Research on Surface Flaw Detection of Industrial Components Based on Deep Learning Algorithm

1 Wei Li, 2 Mahmud Iwan Solihin, 3 Hanung Adi Nugroho, 4 Valliappan Raju

1 Faculty of Engineering, Technology and Built Environment, UCSI University, Kuala Lumpur 56000, Malaysia; 1002162124@ucsiuniversity.edu
2 School of Computer Science and Technology (School of Software), Guangxi University of Science and Technology, Guangxi 545006, China; liwei_happyman@gxust.edu.cn
3 Department of Electrical and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia; adinugroho@ugm.ac.id
4 Professor, Research Management Centre, Perdana University, Malaysia

*Corresponding Author: Mahmud Iwan Solihin. Email: mahmudis@ucsiuniversity.edu.my

Abstract: This paper primarily focuses on the research of industrial component surface defect detection algorithms using deep learning techniques. Taking the example of defect detection in the injector valve seat of automotive engine components, currently, the inspection of such small components heavily relies on manual visual inspection or traditional machine vision methods. Manual visual inspection is inefficient and cannot guarantee the speed and accuracy of detection. Traditional machine vision methods, while efficient, are sensitive to harsh environmental conditions and can be influenced by factors such as lighting, camera positioning, and background, resulting in poor robustness and limited feature extraction capabilities. To address these issues, this paper conducts research on a deep learning-based surface defect detection algorithm for injector valve seats.

Keywords: Faster R-CNN; YOLO; Industrial Components; Flaw Detection; Deep Learning

1 Introduction

In the industrial manufacturing process, the level of automation for the production of mechanical product components is increasing. Surface defect detection of industrial components is becoming increasingly important and requires continuous technological advancement. Taking the example of the injector valve seat of engine components, due to the unique properties of the valve seat material, it is susceptible to various degrees of compression and cutting during the injection molding process, leading to damage on the surface of the component. If these small defects are not detected during subsequent inspections, they can pose safety risks during the future use of the products.

With the rapid advancement of industrial production, the manual visual inspection method is gradually unable to meet the demands of rapid inspection on the production line. Currently, the detection of surface defects on injector valve seats primarily relies on machine vision. High-precision industrial cameras are used to capture images of defective parts, and well-established image processing methods are employed to extract defect feature information for defect detection. Due to the small surface area of defects on injector valve seats and the relatively small size of valve seat components, the use of machine vision incurs relatively high costs. As a result, manufacturers still require manual quality inspections. Similar complex components, such as magnetic rings and screens, also require manual visual inspections[1]. Traditional machine learning algorithms, such as those based on feature extraction and classifiers like Fourier transform[2], Haar[3], HOG+SVM[4], [5], and AdaBoost [6] methods, can be used for internal and external
surface defect detection. These methods involve extracting texture, shape, or statistical features from images and using classifiers for defect identification. However, they have some drawbacks, including slow detection speed, poor detection accuracy, and low robustness.

In recent years, with the widespread application of deep learning in computer vision tasks, deep learning-based methods for industrial defect detection have also seen rapid development and have gradually taken a dominant position[7]. Thanks to the powerful feature extraction capability and high-dimensional data representation of Convolutional Neural Networks (CNN) )[8], deep learning-based methods can automatically learn features that are difficult for humans to design. This not only saves the cost of manually designing features but also significantly improves detection accuracy. In contrast to traditional machine vision detection methods, deep neural network algorithms[9] can automatically extract features from input sample data. Traditional machine vision detection methods often suffer from interference from uncertain factors such as lighting and camera position when processing images, resulting in subpar detection performance. In contrast, deep learning does not require the manual selection of target features. It adapts better to complex detection environments through continuous learning of target data, exhibiting strong robustness. These automatically extracted features have a better effect on detection tasks, making deep learning-based detection in the field of defect detection significantly superior to traditional machine vision detection methods.

This paper takes the inner surface of the automotive fuel injector valve seat as an example. Leveraging the advantages of deep learning, it applies Convolutional Neural Networks (CNN) to the complex defect detection environment. Furthermore, it systematically improves and optimizes existing deep learning algorithms, providing a high-precision and fast detection method for identifying defects on the inner surface of the fuel injector valve seat. This method also enables precise grading and classification of detected defects. With the advent of the Internet of Things era, artificial intelligence has rapidly advanced, relying on the fast development of embedded technologies. Currently, CNNs can be applied to embedded devices, but the requirements for network models on embedded devices and algorithms are quite stringent. The deep learning-based industrial component defect detection method proposed in this paper offers a novel approach for applying Convolutional Neural Networks to embedded devices. This is achieved by compressing the volume of CNNs and reducing the computational load, allowing high-performing CNNs to be applied to embedded devices[10].

