Article

Automated Surface Defect Inspection Based on Autoencoders and Fully Convolutional Neural Networks

Cheng-Wei Lei †, Li Zhang †, Tsung-Ming Tai †, Chen-Chieh Tsai †, Wen-Jyi Hwang * †, and Yun-Jie Jhang †

Department of Computer Science and Information Engineering, National Taiwan Normal University, Taipei 117, Taiwan; 60747052s@ntnu.edu.tw (C.-W.L.); 60747033s@ntnu.edu.tw (L.Z.); 60547048s@ntnu.edu.tw (T.-M.T.); 60847075s@gapps.ntnu.edu.tw (C.-C.T.); 60547051s@ntnu.edu.tw (Y.-J.J.)
* Correspondence: whwang@ntnu.edu.tw
† These authors contributed equally to this work.

Abstract: This study aims to develop a novel automated computer vision algorithm for quality inspection of surfaces with complex patterns. The proposed algorithm is based on both an autoencoder (AE) and a fully convolutional neural network (FCN). The AE is adopted for the self-generation of templates from test targets for defect detection. Because the templates are produced from the test targets, the position alignment issues for the matching operations between templates and test targets can be alleviated. The FCN is employed for the segmentation of a template into a number of coherent regions. Because the AE has the limitation that its capacities for the regeneration of each coherent region in the template may be different, the segmentation of the template by FCN is beneficial for allowing the inspection of each region to be independently carried out. In this way, more accurate detection results can be achieved. Experimental results reveal that the proposed algorithm has the advantages of simplicity for training data collection, high accuracy for defect detection, and high flexibility for online inspection. The proposed algorithm is therefore an effective alternative for the automated inspection in smart factories with a growing demand for the reliability for high quality production.

Keywords: surface inspection; defect detection; artificial intelligence; autoencoder; convolutional neural networks

1. Introduction

Surface quality inspection is an important process in an industrial production system. Basic approaches for inspection are mostly by skilled inspectors, which may be time-consuming and laborious. Furthermore, it would be difficult to meet the requirements of reliability and robustness. With the advent of computer vision [1] and artificial intelligence techniques [2], automated computer visual inspection methods are found to be beneficial for improving performance for industrial production.

One way to carry out surface inspection is by analyzing textures to find patterns without normal features on the test targets. When the surface texture distribution is known a priori, the features associated with local abnormalities can be extracted [3,4]. For example, a Haar–Weibull-variance model [5] has been found to be effective for the extraction of features for defect detection on strip steel surfaces. In frequency domain, spectral features are usually extracted by Fourier transform [6]. Although some results are promising, the local abnormalities-based methods lack the effective use of existing normal-pattern data. The occurrence of false alarms is likely. Some alternative approaches take normal and/or abnormal patterns into consideration [7–10] by deep convolutional neural networks (CNN). For applications such as building defect detection, high classification accuracy can be achieved [10]. The limitation of these methods is that the number of training samples should be adequate and balanced enough to achieve a desirable performance. However,
for scenarios where defective samples are scarce, effective training for a CNN may be a challenging task.

Template-based methods can be employed for alleviating the requirements for the collection of defective samples for surface inspection. The methods introduce defect-free template images into the detection procedure so that no prior knowledge on defects is required. Basic template-based approaches accomplish defect detection by measuring the similarity (or dissimilarity) between the given test image and defect-free template. The normalized cross correlation is classical for dissimilarity measurement. Its improved versions have been proposed, including the partial information correlation coefficient [11] and asymmetric correlation [12]. The distribution-based template establishment procedure [13] is also found to be effective for enhancing detection accuracy. A common drawback of some template matching approaches is that proper alignment between the test image and template is desired for the correlation computation. However, for many applications, the enforcement of alignment operations may be difficult, resulting in degradation of detection accuracy. An alternative template-based method is to adopt the template images as the training images for an autoencoder (AE) for dimension reduction and feature extraction [14,15]. Defect detection can be accomplished by simply comparing the input and output of the AE. No precise alignment is required before the inspection. The accuracy can be further improved by carrying out the AE-based reconstruction in a multiscale fashion [16].

A target surface to be inspected can usually be viewed as an image consisting of a number of coherent regions, where each region is a set of connected pixels sharing common characteristics such as texture or color [17,18]. Although the AEs are promising for surface quality inspection, they may only be suited for surfaces with only a single coherent region. For many real-world applications, inspection of surfaces with multiple coherent regions is usually desired. Because different regions may have different features, it would then be difficult for an AE to extract a feature match to all the regions. As a result, the AE may have different capabilities for each region for defect detection. A unified approach for surface inspection over different homogeneous regions may result in high miss rates in some regions and/or high false alarm rates in others.

The objective of this paper is to develop a novel automated computer vision algorithm for quality inspection of surfaces with multiple coherent regions. The proposed algorithm is a template-based algorithm for defect detection. The algorithm contains two neural networks. The first network is an AE for template generation of an input test target. The second network is a fully convolutional network (FCN) [19,20] for the segmentation of the template into a number of homogeneous regions. Each region of the template is then compared with the corresponding region of the test target for the surface inspection. Because different regions have different features, each region is inspected independently according to its own criteria, different from the other ones. In this way, defects can be accurately identified on all the regions.

The proposed algorithm has a number of advantages. First of all, it does not need defective patterns as training samples. Only a small number of normal surface patterns may suffice for training. A data augmentation scheme is adopted for the generation of defective images. This could facilitate the training operations. It is especially beneficial for cases when the collection of defective samples is difficult, and/or there is no prior knowledge about the surface defects. Furthermore, it is not necessary to carry out the inspection with precise alignment to the template by the proposed algorithm. The surface inspection process can then be effectively simplified.

