Complex & Intelligent Systems (2022) 8:1525–1537
https://doi.org/10.1007/s40747-021-00600-w

ORIGINAL ARTICLE

Deep learning based solder joint defect detection on industrial printed circuit board X-ray images

Qianru Zhang[1] · Meng Zhang[1] · Chinthaka Gamanayake[2] · Chau Yuen[2] · Zehao Geng[3] · Hirunima Jayasekara[2] · Chia-wei Woo[4] · Jenny Low[4] · Xiang Liu[3] · Yong Liang Guan[5]

Received: 24 July 2021 / Accepted: 26 November 2021 / Published online: 4 January 2022
© The Author(s) 2021

Abstract
With the improvement of electronic circuit production methods, such as reduction of component size and the increase of component density, the risk of defects is increasing in the production line. Many techniques have been incorporated to check for failed solder joints, such as X-ray imaging, optical imaging and thermal imaging, among which X-ray imaging can inspect external and internal defects. However, some advanced algorithms are not accurate enough to meet the requirements of quality control. A lot of manual inspection is required that increases the specialist workload. In addition, automatic X-ray inspection could produce incorrect region of interests that deteriorates the defect detection. The high-dimensionality of X-ray images and changes in image size also pose challenges to detection algorithms. Recently, the latest advances in deep learning provide inspiration for image-based tasks and are competitive with human level. In this work, deep learning is introduced in the inspection for quality control. Four joint defect detection models based on artificial intelligence are proposed and compared. The noisy ROI and the change of image dimension problems are addressed. The effectiveness of the proposed models is verified by experiments on real-world 3D X-ray dataset, which saves the specialist inspection workload greatly.

Keywords Joint defect detection · Deep learning · Automated X-ray inspection · Quality control

Introduction

With the improvement of electronic circuit production methods, the production speed of electronic products, such as mobile phones and notebook computers, is also increasing. At the same time, in the printed circuit board (PCB) assembly process, due to the reduction of component size and the increase of component density, the risk of defects is increasing [1]. Therefore, an efficient and accurate quality control system is essential.

✉ Meng Zhang
zmeng@seu.edu.cn

1 National ASIC Research Center, Southeast University, 2 Sipailou , Nanjing 210096, China

2 Singapore University of Technology and Design, Singapore 487372, Singapore

3 Peking University, Beijing 100871, China

4 Keysight Technologies, Singapore 768923, Singapore

5 Nanyang Technological University, Singapore 639798, Singapore

Structural defects are one of the main types of defects in assembled circuit boards, which includes insufficient solder, voids, and short circuits [2]. With the help of automatic detection method, it decreases the manual workload and mitigates the errors brought by specialist subjective judgments and specialist fatigue. Imaging-based automatic inspection methods such as optical imaging and thermal imaging are often used in the production line for quality control [3–6].

Compared with optical and thermal imaging, X-ray imaging can better reflect not only external but also internal defects. To simply put the mechanism, solder uses heavy materials such as silver, copper and lead, which are easily imaged under X-ray, while the light materials that form other components are not easily imaged. The amount of heavy materials can also be reflected from the X-ray imaging. Therefore, the solder quality can be checked on the X-ray images, externally and internally. Therefore, automated X-ray inspection (AXI) has been widely used in electronics manufacturing to monitor the quality of printed circuit board. For example, X-ray image features can be modeled and visualized in 2D space for defect detection. The detection was based on visualizing the deviation of the defect joint fea-

tures from the normal joint [7]. The X-ray images can also be used in 3D space to form a spatial model. By comparing spatial characteristics such as volume and propagation direction, defect solder joints can be filtered out [8]. However, such computerized tomography images require high computation complexity and take time to check one sample.

In recent years, deep convolutional neural networks (CNNs) have shown competitive performance in many fields such as image classification and object detection [9–11]. Image based features can be extracted efficiently by CNNs. Feature extraction is very important for imaging-based methods whether it is optical imaging, thermal imaging or X-ray imaging. In the field of solder joint inspection, some researches emphasize on extracting accurate features and propose innovative extracting methods on optical images, such as line-based-clustering method proposed by Gao et al. [4] and the reference-free path-walking method proposed by Jin et al. [5]. In terms of CNN-based methods, Wu et al. [12] propose to use mask R-CNN method for defect joint localization and detection. Cai et al. [13] propose to use three cascaded levels of CNNs for solder joint defect detection. As the number of levels increase, the performance gets better. However, the more complicated levels require more manual labeling and increase the manual workload. Goto et al. [14] propose to use adversarial convolutional autoencoder in combination with traditional anomaly detection method for defect detection. Their method performs well on an X-ray dataset with eight depths of imaging. But some real-world problems are not addressed such as varying number of slices and noisy X-ray images. In this paper, we focus on defect detection problem to reduce manual workload. Real-world problems are addressed during the detection process. The specific problem formulation is illustrated in the following section.

Problem formulation

In the AXI, advanced technologies are used for solder joint localization and inspection. When defect solder joints are detected by the machine, they are sent to specialists for a second round of inspection. However, sometimes the machine inspection is not accurate, which sends a large number of normal solder joints to the specialists that increases the manual workload. The merchandised AXI machine statistics is rarely discussed in the researches. We calculated the statistics based on a real-world dataset we obtained. Among 518,292 through hole solder images that are labeled by an AXI machine as defects, only around 15% are truly defects as labeled by specialist. Around 85% of solder joints are normal, but also sent to the specialists for further inspection. In this work, the main problem is to detect true defect solder joints from misclassified dataset and to reduce the manual workload.

To achieve the target, two problems need to be addressed, namely noisy region of interest (ROI) and varying number of slices. Current researches for X-ray imaging-based solder defect detection rarely address the noises in the X-ray imaging dataset. Some of the ROIs extracted by AXI are incorrect and fail to enclose the solder joint area, which deteriorates solder defect detection. In addition, there are varying numbers of slices. Some depths of the solder are ignored during data collection, as they may not be necessary for human inspection. Therefore, the number of slices is not the same for all solder joints. Although there are some deep learning-based methods implemented in PCB defect detection as mentioned, few researches focus on X-ray imaging and addressing the varying number of slices.

In general, the problems can be summarized as follows:

– AXI false call increases specialist workload.
– Incorrect ROI from AXI deteriorates solder defect detection.
– Varying number of slices makes it difficult to model.

In this work, deep learning methods are incorporated since they are suitable for high-resolution image-based tasks. We mitigate the AXI machine inefficiency problem by proposing deep learning-based inspection module in the production line. There are two ways of viewing the 3D X-ray images, which decides the way of feature extraction. First one is to view each slice separately and to straightforwardly extract features from each slice. Second one is to extract features from all slices and conduct defect detection-based on the extracted features.

Two pre-processing methods are proposed. First one is to straightforwardly stack each slice of a solder joint. Second one is to treat all slices of a solder joint together and to manipulate them in a statistic way. The dedicatedly designed pre-processing methods can address the varying number of slices problem and the incorrect ROI problem.

