



Article

Development of an Algorithm for Detecting Real-Time Defects in Steel

Jiabo Yu ¹, Cheng Wang ^{1,*}, Teli Xi ², Haijuan Ju ¹, Yi Qu ¹ , Yakang Kong ¹  and Xiancong Chen ¹

¹ Fundamentals Department, Air Force Engineering University, Xi'an 710051, China; b2283216046@163.com (J.Y.); jhjcumtgx@163.com (H.J.); strsky778@163.com (Y.Q.); kongyakang@126.com (Y.K.); cxcaimath@163.com (X.C.)

² School of Optoelectronic Engineering, Xidian University, Xi'an 710126, China; tlxi@xidian.edu.cn

* Correspondence: valid_01@163.com

Abstract: The integration of artificial intelligence with steel manufacturing operations holds great potential for enhancing factory efficiency. Object detection algorithms, as a category within the field of artificial intelligence, have been widely adopted for steel defect detection purposes. However, mainstream object detection algorithms often exhibit a low detection accuracy and high false-negative rates when it comes to detecting small and subtle defects in steel materials. In order to enhance the production efficiency of steel factories, one approach could be the development of a novel object detection algorithm to improve the accuracy and speed of defect detection in these facilities. This paper proposes an improved algorithm based on the YOLOv5s-7.0 version, called YOLOv5s-7.0-FCC. YOLOv5s-7.0-FCC integrates the basic operator C3-Faster (C3F) into the C3 module. Its special T-shaped structure reduces the redundant calculation of channel features, increases the attention weight on the central content, and improves the algorithm's computational speed and feature extraction capability. Furthermore, the spatial pyramid pooling-fast (SPPF) structure is replaced by the Content Augmentation Module (CAM), which enriches the image feature content with different convolution rates to simulate the way humans observe things, resulting in enhanced feature information transfer during the process. Lastly, the upsampling operator Content-Aware ReAssembly of Features (CARAFE) replaces the "nearest" method, transforming the receptive field size based on the difference in feature information. The three modules that act on feature information are distributed reasonably in YOLOv5s-7.0, reducing the loss of feature information during the convolution process. The results show that compared to the original YOLOv5 model, YOLOv5s-7.0-FCC increases the mean average precision (*mAP*) from 73.1% to 79.5%, achieving a 6.4% improvement. The detection speed also increased from 101.1 f/s to 109.4 f/s, an improvement of 8.3 f/s, further meeting the accuracy requirements for steel defect detection.

Keywords: receptive field; YOLOv5s-7.0; feature extraction; surface defect detection; attention weight



Citation: Yu, J.; Wang, C.; Xi, T.; Ju, H.; Qu, Y.; Kong, Y.; Chen, X. Development of an Algorithm for Detecting Real-Time Defects in Steel. *Electronics* **2023**, *12*, 4422. <https://doi.org/10.3390/electronics12214422>

Academic Editor: Spyridon Nikolaidis

Received: 13 September 2023

Revised: 15 October 2023

Accepted: 23 October 2023

Published: 27 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Steel serves as a critical material in the manufacturing industry, infrastructure sector, and other fields. The quality of steel directly impacts both the quality of the finished products and the construction of infrastructure. Therefore, ensuring strict quality control for steel is exceptionally important, as it constitutes the initial guarantee for product qualification.

In the production and processing of steel products, defects such as pitting, scratches, and patches often occur on the surface of the steel. These defects can lead to a decrease in the quality and performance of steel products, thereby reducing the reliability and service life of the steel. Therefore, measures must be taken during steel production and usage to accurately and quickly detect and eliminate these non-compliant steel materials. The methods for detecting surface defects in steel can be categorized as manual inspection and

machine inspection. Manual inspection is prone to randomness, and the accuracy is heavily influenced by the experience and attentiveness of the inspectors. Moreover, small defects in steel may not be easily noticed by human workers [1]. On the other hand, machine inspection offers the advantages of low cost, high efficiency, and good stability.

Object detection algorithms serve as the core of machine-based detection. However, there are still two challenges for mainstream object detection algorithms in recognizing surface defects on steel. Firstly, there is a high similarity between the different types of defects on the steel surface, while some similar defects exhibit significant variations [2]. Secondly, the multitude of defect types on the steel surface leads to imprecise classification results [3]. These two challenges result in decreased precision and a slower detection speed of the object detection algorithms. And mainstream object detection algorithms can no longer meet the strict defect detection requirements of factories [4]. Therefore, there is an urgent need for more advanced algorithms with improved performance in order to meet the production demands of factories.

As a solution to this problem, the architecture of the YOLOv5 model was introduced. However, it should be noted that in the YOLOv5 model architecture, the propagation of convolution calculations leads to the loss of feature information. Therefore, it is crucial to focus on the rational distribution of structures and module replacements that are more suitable for machine computations. Zhang et al. [5] proposed an improved algorithm based on YOLOv5 by incorporating deformable modules to adaptively adjust the perception field scale. They also introduced the ECA-Net attention mechanism to enhance feature extraction capabilities. The improved algorithm achieved a 7.85% increase in the mean average precision (*mAP*) compared to the original algorithm. Li et al. [6] put forward a modified algorithm based on YOLOv5, where they integrated the Efficient Channel Attention for Deep Convolutional Neural Networks (ECA-Net)—an attention mechanism to emphasize feature extraction in defective regions. They replaced the PANet module with the Bidirectional Feature Pyramid Network (BiFPN) module to integrate feature maps of different sizes. The results showed that compared to the original YOLOv5 model, the *mAP* increased by 1% while the computation time decreased by 10.3%. Guizhong Fu et al. [7] proposed a compact Convolutional Neural Network (CNN) model that focused on training low-level features to achieve the accurate and fast classification of steel surface defects. This model demonstrated high precision performance with a small training dataset, even under various modes of interference such as non-uniform illumination, motion blur, and camera noise. Yu He et al. [3] proposed a fusion of multi-level feature maps approach, enabling the detection of multiple defects on a single image. They utilized a Region Proposal Network (RPN) to generate regions of interest (ROI), and the final conclusions were produced by the detector. The results showed an accuracy of 82.3% on the MEU-DET dataset.

Real-Time Object Detection is also an important requirement for the industrialization of steel defect detection. Qinglang et al. [8], focusing on the elongated nature of road cracks, proposed an improved algorithm based on YOLOv3. They fused high-level and low-level feature maps to enhance feature representation and achieve real-time detection of road surfaces. To achieve faster object detection, Jiang et al. [9] introduced the YOLOv4-tiny. This model replaced two (CSPBlock) modules with two ResnetBlock-D modules to improve computation speed. Furthermore, residual network blocks were utilized to extract more feature information from images, thus improving the detection accuracy. The results showed that the improved algorithm achieved a faster detection rate without sacrificing accuracy.