The final and the most important feature is that the proposed algorithm is able to achieve high detection accuracy even when multiple coherent regions are presented on the surface. Because each region can be independently inspected for attaining the optimal accuracy, the proposed algorithm is beneficial for providing reliable and effective defect detection over surfaces of large varieties of objects.
The novelty and contribution of this work is to propose a novel algorithm combining both AE and FCN for defect detection. Most of the existing AE-based approaches [14–16] detect defects from the reproduced images by AEs in a unified manner. By contrast, our method is able to separate reproduced templates into different regions by FCN and inspect each region independently. To improve segmentation accuracy, a novel two-stage training process is presented, where the first stage and the second stage are for AE and FCN, respectively. The defects are regarded as noises in our model. The training at the first stage takes the denoising processes into consideration so that the AE is able to remove defects for template generation. The second stage training is based on the training results from the first stage so that templates can be accurately segmented. The proposed technique provides higher flexibility and better accuracy for defect detection. Furthermore, the technique may also be beneficial for other detection applications such as slug velocity detection in microchannels [21].

The remaining parts of this study are organized as follows. Section 2 presents the proposed automated surface inspection algorithm in detail. Experimental results of the proposed algorithm are then presented in Section 3. Finally, Section 4 includes some concluding remarks of this work.

2. The Proposed Algorithm

In this section, we first provide a brief introduction of CNN, AE, and FCN. An overview of the proposed algorithm then follows. We then discuss the operations for each neural network of the algorithm. The training procedures for the neural networks are then presented in detail. The online inspection system based on the proposed algorithm is also presented so that the results of this study can be effectively applied for a field test. To facilitate understanding of the discussions in this study, Table 1 includes a list of frequently used symbols.

Table 1. A list of frequently used symbols in this study.

| Symbol | Meaning |
|--------|---------|
| C      | The set of defective blocks identified by the proposed algorithm. |
| D      | The L2 distance between two matrices with the same size. |
| F      | The AE in the algorithm. |
| G      | The FCN in the algorithm. |
| K      | The number of training images. |
| M      | The dimension of images $S_k, X_k, Y_k$. |
| N      | The number of coherent regions. |
| $P^i(i)$ | The probability that pixel $y^i$ belongs to the i-th region of Y. |
| $P^i_k(i)$ | The probability that pixel $y^i_k$ belongs to the i-th region of $Y_k$. |
| $S_k$ | The ground truth of $X_k$ for training. |
| $T_i$ | The threshold for defect detection for the i-th region of $X$. |
| $X(i)$ | The i-the region of image X. |
| $X_k$ | The k-th training image for the AE. |
| $x$ | A block of image $X$. |
| $x_j^i$ | The j-th pixel of image $X_k$. |
| $Y$ | The reproduced image by the AE. It serves as the template for $X$. |
| $Y(i)$ | The i-th region of image $Y$. |
| $Y_k$ | The k-th training image for FCN. |
| $y$ | A block of image $Y$. Both $y$ and $x$ have the same size. |
| $y^i$ | The location of $y$ in $Y$ is also the same as that of $x$ in $X$. |
| $y_j^i$ | The j-th pixel of image $Y$. |
| $y_j^k$ | The j-th pixel of image $Y_k$. |
2.1. Basic CNN, AE, and FCN

A commonly used deep learning technique is CNN [2], where convolutional layers are included as hidden layers of the neural network. A convolutional layer convolves its input channels with a set of kernels and passes the results through an activation function as output channels to the next layer. A commonly used activation function for CNN is rectified linear unit (ReLU) [2]. In addition to convolutional layers, fully connected layers containing a number of fully connected neurons are also commonly used in CNN. A CNN network may support pooling or upsampling operations. A pooling operation reduces the dimension of its input channels. Maximum pooling is a typical example for a pooling operation. In contrast to a pooling operation, the goal of an upsampling operation is to increase the dimension of the input channels. Consider a CNN with Q layers. Because convolutional or fully connected operations can be realized by matrix multiplications [2,22], each layer \( i, i = 1, ..., Q \), can be defined as

\[
\begin{align*}
a_i & = W_i u_i + b_i, \\
v_i & = z(a_i),
\end{align*}
\]

where \( u_i \) is the vectorized input of the layer \( i \), \( a_i \) is the results of matrix multiplications, and \( v_i \) is the output of the layer \( i \). The function \( z \) for producing \( v_i \) from \( a_i \) in (2) is the activation function for the layer \( i \). When ReLU is the activation function for the CNN, the corresponding function \( z \) is given by

\[
z(a_i) = \max(0, a_i).
\]

The function \( z \) for other types of activation functions can be found in [2]. The matrix \( W_i \) is determined by the weights associated with the convolutional or fully connected layers. When layer \( i \) is a convolutional layer, the matrix \( W_i \) is a Toeplitz matrix [22,23] obtained from the weights of kernels associated with layer \( i \). The vector \( b_i \) denotes the bias vector. Let \( U = u_1 \) and \( V = v_Q \). The \( U \) and \( V \) are then the input and output of the CNN network, respectively. Depending on the architecture of the CNN, the input \( u_i \) at layer \( i \), \( i = 2, ..., Q \) could be obtained directly from the output \( v_{i-1} \) at layer \( i - 1 \). Alternately, the \( u_i \) may also be a concatenation of the outputs from some of its previous layers.

A basic AE is a neural network that is trained to replicate its input to its output [2]. As shown in Figure 1a, the network contains two parts: an encoder for feature extraction of its input and a decoder for reconstruction from the feature. Both the encoder and decoder contain convolutional layers and/or fully connected layers with operations shown in (1) and (2). Therefore, the basic AE can be regarded as a CNN with Q layers, where each layer \( i \) is defined in (1) and (2). In this study, the autoencoder is not trained to replicate its input \( U \) perfectly. It is restricted to ignoring defective portions of the input image \( U \) for image reconstruction. Only normal portions are copied to the output \( V \).

![Figure 1](image1.png)

*Figure 1.* The structure of basic AE and FCN: (a) Basic AE contains an encoder and a decoder; (b) Basic FCN consists of analysis network and synthesis network.