In the deep learning structure design, three deep learning model structures based on CNN, 3D CNN and long short time memory (LSTM) are proposed that suit the pre-processing methods. The combination of pre-processing methods and deep learning model structures results in four deep learning-based models. Since few deep learning-based methods address the varying numbers of slices problem and the incorrect ROI problem in PCB X-ray imaging, the performance is compared among each other, which can help with the researches with similar problems. Our features of the proposed method are summarized as follows:

– An AI-assisted inspection in production line is proposed that can dramatically reduce specialist workload.
– Varying number of slices problem and the incorrect ROI problem of X-ray imaging are addressed.

– Activation feature map is presented for visualizing and analyzing the pre-processing method.
– Three deep learning structures are proposed that suit the pre-processing methods.

The proposed methods have good generalization performance and its efficacy is verified through experimenting on a very large real-world 3D X-ray dataset. By incorporating the AI-based models, 73.29% normal joints are filtering out and only 26.71% of normal joints are sent to the specialists for inspection, which reduces the specialist workload dramatically. The proposed methods are introduced in Section “Methodology”. The methods detail in Sections “Pre-processing Methods” and “Deep Learning Model structures”. Experiment and performance results are presented in Section “Experiment and results” and we conclude the work in Section “Conclusion”.

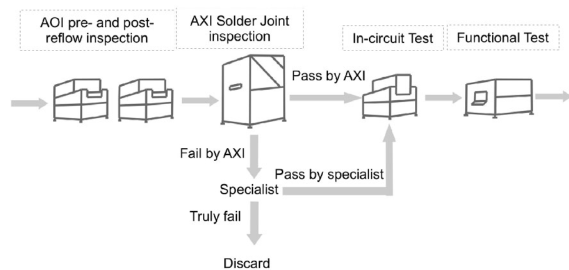
Methodology

An electronic production line is shown in Fig. 1. Automated optical inspection (AOI), automated X-ray inspection (AXI), in-circuit test and functional test are common tests for quality control. Our work focuses on AXI, which detects defect solder joints based on X-ray imaging. Usually, solder joints that are detected as defect ones by AXI are sent to specialist for follow-up inspection as shown in Fig. 1a. However, when AXI generates too many false calls, specialists need to check many normal solder joints that are misclassified as defect. Our work mitigates the inefficiency problem of AXI by introducing deep learning-based methods. The new flow is as shown in Fig. 1b. For the solder joints that are detected as defect by the AXI, they are sent to our methods first. The detected defect solder joints by our methods are sent to specialists, while normal joints are sent for next-step tests.

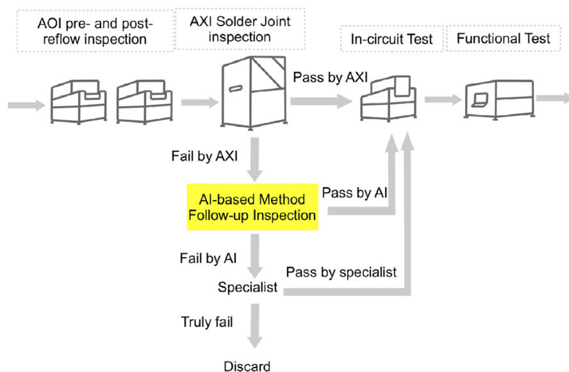
3D automatic X-ray inspection

To better understand the reasoning of the methods, some background of the 3D automatic X-ray inspection (AXI) system is introduced.

The 3D X-ray system can obtain different imaging depths of the PCB underneath the X-ray source with synchronized rotating source and the detector. The component joints on the focus plane would be sharp, while the ones above or below the focus plane would be blurred. Therefore, different slices regarding different depths of the PCB are obtained. These slices are important for joint defect detection as they show the hidden features of the joints. An example with four slices is shown in Fig. 2. Different slices are for different blur levels. Although the larger slice number does not necessarily mean deeper depth, the depth and slice numbering are consistent. There is a maximum number of slices, which means during data collection, all depths of imaging slices are collected from the AXI machine. However, for some solder joints, all slices

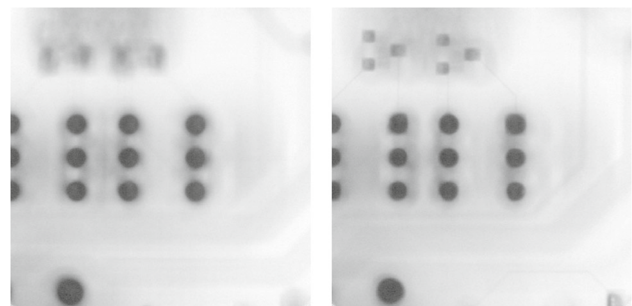


(a) Current inspection in production line

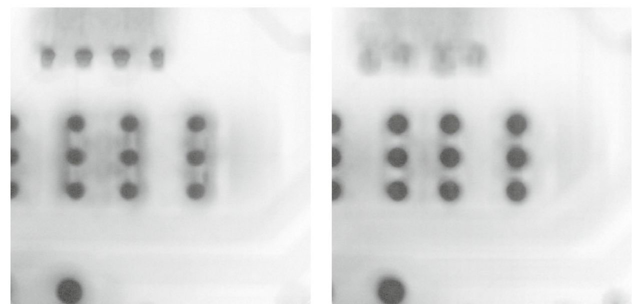


(b) Proposed AI-assisted inspection in production line

Fig. 1 A production line with follow-up inspection after AXI



(a) Slice 00 **(b)** Slice 01



(c) Slice 02 **(d)** Slice 03

Fig. 2 An example of 3D X-ray imaging with four slices

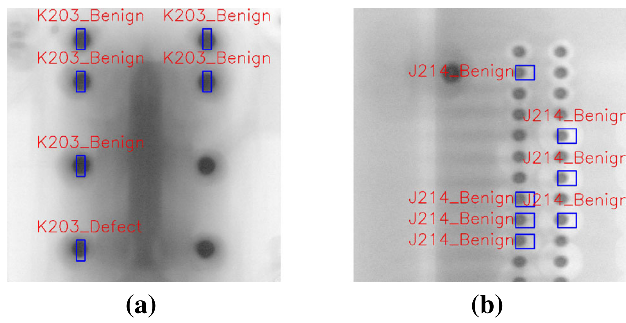


Fig. 3 An example of AXI with ROIs

are not necessary for specialist inspection. Therefore, fewer numbers of slices are collected. For example, the maximum number of six imaging slices could be obtained from the AXI machine, namely [00, 01, 02, 03, 04, 05]. A collected solder joint slices only have three of them, namely [00, 01, 02]. In our case, although some solder joint slices are ignored, the ignored slices would always be the last slices.

More slices provide more information for one solder joint. However, the label is given based on solder joint. That is to say, if one of the slices for the solder joint is defective, all the slices are labeled as defective. It is impossible to know which slices are defective. Therefore, all slices for one solder joint need to be treated together as one input. And different inputs have different number of slices.