There are two main categories of deep learning-based object detection methods: one-stage and two-stage. The one-stage approach directly utilizes convolutional neural networks to extract image features, and perform object localization and classification. Classic algorithms in this category include the YOLO [10–13] series and SSD [14,15] series. In contrast, the two-stage approach generates candidate regions before performing the aforementioned processes. Popular algorithms in this category include the RCNN [16,17] series, SPPNet [18], and R-FCN [19]. Considering its practicality within factories, the YOLOv5

model from the one-stage category is commonly used. It offers a faster detection speed but suffers from a lower accuracy. To address this problem, the authors made improvements to certain structures in YOLOv5 specifically for steel training datasets. These modifications made the algorithm's structure more suitable for machine feature extraction, resulting in an improved detection speed and an increased average accuracy.

This article begins by introducing the basic architectural features of the YOLOv5-7.0 algorithm. Afterwards, the author addresses several issues affecting the measurement accuracy in the original algorithm and proposes three improved modules to replace the problematic ones. The structure and distinguishing characteristics of these replacement modules are emphasized. Subsequently, details regarding the experimental setup, dataset, evaluation metrics, and other relevant information are provided.

The article then presents comparative experimental results, including comparisons of six different module replacements for the c3 module, three different forms of CAM for replacing the SPPF module, and eight different forms of Carafe for replacing the nearest module. Additionally, a comparative experiment is conducted using the three selected optimal modules in combination. Furthermore, the improved algorithm is compared with mainstream detection algorithms.

Finally, the article concludes by presenting comparative visual results of the detection performance between the improved algorithm and the original algorithm.

2. Materials and Methods

2.1. YOLOv5-7.0 Algorithm

The YOLOv5 algorithm is one of the typical one-stage algorithms that utilizes a series of convolutional computations to extract hierarchical features from images within the backbone network. By fusing high-level semantic information with low-level details, it facilitates effective classification and localization, ultimately performing object detection in the "detect" stage.

Version 7.0 of YOLOv5 represents the latest release in the YOLOv5 series and brings significant improvements to instance segmentation performance. YOLOv5-7.0 offers five different models, namely YOLOv5s, YOLOv5m, YOLOv5n, YOLOv5l, and YOLOv5x, arranged in increasing order of module size. These models exhibit varying speeds and accuracy levels.

The network structure of YOLOv5-7.0 consists of three components: the backbone, neck, and head. The backbone is responsible for extracting high-level semantic features from input images. The default backbone network in YOLOv5-7.0 is CSPDarknet, which employs stacked convolutional and pooling layers to reduce the image resolution while capturing essential features. After the processing by the backbone, the neck module in YOLOv5-7.0 performs feature fusion and processing, combining features from different levels to extract more comprehensive information. A Feature Pyramid Network (FPN) is commonly used as the fusion method in YOLOv5-7.0, enabling the extraction of scale information from multiple feature maps. The processed information from the neck module is then fed into the head module, which employs non-maximum suppression to generate three prediction results for predicting object locations and categories.

The algorithm proposed in this paper aims to enhance the YOLOv5s architecture of version 7.0, with the goal of improving detection efficiency and reducing error rates in practical applications. This enhancement seeks to achieve both rapid detection and increased accuracy, catering to the requirements of factories.

2.2. YOLOv5-7.0 Improvement

2.2.1. C3F Operator

C3 is derived from the Cross Stage Partial Networks (CSPNet) architecture. C3 has two variations: one in the backbone of YOLOv5-7.0 as shown in Figure 1a, and another in the head of YOLOv5-7.0 as shown in Figure 1b. The difference between BottleNeck1 and

BottleNeck2 lies in their input processing. In BottleNeck1, the result is obtained by adding the output of Conv applied twice to the initial input.

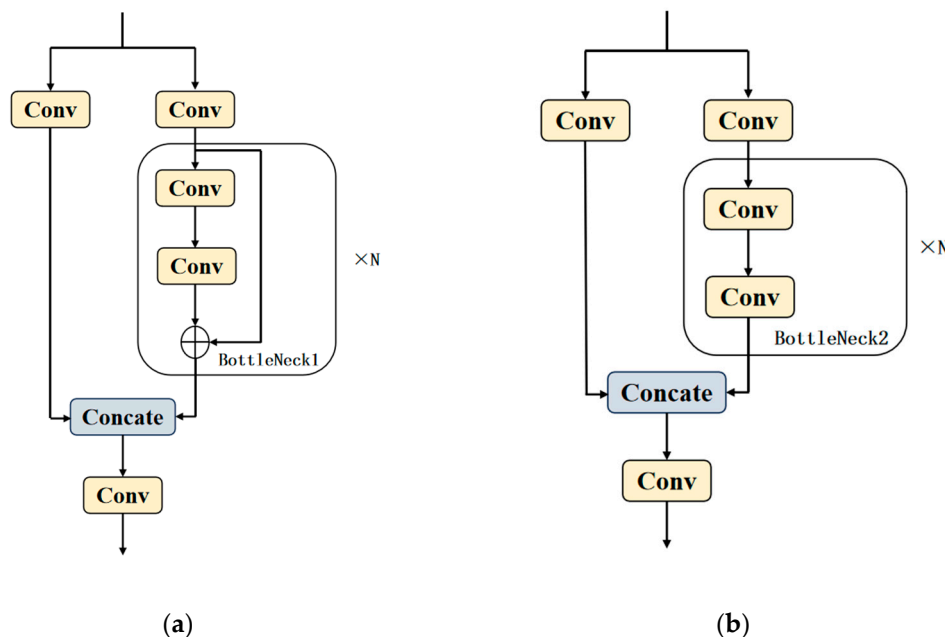


Figure 1. Two forms of the C3 module: (a) the structure of BottleNeck1, (b) the structure of BottleNeck2.

Many studies have shown that the differences in feature maps across different channels of the same image are minimal [20,21]. While most algorithms aim to reduce computational complexity and improve accuracy, they have not effectively addressed the issue of computing redundant features across different channels. The C3 structure, which follows traditional methods for processing feature maps in each channel, inevitably results in redundant computations between similar feature maps.

An improved version of the C3 module in YOLOv5s-7.0, known as C3-Faster (C3F), has been introduced to effectively address the aforementioned issues. Its design concept is derived from the PConv module used by Jierun Chen [22] in FasterNet. In C3F, the unprocessed data are concatenated with the PConv module for further computation. This approach significantly reduces the computational workload while enhancing the accuracy. The structure of the PConv module can be seen in Figure 2a, while the structure of C3F is depicted in Figure 2b.