A basic FCN is a neural network for the segmentation of the input image to a number of coherent regions [19]. The FCN network is also a CNN network relying only on convolutional layers for the exploitation of the correlation among local pixels. No fully connected layers are needed. The FCN can also be separated into two parts: an analysis network for
correlation exploitation and a synthesis network for producing segmentation results, as revealed in Figure 1b. The basic FCN can also be viewed as a CNN with Q layers, where each layer \( i \) is defined in (1) and (2). Furthermore, because all the layers are convolutional layers, the matrix \( W_i \) for each layer \( i, i = 1, ..., Q \), is a Toeplitz matrix. Given an input image \( U \), the FCN produces output \( V = \{B_1, ..., B_N\} \), where \( N \) is the number of coherent regions for segmentation, and \( B_i, i = 1, ..., N \), is the mask image for the \( i \)-th coherent region on \( U \), denoted as \( U(i) \). That is, \( U(i) \) is the set of pixels in \( U \), where the locations of the pixels are identified by \( B_i \).

2.2. Procedure for Defect Detection

The proposed algorithm is a template-based algorithm for defect detection. Figure 2 shows the block diagram of the proposed algorithm, which contains two neural networks: an AE and an FCN. Given a test image \( X \), the AE, denoted by \( F \), reproduces the test image \( X \). That is,

\[
Y = F(X),
\]

where \( Y \) is the image reproduced by the AE. In the proposed system, the AE is expected to remove the defects of the input test image \( X \). Therefore, defects may not be reproduced by the AE when image \( X \) is defective. We view the image \( Y \) reproduced by the AE as the template for the image \( X \).

![Figure 2. The overview of the proposed quality inspection of surface algorithm.](image)

By comparing the test target \( X \) with its template \( Y \) reproduced by the AE, it is then possible to identify defect regions. To carry out the comparison, the input image \( X \) is first separated into a set of non-overlapping blocks with equal size. Let \( x \) be a block of \( X \). To determine whether there exists defects in \( x \), we compute L2 distance between \( x \) and its counterpart \( y \) in the template \( Y \). As shown in Figure 3, the blocks \( x \) and \( y \) have the same size. The location of \( x \) in \( X \) is also the same as that of \( y \) in \( Y \). A defect is detected when the L2 distance, denoted by \( D(x, y) \), is larger than a pre-specified threshold \( T \).

One issue in this approach is that the AE may not have the same capacity for the reconstruction of different blocks in \( X \). This is because local features for an image may vary. It is more difficult to reconstruct areas containing complex patterns. As a result, for a block \( x \) in the areas with large variations, the discrepancy between \( x \) and its counterpart \( y \) would be high even if the input image \( X \) is not defective. In these cases, it may be necessary to adopt a higher threshold value \( T \) for determining a defective block. A single threshold \( T \) for defect detection may not be appropriate for all the blocks from an input image.
Figure 3. An example showing the relationship between a block \( x \in X \) and its counterpart \( y \in Y \). In the example, the input test image is separated into 16 non-overlapping blocks with equal size. Each block \( x \in X \) and its counterpart \( y \in Y \) have the same size. The location of \( x \) in \( X \) is also the same as that of \( y \) in \( Y \).

In this study, the FCN is adopted to solve the issue stated above. It is used to segment the template \( Y \) into \( N \) regions \( Y(1), ..., Y(N) \), where \( N \) is the number of coherent regions in \( Y \). That is,

\[
\{Y(1), ..., Y(N)\} = G(Y),
\]

where \( G \) denotes the FCN operations. Each region \( Y(i) \) produced by the FCN is a set of pixels sharing common features such as colors or textures. Each \( Y(i) \) can be associated with a threshold \( T_i \), \( i = 1, ..., N \). For a block \( x \), when its counterpart \( y \) belongs to \( Y(i) \), we then adopt the threshold \( T_i \) for the defect detection. That is, we first define the sets

\[
X(i) = \{x : y \in Y(i)\}, i = 1, ..., N.
\]

In this case, the block \( x \in X(i) \) is said to be defective when \( D(x, y) > T_i \). In this way, different threshold values can be selected for defect detection in accordance with the local features for different regions. The summary of the proposed algorithm is also provided in Algorithm 1, where the set of defective blocks, denoted by \( C \), contains the final results of the proposed algorithm. Based on the final \( C \), the locations of defective blocks can be easily identified. The defect attributes such as their patterns and areas can then be effectively visualized and measured.

2.3. The Operations of AE and FCN

The proposed algorithm is not restricted to any specific types of AEs and FCNs. However, for demonstration purposes, design examples of AE and FCN are revealed in Figures 4 and 5 and Tables 2 and 3. Based on the examples, the operations of each network are then presented.

We can see from Figure 4 that an AE contains an encoder and a decoder. The goal of the encoder is to perform the feature extraction of the input test image. It contains a number of convolution layers with maximum pooling operations. Based on the features produced by the encoder, the decoder carries out the image reconstruction operations so that the test image can be reproduced at the output of the AE. The decoder also consists of a number of convolution layers, which are followed by upsampling operations for the image reconstruction. The activation functions for all the convolution operations are relu, as shown in Table 2.
Algorithm 1: Proposed quality inspection of surfaces algorithm.

Require: A trained AE $F$.
Require: A trained FCN $G$.
Require: An inspection target $X$.
Require: Thresholds $T_i, i = 1, ..., N$.