Along with the X-ray images, AXI also provides a bounding box of the inspected solder joint called region of interest (ROI). The ROI consists of two sets of coordinates of the upper-left and lower-right corners of the ROI. Ideally, ROIs not only provide the location of the inspected solder joint, but also the area of the inspected solder joint. However, some ROIs are not reliable. For example, ROIs fail to enclose the whole part of the solder joint due to the small size of ROI in Fig. 3a. It is supposed to be a square and the short side should have been the same length as the long side. Or the ROI shifts horizontally in Fig. 3b. The size of the ROI is correct, but due to the shift, the ROI could not enclose the whole part of solder joint. Although one edge of the ROI cuts through the solder joint, most area of the solder joint is within the ROI.

Proposed framework

There are two phases, namely training phase and implementing phase, in the proposed framework as shown in Fig. 4.

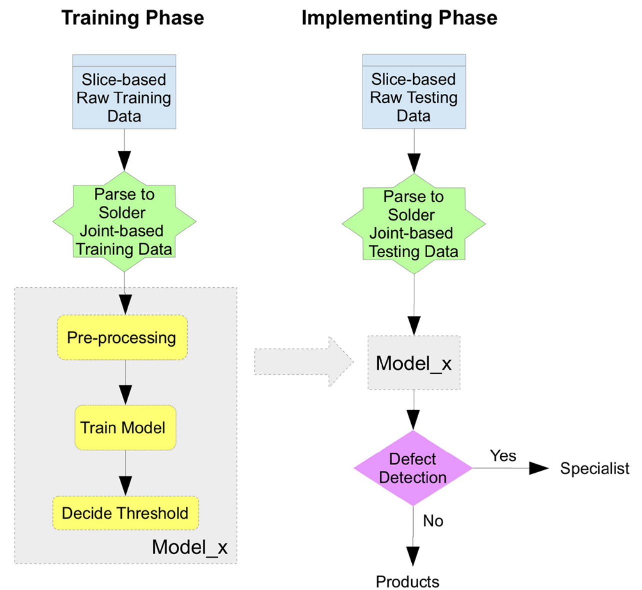


Fig. 4 Proposed framework

During the training phase, raw mega data, such as joint type, board type, ROI, and X-ray image file path, from the machine is parsed first and prepared. Also, slice-based image data are parsed into solder joint-based image data. Then the image file is pre-processed before model training. To reduce false normal solders that should have been detected as defect solders, thresholding is empirically selected.

During the implementing phase, the new incoming parsed and pre-processed data are sent to the trained model and compared with the threshold. Detected defects are passed to the operator for further analysis and decision-making.

Under this framework, four models are proposed. The proposed models have different pre-processing methods and different AI model structures as shown in Table 1. Two pre-processing methods are proposed and dedicated designed, namely ROI mask pre-processing and channel-wise preprocessing. ROI mask pre-processing method treats all slices of a solder joint together and to manipulate them in a statistic way. Channel-wise preprocessing method straightforwardly stacks each slice of a solder joint. Three deep learning structures are proposed that fall into the categories of CNN, 3D CNN and, LSTM. In Model A, the slices are merged during pre-processing and features are extracted based on the statistics of the slices. For Model B and Model C, the Channel-wise

Table 1 Model description

Model name	Pre-processing method	Deep learning model structure
Model A	ROI mask	CNN structure
Model B	Channel-wise	CNN structure
Model C	Channel-wise	3D CNN structure
Model D	Channel-wise	LSTM-based structure

pre-processing method contains the original images and features of all slices are extracted together by CNN. For Model D, each slice feature is extracted by one CNN. The extracted features are combined together using LSTM module. In the following sections, each pre-processing method and model structure are illustrated in detail.

Pre-processing methods

Varying numbers of slices and noised ROIs are the two problems to handle. Based on the AXI characteristics mentioned in Section "3D Automatic X-ray Inspection", there are different numbers of slices for different solder joints, since some can be detected with deep depth of imaging, while some need both deep and shallow depths of imaging. The other problem is that some ROIs are not correct, which cannot surround the area of the joint of interest. We address the two problems in different ways and propose two pre-processing methods, ROI mask pre-processing and Channel-wise pre-processing.

ROI mask pre-processing

During pre-processing step, we propose to do third-dimension padding, variance feature, ROI mask, and cropping. We pad slices with zero value, add a variance feature image, add a ROI mask, and crop to generate the three-channel input.

Since there are varying numbers of slices, to incorporate all the information from the slices, we propose to pad zero-slices on the third-dimension to the maximum number of slices as shown in Fig. 5. The reason for padding zeros is that it gives all the solder joint a base-line slice, which helps in the following variance feature pre-processing.

Variance feature is extracted by calculating pixel-wise variance across the third dimension. The form of variance feature is also an image such that each pixel in the corresponding location is the brightness variance for all the slices. By introducing the variance feature image, the information from all the slices can be used.

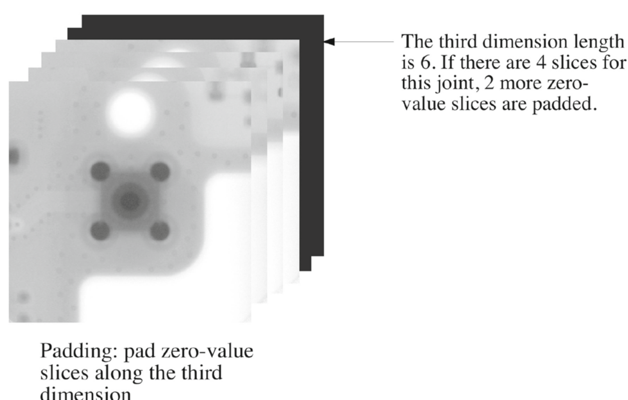


Fig. 5 Third-dimension padding

There are mostly more than one solder joint on one image and ROI is an indicator for the location of the solder joint. As some ROIs are not reliable and cannot enclose the whole solder joint, solder joint patches cannot be correctly cropped using ROI. As a result, we include an ROI mask in the input to indicate the solder joint location.

Input: Original image dataset
$\mathcal{D}_{image} = \{s_n^i; i\ is\ slice\ number,\ n\ is\ solder\ number\}$,
ROI = $\{xmin, xmax, ymin, ymax\}$
Output: Solder joint patch dataset $\mathcal{D}_{patch}$, cropping ROI = $\{cxmin, cxmax, cymin, cymax\}$

1 $\mathcal{D}_{patch} \leftarrow \emptyset$;
2 **for** $k \leftarrow 1\ to\ n$ **do**
3 **if** $s_k^i, i \in [1, \text{SLICE}_{\text{MAX}}]$ *not exists* **then**
4 Pad zero-slice to s_k^i;
5 **end**
6 $channel_1 = s_k^1$;
7 $channel_2 = Variance(s_k^1, s_k^2, ..., s_k^6)$;
8 $channel_3 = 0$;
9 $channel_3[ymin : ymax, xmin : xmax] = D_{ROI}$;
10 $center_k(x, y) = (\frac{xmax-xmin}{2} + xmin, \frac{ymax-ymin}{2} + ymin)$;
11 $cropLength = \text{L}_{\text{CROP}}$;
12 **if** $\max(xmax - xmin, ymax - ymin) \geq \text{T}_{\text{SMALL}}$ **then**
13 $cropLength =$ $\alpha \times \max(xmax - xmin, ymax - ymin)$
14 **end**
15 $(cxmin, cymin) = center_k(x, y) - \frac{cropLength}{2}$;
16 $(cxmax, cymax) = center_k(x, y) + \frac{cropLength}{2}$;
17 **if** $cxmin < 0$ **then**
18 $cxmin = 0$;
19 $cxmax = cxmax + |cxmin|$
20 **end**
21 **if** $cxmax > \text{L}_{\text{SLICE}}$ **then**
22 $cxmax = \text{L}_{\text{SLICE}}$ $cxmin = cxmin - |cxmax - \text{L}_{\text{SLICE}}|$
23 **end**
24 Repeat step $17 - 23$ for y axis cropping ROI;
25 $patch_k$ = Crop $channel_{1-3}$ according to cropping ROI=$\{cxmin, cxmax, cymin, cymax\}$;
26 Resize $patch_k$ to $S_{input} \times S_{input}$;
27 Add $patch_k$ to $\mathcal{D}_{patch}$;
28 **end**