2 Dataset Construction

In the detection of fuel injector valve seats, due to the small size of defects on the valve seats, they cannot be visually identified by the naked eye. Therefore, it is necessary to magnify the images of the fuel injector valve seats. Additionally, interference from external light sources must be overcome. This involves adjusting and setting parameters such as focal length to obtain clear images of defects and ensure the quality of captured images. During the inspection, the precision of the images directly affects the final determination of defects. CCD industrial cameras, known for their excellent performance, offer good color reproduction and pre-calibration functions for light sources. These cameras include input and output units as well as photosensitive units. By matching the magnifying lens with the selected camera interface and chip size, image magnification and precision control are achieved. The selection of appropriate light sources and fixtures, along with the design of a light-shielding device, helps eliminate external light source interference and effectively overcomes light source divergence issues. This results in clearer captured images and high-quality image information. Finally, secondary development of the image acquisition card is performed, and image capture and storage are completed through the coordination of the camera and motor. The image capture environment and process are depicted in Figure 1 for reference.
Due to the fact that images captured by cameras are typically in color and contain rich color information, redundant information can increase the difficulty of subsequent image processing. Therefore, it is necessary to apply effective methods for image preprocessing in practical production. This operation aims to weaken or eliminate irrelevant information in the images while preserving useful, real information related to the detection target. Currently, the collected 2100 images are insufficient for model training in the sample dataset, and the quantity of images directly impacts the results of detection and recognition. To improve the generalization capability of the detection model, data augmentation is commonly employed to address data scarcity. This method effectively enhances object recognition rates and often involves dataset expansion through techniques such as grayscale conversion, saturation enhancement, saturation reduction, and flipping. This augmentation process results in a database of 7500 images. Simultaneously, the original captured images have a resolution of 5678x2948 pixels. Such high pixel counts not only consume memory space but are also unfavorable for later model training. Therefore, the images are compressed before training. Next, the samples are shuffled, and 20% (1500 images) are extracted as a test set, while 80% (6000 images) are designated as a training set. Afterward, the defects in the training images are categorized and their positions marked. Once the labeling is complete, each annotated image is saved as an XML file, serving as training samples.
Fig. 2. Image data of injector valve seat collected by visual data acquisition system. (a) rags + scratches; (b) rags; (c) rags; (d) rags + black areas; (e) scratches; (f) rags + scratches.

3 Improved Faster R-CNN Algorithm

The Faster R-CNN architecture consists of three main components: the feature extraction layer, the Region Proposal Network (RPN) layer, and the Fast R-CNN layer. The feature extraction layer in Faster R-CNN uses the VGG16 network model. It extracts features from the image using Convolution, ReLU, and Pooling layers. VGG16 is considered an excellent network model for classification within convolutional neural networks. The overall structure of the Faster R-CNN network is illustrated in Figure 5, providing an overview of the entire detection process. Given an arbitrary-sized image of dimensions P×Q, it is scaled to a fixed size of M×N and then fed into the CNN network. The CNN network includes 13 Convolution layers, 13 ReLU layers, and 4 Pooling layers. The RPN network begins with a 3×3 convolution, which generates Positive Anchors and corresponding Bounding Box Regression offsets, followed by the calculation of Proposals. The ROI Pooling layer is then used to extract Proposal Features from Feature Maps, which are subsequently input into fully connected and SoftMax networks for classification.

In summary, Faster R-CNN builds upon the Fast R-CNN framework by introducing the RPN (region proposal network) [11]. This network replaces the selective search used in Fast R-CNN. It effectively adjusts the target positions through techniques like region merging and redundancy removal, producing high-quality region proposals. By sharing the feature extraction network, it significantly reduces the time required for region detection, resulting in more precise results. Finally, Fast R-CNN is responsible for learning region features, classifying objects, and regressing bounding boxes. The candidate region (RPN) is a fully convolutional neural network. During training, it employs non-maximum suppression to obtain candidate regions based on object/non-object classification probabilities within windows. These regions are randomly selected in specific ratios for training, using Intersection over Union (IoU) as the criterion for determining positive and negative samples. IoU measures the overlap between bounding box regression (BBox) and real target window. Experimental findings suggest that IoU values significantly affect detection performance. If set too large, it can lead to inaccurate detections and a high rate of false positives. If set too small, it might result in incomplete detections and false negatives. Therefore, for the detection of defects in automobile fuel injector valve seats, IoU is set to consider a prediction as correct (positive sample) when the IoU overlap with the ground truth box is greater than 0.7, and as incorrect (negative sample) when the IoU overlap is less than 0.3.
3.1 Improve the Number and Size of Anchor Boxes