1: Initialize $C \leftarrow \emptyset$.
2: Compute $Y = F(X)$ by (4).
3: Compute $\{Y(1), ..., Y(N)\} = G(Y)$ by (5).
4: Separate $X$ into blocks $\{x\}$.
5: Find $y$ from $Y$ for each $x \in X$.
6: Form $X(i) = \{x : y \in Y(i)\}, i = 1, ..., N$, by (6)
7: for $i \leftarrow 1, N$ do
8:     repeat
9:         Get a block $x$ from $X(i)$.
10:        Compute $D(x, y)$, the L2 distance between $x$ and $y$.
11:        if $D(x, y) > T_i$ then
12:            $C \leftarrow C + x$.
13:        end if
14:     until All blocks $x \in X(i)$ are searched.
15: end for
16: return $C$

Table 2: Details of the convolution layers of the AE.

| Convolution Layers | Conv 1 | Conv 2 | Conv 3 | Conv 4 | Conv 5 |
|--------------------|--------|--------|--------|--------|--------|
| Number of Input Chan. | 1 | 16 | 32 | 32 | 16 |
| Number of Output Chan. | 16 | 32 | 32 | 16 | 1 |
| Activation Function | ReLU | ReLU | ReLU | ReLU | ReLU |

Table 3: Details of the convolution layers of the FCN.

| Convolution Layers | Conv 1 | Conv 2 | Conv 3 | Conv 4 | Conv 5 | Conv 6 | Conv 7 | Conv 8 | Conv 9 | Conv 10 | Conv 11 |
|--------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|----------|---------|
| Number of Input Chan. | 1 | 64 | 64 | 128 | 128 | 256 | 256 | 128 | 128 | 64 | 64 |
| Number of Output Chan. | 64 | 64 | 128 | 128 | 256 | 128 | 128 | 64 | 64 | 64 | 1 |
| Activation Function | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | ReLU | Soft-max |

An important aspect of the AE shown in Figure 4 is that it is based solely on convolutional layers. The fully connected layers are not included. This is beneficial for reducing the number of weights and computation complexities of the algorithm. Furthermore, the convolution operations are able to effectively extract local features of input images.
Therefore, the reconstructed images are less sensitive to the variations of global features such as positions of objects on the test images. This could be beneficial for reducing the efforts for alignment.

The example of the FCN network shown in Figure 5 is used for image segmentation. As revealed in Figure 5, the FCN network is actually a simplified version of the U-Net [20]. It contains the analysis operations for feature extraction and synthesis operations for producing the segmented images. In addition to convolution operations, the U-Net also contains max-pooling, up-sampling, and concatenation operations so that features at different resolutions can be captured for image segmentation. It can be observed from Table 3 that the activation function at the final layer is the Softmax. The FCN produces \( N \) binary output images \( B_1, ..., B_N \). Each binary image \( B_i \) serves as a mask revealing the region \( Y(i) \). That is, all the locations of pixels in \( B_i \) with value 1 indicate the area covered by \( Y(i) \).

Figure 4. An example of the AE for the proposed algorithm. The AE in this example contains 5 convolution layers, which are denoted by Conv \( i, i = 1, ..., 5 \).

Figure 5. An example of the FCN for the proposed algorithm. The FCN in this example contains 11 convolution layers, which are denoted by Conv \( i, i = 1, ..., 11 \).

2.4. The Training of AE and FCN

Figure 6 shows the procedure for the training of the AE and FCN. As shown in Figure 6, there are two training stages. The first stage is the training for the AE. After the training process in the first stage is completed, we use the resulting AE network model to generate the training images for the FCN in the second stage of the training process.
Figure 6. Procedure for the training of AE and FCN.

Let $X_k, k = 1, \ldots, K$, be the $k$-th images for the training of the AE, where $K$ is the number of training images. All the training images are defective images. Furthermore, let $Y_k, k = 1, \ldots, K$, be the image at the output of the AE when its input is $X_k$. Given a training image $X_k$, let $S_k$ be the ground truth of $X_k$. That is, $S_k$ is the defect free version of $X_k$. $S_k$ can be regarded as images from a normal sample. The loss function, denoted by $J$, for the training of the AE is given by

$$ J = \frac{1}{K} \sum_{k=1}^{K} D(S_k, Y_k). \quad (7) $$

Note that $Y_k$ and $S_k, k = 1, \ldots, K$, are the reconstructed images and their ground truth, respectively. Therefore, the goal of the training is to guide the AE to effectively remove defective parts of input samples $X_k$ so that the discrepancy between $S_k$ and $Y_k$ can be minimized. In this way, the AE is only able to reproduce normal patterns of the input images. The images produced by the AE can then be viewed as the templates of the corresponding input images for defect detection.

For applications where only normal samples are available, it is only possible to acquire $S_k, k = 1, \ldots, K$, from the normal samples for training. In these cases, it may be necessary to obtain $X_k$ from $S_k$ by data augmentation. One simple approach for the augmentation is by adding a zero mean Gaussian noise to $S_k$. We then view the corresponding image after noise corruption as $X_k$. That is,

$$ X_k = S_k + \eta_k, \quad (8) $$

where $X_k$, $S_k$ and $\eta_k$ have the same dimension, denoted by $M$. Each element $\xi$ of $\eta_k$ is drawn from a zero-mean Gaussian distribution with variance $\sigma^2$. That is, the density function for each element $\xi$ of $\eta_k$ is given by $\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\xi^2}{2\sigma^2}\right)$. In this way, the template generation process can be regarded as the denoising process [24], where the defective pixels are those corrupted by noises. The input image $X_k$ and output image $Y_k$ to the AE are the corrupted and restored versions of $S_k$, respectively. The AE in the proposed algorithm is then equivalent to a denoiser, where the ground truth $S_k$ is available for each corrupted observation $X_k$ for the training.

An advantage of the proposed AE training approach is that a single image from normal sample can be used to generate multiple defective images. That is, different training images $X_k$ may have the same ground truth $S_k$. Therefore, even for the cases where only a small number of normal samples are available, a large number of training images can still be produced. This could be beneficial for the avoidance of overfitting for the training of the AE.