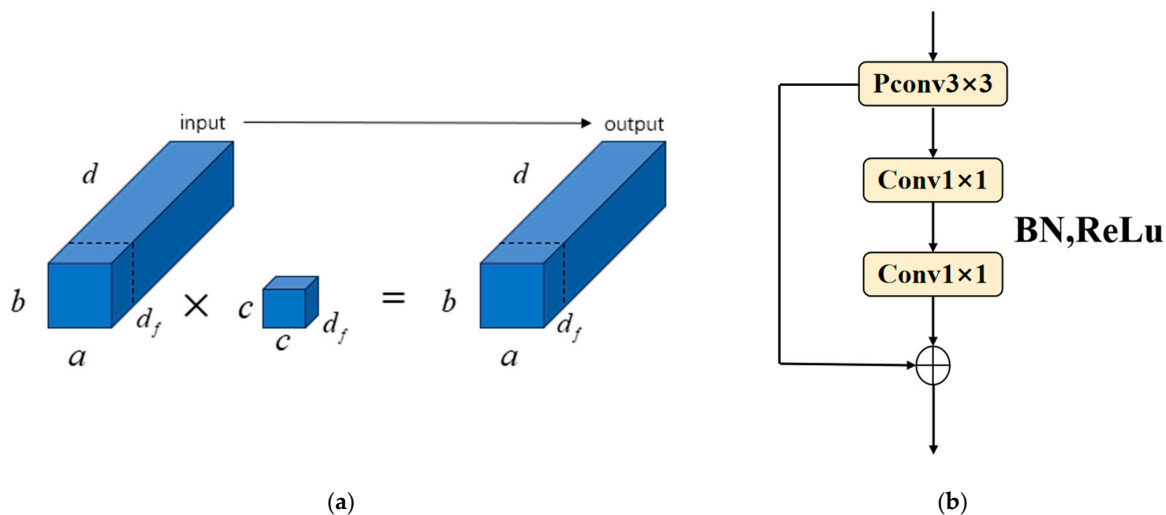


Figure 2. Overview diagram of the C3F model: (a) the structure of PConv, (b) the structure of C3F.

C3F is a fundamental operator that can be embedded into various neural networks to address the issue of redundant convolutions that often occur in neural network computations. By reducing memory access, C3F performs conventional Conv convolutions on only a portion of the input data, typically treating either the first or last channel as the representation of the entire image. The floating point operations (FLOPs) for conventional Conv are

$$a \times b \times c^2 \times d^2 \quad (1)$$

The corresponding amount of memory access is

$$a \times b \times 2d + c^2 \times d^2 \quad (2)$$

In a contrasting manner, after replacing the C3 module with a C3F module, the FLOPs are reduced to

$$a \times b \times c^2 \times d_f^2 \quad (3)$$

The corresponding amount of memory access is

$$a \times b \times 2d_f + c^2 \times d_f^2 \quad (4)$$

where $d/d_f \geq 2$ is the number of channels.

This means that the FLOPs for Conv are at least four times greater than the FLOPs for C3F, and the memory access for Conv is more than double that of C3F.

In order to fully utilize the unused channels ($d - d_f$) in the feature maps after the aforementioned operations, these channels are transformed into pointwise convolutions (Conv1 × 1) and added to the center position of the PConv module. This results in a convolutional layer with efficient computational capabilities. Batch normalization (BN) is then applied to further improve the convergence speed. To avoid the issues of gradient vanishing or exploding during computation, a Rectified Linear Unit (ReLU) is used as the activation function to enhance the non-linear fitting ability of the upper and lower layer function values. Subsequently, pointwise convolutions, average global pooling, and fully connected layers are employed to merge and output the final results.

This approach of processing digital images allows for a reduction in computational workload, thereby enhancing the speed of computation without sacrificing accuracy. By intelligently combining convolutional layers (attaching Conv1 × 1 layers to the center position of the PConv module, forming a T-shaped structure), greater attention is given to this central position, which has the highest Frobenius norm. This arrangement aligns with the pattern of feature extraction in images and can even reduce the computational workload while improving the precision.

2.2.2. Information Feature Enhancement Module—CAM

To address the issue of the varying candidate box sizes after the first stage of detection in RCNN, He, Kaiming et al. [23] proposed the spatial pyramid pooling (SPP) structure to fix the detection box size. The SPP structure incorporates parallel operations of MaxPool2d with 5×5 , 9×9 , and 13×13 modules, as shown in Figure 3a, to ensure fixed-size outputs. Subsequently, He, Kaiming further improved the SPP structure by introducing the spatial pyramid pooling-fast (SPPF) structure. The innovation in the network architecture is due to the replacement of the MaxPool2d operation in SPP with three consecutive 5×5 modules, as depicted in Figure 3b. This modification leads to outputs of the same size while reducing the computational workload and improving the detection speed. However, it is important to note that the SPPF structure inherently involves the loss of partial information during the pooling process. If the convolutional operations prior to pooling fail to learn sufficient features, it can have a significant impact on the detection results.

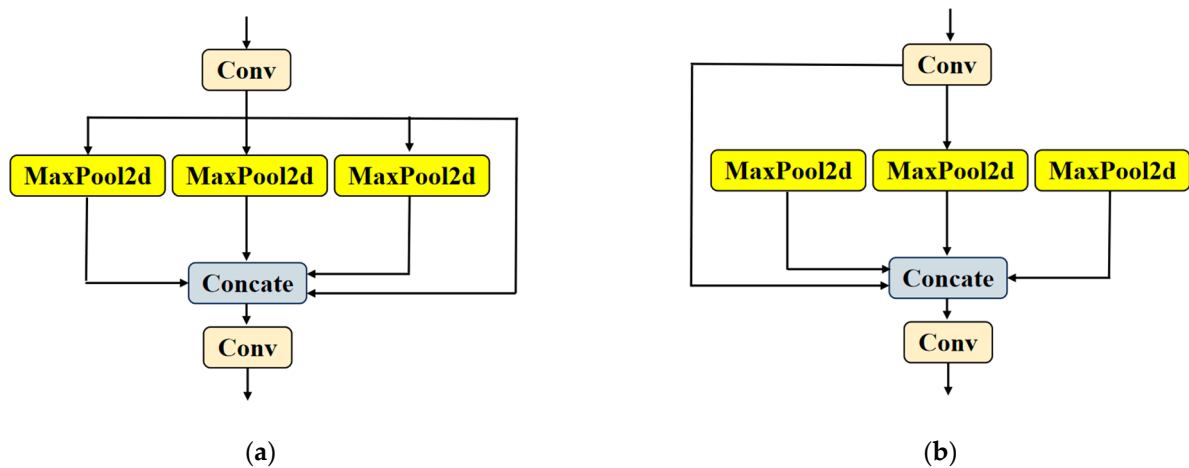


Figure 3. Structure of pooling layer in YOLOv5: (a) the structure of SPP, (b) the structure of SPPF.