The anchor box is a crucial parameter in RPN (Region Proposal Network) operations and has a decisive impact on the quantity and size of candidate boxes generated in subsequent steps. In the original Faster R-CNN, the RPN uses three different sizes and aspect ratios (1:1, 1:2, 2:1), creating nine different-sized anchor boxes. These anchor boxes are used to predict the positions of windows containing objects. Assuming a feature map with a height of \( H \) and width of \( W \), it would generate \( H\times W\times 9 \) candidate boxes. These candidate regions are then refined using non-maximum suppression to remove redundant areas. However, in the case of defect detection in fuel injector valve seats, it differs from typical object detection tasks like pedestrians or vehicles. Fuel injector valve seat defects tend to be small and diverse in nature. Using existing algorithm models may result in missed detections. To address this, modifications are made to the anchor boxes tailored to the characteristics of this specific object. The aspect ratios are adjusted to 0.5, 1, 2, and 4, and the scaling ratios become 4, 8, 16. This modification increases the number of candidate windows from the original nine to twelve. The improved anchor windows enhance the detection performance of the Faster R-CNN model, particularly for smaller-sized fuel injector valve seats, providing more accurate detection rates. Through calculations, it can be observed that the improved anchor window sizes closely match the defect sizes. The implementation steps for this improvement are as follows:

**Step 1** Initialization of Anchors, initialize anchor_base as 16, set anchor_middle as (7.5, 7.5), define size as 16*16, specify the aspect ratios as ratio=[0.5, 1, 2, 4]. Here, anchor_base is the initial value, anchor_middle is the center for initialization, size represents the area, and ratio denotes the aspect ratios.

**Step 2** Calculate the initial anchor areas for different aspect ratios as formula (1):

\[
s_{1,2,3,4} = \frac{\text{size}}{\text{ratio}} = [512, 256, 128, 64]
\]

(1)

**Step 3** Compute different widths and heights for the anchors by taking the square root, as shown in Formula (2):

\[
w_{1,2,3,4} = \sqrt{s_{1,2,3,4}} = [23, 16, 11, 8]
\]

(2)

**Step 4** Calculate anchors_ratio based on the anchor center and various width and height values, as depicted in Formulas (3) and (4):
\[ x_{\text{left}} = x_1 - \frac{(w_1 - 1)}{2} = -3.5, \quad y_{\text{left}} = y_1 - (h_1 - 1)/2 = 2 \]

\[ x_{\text{right}} = x_1 + \frac{(w_1 - 1)}{2} = 18.5, \quad y_{\text{right}} = y_1 + (h_1 - 1)/2 = 13 \]

\[
\text{anchors} = \{[-3.5, 2, 18.5, 13], [0, 0, 15, 15], [2.5, -3, 12.5, 18], [4, -8, 11, 23]\}
\]

**Step 5** Expand the anchors using three different scales \([4, 8, 16]\). To expand, first calculate the center and dimensions of the anchors from the previous step. Multiply the width and height by the scale factor. Then, use this center and the new width and height to calculate the final desired anchors, as illustrated in Formulas (6) and (7):

\[ w^*_1 = w_1 \times \text{scale}_1, \quad h^*_1 = h_1 \times \text{scale}_1 \]

\[ x_{\text{min}} = x_1 - \frac{(w^*_1 \times \text{scale}_1 - 1)}{2}, \quad y_{\text{min}} = y_1 - \frac{(h^*_1 \times \text{scale}_1 - 1)}{2} \]

In the end, anchors of various sizes will be obtained, as presented in Formula (8):

\[
\text{anchors}_{1,2,3,\ldots,12} = \{[-37, -15, 54, 32] \ldots [-55, -247, 72, 264]\}
\]

### 3.2 Improved Feature Extraction Network

While VGG-16 is an excellent model in terms of classification performance within convolutional neural networks, it suffers from significant feature loss during image feature extraction. This limitation results in insufficient feature extraction, ultimately impacting the effectiveness of object detection, especially for small objects. As network models continuously evolve, deeper models such as Inception[12], ResNet[13] have emerged, demonstrating that increasing the depth of networks can enhance performance. Therefore, by incorporating the advantages of networks like Inception and ResNet and making improvements to the feature extraction network, both in terms of depth and width, it becomes possible to extract features more effectively. Since the first three layers of convolutional networks in the network are proficient at feature extraction, leveraging this characteristic to enhance the VGG16 network is a viable approach. The modified network is illustrated in Figure 4.

**Fig.4.** Feature extraction network.
The Conv4 convolutional network module is replaced with inspiration from the construction of deep networks, such as ResNet50, and the inner product strategy of splitting-transforming-merging in the Inception model. Simultaneously, the depth and width of this layer’s network are increased to enhance its expressive capability. Increasing the depth and width of the network is the most direct method for improving neural network performance [12]. Since Inception modules possess excellent local topological structures and perform parallel convolution operations on input images, eventually concatenating features with different receptive fields, the design of this network module incorporates a fusion of ResNet and Inception characteristics. In this configuration, the Inception module is used to replace convolution layers in the residual connections, forming a new structure. This enables nodes to learn the difference mapping between input and output, mitigating the need to fit input-output features. This eliminates gradient dispersion and explosion, while also speeding up network training.