Algorithm 1: ROI mask pre-processing algorithm

The ROI mask pre-processing method details in Algorithm 1. The slices are first padded zero-slices on the third-dimension to the maximum number of slices $\text{SLICE}_{\text{MAX}}$ ready for variance feature extraction. Then the first slice, the variance feature, and the ROI mask are assigned to the three channels, respectively. In $channel_3$, the density of the ROI mask $\text{ROI}_{\text{density}}$ is set to 10 empirically. Cropping needs to be conducted on the three channels to obtain small patches. On the one hand, one image is very big for the joint of interest and

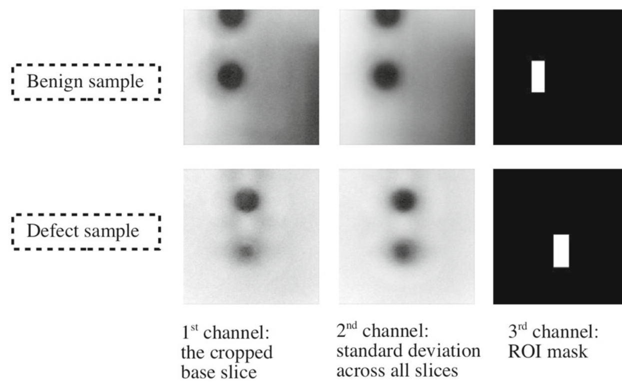


Fig. 6 ROI mask pre-processing sample

contains some background that will deteriorate the detection process. On the other hand, adjacent area for the joint of interest can provide information for detection. Therefore, small patches that contain the joint of interest as well as the adjacent area cropping from the image is helpful. The images are cropped into patches that can contain adjacent solder joints. For each X-ray image, we obtain in the dataset, the number of patches is dependent on the number of machine ROI. For example, in Fig. 3a in Section "3D Automatic X-ray Inspection", there are six solder joints of interests. Therefore, six patches are cut from this image. For the small solder joint of interest in the area of high solder joint density, the cropped size $L_{CROP} = 340$ pixels based on the dataset statistics and the suggestions from the specialists. The cropped patch may contain the adjacent solder joint. The way to distinguish the solder joint of interest and the adjacent solder joint is the ROI mask, which is $channel_3$ in the pre-processing method. The ROI mask helps locate the solder joint of interest by highlighting the original ROI area. For a few large joints of interest, the image is cropped based on the enlarged length of ROI with a factor α, which is an empirical factor provided by the specialist. The cropping strategy is solder joint center-priority, but the solder joint is not necessarily in the center of the cropped patch. From step 17–23, the cropping ROIs are calibrated if they are outside the range of the slice image. When the cropping area is outside the image, cropping coordinates are moved towards the center so that no padding for the original image is needed. At last, $patch_k$ is cropped based on the cropping ROI=$\{cxmin, cxmax, cymin, cymax\}$, and resized to 224×224, which is an empirical input size of the CNN model structure.

The output of pre-processing consists of three channels as shown in Fig. 6. The first channel is the common base slice that is called "slice 00" mentioned in Section "3D Automatic X-ray Inspection". The second channel is the variance for all the slices including the padded zero-value slices and the third channel is the ROI mask.

Channel-wise pre-processing

Different from the ROI mask pre-processing that incorporates an ROI mask and the variance features, a straightforward pre-processing method is proposed by concatenating all the slices in channel-wise direction before sent into the model.

The channel-wise pre-processing algorithm is shown in Algorithm 2. At first, since there are varying number of slices, zero-slices are padded to the maximum number $\mathrm{SLICE}_{\mathrm{MAX}}$, which is six in our case. As the dataset not only include the focused solders, but also the surrounding solders as well as the background, an ROI-based cropping is implemented during the pre-processing step to reduce the size of input. The cropping length $cropLength$ is obtained based on the enlarged length of ROI with a factor α, and in this case, it is 1.5 based on specialist suggestion. Then the slices are cropped based on cropping ROI =$\{cxmin, cxmax, cymin, cymax\}$ that is calculated by step 6–9. Different from ROI mask pre-processing method, if the cropping area is outside the slice s_k^i, zeros are padded around the empty area. In this way, the solder joint of interest is centered, which is better for detection since there is no information to indicate solder joint location. At last, patch $patch_k$ is resized into 128×128 that can be efficiently used by the CNN structure proposed in the following sections.

The output of pre-processing consists of six channels as shown in Fig. 7. Each channel represents one slice of the

Input: Original image dataset
$\mathcal{D}_{image} = \{s_n^i; i \text{ is slice number}, n \text{ is solder number}\}$,
ROI = $\{xmin, xmax, ymin, ymax\}$
Output: Solder joint patch dataset $\mathcal{D}_{patch}$, cropping ROI = $\{cxmin, cxmax, cymin, cymax\}$

1 $\mathcal{D}_{patch} \leftarrow \emptyset$;
2 **for** $k \leftarrow 1$ *to* n **do**
3   **if** $s_k^i, i \in [1, \mathrm{SLICE}_{\mathrm{MAX}}]$ *not exists* **then**
4   Pad zero-slice to s_k^i;
5   **end**
6   $center_k(x, y) = (\frac{xmax-xmin}{2} + xmin, \frac{ymax-ymin}{2} + ymin)$;
7   $cropLength = \alpha \times \max(xmax - xmin, ymax - ymin)$;
8   $(cxmin, cymin) = center_k(x, y) - \frac{cropLength}{2}$;
9   $(cxmax, cymax) = center_k(x, y) + \frac{cropLength}{2}$;
10   $patch_k$ = Crop s_k^i according to cropping ROI=$\{cxmin, cxmax, cymin, cymax\}$, $i \in [1, \mathrm{SLICE}_{\mathrm{MAX}}]$;
11   **if** $(cxmin \text{ or } cymin) \leq 0$ *or* $(cxmax \text{ or } cymax) \geq \mathrm{L}_{\mathrm{SLICE}}$ **then**
12   pad zeros on $patch_k$;
13   **end**
14   Resize $patch_k$ to $S_{input} \times S_{input}$;
15   Add $patch_k$ to $\mathcal{D}_{patch}$;
16 **end**

Algorithm 2: Channel-wise pre-processing algorithm

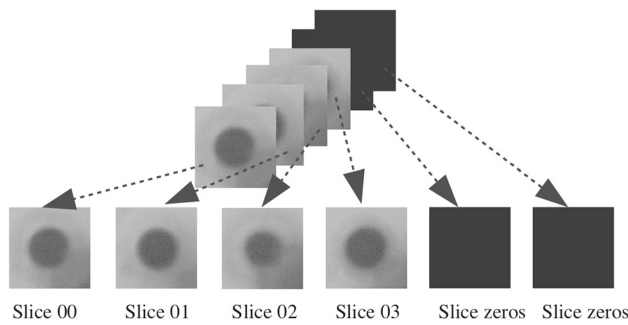


Fig. 7 Channel-wise pre-processing sample

solder joint. For the slice number that is less than six, zero-value slices are padded.