The training of FCN is based on $Y_k, k = 1, \ldots, K$, which are the reconstructed images produced by the AE. Let $y_{jk}^k$ be the $j$-th pixel of the image $Y_k$. Let $p_{jk}^k(n)$ be the estimated
probability that the pixel \(y^j_k\) belongs to the region \(n\). Therefore, for fixed \(k\) and \(j\), it follows that
\[
\sum_{n=1}^{N} P^j_k(n) = 1,
\]
where \(0 \leq P^j_k(n) \leq 1\). Let region \(m\) be the ground truth of the pixel \(y^j_k\). The estimated probabilities are trained by the proposed FCN network. The estimated probability is said to be accurate when
\[
P^j_k(m) = \max_{n=1,...,N} P^j_k(n).
\]

Based on the facts stated above, the corresponding loss function for the training of FCN is
\[
L = \sum_{k=1}^{K} \sum_{j=1}^{M} \log \left( \frac{1}{P^j_k(m)} \right),
\]
where \(M\) is the dimension of \(Y^k\), and \(K\) is the number of training images. Clearly, the loss \(L\) will increase when \(P^j_k(m)\) does not meet the condition in (10). In fact, the loss function will penalize at each pixel \(y^j_k\) the deviation of \(P^j_k(m)\) below 1.0. Therefore, the training of FCN minimizing the loss function \(L\) is able to maximize \(P^j_k(m)\) for each \(k\) and \(j\). After the FCN is trained, given a test image \(Y\), the network is then able to produce \(P^j(n)\), the estimated probability of the \(j\)-th pixel \(y^j\) belonging to region \(n\). The test image \(Y\) is subsequently segmented to regions \(Y(1),...,Y(N)\), where
\[
Y(i) = \{y^j : P^j(i) = \max_{n=1,...,N} P^j(n)\}, i = 1,...,N.
\]

The operations shown in (12) can be viewed as the function \(G(Y)\) defined in (5). In addition to training, the validation is required for the avoidance of overfitting. In the proposed algorithm, the validation operates in conjunction with the training. However, both processes are based on different data sets. After the completion of each epoch during the training process, the values of the loss function for the training set and validation set are measured, respectively. We stop the training process only after the convergence of the loss function values for both the training set and validation set are observed. The samples in the training set and validation set are based on the data augmentation process presented in (8). However, the data set for testing in our experiments consists of real images acquired from a camera without data augmentation. Furthermore, the samples in the testing set are different from those in the training and validation sets. The effectiveness of the proposed algorithm can then be evaluated from the real images for defect detection.

2.5. The Proposed Online Inspection System

In addition to the development of algorithms, the online evaluation of the proposed algorithms in an Internet of Things (IoT) system [25] for manufacturing [26] is also considered in this study. Figure 7 shows the basic architecture of the IoT system, which consists of illumination devices, a surface inspection platform, and a computer server. The trained neural network models for different products are stored in the server. The proposed system is deployed in the surface inspection platform. Given a product, the corresponding model can be downloaded from the cloud server to the inspection platform for the defective detection operations. When a defective sample is identified, the images of the defective samples will also be delivered to the server for subsequent quality management.
3. Experimental Results

This section provides evaluations of the proposed work. The setup of the experiments is the online surface inspection platform shown in Figure 7. The surface inspection platform contains a high resolution industrial camera FLIR Blackfly S USB 3 and a personal computer with NVIDIA RTX 2070X GPU. The development of neural network models is based on Keras [27] built on the top of Tensorflow 2.0. The training and testing images of the inspection targets for the neural network models are acquired from the industrial camera of the online surface inspection platform.

Without loss of generality, examples of the inspection targets are the display cards. The inspection of the backplate of the cards and their gold finger connectors is considered in this study. The backplate of a display card usually contains multiple coherent regions. Each may have different characteristics such as patterns or colors. The backplate inspection would then be beneficial for demonstrating the effectiveness of the proposed algorithm. Furthermore, the defect detection for a gold finger connector is usually the major focus for the inspection of printed circuits. The images of gold finger connectors also contain multiple coherent regions. We therefore include the corresponding inspection in this study as well.

3.1. Surface Inspection for Gold Finger Connectors

The gold finger of a display card is the connector on the edge of the corresponding printed circuit board. Because the gold finger connector is a long and narrow strip, it would be best to acquire portions of the strip one at a time for accurate inspection. Figure 8 shows the examples of images of normal or defective samples of a gold finger connector. Some variations can be observed on the normal samples revealed in Figure 8a, especially for the regions outside the gold finger area. For the defective samples, scratches can be observed. Furthermore, we can see from Figure 8b that there are no regular patterns for the scratches. It would then be difficult to use the classification-based methods or the local abnormalities-based methods for accurate defect detection.

The proposed algorithm is based on the AE shown in Figure 4, which can be trained by the images augmented from only a small number normal samples. In the experiments, 100 training images (i.e., \( K = 100 \)) augmented from 16 normal samples of gold finger are employed for training. All the normal samples, training images, and the reproduced images by AE have the same dimension \( 256 \times 256 \). That is, the \( S_k, X_k, \) and \( Y_k \) have the same dimension of \( M = 256 \times 256 \). The augmentation process is based on noise corruption operations shown in (8). For an input image from a defective sample, it is then expected that the defective parts of the image may not be reproduced by the AE. Figure 9 shows examples of the input test images and their reconstructions by the AE. The test images are outside the training set. There are two scenarios: one is with a normal input sample, and the other is with a defective input sample. All the input test images considered in the examples are outside the training set. We can observe that, for the scenario with a normal sample shown in Figure 9a,b, the input image can be accurately reproduced. On the contrary, for the scenario with a defective sample, revealed in Figure 9c,d, the reconstruction is not...
accurate. In fact, most of the defective regions are removed on the output image. We can then view the image produced by the AE as the template for the defect detection.

Figure 8. Examples of normal or defective samples of gold finger connector images. (a) Normal samples with no defects; (b) defective samples.