In the replaced module, firstly, dimension reduction and splitting of the input are achieved through a 1×1 convolution. Next, multiple 3×3 convolutions are applied for transformation (to reduce computational load, a 5×5 convolution is replaced with two 3×3 convolutions). Subsequently, channel merging is performed along the channel dimension in a concatenated manner, enabling multi-scale detection. Finally, channel consistency is attained through a 1×1 convolution, completing the addition of the residual module to the output linear vector, as depicted in Equation (9).

\[ Z_{\text{add}} = \sum_{i=1}^{c} (X_i + Y_i) * K_i = \sum_{i=1}^{c} X_i * K_i + \sum_{i=1}^{c} Y_i * K_i \]  

In the equation, \(X\) and \(Y\) represent input channels, with both input and output channels set to 512 in the experiments. The \("\"\) denotes convolution. In this module, network module design is typically achieved by modifying the channel numbers of the 3×3 convolution layers. This is because these convolution layers operate independently of the input and output, making it more advantageous for network construction. Both the VGG16 network and the conv4 module of ResNet50 have approximately 5,898k parameters. Each module of ResNet50 has around 983k parameters. Therefore, following the principle of building similar modules, the channel number of the 3×3 convolution layer is set to 172. To expedite convergence, batch normalization (BN) layers are added before each layer[14], which helps correct the output of the previous layer, normalizing its mean to 0 and variance to 1 before input to the next layer. The improved feature extraction network can leverage the strengths of different networks, enhance feature utilization, obtain richer input features, and further enhance performance.

### 3.3 Performance Evaluation

In order to objectively evaluate the detection performance of this improved algorithm, we compared the following algorithms: Faster R-CNN with Feature Network enhancement (Faster R-CNN+FN), Faster R-CNN with Proposal Anchor Boxes enhancement (Faster R-CNN+PAB), and the algorithm that combines both candidate box and feature network enhancements, Faster R-CNN+FN+PAB, with the original Faster R-CNN algorithm. Typically, the integral method is used to calculate the area under the P-R curve to obtain the average precision (AP). The results for the
measurement of common defects in fuel injector valve seats, namely rag, black area, and scratch, are shown in Figure 5.

![Fig. 5. Improved Faster R-CNN model P-R comparison.](image)

From Figure 5 (a), it can be observed that for the detection of burrs on the fuel injector valve seats, the average precision (AP) for Faster R-CNN is 0.7745, for Faster R-CNN+FN algorithm, the average precision is 0.7926, for Faster R-CNN+PAB algorithm, the average precision is 0.8038, and for Faster R-CNN+FN+PAB algorithm, the experimental average precision is 0.8185. This indicates that the proposed Faster R-CNN+FN algorithm, Faster R-CNN+PAB algorithm, and Faster R-CNN+FN+PAB algorithm have improved by 1.81%, 2.93%, and 4.4%, respectively, compared to the base Faster R-CNN. From Figure 5 (b), it can be seen that for the detection of rust spots on the fuel injector valve seats, the average precision (AP) for Faster R-CNN is 0.6989, for Faster R-CNN+FN algorithm, the average precision is 0.7104, for Faster R-CNN+PAB algorithm, the average precision is 0.7037, and for Faster R-CNN+FN+PAB algorithm, the experimental average precision is 0.7197.

It can be deduced that the Faster R-CNN+FN algorithm, Faster R-CNN+PAB algorithm, and Faster R-CNN+FN+PAB algorithm proposed in this paper have improved by 1.15%, 0.48%, and 2.08%, respectively, compared to the base Faster R-CNN, as shown in Figure 5 (c). As for the detection of scratches on the fuel injector valve seats, the average precision (AP) for Faster R-CNN is 0.6242, for Faster R-CNN+FN algorithm, the average precision is 0.6430, for Faster R-CNN+PAB algorithm, the average precision is 0.6357, and for Faster R-CNN+FN+PAB algorithm, the average precision is 0.6819. Therefore, it can be concluded that the Faster R-CNN+FN algorithm, Faster R-CNN+PAB algorithm, and Faster R-CNN+FN+PAB algorithm proposed in this paper have improved by 1.61%, 1.52%, and 4.08%, respectively, compared to the base Faster R-CNN.