The utilization of the context augmentation module (CAM) has demonstrated remarkable effectiveness in handling low-resolution targets, while also providing a robust solution to the aforementioned issues. The conceptual architecture of CAM was a solution to address the imbalanced training dataset and limitations of the network. In this study, the CAM module is incorporated into YOLOv5s-7.0, replacing the SPPF structure and further enhancing the computational precision.

The CAM module uses convolution with varying dilation rates to process images and enrich the contextual information from both the upper and lower regions of the image. By combining the feature information from multiple images with different dilation rates, the expression of the features becomes more evident. The structure of the CAM module is illustrated in Figure 4.

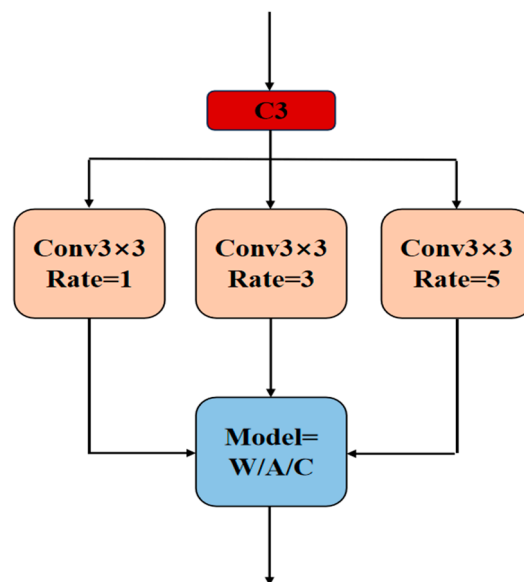


Figure 4. The structure of CAM.

In the above figure, 3×3 convolution kernels are applied with rates of 1, 3, and 5. This approach draws inspiration from the way humans recognize objects, where using a rate of 1 is akin to observing details up close, such as when observing a panda and noticing its creamy white torso, sharp black claws, and black ears. However, these details may not be sufficient to determine the object’s category. By contrast, performing convolution calculations with rates of 3, 5, or even larger rates is akin to viewing an object in its entirety, comparing it to the surrounding environment. Applying this visual approach to machine

learning has demonstrated comparable results. By simulating this method of human observation, machine learning adjusts the rate to obtain different receptive fields and then fuses them for improved accuracy. This learning technique works particularly well for smaller targets at various resolutions.

The CAM module includes three types of weight forms: Weight, Adaptive, and Concatenation, as illustrated in Figure 5.

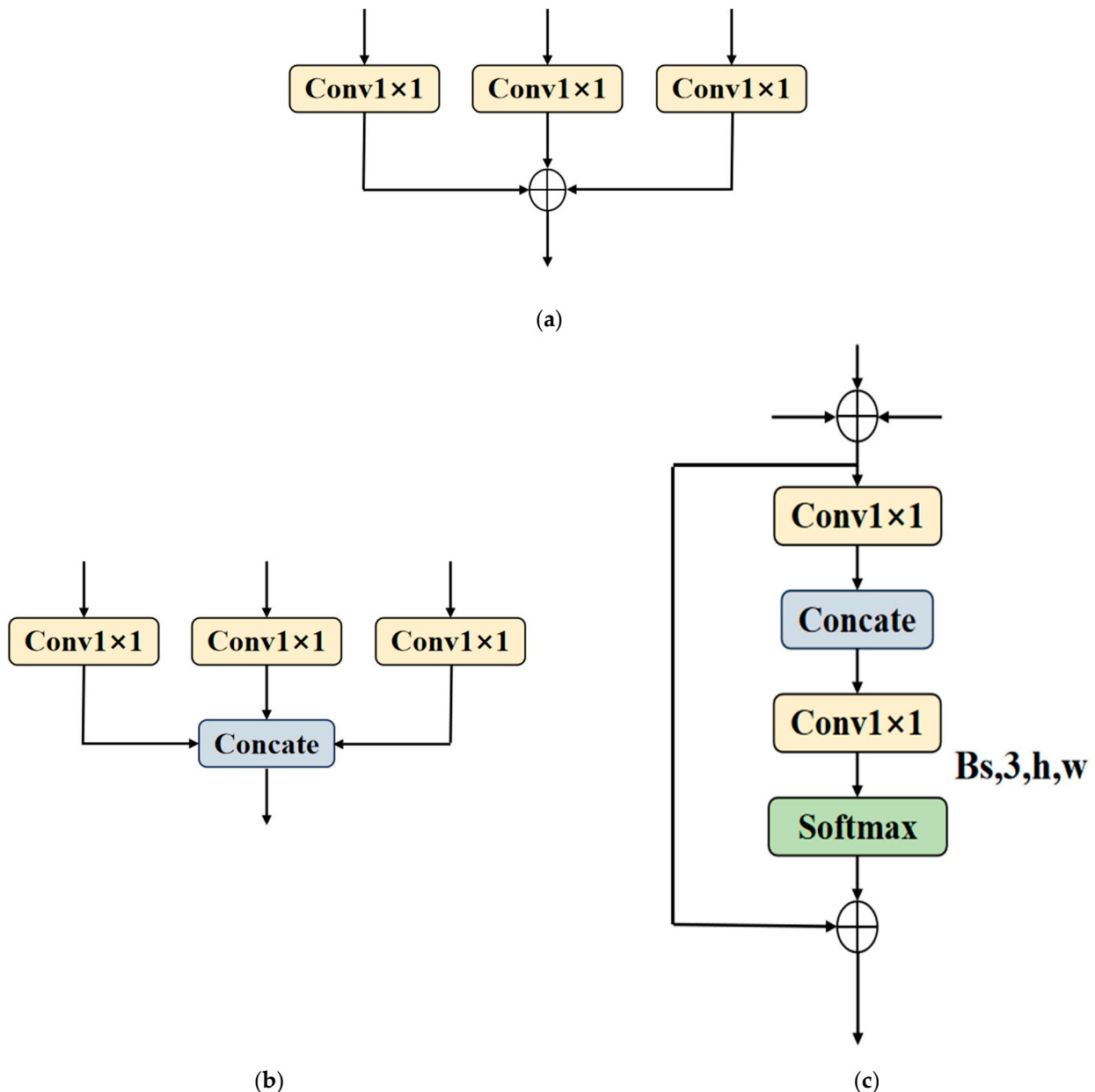


Figure 5. The three modes of CAM: (a) Weight, (b) Concatenation, (c) Adaptive.