Figure 9. Examples of the input test images and their reconstructions by the AE for gold finger connector images: (a) normal test sample; (b) reconstructed normal test sample by the AE; (c) defective test sample; (d) reconstructed defective test sample by the AE.
The reconstructed images produced by the AE can be segmented into two regions (i.e., $N = 2$): one is the gold finger area, and the other is the area outside the gold finger connector. The corresponding FCN for the segmentation is trained by the images reproduced by the AE. There are 100 images for the FCN training. Figure 10 shows the results of image segmentation produced by the FCN. All the test images considered in the examples are outside the training set. It can be observed from Figure 10 that the gold finger areas can be accurately identified for the test images considered in the experiments.

![Figure 10](image1.png)

**Figure 10.** Examples of segmentation results produced by FCN for the images of gold finger connectors. (a) The input images to the FCN; (b) the segmentation results, where the white areas are the gold finger area, and the black areas are the area outside gold finger connector.

The general inspection procedure outlined in Algorithm 1 can be further simplified for the inspection of the gold finger connector. In this case, the focus of the surface inspection is actually on the white areas produced by the FCN shown in Figure 10b, which correspond to the gold finger area. Only the L2 distance of each block $y$ located in the white area of the reconstructed test image produced by the AE, as well as the corresponding block $x$ in the original test image, is measured. When the resulting L2 distance is larger than a pre-specified threshold, the block is said to be defective. In addition to detection, we also provide a simple visualization scheme, as shown in Figure 11, where the original test image is superimposed by the defective blocks. The defective blocks are marked as red, orange, or yellow blocks, depending on their corresponding L2 distance measurements. In this way, the quality of the surface inspection can be directly observed.

To show the effectiveness of the proposed algorithm based on the visualization scheme shown in Figure 11, a number of examples for the defective samples and their detection results are revealed in Figure 12. To facilitate the observation, the ground truth of the samples is also included. It can be observed from Figure 12 that, although the diversities of defects are high, they are effectively identified. This is because the AE is able to reproduce defect-free templates from the defective samples. Furthermore, the proposed FCN can accurately identify the gold finger areas from the templates.
Figure 11. The proposed visualization scheme for the inspection results: (a) visualization results for a normal sample; (b) visualization for a defective sample; (c) colors and their corresponding $L_2$ distances.

Figure 12. Examples of the defective samples, their ground truth, and inspection results for gold finger connectors: (a) defective samples; (b) ground truth; (c) inspection results.
3.2. Surface Inspection for the Backplate of a GPU Card

In addition to a gold finger connector, the experiments for surface inspection for the backplate of a display card are also considered. The training set for the experiments contains 100 images (i.e., $K = 100$) augmented from the 16 normal samples of the backplate of the display card. The dimension of images $S_k$, $X_k$, and $Y_k$ for the experiments is $M = 512 \times 512$. Figure 13 shows examples of normal and defective samples of the backplate and their corresponding templates produced by the AE. We can observe from Figure 13 that accurate reconstruction is possible for normal samples. Furthermore, the AE is able to remove most of the defective regions for the flawed samples. Therefore, similar to the results shown in Figure 9 for the gold finger area, images produced by the AE can also be effectively used as templates for the backplates for surface inspection.

![Figure 13. Examples of the input test images and their reconstructions by the AE for the backplate surface of a display card: (a) normal test sample; (b) reconstructed normal test sample by the AE; (c) defective test sample; (d) reconstructed defective test sample by the AE.](image)

Due to the high complexities of the surface of a backplate, it may be necessary to separate the surface of the backplate into more than two regions. An example is to separate the surface into five regions. An individual region can then be inspected independently with its own threshold value. The segmentation results produced by the FCN for various test images are shown in Figure 14, where each region is associated with a different color.
From Figure 14, we see that the fan of the backplate is separated into blue and yellow areas, where the blue regions indicate the fan blades and fan center. The remaining part of the fan is colored by yellow. The area outside the fan of the shell is segmented into three regions, labeled by green, red, and black colors, respectively. The green region has higher brightness than that of the other two areas. On the contrary, the black region has lower brightness. We can observe from Figure 14 that the FCN is able to effectively separate each test image into these different regions for subsequent inspections.

Figure 15 shows some visualization results for the inspection of the backplate surface. We can observe from Figure 15 that the proposed algorithm is able to identify the defective areas effectively, even though some areas are actually small. In addition to the effectiveness of the AE for producing the templates, the high accuracy for the segmentation of templates by the FCN plays a key role in the surface inspection. The segmentation process allows the inspection for each region to be carried out independently by selecting the threshold best-matched to that region.

Figure 14. Examples of segmentation results produced by FCN for backplate images. (a) The input images to the FCN; (b) the segmentation results, where the surfaces are separated into five regions labeled by blue, yellow, red, green, and black colors, respectively.
Figure 15. Examples of the input samples, their ground truth, and inspection results for the backplate of a display card: (a) input samples; (b) ground truth; (c) inspection results.

3.3. Numerical Evaluation

In addition to the visualization, the numerical evaluation of the proposed algorithm is also included in this study. For the gold finger images, the evaluation is based on the receiver operating characteristic (ROC) curve [28] of a test set containing 64 images of normal or defective samples. The ROC curve is acquired by plotting the true positive rate (TPR) against false positive rate (FPR) at various threshold settings. Let $A$ and $B$ be
the total number of normal and defective samples in the test set, respectively. Among \( A \) normal samples, let \( C \) be the number of samples which are incorrectly found to be defective. In addition, let \( D \) be the number of samples correctly found to be defective among the \( B \) defective samples. We then define \( \text{TPR} = \frac{D}{B} \) and \( \text{FPR} = \frac{A}{C} \), respectively. Figure 16 shows the resulting ROC for the proposed algorithm and the AE algorithms in [14,24]. To achieve fair comparisons, the AEs of all the algorithms are based on the architecture shown in Figure 4. The training set for the AE in [14] contains only images from normal samples. By contrast, the AE in [24] is a denoiser trained by images corrupted by Gaussian noises with noise-free images as ground truth.

\[
\text{TPR} = \frac{D}{B}, \quad \text{FPR} = \frac{A}{C}
\]

**Figure 16.** The ROC curve for the inspection of gold finger area for various algorithms: (a) ROC curve of the proposed algorithm, (b) ROC curve for algorithm in [14], (c) ROC curve for algorithm in [24].