By comparing with the advanced Faster R-CNN algorithm in object detection, the proposed algorithms exhibit significant improvements in detection performance. In the context of fuel injector valve seat defects, the average mean average precision (mAP) is obtained by comparing the accuracy of different defect types under various algorithms, as shown in Table 1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Rag (%)</th>
<th>Black (%)</th>
<th>Scratch (%)</th>
<th>mAP (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN</td>
<td>77.45</td>
<td>69.89</td>
<td>62.42</td>
<td>69.92</td>
</tr>
<tr>
<td>Faster R-CNN+FN</td>
<td>79.26</td>
<td>71.04</td>
<td>64.30</td>
<td>71.53</td>
</tr>
<tr>
<td>Faster R-CNN+PAB</td>
<td>80.38</td>
<td>70.37</td>
<td>63.57</td>
<td>71.44</td>
</tr>
</tbody>
</table>
4 Improved YOLO Algorithm

Although the YOLO network architecture has significantly improved detection speed compared to traditional CNN network structures, the enormous computational load of large network models still makes it challenging to run on embedded operating systems. In order to enhance the detection speed of the YOLO algorithm for surface defects on fuel injector valve seats and reduce the network’s computational complexity, this paper has streamlined and optimized the feature extraction network within the YOLO model. The optimized YOLO-tiny network structure is illustrated in Figure 6.

From Figure 6, it can be observed that in the improved YOLO-tiny feature extraction network, the input image remains at its original size of $416 \times 416$. Firstly, it passes through two CBL modules, each comprising a convolutional layer, a standardization processing function, and a Leaky activation function. Following this, it enters the CSP module. As illustrated in Figure 8, within the structure of CSP_Resblock_body, each image data input into CSP_Resblock_body is divided into two parts after passing through a CBL module. One part goes through the Resblock_body module, as shown in Figure 7. The other part passes through another CBL module, and then the two parts are combined and pass through a CBL module before being output to the next layer. After outputting from the CSP1 module, it undergoes max-1798
pooling and proceeds to the next layer. After passing through two Maxpooling modules and the CSP module, the network outputs a 13×13 feature map. The new YOLO-tiny feature extraction network consists of a total of 25 convolutional layers. Compared to the YOLO v3 feature extraction network with 53 convolutional layers, the network's hierarchy and depth have been significantly reduced. Additionally, the CSPNet module is embedded into the Darknet53 neural network. The CSP strategy effectively reduces the repetition of gradient information during the information integration process, enhancing the network's learning capabilities. Incorporating CSPNet also helps reduce computational bottlenecks and memory costs, making the improved network suitable for use on embedded devices with limited memory resources.

However, the improved YOLO-tiny network exhibits insufficient feature information extraction for defects. It has been observed that the enhanced network structure still faces issues of false positives and false negatives in detecting black area and rag defects. In this paper, we propose to incorporate both channel attention modules and spatial attention modules into the feature extraction network. This dual approach aims to enhance the network's capability for feature information extraction in two dimensions, further improving its detection performance for black area and rag defects.

4.1 Overview of SENet Network

In the field of deep learning, the development of CNN classification networks plays a crucial role in various computer vision tasks such as object detection and semantic segmentation. The fundamental computation in CNN networks is the convolution operation, which combines spatial and channel features. In 2019, X. Zhong et al. took a unique approach and made the first attempt to improve and optimize network performance in the channel dimension[15]. They introduced the SENet embedded network structure, incorporating an attention model called SENet into the channel dimension. The SENet structure is illustrated in Figure 9.

![Fig.9. CSP_Resblock_Body module structure.](image)

The SENet module begins with a serialization (Squeeze) operation, compressing global spatial information into channel descriptors, represented as shown in Equation (10).

\[
 z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) 
\]

(10)

\(F_{sq}\) represents the squeeze operation, \((H, W)\) denotes the width and height of \(u\), and the real number \(c\) represents the channels in \(u\). The main purpose of this step is to fuse positional information from the entire feature map, minimizing the potential for inaccurate evaluations. Following the squeeze operation, an activation (Excitation) operation is performed, recalibrating the channels using channel correlations. This is achieved by generating weights for each feature channel through the parameter \(W\), as shown in Equation (11).

\[
s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \ast \delta(W_1z))
\]

(11)
\( F_{\text{ex}} \) corresponds to the activation operation, where the previously extracted feature \( z \) is used to recalibrate the weights \( W \). Here, \( \delta \) represents the ReLU activation function, and \( \sigma \) represents the sigmoid activation function. The SENet network structure is straightforward and can be embedded into other classification or detection models without the need to introduce new functions or layers.

4.2 Overview of CBAM Network

To enhance the performance of CNN models, researchers have primarily focused on three critical aspects: depth, width, and cardinality. In the previous section, we discussed how SENet attempted to improve network performance by considering channel-wise attention, achieving impressive results. In 2019, C. Wen et al. introduced the CBAM (Convolutional Block Attention Module) network on top of the existing network structure[16]. CBAM is a lightweight, versatile module that enhances network attention without significantly increasing network complexity or computational requirements. It includes both spatial attention and channel attention modules. In comparison to SENet, which primarily focuses on channel-wise attention mechanisms, CBAM has demonstrated superior results on various classification and detection datasets. The improvements made by CBAM over SENet include:

1. An enhanced learning method for channel weights compared to SENet.
2. The introduction of a novel approach for learning the importance of different spatial positions within channels. CBAM incorporates a spatial attention model on top of the SENet's channel attention mechanism. These enhancements have contributed to CBAM's superior performance across different datasets and tasks.

