 defective samples and 5 known types of defects. The cylinder dataset (MCSD-c) included 323 defect-free samples, 66 defective samples and 15 known types of defects. The hook dataset (MCSD-h) included 1094 defect-free samples, 159 defective samples and 7 known types of defects. These samples have complex backgrounds and various types of defects, which makes the defection difficult.

The RSDDs [40] datasets could be classified into Type-I and Type-II datasets. The former was captured from express rails, and included 67 images, whereas the latter was captured from common/heavy haul rails, and included 128 images. In the case of a large-facility inspection, a practical rail line covering 45 km of continuous rail surface images was adopted in RSDDs.

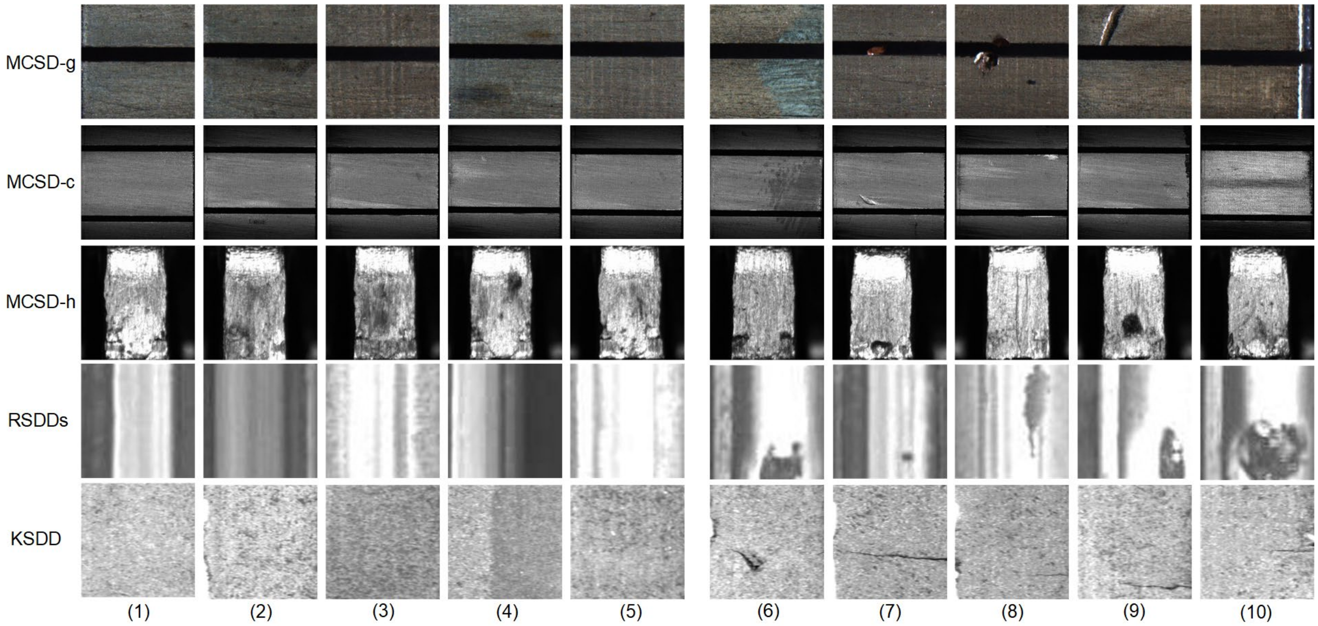


Fig. 8. Images of the four datasets. (1), (2), (3), (4) and (5) represent defect-free samples, whereas (6), (7), (8), (9) and (10) represent defective samples.