The Weight mode involves adding the information after it undergoes $\text{Conv1} \times 1$ processing three times. The Adaptive mode adopts an adaptive approach to match the weights, where $[bs, 3, h, w]$ in the diagram represents spatially adaptive weight values. The Concatenation mode combines the information after it undergoes $\text{Conv1} \times 1$ processing three times through weighted fusion. There is no phenomenon of one mode being better than the other among these three modes. The performance of the CAM module is influenced by factors such as the dimensions of different datasets' images, their characteristic features, and the connections between different modules.

2.2.3. Variable Receptive Field Module—CARAFE

Upsampling is widely used in deep neural networks to enlarge high-level features. In version 7.0 of YOLOv5, the nearest module, which means the nearest neighbor interpolation, is employed for upsampling operations. When using the upsampling module in image processing, each point in the output image can be mapped to the input image. The mapped point takes on the value of the nearest point among the adjacent four points in the input image, assumed as a, b, c, and d. The “nearest” module does not add computational complexity as it only requires passing values and processing data blocks. However, this module also has several drawbacks that significantly affect the results of neural network computations. The data transformation approach of the “nearest” module reduces the gradual correlation between adjacent pixels in the original image. Additionally, its 1×1 perceptual field is very small, and this uniform and simplistic sampling method does not effectively utilize the semantic information of the feature map.

The Content-Aware ReAssembly of FEatures (CARAFE) module, initially proposed by CL Chen et al. [24] can address the limitations of nearest neighbor methods. CARAFE provides a larger receptive field for images by generating corresponding kernels based on the semantic information of the feature map. This allows for a more focused consideration of the content surrounding the image, thereby improving the detection accuracy without significantly increasing the computational requirements. The execution process of the CARAFE module consists of two steps: “Creation of the Upsampling Kernel” and “New Kernel Convolution Calculation”. The creation of the upsampling kernel is illustrated in Figure 6, while the process of new kernel convolution calculation is depicted in Figure 7.

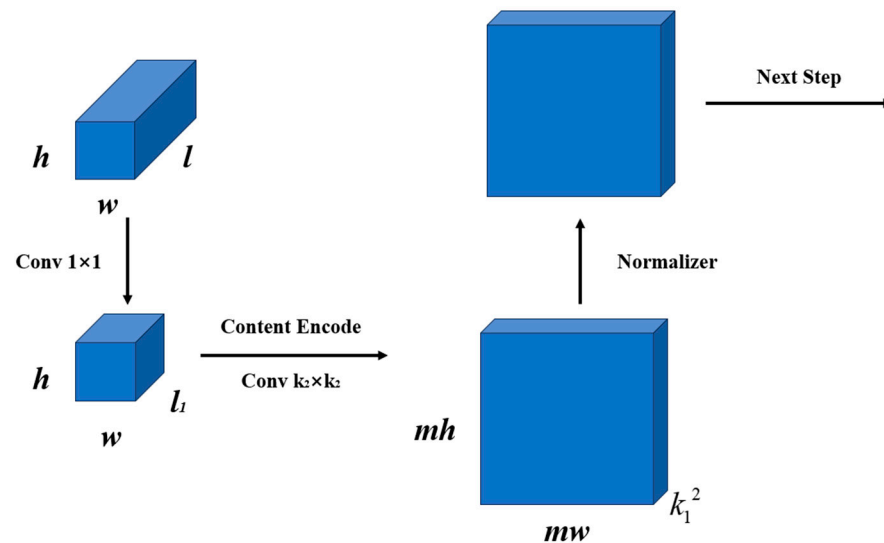


Figure 6. Creation of the upsampling kernel.

The image above depicts the process of creating an upsampling kernel. To meet the requirement of reducing the computational complexity, the input image of size $h \times w \times l$ undergoes calculations using a 1×1 convolutional operation, compressing the channels to l_1 . Next, the content is re-encoded and passed through a $k_2 \times k_2$ convolutional operation, dividing the l_1 channels into m^2 groups of k_1^2 channels. These channels are then rearranged and combined in a $mh \times mw \times k_1^2$ structure after unfolding the spatial dimensions. Finally, the rearranged $mh \times mw \times k_1^2$ structure is normalized, ensuring that the weights of the newly created upsampling kernel sum up to 1.

At each position in the output feature map $h \times w \times l$, there is a mapping to the input feature map. For example, in the rectangular shape shown in the image, the yellow region corresponds to a 6×6 area in the input feature map. Then, the upsampling kernel $mh \times mw \times k_1^2$ is rearranged into a $k_1 \times k_1$ structure within the red region, and the dot product between this rearranged kernel and the 6×6 input area yields an output

value. The yellow region of the rectangular shape determines the corresponding positional coordinates of the red region in the upsampling kernel, and a single upsampling kernel can be shared among all channels at that corresponding position.

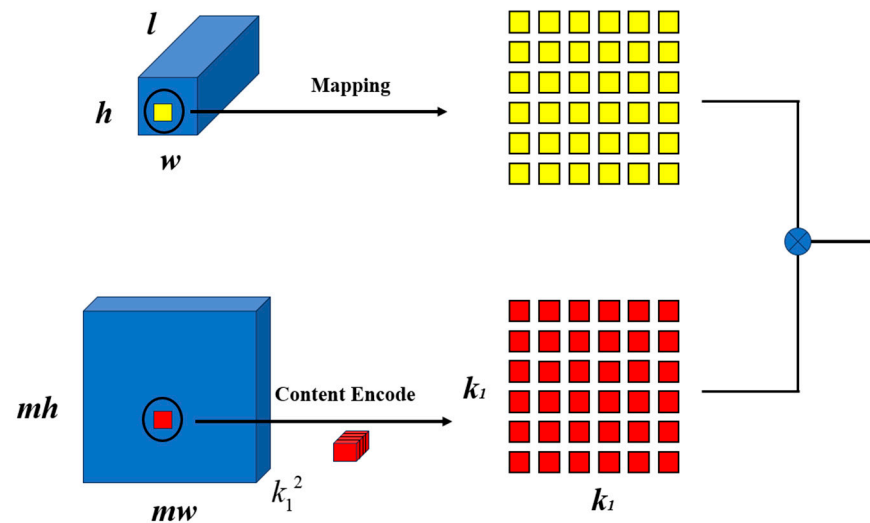


Figure 7. New kernel convolution calculation.