The CBAM network structure represents an attention mechanism module. CBAM's most significant feature is that it combines the advantages of both spatial attention modules and channel attention modules, optimizing the network in both aspects. Unlike SENet, which includes only channel attention modules and has limited filtering capabilities for feature information, CBAM is a lightweight and versatile module that can be seamlessly integrated into any CNN network structure without introducing significant overhead. The network structure of CBAM is depicted in Figure 10.

From Figure 10, it can be observed that the input feature map matrix first passes through the channel attention module.
and then through the spatial attention model, ultimately producing a reconstructed feature map. Below, we will provide a detailed introduction to the channel attention module and the spatial attention module.

The structure of the channel attention module is depicted in Figure 11.

![Fig.11. Channel attention module structure.](image)

In convolutional neural networks (CNNs), feature extraction primarily relies on convolutional layers. Depending on the number of convolutional filters, each image processed through these layers results in a feature map with the same number of channels as the filters. Throughout the neural network, as the number of convolutional layers and filters increases, the number of channels generated also increases. However, since defects in the image occupy a very small proportion, not every channel conveys useful information. Therefore, a channel attention model is needed to determine the effective channels and filter out the ineffective ones. When extracting channel attention, spatial information in the image is briefly ignored. Max-pooling and average-pooling are performed in the spatial dimension, resulting in two-dimensional vectors. These vectors are then individually processed through a two-layer neural network, and the two features are added together. After passing through a loss function (sigmoid), weights are assigned to each channel. This process can be represented as Equation (12).

$$M_c(F) = \sigma\left(MLP(AvgPool(F)) + MLP(maxPool(F))\right)$$

$$= \sigma\left(W_1(W_0(F_{avg}^c)) + W_1(F_{max}^c)\right)$$

(12)

The structure of the spatial attention module is depicted in Figure 11.

![Fig.12. Spatial attention module structure.](image)
The spatial attention module focuses on valuable information within each channel. When extracting spatial attention information, channel information needs to be temporarily masked. Then, average pooling and max-pooling operations are applied in the channel dimension, and the results are concatenated. Afterward, a convolutional operation reduces it to a single channel. Finally, it passes through a sigmoid activation function to obtain the expression of the spatial attention model. The formula for the spatial attention structure is as shown in Equation (13).

\[
M_s(F) = \sigma(f^{7 \times 7}([\text{AvgPool}(F); \text{MaxPool}(F)])) = \sigma(f^{7 \times 7}(F_{\text{max}}; F'_{\text{max}}))
\]  

(13)

The results of the channel attention and spatial attention are presented in Tables 2 and 3.

### Table 2. Result of channel attention.

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
<th>GFLOPs</th>
<th>Top-1 (%)</th>
<th>Top-5 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet50 (baseline)</td>
<td>25.56M</td>
<td>3.86</td>
<td>24.56</td>
<td>7.50</td>
</tr>
<tr>
<td>ResNet50 + AvgPool (SE)</td>
<td>25.92M</td>
<td>3.94</td>
<td>23.14</td>
<td>6.70</td>
</tr>
<tr>
<td>ResNet50 + MaxPool</td>
<td>25.92M</td>
<td>3.94</td>
<td>23.20</td>
<td>6.83</td>
</tr>
<tr>
<td>ResNet50 + AvgPool &amp; MaxPool</td>
<td>25.92M</td>
<td>4.02</td>
<td>22.80</td>
<td>6.52</td>
</tr>
</tbody>
</table>

### Table 3. Result of spatial attention.

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
<th>GFLOPs</th>
<th>Top-1 (%)</th>
<th>Top-5 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet50 + channel (SE)</td>
<td>28.09M</td>
<td>3.860</td>
<td>23.14</td>
<td>6.70</td>
</tr>
<tr>
<td>ResNet50 + channel</td>
<td>28.09M</td>
<td>3.860</td>
<td>22.80</td>
<td>6.52</td>
</tr>
<tr>
<td>ResNet50 + channel + spatial (1x1 conv, k=3)</td>
<td>28.10M</td>
<td>3.862</td>
<td>22.96</td>
<td>6.64</td>
</tr>
<tr>
<td>ResNet50 + channel + spatial (1x1 conv, k=7)</td>
<td>28.10M</td>
<td>3.869</td>
<td>22.90</td>
<td>6.47</td>
</tr>
<tr>
<td>ResNet50 + channel + spatial (avg &amp; max, k=3)</td>
<td>28.09M</td>
<td>3.863</td>
<td>22.68</td>
<td>6.41</td>
</tr>
<tr>
<td>ResNet50 + channel + spatial (avg &amp; max, k=7)</td>
<td>28.09M</td>
<td>3.864</td>
<td>22.66</td>
<td>6.31</td>
</tr>
</tbody>
</table>