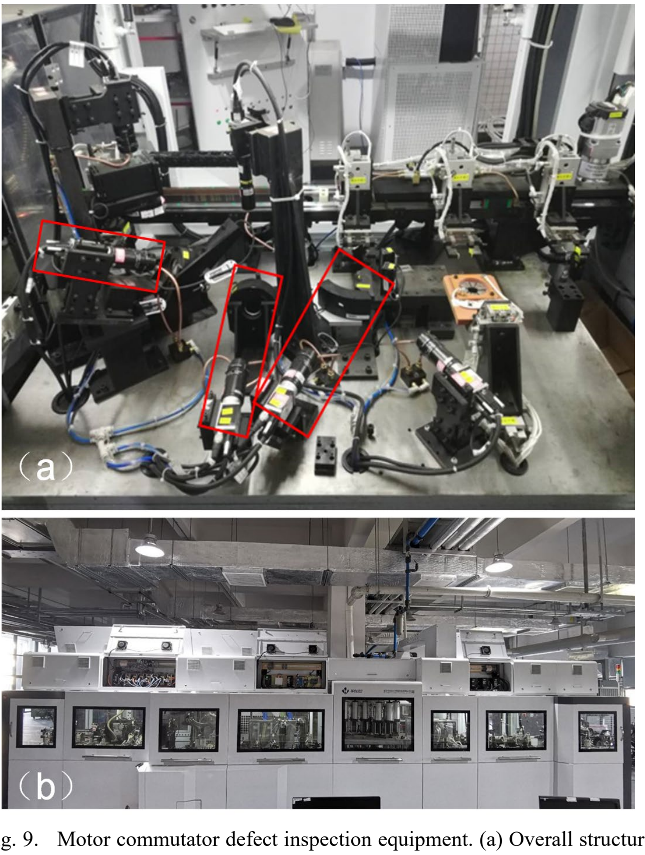


Fig. 9. Motor commutator defect inspection equipment. (a) Overall structure design of inspection table, the cameras marked with the red boxes capture the images; from left to right: MCSD-h, MCSD-c, and MCSD-g. (b) Motor commutator production line.

We used the second type of data set and cropped the track image into small square blocks. This resulted in a total of 2794 images, including 2538 defect-free samples and 256 defective samples.

KSDD [41] comprised 50 defective electrical commutators, each with up to eight relevant surfaces. This resulted in a total of 399 images, including 347 defect-free samples and 54 defective samples.

B. Experimental Setup

The experimental setup covered three main aspects as follows:

1) Detection capabilities of the proposed MAA model was evaluated and compared with the state-of-the-art anomaly detection models on five datasets.
2) The localization capability was evaluated and compared with three methods in the literature on five datasets.
3) Three different distance measurement methods were evaluated and compared in five datasets.
4) A run-time analysis of our approach was conducted to verify the real-time processing capabilities of the proposed MAA.

In this experiment, we trained the learning model at the learning rate of 10e-5 and Adams was to optimize the MAA. More specific parameters are listed in Table I.

C. Defect Detection Results

Defect detection was evaluated using six state-of-the-art deep learning-based approaches and VISIONPRO VIDI [42], which is a deep learning-based software for industrial image analysis. The selected deep learning-based approaches for comparison purposes, mentioned in section I and II are as follows.

The first is the three methods used in industrial image detection. The first method is DAE [6], as depicted in Fig. 2 (1). For actual comparison, we added DAE with skip connection, known as DAE (skip). The second method [7] is based on GAN, as depicted in Figure2 (2)-(3), which is similar to AnoGAN [31]. The third method [8] is to improve the detection efficiency based on GAN, depicted in Figure 2 (4), which is similar to Efficient-GAN [32] and f-AnoGAN [33]. To further prove the advantage of our method, we compared our experimental results with those of other advanced methods, namely GANomaly [34], CVAE-GAN [35] and ALOCC [36].

We compared the results of the respective models using the ROC curves, corresponding area under the curve (AUC), accuracy, and recall of defective samples.