The sizes $k_1 \times k_1$ and $k_2 \times k_2$ represent the reorganized upsampling kernel and the receptive field, respectively. A larger receptive field means a larger range of content perception, requiring a larger reorganized upsampling kernel to match the increased receptive field. CL Chen introduced and explained the permutation patterns of $k_1 k_2$ in the Carafe module, which include [1, 3], [1, 5], [3, 3], [3, 5], [3, 7], [5, 5], [5, 7], and [7, 7]. Additionally, it was noted that the parameter size increases as the value of k increases.

2.2.4. The Architecture of the Improved Model Based on YOLOv5s-7.0

The proposed improved algorithm is based on YOLOv5s-7.0. The purpose of the improvement is to improve the detection performance of small target objects, such as those with low pixel clarity. The structure of the improved algorithm is depicted in Figure 8. In this improvement, all seven C3 modules in the original algorithm's backbone structure and four C3 modules in the neck structure are replaced with C3F modules. The C3F modules reduce redundant computations and their T-shaped pattern allows the receptive field to focus more on the center position with the maximum Frobenius norm.

Additionally, the SPPF in the original algorithm's backbone is replaced with a CAM. The CAM obtains three different receptive fields with rates of 1, 3, and 5 when processing an image. This allows the algorithm to focus more on contextual information, reducing the impact of low pixel clarity features. Moreover, three fusion methods (weight, adaptive, and concatenation) are proposed for combining the obtained receptive fields.

Furthermore, the two nearest neighbor upsampling operators—the “nearest” modules in the original algorithm's neck—are replaced with a feature recombination module called CARAFE, which focuses on semantic information. Based on the values of k_1 and k_2 ([1, 3], [1, 5], [3, 3], [3, 5], [3, 7], [5, 5], [5, 7], and [7, 7]), a total of 16 combinations can be obtained from the two “nearest” modules. This approach of increasing the receptive field differs from directly expanding it with CAM, as it reconstructs the upsampling kernel based on the feature information to enhance the receptive field.

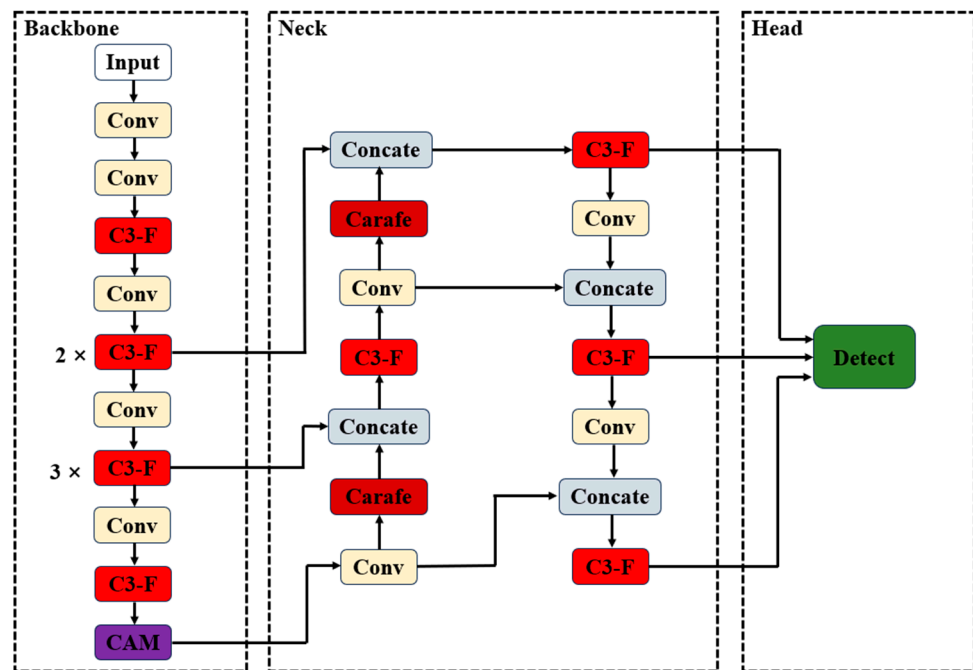


Figure 8. The improved architecture based on version 7.0 of YOLOv5s.

By combining these three improvement methods without significantly increasing the computational complexity, the algorithm achieves a significant improvement in its accuracy. A comparative analysis will be presented in the following sections.

3. Experiments and Analysis

3.1. Experimental Environment

The experimental platform used in this study is the AutoDL cloud service platform provided by SeetaTech Technology Ltd. The basic configuration is shown in Table 1. During the training process using the improved algorithm based on YOLOv5s-7.0, no pre-trained weights were used. The basic parameters were set as follows: epochs = 300, batch-size = 32, workers = 22, initial learning rate = 0.001, and confidence threshold = 0.5.

Table 1. Setup and parameters of the experimental platform.

Name	Type of Configuration
Mirror image environment	PyTorch 1.11.0 + Cuda 11.3
Deep learning framework	ubuntu20.04
Program	Python 3.8
GPU	RTX 4090 (24 GB)
CPU	22 vCPU AMD EPYC 7T83 64-Core Processor
Memory capacity	90 GB

3.2. Dataset and Evaluation Indicators

3.2.1. Dataset

This article utilizes a publicly available dataset called “NEU-DET” [25] provided by the Northeastern University of China for training and testing purposes. A total of six types of damage patterns shown in Figure 9 were selected, including scratches, crazing, inclusions, patches, pitting, and rolled-in scales, which are common defects in steel. The dataset consists of 1800 images, each with a resolution of 200 × 200 pixels, and contains 300 images for each defect category. To facilitate effective training, validation, and testing, the dataset was divided in an 8:1:1 ratio. This means that 1440 images were used for training, while the remaining 360 images were evenly distributed between the validation and testing sets.