Table 2 demonstrates that using average pooling and Maxpooling can significantly reduce detection errors. Table 3 indicates that the spatial attention module performs best when it includes parameters "avg" and "max" with a kernel size of 7. Therefore, it is evident that the integration of the CBAM module into deep learning networks leads to varying degrees of performance improvement. Consequently, this paper incorporates the CBAM module into the enhanced YOLO-tiny network to further enhance its feature extraction capabilities.

### 4.3 Embedding CBAM in YOLO-tiny Feature Extraction Network

In the improved YOLO-tiny feature extraction network, due to the fewer convolutional layers, the network's ability to
extract defect features is limited, resulting in significant instances of false positives and false negatives in the detection results. This paper embeds the CBAM network into the feature extraction network of YOLO-tiny. When combined with the existing CSP module, the weights of the feature maps are dynamically reassigned through continuous learning. This filtering process eliminates irrelevant feature information, ultimately yielding feature maps with distinct defect information. The CBAM module is incorporated into the end of each CSP_Resblock_body in this paper. The YOLO-tiny feature extraction network with the embedded CBAM module is the same as the enhanced YOLO-tiny feature extraction network described in Chapter Three, with the only difference being the CSP module. The improved feature extraction network is depicted in Figure 6, and the structure of the CSP_Resblock_body with the embedded CBAM module is illustrated in Figure 13.

One significant advantage of the CBAM attention module is its ability to be seamlessly integrated into a network without altering the original network structure and without incurring additional computational overhead, enabling end-to-end training. It filters out redundant feature information before passing image data to the next residual module. After each CSP_resblock_body undergoes filtering with the CBAM structure, the resulting feature information becomes more concentrated and accurate.

For the purpose of distinguishing and comparing the results, the network with the added CBAM module is referred to as YOLO-CBAM-tiny. It was trained using the injector valve seat dataset. Subsequently, a comparison was made in terms of model size, detection speed for individual images, and the precision of detecting various defects. The experimental results are shown in Figure 13.
Fig.13. Improved YOLO model P-R comparison.

Through the comparison of P/R curves for the three types of defects mentioned above, it can be observed that the AP values for all three defect categories have shown slight improvements. The improved YOLO-CBAM-tiny network demonstrates a 7.4% increase in AP for point defects, a 0.43% increase for Scratch defects, and a 2.96% increase for rag defects. This indicates that the inclusion of the CBAM module enhances the network’s capability to extract feature information, making it better at distinguishing between the three defect categories. This study conducted separate statistical analyses for black area defects, scratch defects, and rag defects, and the detection results for each defect type are presented in Table 4.

### Table 4. Comparison of Model Detection Speed and Accuracy

<table>
<thead>
<tr>
<th>Model</th>
<th>Rag (%)</th>
<th>Black (%)</th>
<th>Scratch (%)</th>
<th>mAP (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YOLO-tiny</td>
<td>81.39</td>
<td>75.23</td>
<td>83.78</td>
<td>80.13</td>
</tr>
<tr>
<td>YOLO-CBAM-tiny</td>
<td>84.35</td>
<td>82.63</td>
<td>84.21</td>
<td>83.73</td>
</tr>
</tbody>
</table>

As shown in Table 4, it can be concluded that the YOLO-CBAM-tiny model achieves an mAP of 83.73% on the test dataset. In contrast, the improved YOLO-tiny model achieves an mAP of 80.13% on the same test dataset. With the inclusion of the CBAM attention module, the network’s mAP increases by 3.24%. Additionally, when comparing the detection performance of black area defects and rag defects between the two models, it becomes evident that the YOLO-CBAM-tiny network performs better in detecting rag defects and scratch defects.