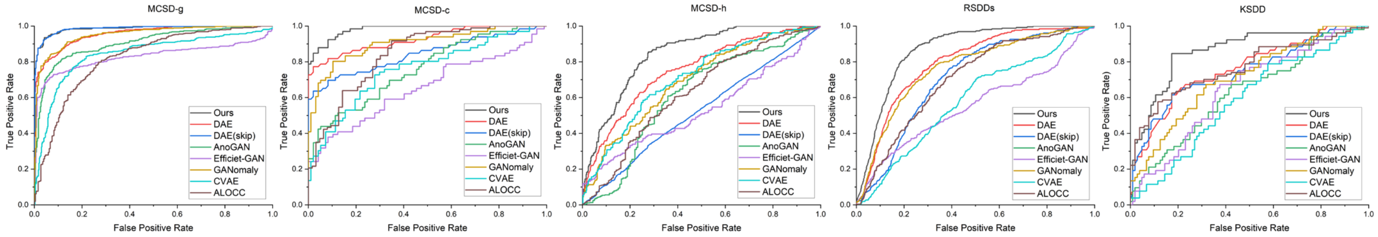


Fig. 10. ROC curves of anomaly detection results; from left to right: the results of MCSD-g, MCSD-c, MCSD-h, RSDDs, and KSDD datasets.

TABLE II
Result Of Defects Detection

		Ours	DAE	DAE(skip)	AnoGAN	Efficient-GAN	GANomaly	CVAE-GAN	ALOCC	VIDI
MCSD-g	AUC	0.980	0.913	**0.985**	0.905	0.829	0.953	0.843	0.822	0.940
	Accuracy	0.832	0.837	**0.947**	0.855	0.836	0.892	0.804	0.774	0.855
	Recall	**0.983**	0.933	0.922	0.818	0.729	0.842	0.831	0.730	0.926
MCSD-c	AUC	**0.977**	0.906	0.793	0.761	0.655	0.903	0.757	0.840	0.970
	Accuracy	0.855	0.865	0.785	0.730	0.699	0.854	0.724	0.813	**0.926**
	Recall	**1.000**	0.773	0.591	0.515	0.379	0.803	0.727	0.651	0.929
MCSD-h	AUC	**0.844**	0.720	0.756	0.615	0.534	0.694	0.711	0.629	0.470
	Accuracy	0.663	**0.748**	**0.748**	0.679	0.731	0.694	0.692	0.698	0.407
	Recall	**0.918**	0.634	0.629	0.535	0.226	0.648	0.446	0.432	0.603
RSDDs	AUC	**0.941**	0.829	0.703	0.703	0.562	0.792	0.535	0.723	0.880
	Accuracy	0.815	0.772	0.624	0.644	0.723	0.777	0.513	0.661	**0.801**
	Recall	**0.934**	0.750	0.824	0.751	0.258	0.688	0.711	0.754	0.739
KSDD	AUC	**0.857**	0.711	0.699	0.630	0.645	0.711	0.586	0.768	0.750
	Accuracy	**0.731**	0.679	0.718	0.609	0.641	0.679	0.545	0.679	0.679
	Recall	**0.904**	0.673	0.558	0.673	0.647	0.673	0.788	0.885	0.692