Deep learning model structures

Three deep learning model structures are proposed in this work. First one is designed as a regular CNN structure. Second one is designed based on 3D convolution and third one is designed with LSTM module.

CNN structure

For a regular CNN structure, 2D convolution with activation function is used to extract features from high-dimension images as shown in Eq. (1), where C is the input feature channel size, K is the kernel weight size, b is the bias and σ is the activation function. The architecture of the neural network is illustrated in Table 2. It consists of convolutional layers, batch normalization layers, max pooling layers, and global average pooling layers. The concrete structure is shown in Fig. 8. 3×3 kernel has few parameter size and is used in the convolutional layer of the proposed structure. As by stacking 3×3 convolution, various reception fields can be achieved [15]. As the structure depth becomes deeper, parameters become more and training time becomes longer. Batch normalization layer is used to increase training speed since distribution shifting problem in each layer is mitigated. It is defined in Eq. (2) and Eq. (3), where μ_{B} and σ_{B}^2 are the mean and variance for each batch training, and γ and β are trained parameters [16]. The global average pooling layer [17] is introduced. It helps reduce a large amount of parameters in the following fully connected layers and helps for visualizing the feature maps that will be discussed in Section "Results". The kernel size of the convolutional layer is 3×3, which is the smallest size of capturing all the directions [15]. Its input channel size is designed as three for ROI mask pre-processing and six for channel-wise pre-processing.

$$f_{2\mathrm{D}} = \sigma\left(\sum_{n=0}^{C}\sum_{i=0}^{K}\sum_{j=0}^{K} w_{ij,c}x + b\right) \tag{1}$$

$$\overline{x_i} = \frac{x_i - \mu_{\mathrm{B}}}{\sqrt{\sigma_{\mathrm{B}}^2 + \epsilon}} \tag{2}$$

$$y_i = \gamma\overline{x_i} + \beta. \tag{3}$$

3D CNN Structure

For the channel-wise pre-processing method, since the correlation information is not extracted during the pre-processing

Table 2 CNN structure

Layer name	Output shape	Kernel/weight size
Conv2D×2	224 × 224 × 64	3 × 3 × 64
Batch normalization	224 × 224 × 64	–
Max pooling	112 × 112 × 64	–
Conv2D×2	112 × 112 × 128	3 × 3 × 128
Batch normalization	112 × 112 × 128	–
Max pooling	56 × 56 × 128	–
Conv2D×3	56 × 56 × 256	3 × 3 × 256
Batch normalization	56 × 56 × 256	–
Max pooling	28 × 28 × 256	–
Conv2D×3	28 × 28 × 512	3 × 3 × 512
Batch normalization	28 × 28 × 512	–
Conv2D×3	28 × 28 × 512	3 × 3 × 512
Global average pooling	512	–
Fully connected	2048	512 × 2048
Dropout	2048	–
Fully connected	1	2048 × 1

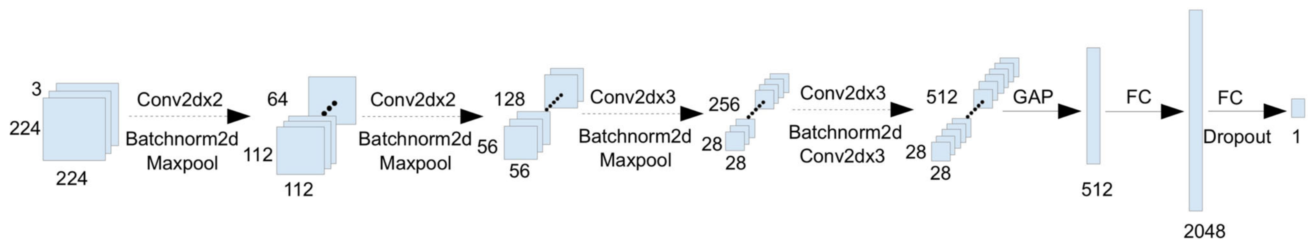


Fig. 8 CNN structure

Table 3 3D CNN structure

Layer name	Output shape	Kernel/weight size
Input	$128 \times 128 \times 6 \times 1$	–
Conv3D	$126 \times 126 \times 5 \times 8$	$3 \times 3 \times 2 \times 8$
Conv3D	$124 \times 124 \times 4 \times 16$	$3 \times 3 \times 2 \times 16$
Max pooling	$62 \times 62 \times 2 \times 16$	–
Conv3D	$60 \times 60 \times 1 \times 32$	$3 \times 3 \times 1 \times 32$
Conv3D	$58 \times 58 \times 1 \times 64$	$3 \times 3 \times 1 \times 64$
Max pooling	$29 \times 29 \times 1 \times 64$	–
Batch normalization	$29 \times 29 \times 1 \times 64$	–
Dropout	53824	–
Fully connected	1024	53824×1024
Dropout	1024	–
Fully connected	2	1024×2

phase, it is more reasonable to increase the feature extraction among channels for the CNN structure. Compared with a 2D convolution, 3D convolution uses a 3D tensor and is more flexible for third-dimension feature extraction. A 3D convolution is defined in Eq. (4), where K is the kernel weight first- and second-dimension size that shares the same meaning with that of the 2D kernel, and Q is the kernel weight third-dimension size. For a 3D kernel, its first and second dimensions usually remain the same, which extract the spatial feature, while the third-dimension size is not necessarily the same. In 3D convolution, kernel w_{ijq} is a 3D tensor that moves along three axes to extract features of the input. C is the input feature channel size, and b is the bias and σ is the activation function.