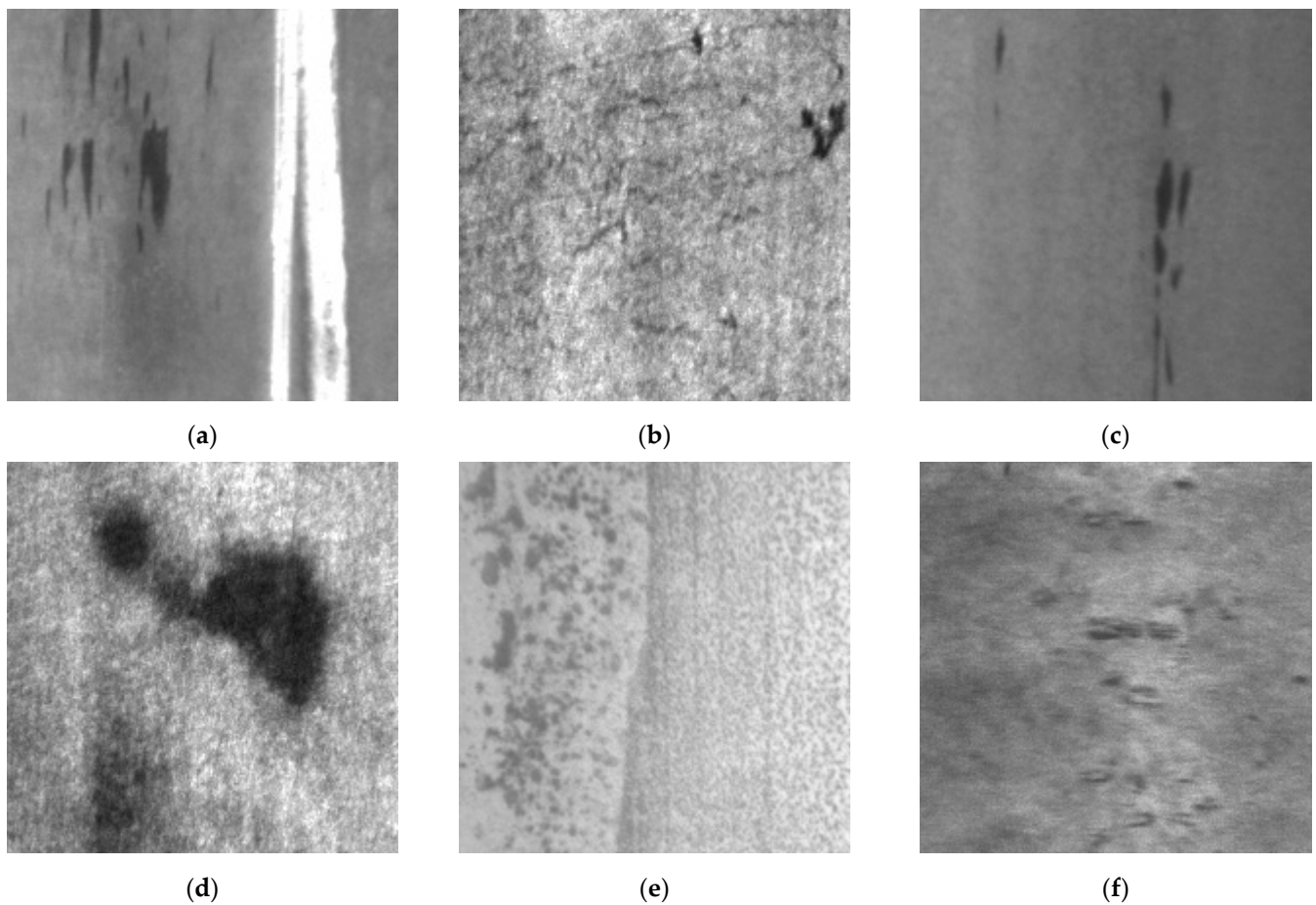


Figure 9. Six kinds of defect pictures: (a) scratches, (b) crazing, (c) inclusions, (d) patches, (e) pitting, (f) rolled-in scales.

3.2.2. Evaluation Indicators

In the object detection experiments, the algorithm's performance is evaluated using precision (P), recall (R), average precision (AP), mean average precision (mAP) and frames per second (FPS). The corresponding expressions are as follows:

$$P = \frac{TP}{TP + FP} \quad (5)$$

$$R = \frac{TP}{TP + FN} \quad (6)$$

$$AP = \int_0^1 P(R) dR \quad (7)$$

$$mAP = \frac{\sum_{i=0}^n AP = (i)}{n} \quad (8)$$

In these expressions, TP is expressed as the number of pictures with positive samples for both labeling and detection; FP represents the number of images where the ground truth is negative but the detection is positive; and FN represents the number of images where the ground truth is positive but the detection is negative. FPS is an indicator of detection speed, representing the number of images that can be detected per second.

3.3. Module Replacement Comparative Experiment

In order to analyze and demonstrate the impact of different improvement methods on the performance of defect detection algorithms on the “Steel Strip NEU-DET”, we will conduct individual or combined comparative analyses of three modules replaced in the improved algorithm. The following sections will experimentally evaluate the comparative effects of replacing several common C3 modules of the same type, compare the performance of three weight options in the CAM module, and assess the comparative effects of varying k_1k_2 parameter combinations in the Carafe module.

3.3.1. Comparative Experiment of C3 Module Replacement

In addition to the C3F module being able to replace C3, in recent years, many excellent papers have proposed convolutional kernels that can improve the algorithm’s performance. For example, Yolov8 introduces a module called cross stage partial feature fusion (C2f), which merges high-level and low-level features to enhance accuracy. The deformable convolution V2 (DCN) [26] is a variable convolution that adapts the kernel size according to the network layer size. The switchable atrous convolution (SAC) [27] by Google’s team is used for tiny object detection. The distribution shifting convolution (DSCConv) [28] quantizes the original convolutional parameters and shifts the distribution to reduce the memory usage and accelerate the computation speed. The diverse branch block (DBB) [29], proposed in CVPR2021, combines multiple branches with rich features into a main branch through reparameterization. These convolutional kernels will be incorporated into YOLOv5s-7.0, and their performance will be compared and evaluated, as shown in Table 2.

Table 2. Comparison of parameters of replacing the C3 module.

Model	Params ($\times 10^6$)	FPS (f/s)	mAP (%)
YOLOv5s-7.0(C3)	7.03	101.1	73.1
YOLOv5s-7.0(C3F)	5.8	111.2	76.3
YOLOv5s-7.0(C2f)	9.29	119	76.2
YOLOv5s-7.0(SAC)	8.29	85.7	76
YOLOv5s-7.0(DBB)	7.03	56.8	74.1
YOLOv5s-7.0(DCN)	7.09	121.6	73.9
YOLOv5s-7.0(DSCConv)	7.03	129.4	73.8

The improved algorithms that replaced the convolutional kernels with C3F, C2f, SAC, DBB, DCN, and DSCConv have increased the mAP compared to the original algorithm by 3.4%, 3.3%, 3.1%, 1.2%, 1%, and 0.9%, respectively. All six improved algorithms of YOLOv5s-7.0 can enhance the detection accuracy. However, only the algorithm, which incorporates C3F, C2f, DCN, and DSCConv, improves both the detection accuracy and speed. The algorithm that incorporates DCN and DSCConv has a weaker ability to improve the detection accuracy compared to the other two convolutional kernels. Additionally, considering that C2f has approximately 3.49×10^6 more computational parameters than C3F while achieving a similar detection accuracy, in the subsequent ablation experiments, only the case of integrating C3F will be considered.