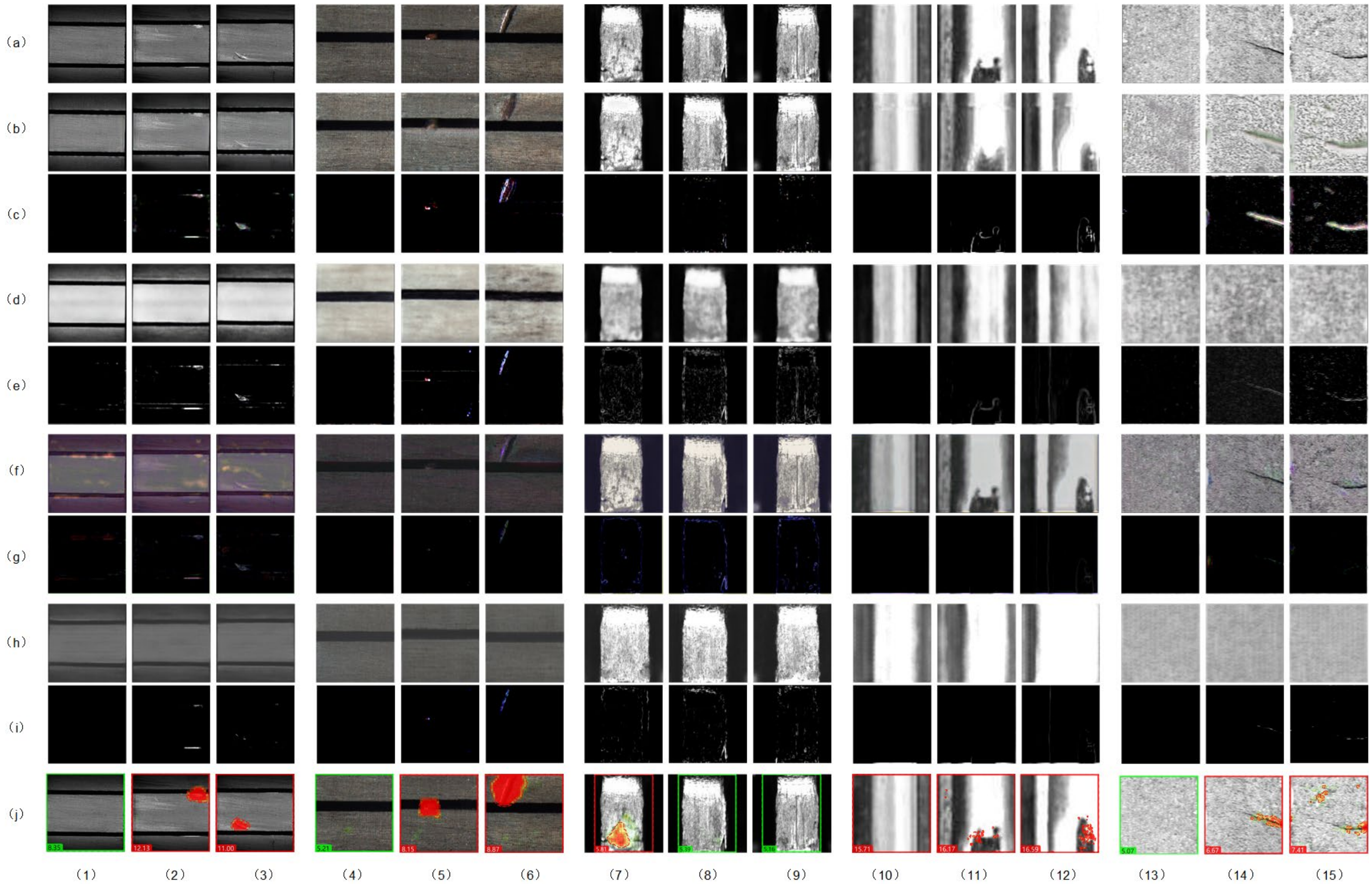


Fig. 11. Results of defect location. Row (a) represents the original samples, (b) and (c) represents the reconstructed and location images of our method, (d) and (e) represents the results of DAE, (f) and (g) represents the results of DAE(skip), (f) and (g) are the results AnoGAN, (h) represents the result of ViDi. Columns (1), (2) and (3) represents the MCSD-g datasets, where (1) represents the defect-free samples, and (2) and (3) represents the defective samples. (4), (5), and (6) represents the MCSD-c datasets, (7), (8) and (9) represents the MCSD-h datasets, (10), (11), and (12) represents the RSDDs datasets, (13), (14), and (15) represents the KSDD datasets.