There are multiple slices for one solder joint, and the solder joint slices are stacked along the third dimension during the pre-processing process. The 3D CNN architecture is designed to extract features that is fit for the pre-processing. It is detailed in Table 3. During the first two stacked 3D convolutional layers, the kernel is 3×3 in the spatial dimension and is 2 in the third dimension. 3×3 kernel is commonly used for extracting spatial feature and by stacking them, larger receptive field can be achieved. In the third dimension, kernel size is 2, which means the adjacent slice information is focused. But the extraction is not limited to adjacent slices, because the kernel moves along the third dimension and convolves with each slice as it does in the 2D space. In the following two stacked 3D convolutional layers, $3 \times 3 \times 1$ kernel is used, because it needs to fit for the output feature from the previous layer. For example, the output of the first max-pooling layer is $60 \times 60 \times 1 \times 32$, which is also the input of the following 3D convolutional layer. Its third dimension is 1, and therefore, the third dimension size of the 3D kernel needs to be designed as 1. Besides 3D convolutional layer, max-pooling and batch normalization layers are also introduced. The 3D CNN structure not only extracts the spatial features from each slice in our case, but also the relationship among multiple slices are reserved.

$$f_{3\mathrm{D}} = \sigma\left(\sum_{n=0}^{C}\sum_{q=0}^{Q}\sum_{i=0}^{K}\sum_{j=0}^{K} w_{ijq,c}x + b\right) \tag{4}$$

LSTM-based structure

LSTM neural network is one of the recurrent neural networks that can model a sequence of inputs. By inserting the LSTM module with a forget gate, an input gate, and an output gate that are defined in Eqs. (5), (6) and (7) respectively, the module cell can implement discarding, storing, and activating information from the last timestamp state. The module cell state is updated according to Eqs. (8), (9) so that it is able to capture the relationship within the sequential inputs. And the

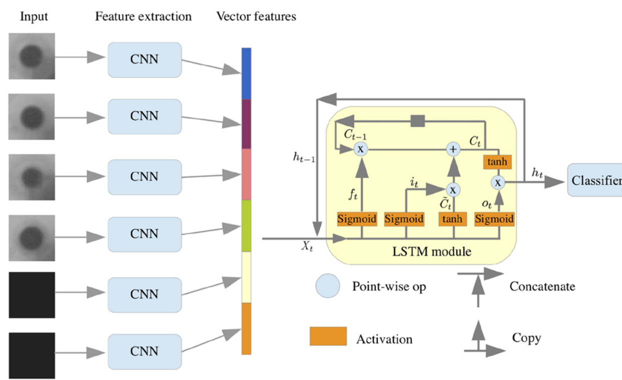


Fig. 9 LSTM-based structure illustration

Table 4 LSTM-based structure

Layer name	Output shape	Kernel/Weight size
LSTM	2048	–
Fully connected	512	2048 × 512
Dropout	512	–
Fully connected	2	512 × 2

final output of the LSTM module is defined as Eq. (10). The gates inside the module help remember not only the adjacent relationship but also the long term relationship between the inputs.

$$f_t = \text{Sigmoid}\left(W_f \cdot \left[h_{t-1}, X_t\right] + b_f\right) \quad (5)$$

$$i_t = \text{Sigmoid}\left(W_i \cdot \left[h_{t-1}, X_t\right] + b_i\right) \quad (6)$$

$$o_t = \text{Sigmoid}\left(W_o \left[h_{t-1}, X_t\right] + b_o\right) \quad (7)$$

$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, X_t\right] + b_C\right) \quad (8)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (9)$$

$$h_t = o_t * \tanh\left(C_t\right). \quad (10)$$

For our case, slices from 00 to 05 are the sequential inputs. Since there are six slices, the LSTM module continuously computing six times until all the slices are processed. Specialist compares the solder joint difference among different slices and the changes can indicate the joint defection. Therefore, it is expected that LSTM based model structure can capture the relationship within different slices for defect detection.

Since images are high-dimension inputs, features are extracted by CNNs from each slice before sent into the LSTM as shown in Fig. 9. The output size of the features extracted from the CNNs is 2048×6. Each vector consists of the information from each slice with dimension 2048. The sequence length of the LSTM is 6, as six slices compose of the solder joint and share one label. The output The LSTM module is followed a classifier composed with two fully connected layers, which are used for classification as shown in Table 4. The output size of the LSTM is designed empirically as 2048, which squeezes the information from six successive slices. The classifier is composed of two fully connected layers. Dropout is introduced to increase the generalization performance.

Experiment and results

We implement the method using Python Keras and TensorFlow, and the parameter settings are summarized in Table 5. Adam is used as our optimization method. The weights are initialized using Glorot uniform distribution to get a faster convergence, where the upper and lower limits of the uniform distribution are based on the number of input and output unit amount of the current layer [18]. The activation function is the rectified linear function (ReLU) [9]. The learning rate for Model A and Model B in Table 1 is set $1e-4$. And the learning rate for Model C and Model D in Table 1 is set $1e-5$ with decay $1e-6$. After several trials reducing from $1e-2$ [9], the learning rate is set based on the each model's performance. The trainable parameters of the models are summarized in Table 6.

Dataset and metrics

The dataset is in two forms, which are image dataset and the mega dataset. In the image dataset, each image is a grayscale X-ray image of a part of PCB as shown in Fig. 2. Each image is uniquely identified by the image path. In the mega dataset, it consists of information, such as image path, board number, slice number, joint type, machine defect, ROI, label, etc. Each solder joint can be uniquely identified by the image path together with the ROI. Each sample is a solder joint that is

Table 5 Training setting

Model	Weight initialization	Activation function	Optimizer	Learning rate
Model A	Glorot uniform distribution	ReLU	Adam	$1e-4$
Model B				$1e-4$
Model C				$1e-5$
Model D				$1e-5$

Table 6 Trainable parameters

Model	# Trainable parameters
Model A	15, 769, 281
Model B	15, 771, 009
Model C	55, 149, 194
Model D	34, 612, 738

Table 7 Dataset slice distribution

Pin through hole solder	
Slice distribution	1 slice: 763
	2 slices: 1694
	3 slices: 7576
	4 slices: 466029
	5 slices: 41345
	6 slices: 945
Normal-defect ratio	1:0.18

Table 8 Dataset statistics

Dataset	Benign:Defect
Training	37579:37948
Testing	206800:33177
Validation	2000:2000

cropped according to the proposed pre-processing algorithm. It is noted that the specialist labels all slices of one solder joint as either defective or not. There is no granularity whether or not the individual slices are defective or not.

There are two problems to be solved in our work. First one is that the number of slices vary for different solder joints. In this dataset, different slices represent different blur levels. The slice numbering is not necessarily in any order like top down. But the depth and slice numbering are consistent. The maximum number of slices is six. There are varying numbers of slices for some solder joints, and the ignored slices are always the last ones. The second problem is that ROI provided by the machine is noisy. Some ROIs are rectangular. One side of the ROI cuts through the solder joint. Some ROIs shift from the center of the solder joints, which also results in failing to enclose the whole solder joint. The noisy ROIs usually happen in small solder joints with incorrect ROI less than around 100 pixels.

The dataset slice distribution and normal–defect ratio shown in Table 7, and the dataset statistics is summarized in Table 8. Training dataset, validation dataset and testing dataset are divided based on board types for testing generalization performance. That is to say the board types in the testing phase are not seen during training the models to mimic the real-world case. Since validation dataset is different from testing dataset, if the model performance on validation dataset is similar to that on testing dataset, its generalization performance is better. In the real-world scenario, benign solder joints are far more than defect solder joints. Training on the imbalanced dataset leads to poor performance as there would be a bias towards the majority class, which makes it difficult to detect the minority class [19,20]. To deal with data imbalanced problem, we down-sample the benign data to get a balanced training dataset. As the absolute amount of the defect training data is very large and there are some repeated patterns in the dataset, down-sampling is implemented. In the future work, dedicated designed data augmentation method could be further explored for better performance.