### Table 5. Comparison of Model Detection Speed and Accuracy

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (%)</th>
<th>Recall (%)</th>
<th>mAP (%)</th>
<th>Speed (ms)</th>
<th>Weight (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YOLO-tiny</td>
<td>83.20</td>
<td>76.60</td>
<td>80.13</td>
<td>5.5</td>
<td>23.1</td>
</tr>
<tr>
<td>YOLO-CBAM-tiny</td>
<td>85.83</td>
<td>80.60</td>
<td>83.73</td>
<td>6.3</td>
<td>25.4</td>
</tr>
</tbody>
</table>

By comparing the results in Table 5, it is evident that the inclusion of the CBAM module in the YOLO-CBAM-tiny network has led to an improvement in defect detection performance, with a 3.6% increase in mAP. Importantly, the addition of the CBAM module has not resulted in a bloated network; the model's average detection time per image remains almost unchanged, and the model's weight file has only increased by 2.3 megabytes.
5 Experimental Results and Analysis

As shown in Table 6, the accuracy data for injector valve seat object detection indicate improvements in the candidate box refinement algorithm and the feature extraction network algorithm when compared to the original algorithm model. Moreover, the combination of both algorithms has significantly enhanced the original model, resulting in an approximate 4% increase in average accuracy. The accuracy of burr detection is relatively higher compared to the detection of other defects. This is because burrs are more prevalent in the defect samples, leading to a higher detection accuracy. Additionally, it can be observed that the improved YOLO algorithms (YOLO-tiny and YOLO-CBAM-tiny) outperform the improved Faster R-CNN algorithm (Faster R-CNN+FN+PAB) in terms of mAP. Although both sets of algorithms exhibit similar performance in detecting rag defects, the improved YOLO algorithms excel in detecting black area defects and scratch defects.
Table 6. AP of Different Defect Types of Improved Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Rag (%)</th>
<th>Black (%)</th>
<th>Scratch (%)</th>
<th>mAP (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN</td>
<td>77.45</td>
<td>69.89</td>
<td>62.42</td>
<td>69.92</td>
</tr>
<tr>
<td>Faster R-CNN+FN</td>
<td>79.26</td>
<td>71.04</td>
<td>64.30</td>
<td>71.53</td>
</tr>
<tr>
<td>Faster R-CNN+PAB</td>
<td>80.38</td>
<td>70.37</td>
<td>63.57</td>
<td>71.44</td>
</tr>
<tr>
<td>Faster R-CNN+FN+PAB</td>
<td>81.85</td>
<td>71.97</td>
<td>68.19</td>
<td>74.00</td>
</tr>
<tr>
<td>YOLO-tiny</td>
<td>81.39</td>
<td>75.23</td>
<td>83.78</td>
<td>80.13</td>
</tr>
<tr>
<td>YOLO-CBAM-tiny</td>
<td>84.35</td>
<td>82.63</td>
<td>84.21</td>
<td>83.73</td>
</tr>
</tbody>
</table>

The experimental results demonstrate the effectiveness of the improved algorithm. Firstly, for the recognition of the three common types of defects, as shown in Figure 15(a) and Figure 15(e), it can be observed that the rag defects are relatively small, and the black areas are very similar to the background of the injector valve seat surface, making them easily overlooked by the human eye. However, the proposed algorithm achieves good recognition results for these defects. In actual production, injector valve seats often have multiple types of defects simultaneously. By using the improved Faster R-CNN detection, most defects can be detected. However, as shown in Figure 15(b) and Figure 15(f), there are cases of missed detections with the improved Faster R-CNN. In contrast, using the YOLO-tiny and YOLO-CBAM-tiny algorithms, the detection results shown in Figure 15(c)-(d) and Figure 15(g)-(h) indicate improved performance, especially for black area and scratch defects. The YOLO-CBAM-tiny network, with the addition of the CBAM module, demonstrates notably superior detection performance for point defects compared to the YOLO-tiny network. This is primarily reflected in the higher prediction confidence and detection count for rag defects, indicating that the CBAM module can enhance the network's detection accuracy and precision to a certain extent. Specific test results are presented in Figure 15.
6 Conclusion

This study focuses on injector valve seat image data and applies deep learning-based object detection methods to the field of internal defect detection in injector seats. Improvements are made to both the Faster R-CNN detection model and the YOLO detection model. The experiments are conducted on three common objects found in injector valve seats: burrs, scratches, and black areas. The experimental results indicate a significant improvement in accuracy compared to the original algorithms. However, it's worth noting that the number of samples for defects other than burrs is relatively limited. Future work will involve enhancing both the sample dataset and algorithm models to further improve accuracy. It's observed that deepening the network in the experiments can lead to issues such as memory shortages and slow response times. To better apply these techniques in industrial settings, future research can explore training lightweight networks that reduce model size while maintaining performance, thus better meeting the valve seat detection requirements in industrial applications. Investment in this technology can significantly enhance the reliability and
precision of detection, contributing to the advancement of automated production processes and holding practical significance in injector production.

Reference


DOI: [https://doi.org/10.15379/ijmst.v10i4.2324](https://doi.org/10.15379/ijmst.v10i4.2324)

This is an open access article licensed under the terms of the Creative Commons Attribution Non-Commercial License ([http://creativecommons.org/licenses/by-nc/3.0/](http://creativecommons.org/licenses/by-nc/3.0/)), which permits unrestricted, non-commercial use, distribution and reproduction in any medium, provided the work is properly cited.