TABLE III
RESUALT OF DIFFERENT DISTANCE MEAUREMENT

	L1 Normal	Markov+L1 Normal	FMD
MCSD-g	0.649	**0.986**	0.980
MCSD-c	0.732	**0.982**	0.977
MCSD-h	0.623	0.843	**0.844**
RSDDs	0.603	0.909	**0.941**
KSDD	0.606	0.779	**0.857**

The ROC curve is illustrated in Fig. 10, and the AUC is presented in Table II. Our method outperforms all other methods on the five datasets. The highest AUC value implies that our method has the best classification ability in comparison with those of the other methods, which confirms that the proposed network structure of the MAA and the distance measurement method of FMD perform well in this type of task.

If it is the same threshold determination method, an improved classification ability implies that our method must theoretically achieve a higher accuracy. However, as can be seen in Table II, the accuracy obtained by our method in four datasets was lower than those obtained by the other methods. However, the recall rates achieved by our method were higher than those of the other methods in majority of the datasets. In particular, in the MCSD-t and MCSD-c datasets, the recall rates were higher than 99.7%, with an accuracy rate of not less than 80%, which is consistent with the industrial-level defect detection accuracy. In fact, this experimental phenomenon is consistent with our threshold determination strategy. As a result, at the cost of lower accuracy, a higher recall rate was obtained.

Furthermore, the performance of our method was better than that of ViDi, implying that our method has a higher practical application value.

D.Localization Results

The qualitative analysis of defect localization is depicted in Fig. 11. The defect localization was evaluated using AnoGAN [31], GANomaly [34] and ViDi [42] methods. on five datasets, namely, MSCD-g, MCSD-c, MCSD-h, RSDDs and KSDD.

According to Fig. 10, in comparison with the original image, the reconstruction effect of our method was better than those of the other methods in generating defect-free regions; And our method was inferior in defect region generation. The defect location obtained by residual is clear. And compared with AnoGAN and DAE (skip), our method is clearer in the division of anomalous regions. In addition, our method results in smaller noise in comparison with DAE, and more accurate positioning in comparison with ViDi. In summary, our method can be used to localize the anomalous area to a certain extent as a reference for defect detection.

E. Comparison of distance measurement methods

A distance measurement experiment was added to the revised manuscript. Under the same network structure, we replaced the distance measurement method, and got the table III. It can be found that the result is the worst when only the L1 distance is used, which proves that the Euclidean space distance measurement is often invalid. The AUC value is greatly improved when Markov+L1 distance is used, which proves the effectiveness of the Markov random field. Finally, FMD has been further improved compared to Markov+L1, which proves that Fréchet Distance is more accurate.

TABLE IV
REAL-TIME ANALYSIS

	SPEED(MS)	FPS	MEMORY(MB)
Ours	133.92/4	33.48	4729/4
AnoGAN	47105.79/1	0.02	1912/1
DAE	28.73/4	139.23	3428/4
Efficient-GAN	17.29/4	231.35	1476/4
GANomaly	41.61/4	96.13	5163/4
CVAE-GAN	\	\	\
ALOCC	99.86/4	40.06	4497/4

F. Run-Time Analysis Results

To simulate the production line environment more realistically, we conducted a real-time testing on a personal computer to ensure that it can meet the industrial production cycle requirements. We used an NVIDIA GeForce GTX 1060 graphics card for parallel calculations. Because of the limited memory, four samples were evaluated simultaneously. The test results are presented in Table IV. Achieving approximately 33 FPS, our method demonstrated its ability for defect detection in real-time. Moreover, CVAE-GAN occupies more memory than the maximum memory of the graphics card.

V.CONCLUSION

In this article, we proposed an innovative anomaly detection approach for surface defects detection using memory-augmented adversarial autoencoders, which can detect and localize defects for real-time with only defect-free samples for training.

The results of the experiments conducted on two benchmark datasets and three datasets collected from the production line prove the accuracy, robustness, and low computational overheads of the proposed approach, confirming its applicability in the industrial field. In terms of surface defect detection, the proposed MAA requires only defect-free samples as the training set, which saves the cost of data collection and marking. In addition, it provides an easy to train algorithm model capable of end-to-end detection. Furthermore, it has a high recall rate and can meet the industrial production cycle requirements.