Accuracy is one of the commonly used metrics. However, under the condition of imbalanced problem, as the test data include large amount of benign data and small amount of anomaly data, accuracy is not fair for performance checking. For example, one method can classify all the solder joints as benign, which will give a very high accuracy rate. But it cannot detect any defect solder joint. Instead, Recall, false positive rate (FPR), and area under the receiver operating characteristic (AUROC) are commonly used.

For a binary classifier labeled as either positive and negative, there are four outcomes given an instance: true positive (TP); false positive (FP); false negative (FN); true negative (TN). The true positive rate (TPR) is defined as correct positive results that happen in all positive samples, which is also called Recall. FPR is defined as incorrect positive results that happen in all negative samples. In our case, defect solder joint is defined as positive sample and normal solder joint is defined as negative one. There is a tradeoff between Recall and FPR. At a given Recall, the smaller FPR the better. Recall and FPR are calculated based on fixed discrimination threshold. Therefore, it cannot reflect the dynamic relationship between each other.

The receiver operating characteristic (ROC) curve is a commonly used tool to analyze and visualize the performance of a binary classifier as its discrimination threshold is varied, which results in a tradeoff between TPR and FPR. ROC curve takes FPR as its horizontal axis and takes TPR as its vertical axis. The points in the graph depict the relationship between TPR and FPR. Classifier with upper left conner points has better overall performance. The area under the ROC is AUROC that measures the general performance regardless of the thresholds.

Compared with metric of accuracy, which measures the proportion between the sum of TP and TN and the total number of instances, Recall, FPR, and AUROC are more precise as they take imbalance classes of data into consideration.

Results

Results of four models shown in Table 9. From the table, there is a tradeoff between Recall and FPR. As threshold decreases, the model can achieve higher Recall and FPR increases, respectively. For different models, the threshold has different impacts. For example, for Model A, when threshold is 0.1, on validation dataset, its Recall = 0.9376, while for Model D, Recall = 0.9826.

As the model generalizes from validation dataset to the testing dataset, there is around 2–5% drop for Recall for

Table 9 Model performance

Threshold	Metrics	Model A		Model B		Model C		Model D	
		Validation	Testing	Validation	Testing	Validation	Testing	Validation	Testing
0.1	Recall	0.9376	0.9027	0.9877	0.9293	0.9795	0.9200	0.9826	0.9556
	FPR	0.2900	0.3247	0.4575	0.4799	0.4204	0.4240	0.4745	0.5568
0.2	Recall	0.9020	0.8713	0.9677	0.9124	0.9457	0.8884	0.9535	0.9134
	FPR	0.2155	0.2415	0.3365	0.3370	0.3018	0.2798	0.3597	0.3574
0.3	Recall	0.8742	0.8525	0.9456	0.8990	0.9149	0.8654	0.9213	0.8754
	FPR	0.1772	0.2003	0.2547	0.2611	0.2248	0.2058	0.2564	0.2433
0.4	Recall	0.8499	0.8369	0.9222	0.8867	0.8797	0.8402	0.8830	0.8508
	FPR	0.1521	0.1729	0.2003	0.2127	0.1729	0.1582	0.1849	0.1800
0.5	Recall	0.8251	0.8198	0.8969	0.8719	0.8445	0.8146	0.8480	0.8261
	FPR	0.1316	0.1492	0.1609	0.1740	0.1365	0.1255	0.1378	0.1362

Table 10 Threshold comparison

Threshold (val)	Model	Recall	FPR
0.20	Model A	0.8713	0.2415
0.48	Model B	0.8749	0.1809
0.34	Model C	0.8556	0.1849
0.28	Model D	**0.8836**	0.2908

Bold value indicates best result in the comparison

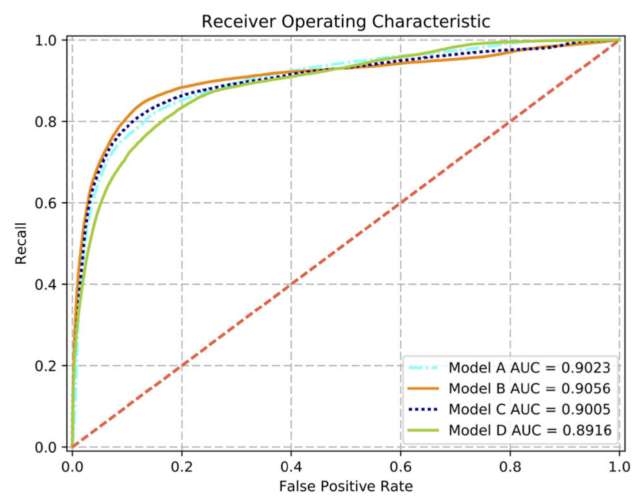


Fig. 10 ROI mask model overall performance

all the models. As for FPR, Model A and Model B have around 1–3% increase, while Model C and Model D have both increase and decrease statistics when generating from validation dataset to the testing dataset. The empirical threshold can be picked based on the validation dataset. In Table 10, given the threshold that makes the validation dataset Recall = 0.90, the highest Recall = 0.8836 for testing dataset is achieved by Model D with a drop of $\sim 2\%$. Among the four models, when the validation Recall = 0.90, Model D has the best detection generalization performance.

The overall performance of the proposed models can be seen in Fig. 10. Regardless of thresholds, all the models perform similarly with AUROC ranging from 0.8916 to 0.9056. However, there are some fine-grained differences among the four models. As Recall increases to 1.00, Model A and Model D outperform the other two models with lower FPR. And as Recall decreases, Model B and Model C perform better with lower FPR.

For the deep learning model structure comparison, since Model B, Model C, and Model D share the same preprocessing method (channel-wise preprocessing method), the comparison among the three models indicates the performance comparison among CNN structure, 3D CNN structure, and LSTM-based structure. From Fig. 10, we can see that Model B performs the best with lower Recall, and Model D performs the best with higher Recall among Model B, Model C and Model D. Therefore, CNN structure has the best performance when Recall is lower and LSTM-based structure has the best performance as Recall becomes higher.

For the pre-processing method comparison, since Model A and Model B share the same deep learning model structure (CNN structure), comparison between Model A and Model B indicates the performance comparison between ROI mask pre-processing method and channel-wise preprocessing. From Fig. 10, since Model B has lower FPR when Recall is low, channel-wise pre-processing is better for lower Recall. When Recall becomes higher, since Model A has lower FPR, ROI mask pre-processing method has better performance.

In reality, some industrial requirement is expected such as Recall $\geq$ 0.90 and Recall $\geq$ 0.95.