Based on the ROC curves, the area under ROC (AUROC) of each algorithm is also measured. As shown in Table 4, the AUROC for the proposed algorithm and algorithms in [14,24] are 0.978, 0.690, and 0.859, respectively. The study in [14] does not perform well because the AE is trained by only normal images. It may not be able to effectively remove defective parts of test images for template matching. Based on the denoiser AE and the FCN-based template matching, the proposed algorithm has better AUROC performance over the algorithms in [14,24].

Because the diversities for the region outside gold finger area may be large even for the normal samples, the same AE may then have different capacities for the reconstructions of the regions inside and outside the gold finger area. A unified treatment for all the regions may then introduce higher FPRs and/or lower TPRs. As a result, although the algorithms in [14,24] are also based on AEs, the algorithms inspect all the regions by the same threshold. They may then have inferior AUROC performance. On the contrary, the proposed algorithm leverages the results of the FCN so that the template matching can be carried out only for the gold finger area. An accurate detection with a superior ROC curve can then be attained.

The ROC curve for the inspection of the backplate surface for various techniques is revealed in Figure 17. The numerical evaluation is based on a test set containing 64 normal or defective backplate surface images. We can see from Figure 17 that the proposed algorithm has superior performance. Without the employment of FCN, it would be difficult to find a threshold well-suited for all regions on the surface of the backplate for the defect detection. Consequently, we can see from Table 4 that the AUROCs of the algorithms in [14,24] are only 0.674 and 0.886, respectively. By contrast, the proposed algorithm is still able to achieve a high AUROC of 0.983 even for the inspection of the backplate surface. It can then be concluded that the proposed algorithm offers reliable inspection results for complex surfaces.
The ROC curve for the inspection of backplate surface for various algorithms: (a) ROC curve of the proposed algorithm; (b) ROC curve for algorithm in [14]; (c) ROC curve for algorithm in [24].

In addition to the AUROC, the latency for the inference operations for the algorithms is also included in Table 4. In the experiments, the inference operations are carried out by the NVIDIA RTX 2070 GPU platform. It can be observed from Table 4 that the latency of the inference operations over the gold finger images is lower than that over the shell surface images for a given algorithm. The inspection for gold finger images can be faster because they have smaller image sizes as compared with the backplate surface images (i.e., $256 \times 256$ vs. $512 \times 512$). We can also see from Table 4 that the proposed algorithm has larger latency for the inference operations. This is because the proposed algorithm requires additional FCN-based template matching operations. Nevertheless, based on the latency, high throughput inspection can still be attained. In fact, for the gold finger images, the latency is 4.1 ms. The maximum throughput for the inspection would then be 243 frames per second (fps). For the backplate surface, the latency is increased to 47.3 ms. The maximum throughput could still achieve 21 fps for the inspection. All these facts show the effectiveness of the proposed algorithm.

Table 4. Summary of AUROC and inference latency for various algorithms.

|                          | The Proposed Algorithm | Algorithm in [14] | Algorithm in [24] |
|--------------------------|------------------------|-------------------|-------------------|
| **Gold Finger**          | AUROC                  | 0.978             | 0.690             | 0.859             |
| (256 × 256)              | Latency                | 4.1 ms            | 1.7 ms            | 1.7 ms            |
| **Backplate Surface**    | AUROC                  | 0.983             | 0.674             | 0.886             |
| (512 × 512)              | Latency                | 47.3 ms           | 7.3 ms            | 7.3 ms            |

4. Conclusions

The experimental results have revealed the effectiveness of the proposed algorithm for surface inspection. Only normal samples are required for the proposed algorithm. A simple data augmentation scheme is adopted for the generation of defective images for the training of the neural networks. This could facilitate the collection of a training set for the algorithm. In addition, the ability for the self-generation of the template by the AE for an input test image is beneficial for lifting the restriction on the synchronization between the position of the test image and the template. Flexibility for the inspection process can be improved. The segmentation operations carried out by the FCN can separate the templates into different regions for independent inspection. Both the self-generation and segmentation operations for templates could effectively enhance both the robustness and accuracy for defect detection. Experiments on the gold finger areas and the backplate surface of a display card have been conducted. Both the visualization and numerical results are provided. We conclude from the results that the proposed algorithm provides an effective solution for defect detection applications where flexibility, reliability, and accuracy of the inspection are important concerns.
Regarding future research of the proposed study, labeling would be a potential extension. In the proposed algorithm, the coherent regions of the test targets are specified and labeled by direct visual observation. Efforts are therefore required for accurate labeling. Degradation in performance may be possible with improper labeling. Semi-supervised techniques for the proposed system are then desired for alleviating the efforts for labeling. The techniques are expected to provide accurate detection even with noisy labels. The higher robustness against noisy labels would be beneficial for the deployment of the proposed algorithm for new inspection targets with minimal labeling efforts.

Author Contributions: Conceptualization, T.-M.T., W.-J.H. and Y.-J.J.; Methodology, C.-W.L. and L.Z.; Software, C.-W.L., L.Z. and C.-C.T.; Visualization, C.-W.L., L.Z and C.-C.T.; Validation, C.-C.T., T.-M.T. and Y.-J.J.; Supervision, T.-M.T. and W.-J.H.; Project administration, W.-J.H.; Writing—original draft, W.-J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was jointly funded by the Ministry of Science and Technology, Taiwan, and TUL Corporation under Grant MOST 110-2622-E-003-003.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations
The following abbreviations are used in this manuscript:

AE       Autoencoder
AUROC   Area Under Receiver Operating Characteristic
CNN     Convolutional Neural Network
FCN     Fully Convolutional Network
FPR     False Positive Rate
IoT     Internet of Things
ReLU    Rectified Linear Unit
ROC     Receiver Operating Characteristic
TPR     True Positive Rate