We will combine the residuals of the feature matrix with those of the image to localize the defect and outline the defect contour more accurately in our future works.

REFERENCES

[1] Ding, Han, et al. "State of AI-Based Monitoring in Smart Manufacturing and Introduction to Focused Section." IEEE/ASME Transactions on Mechatronics 25.5 (2020): 2143-2154.

[2] Wang, Tian, et al. "A fast and robust convolutional neural network-based defect detection model in product quality control." The International Journal of Advanced Manufacturing Technology 94.9-12 (2018): 3465-3471.

[3] Wang, Yuanbin, et al. "A CNN-based Adaptive Surface Monitoring System for Fused Deposition Modeling." IEEE/ASME Transactions on Mechatronics (2020).

[4] Qiu, Lingteng, Xiaojun Wu, and Zhiyang Yu. "A high-efficiency fully convolutional networks for pixel-wise surface defect detection." IEEE Access 7 (2019): 15884-15893.

[5] Li, Jiangyun, et al. "Real-time detection of steel strip surface defects based on improved yolo detection network." IFAC-PapersOnLine 51.21 (2018): 76-81.

[6] An, Jinwon, and Sungzoon Cho. "Variational autoencoder based anomaly detection using reconstruction probability." Special Lecture on IE 2.1 (2015): 1-18.

[7] Lai, Y. T. K., et al. "Industrial Anomaly Detection and One-class Classification using Generative Adversarial Networks." 2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM).

IEEE, 2018.

[8] Hu, Guanghua, et al. "Unsupervised fabric defect detection based on a deep convolutional generative adversarial network." Textile Research Journal (2019): 0040517519862880.

[9] Gong, Dong, et al. "Memorizing Normality to Detect Anomaly: Memory-augmented Deep Autoencoder for Unsupervised Anomaly Detection." arXiv preprint arXiv:1904.02639 (2019).

[10] Zong, Bo, et al. "Deep autoencoding gaussian mixture model for unsupervised anomaly detection." (2018).

[11] Lei, Na, et al. "A Geometric Understanding of Deep Learning." Engineering (2020).

[12] Dai, Jun, et al. "Machinery Health Monitoring Based on Unsupervised Feature Learning via Generative Adversarial Networks." IEEE/ASME Transactions on Mechatronics 25.5 (2020): 2252-2263.

[13] Heusel, Martin, et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium." Advances in neural information processing systems. 2017.

[14] Dowson, D. C., and B. V. Landau. "The Fréchet distance between multivariate normal distributions." Journal of multivariate analysis 12.3 (1982): 450-455.

[15] Le Cun, Yann, and Françoise Fogelman-Soulié. "Modèles connexionnistes de l'apprentissage." Intellectica 2.1 (1987): 114-143.

[16] Makhzani, Alireza, and Brendan Frey. "K-sparse autoencoders." arXiv preprint arXiv:1312.5663 (2013).

[17] Vincent, Pascal, et al. "Extracting and composing robust features with denoising autoencoders." Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[18] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).

[19] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.

[20] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).

[21] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.

[22] Graves, Alex, Greg Wayne, and Ivo Danihelka. "Neural turing machines." arXiv preprint arXiv:1410.5401 (2014).

[23] Weston, Jason, Sumit Chopra, and Antoine Bordes. "Memory networks." arXiv preprint arXiv:1410.3916 (2014).

[24] Santoro, Adam, et al. "One-shot learning with memory-augmented neural networks." arXiv preprint arXiv:1605.06065 (2016).

[25] Kim, Youngjin, Minjung Kim, and Gunhee Kim. "Memorization precedes generation: Learning unsupervised gans with memory networks." arXiv preprint arXiv:1803.01500 (2018).

[26] Li, Chongxuan, Jun Zhu, and Bo Zhang. "Learning to generate with memory." International Conference on Machine Learning. 2016.

[27] Kwon, Donghwoon, et al. "A survey of deep learning-based network anomaly detection." Cluster Computing (2019): 1-13.