For requirement of Recall $\geq$ 0.90, comparing among the four models, Model B achieves the lowest FPR = 0.2671 as shown in Table 11. That is to say, in the production line, com-

Table 11 Performance comparison

Model	AUROC	FPR@90% Recall	FPR@95% Recall
Model A	0.9023	0.3247	**0.5464**
Model B	0.9056	**0.2671**	0.7073
Model C	0.9005	0.3349	0.6040
Model D	0.8916	0.3637	0.5872

Bold values indicate best result in the comparison

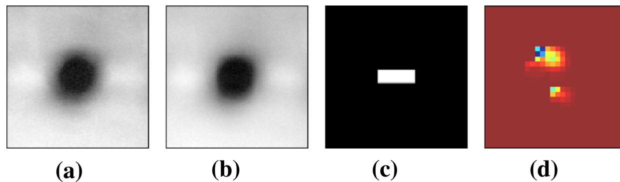


Fig. 11 Activation feature map samples

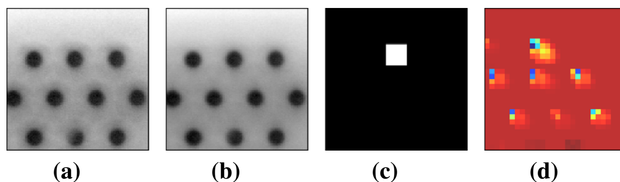


Fig. 12 Activation feature map samples

pared with sending all the normal solder joints that detected as defects by the AXI machine to the specialist for manual inspection, only 26.71% of normal joints are sent to the specialists and 73.29% of normal joints are filtering out, which reduces the specialist workload dramatically. For requirement of Recall $\geq$ 0.95, Model A has the lowest FPR = 0.5464 among all the models that almost reduces the specialist workload by 50%.

For the ROI mask pre-processing method, some pre-processed input patches include ROI mask that is incorrect and some patches include multiple solder joints besides the joint of interest. We visualize the CNN activation feature map before the final classification of the CNN to see which areas contribute to the CNN classification decision.

Although some ROI masks are rectangular that cannot include the whole part of the circular joint of interest, the activation feature map of CNN can include the areas outside the incorrect ROI mask. For example, Fig. 11a–c are the base slice channel, variance channel, and the ROI mask channel of the input patch. Fig. 11d is the activation feature map of the joint. Although the ROI mask in Fig. 11c cannot include the whole joint, the activation feature still covers the most area of the joint.

The patches achieved from the ROI mask pre-processing method include not only the joint of interest, but they can also include the adjacent joints that are outside of the ROI mask. In Fig. 12, there are many joints in one input patch, the joint of interest is indicated by the ROI mask in Fig. 12c. From the activation feature in Fig. 12d, we can see that not only the joint of interest is activated, but also the adjacent joints are activated, which mimics the specialist inspection that compares the joint of interest with joints in other images as well as the adjacent joints within one image.

Conclusion

In this work, four models are proposed for follow-up defect inspection. Two different pre-processing methods are proposed to address the varying numbers of slices problem and the incorrect ROI problem, and three deep learning model structures are proposed that suit the pre-processing methods. Channel-wise pre-processing method used in Model B, Model C, and Model D is better for lower Recall, while ROI mask pre-processing method used in Model A is better for higher Recall requirement. CNN model structure used in Model A and Model B is the best for for lower Recall, and LSTM-based structure used in Model D is the best for higher Recall requirement among the three deep learning model structures. The proposed four models have similar general performances in terms of AUROC. For requirement of Recall $\geq$ 0.90, the best model is achieved by Model B. And for requirement of Recall $\geq$ 0.95, the best model is Model A. By introducing AI into the production line, specialist workload can be dramatically reduced. The performance can be the reference for future researches on the X-ray imaging defect detection problems. In the future, model fusion can be implemented on the four models to increase the general performance.

Acknowledgements This research work was in part supported by the China Scholarship Council, Keysight Technologies, the Key R&D Program of China (Project No. 2018YFB2202703), and the Natural Science Foundation of Jiangsu Province (Project No. BK20201145).

Declaration

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copy-

right holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Liukkonen M, Havia E, Hiltunen Y (2012) Computational intelligence in mass soldering of electronics-a survey. Expert Syst Appl 39(10):9928–9937
2. Leinbach G, Oresjo S (2001) The why, where, what, how, and when of automated x-ray inspection. Agilent Technologies, Loveland, Colorado
3. Huang CY, Hong JH, Huang E (2019) Developing a machine vision inspection system for electronics failure analysis. IEEE Trans Compon Packag Manuf Technol 9(9):1912–1925
4. Gao H, Jin W, Yang X, Kaynak O (2016) A line-based-clustering approach for ball grid array component inspection in surface-mount technology. IEEE Trans Ind Electron 64(4):3030–3038
5. Jin W, Lin W, Yang X, Gao H (2017) Reference-free path-walking method for ball grid array inspection in surface mounting machines. IEEE Trans Ind Electron 64(8):6310–6318
6. Lu X, He Z, Su L, Fan M, Liu F, Liao G, Shi T (2018) Detection of micro solder balls using active thermography technology and k-means algorithm. IEEE Trans Ind Inf 14(12):5620–5628
7. Jewler SJ (2019) High resolution automatic X-Ray inspection for continuous monitoring of advanced package assembly. In: 2019 International wafer level packaging conference (IWLPC), pp 1–5. https://doi.org/10.23919/IWLPC.2019.8914113
8. Yung LC (2018), Investigation of the solder void defect in IC semiconductor packaging by 3D computed tomography analysis. In: 2018 IEEE 20th Electronics packaging technology conference (EPTC), 2018, pp 886–889. https://doi.org/10.1109/EPTC.2018.8654370
9. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, vol 25, pp 1097–1105
10. Zhang Q, Zhang M, Chen T, Sun Z, Ma Y, Yu B (2019) Recent advances in convolutional neural network acceleration. Neurocomputing 323:37–51
11. Han Q, Yin Q, Zheng X, Chen Z (2021) Remote sensing image building detection method based on mask r-cnn. Complex Intell Syst, https://doi.org/10.1007/s40747-021-00322-z
12. Wu H, Gao W, Xu X (2019) Solder joint recognition using mask r-cnn method. IEEE Trans Compon Packag Manuf Technol 10:525-530
13. Cai N, Cen G, Wu J, Li F, Wang H, Chen X (2018) Smt solder joint inspection via a novel cascaded convolutional neural network. IEEE Trans Compon Packag Manuf Technol 8(4):670–677
14. Goto K, Kato K, Nakatsuka S, Saito T, Aizawa H (2019) Anomaly detection of solder joint on print circuit board by using adversarial autoencoder. In: Fourteenth International Conference on quality control by artificial vision, vol. 11172, p. 111720T. International Society for Optics and Photonics
15. Simonyan K, Zisserman A (2015) Very deep convolutional networks for largescale image recognition. In: International conference on learning representations (ICLR), pp 1–14. https://arxiv.org/abs/1409.1556
16. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning (ICML), vol 37, pp 448–45
17. Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A (2016) Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on computer vision and pattern recognition, pp 2921–2929
18. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on artificial intelligence and statistics, pp 249–256
19. Roshan SE, Asadi S (2020) Improvement of bagging performance for classification of imbalanced datasets using evolutionary multi-objective optimization. Eng Appl Artif Intell 87:103319
20. Priya S, Uthra RA (2021) Deep learning framework for handling concept drift and class imbalanced complex decision-making on streaming data. Complex Intell Syst, https://doi.org/10.1007/s40747-021-00456-0

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.