References
1. Szeliski, R. Computer Vision: Algorithms and Applications; Springer: London, UK, 2011.
2. Goodfellow, I.; Bengio, Y.; Courville, A. Deep Learning; MIT Press: Cambridge, MA, USA, 2016.
3. Xu, L.; Huang, Q. Modeling the interactions among neighboring nanostructures for local feature characterization and defect detection. IEEE Trans. Autom. Sci. Eng. 2012, 9, 745–754. [CrossRef]
4. Ren, R.; Hung, T.; Tan, K. C. A Generic Deep-Learning-Based Approach for Automated Surface Inspection. IEEE Trans. Cybern. 2018, 48, 929–940. [CrossRef] [PubMed]
5. Liu, K.; Wang, H.; Chen, H.; Qu, E.; Tian, Y.; Sun, H. Steel surface defect detection using a new Haar–Weibull-Variance model in unsupervised manner. IEEE Trans. Instrum. Meas. 2017, 66, 2585–2596. [CrossRef]
6. Bai, X.; Fang, Y.; Lin, W.; Wang, L.; Ju, B.F. Saliency-based defect detection in industrial images using phase spectrum. IEEE Trans. Ind. Inform. 2014, 10, 2135–2145. [CrossRef]
7. Heydarzadeh, M.; Nourani, M. A two-stage fault detection and isolation platform for industrial systems using residual evaluation. IEEE Trans. Instrum. Meas. 2016, 65, 2424–2432. [CrossRef]
8. Lee, T.; Lee, K. B.; Kim, K.O. Performance of machine learning algorithms for class-imbalanced process fault detection problems. IEEE Trans. Semicond. Manuf. 2016, 29, 436–445. [CrossRef]
9. Qiu, L.; Wu, X.; Yu, Z. A high-efficiency fully convolutional networks for pixel-wise surface defect detection. IEEE Access 2019, 7, 15884–15893. [CrossRef]
10. Perez, H.; Tah, J.H.M.; Mosavi, A. Deep Learning for Detecting Building Defects Using Convolutional Neural Networks. Sensors 2019, 19, 3556. [CrossRef] [PubMed]
11. Wang, C.C.; Jiang, B.C.; Lin, J.Y.; Chu, C.C. Machine vision-based defect detection in IC images using the partial information correlation coefficient. *IEEE Trans. Semicond. Manuf.* 2013, 26, 378–384. [CrossRef]

12. Elboher, E.; Werman, M. Asymmetric correlation: A noise robust similarity measure for template matching. *IEEE Trans. Image Process.* 2013, 22, 3062–3073. [CrossRef] [PubMed]

13. Wang, H.; Zhang, J.; Tian, Y.; Chen, H.; Sun, H.; Liu, K. A Simple Guidance Template-Based Defect Detection Method for Strip Steel Surfaces. *IEEE Trans. Ind. Inform.* 2019, 15, 2798–2809. [CrossRef]

14. Sakurada, M.; Yairi, T. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, Gold Coast, Queensland, Australia, 2 December 2014.

15. Youkachen, S.; Ruchanurucks, M.; Phatrapomnant, T.; Kaneko, H. Defect segmentation of hot-rolled steel strip surface by using convolutional auto-encoder and conventional image processing. In Proceedings of the 2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES), Bangkok, Thailand, 25–27 March 2019; pp. 1–5.

16. Yang, H.; Chen, Y.; Song, K.; Yin, Z. Multiscale Feature-Clustering-Based Fully Convolutional Autoencoder for Fast Accurate Visual Inspection of Texture Surface Defects. *IEEE Trans. Autom. Sci. Eng.* 2019, 16, 1450–1467. [CrossRef]

17. Corso, J.J.; Hager, G.D. Coherent Regions for Concise and Stable Image Description. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), San Diego, CA, USA, 20–25 June 2005.

18. Gould, S.; Gao, T.; Koller, D. Region-Based Segmentation and Object Detection. In Proceedings of Annual Conference on Neural Information Processing Systems (NIPS), Vancouver, Canada, 7–9 December 2009.

19. Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In Proceedings of the Computer Vision Pattern Recognition Conference (CVPR), Boston, MA, USA, 7–12 June 2015.

20. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015.

21. Gagliano, S.; Stella, G.; Bucolo, M. Real-Time Detection of Slug Velocity in Microchannels. *Micromachines* 2020, 11, 241. [CrossRef] [PubMed]

22. Youjachen, S.; Ruchanurucks, M.; Phatrapomnant, T.; Kaneko, H. Defect segmentation of hot-rolled steel strip surface by using convolutional auto-encoder and conventional image processing. In Proceedings of the 2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES), Bangkok, Thailand, 25–27 March 2019; pp. 1–5.

23. Yang, H.; Chen, Y.; Song, K.; Yin, Z. Multiscale Feature-Clustering-Based Fully Convolutional Autoencoder for Fast Accurate Visual Inspection of Texture Surface Defects. *IEEE Trans. Autom. Sci. Eng.* 2019, 16, 1450–1467. [CrossRef]

24. Bengio, Y.; Yao, L.; Alain, G.; Vincent, P. Generalized denoising auto-encoders as generative models. *arXiv* 2013, arXiv:1305.6663.

25. Ma, X.; Yao, T.; Hu, M.; Dong, Y.; Liu, W.; Wang, F.; Liu J. A Survey on Deep Learning Empowered IoT Applications. *IEEE Access* 2019, 7, 181721–181723. [CrossRef]

26. Yun, J. P.; Shin, W. C.; Koo, G.; Kim, M. S.; Lee, C.; Lee S. J. Automated defect inspection system for metal surfaces based on deep learning and data augmentation. *J. Manuf. Syst.* 2020, 317–324. [CrossRef]

27. Chollet, F. Keras. [Online]. Available online: http://github.com/fchollet/keras (accessed on 10 April 2021).

28. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* 2006, 27, 861–874. [CrossRef]