[28] Amer, Mennatallah, Markus Goldstein, and Slim Abdennadher. "Enhancing one-class support vector machines for unsupervised anomaly detection." Proceedings of the ACM SIGKDD workshop on outlier detection and description. 2013.

[29] Duan, Lixiang, et al. "A new support vector data description method for machinery fault diagnosis with unbalanced datasets." Expert Systems with Applications 64 (2016): 239-246.

[30] Zhai, Shuangfei, et al. "Deep structured energy based models for anomaly detection." arXiv preprint arXiv:1605.07717 (2016).

[31] Schlegl, Thomas, et al. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery." International Conference on Information Processing in Medical Imaging. Springer, Cham, 2017.

[32] Zenati, Houssam, et al. "Efficient gan-based anomaly detection." arXiv preprint arXiv:1802.06222 (2018).

[33] Schlegl, Thomas, et al. "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks." Medical image analysis 54 (2019): 30-44.

[34] Akcay, Samet, Amir Atapour-Abarghouei, and Toby P. Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training." Asian Conference on Computer Vision. Springer, Cham, 2018.

[35] Bian, Jiang, et al. "A Novel and Efficient CVAE-GAN-Based Approach With Informative Manifold for Semi-Supervised Anomaly Detection." IEEE Access 7 (2019): 88903-88916.

[36] Sabokrou, Mohammad, et al. "Adversarially learned one-class classifier for novelty detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[37] Xingjian, S. H. I., et al. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting." Advances in neural information processing systems. 2015.

[38] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[39] Rodríguez-Quiñonez, J. C., et al. "Improve 3D laser scanner measurements accuracy using a FFBP neural network with Widrow-Hoff weight/bias learning function." Opto-Electronics Review 22.4 (2014): 224-235.

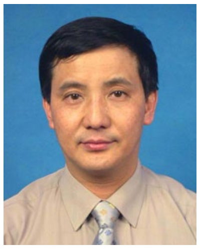
[40] Gan, Jinrui, et al. "A hierarchical extractor-based visual rail surface inspection system." IEEE Sensors Journal 17.23 (2017): 7935-7944.

[41] Tabernik, Domen, et al. "Segmentation-based deep-learning approach for surface-defect detection." Journal of Intelligent Manufacturing 31.3 (2020): 759-776.

[42] Cognex (2018) VISIONPRO VIDI: Deep learning-based software for industrial image analysis. URL https: //www.cognex.com/products/machine- vision/ vision-software/visionpro-vidi

Tongzhi Niu received the B.S. degree in mechanical design, manufacturing, and automation from the Wuhan University of Technology, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan.

His current research interests include intelligent manufacturing, defects detection, image processing, and deep learning.



Bin Li received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Huazhong Universityof Science and Technology, Wuhan, China, in 1982, 1989, and 2006, respectively.

He is currently a Professor with the School of Mechanical Science and Engineering, Huazhong University of Science and Technology.

His currentresearch interests include intelligent manufacturing and computer numerical control (CNC) machine tools.

Weifeng Li received the B.S. degree in mechanical manufacturing and automation from the Qingdao University of Science and Technology, Qingdao, China, in 2018. He is currently pursuing the M.S. degree with the State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan.

His current research interests include intelligent manufacturing, image processing.

Yuanhong Qiu received the B.S., M.S. degree in mechanical manufacturing and automation from the Huazhong University of Science and Technology,Wuhan, China, in 2014, 2017 where he is currently pursuing the Ph.D. degree in mechatronic engineering.

His current research interests include intelligent manufacturing, image processing and deep learning.

Shuanlong Niu received the B.S. degree in mechanical manufacturing and automation from the Huazhong University of Science and Technology, Wuhan, China, in 2017, where he is currently pursuing the Ph.D. degree in mechatronic engineering.

His current research interests include intelligent manufacturing, image processing, pattern recognition, and deep learning.