3.3.2. Comparative Experiment of CAM Module Replacement

The three fusion modes of the CAM module have been described in detail in Section 2.2.2. of this paper. In this paper, when the original algorithm was fused with the CAM module, it was discovered that the Adaptive mode, which allows for the importance analysis of the results through attention mechanisms, performed the best on this dataset. It is more suitable for this dataset, as shown in Table 3.

Table 3. Comparison of the performance of CAM in three modes.

CAM	Params ($\times 10^6$)	<i>mAP</i> (%)
Weight	14.25	75.4
Adaptive	14.25	75.7
Concatenation	14.51	75.2

The three fusion modes of the CAM module, when compared to the original algorithm, have all shown an improvement in the detection accuracy of at least 2.3%. Among them, after multiple experiments on the “NEU-DET” dataset, the Adaptive mode of the CAM module achieved the highest *mAP* of 75.7%. Therefore, in the subsequent ablation experiments, only the Adaptive mode of the CAM module will be considered. Although the CAM module can enhance the detection accuracy, it has a higher computational cost compared to the SPPF in the original algorithm. To further reduce the overall computational cost of the algorithm, future improvements can consider integrating the CAM module in the higher levels of the original algorithm.

3.3.3. Comparative Experiment of Carafe Module Replacement

In version 7.0 of YOLOv5s, this paper replaces both the “nearest” upsampling operators with Carafe. The variation in the *k* values for the higher-level Carafe has a relatively small impact on the overall parameter size of the algorithm. Even when replacing the high-level upsampling operator with Carafe [7, 7], the overall parameter size of the algorithm only increases from 7.03×10^6 to 7.67×10^6 . This is due to the geometric growth in the parameter size resulting from increasing the *k* values of both the higher-level and lower-level Carafe. In this study, the lower-level upsampling operator is set as Carafe [1, 3], and the different combinations of the *k* values for the higher-level Carafe are explored to find the optimal integration of Carafe into YOLOv5s-7.0. The comparison of the parameter size and detection accuracy of the eight combinations of Carafe-enhanced algorithm integration is shown in Table 4.

Table 4. Comparison of the performance of Carafe in eight modes.

$[k_1, k_2]$	Params ($\times 10^6$)	<i>mAP</i> (%)
[1, 3]	7.06	74.6
[1, 5]	7.06	74.2
[3, 3]	7.07	74.1
[3, 5]	7.11	75
[3, 7]	7.17	74
[5, 5]	7.21	74.3
[5, 7]	7.37	75.2
[7, 7]	7.67	73.2

All the fusion combinations incorporating the Carafe module lead to an improved detection accuracy. Through a comparison of the *mAP* results, it was found that the optimal detection accuracy is achieved when the recombination sampling kernel k_1 is set to 1, 3, or 5, and the receptive field k_2 is set to 3, 5, or 7. Among the combination modes [1, 3], [3, 5], and [5, 7], the detection accuracy increases with the increase in the *k* values, with the [5, 7] combination achieving the highest *mAP* of 75.2%. Moreover, the increase in the parameter size is not substantial. Therefore, in the subsequent ablation experiments, only the Carafe mode [5, 7] needs to be considered.

3.4. Ablation Experiment and Analysis

To investigate the impact of the three aforementioned improvement combinations on the detection accuracy of surface defects in steel strips, ablation experiments were conducted as shown in Table 5.

Table 5. Ablation experiment.

Model	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7	Group 8
YOLOv5s-7.0	✓	✓	✓	✓	✓	✓	✓	✓
C3F		✓			✓	✓		✓
CAM(Adaptive)			✓		✓		✓	✓
Carafe [5, 7]				✓		✓	✓	✓
Params ($\times 10^6$)	7.03	5.8	14.25	7.37	13.0	6.14	14.6	13.35
<i>mAP</i> (%)	73.1	76.3	75.7	75.2	77.9	77.1	76.7	79.5

In the table, ‘✓’ indicates the presence of the module Comparing. Group 2 with Group 1, the addition of the C3F module resulted in a reduction in the 1.23×10^6 parameters. However, the *mAP* increased from 73.1% to 76.3%, showing an improvement of 3.2%. When comparing Group 3 with Group 1, the inclusion of the CAM module led to an almost doubling of the parameter count, but the detection accuracy also improved by 2.6%. In the case of Group 4 compared to Group 1, the integration of the Carafe module only increased the parameter count by 0.34×10^6 , yet the detection accuracy improved by 2.1%. Groups 5, 6, and 7 represent paired combinations of C3F, CAM, and Carafe, respectively. When compared to their individual integration with YOLOv5s-7.0, all these combination modes exhibited further improvements in *mAP*, suggesting that there is no adverse reaction when combining C3F, CAM, and Carafe. Group 8 fused C3F, CAM, and Carafe into YOLOv5s-7.0, and created a new model named “YOLOv5s-7.0-FCC”. Furthermore, the code for the YOLOv5s-7.0-FCC model is publicly available and readers are free to use it. The download location can be found in the Supplementary Materials Data File S1. Lastly, Group 8, when compared to Group 1, saw an increase in the parameter count from 7.03×10^6 to 13.35×10^6 , while the detection accuracy soared from 73.1% to 79.5%. Sacrificing some parameters to achieve significant improvements in the detection accuracy proves to be meaningful.

4. Results

4.1. Superior Features of YOLOv5s-7.0-FCC in Mainstream Object Detection Algorithms

Currently, in engineering applications, other mainstream object detection algorithms such as Faster-RCNN, SSD, YOLOv3, and YOLOv4 are widely used. They are comprehensively evaluated and compared with YOLOv5s-7.0-FCC, and the specific results are presented in Table 6.

Table 6. Comparison of performance for Mainstream Object Detection Algorithms.

Algorithm	Params ($\times 10^6$)	FPS (f/s)	<i>mAP</i> (%)
Faster-RCNN	56.81	56.2	61.5
SSD	48.83	216	72
YOLOv3	61.55	83.1	70.6
YOLOx	18.01	98	71.9
YOLOv5s-7.0	7.03	101.1	73.1
YOLOv5s-7.0-FCC	13.35	109.4	79.5

The table above shows the results obtained on the “NEU-DET” dataset by testing various algorithms. YOLOv5s-7.0-FCC has fewer parameters than Faster-RCNN, SSD, YOLOv3, and YOLOv4, and it achieves the highest detection accuracy. In terms of the detection speed, except for being slower than SSD, it is faster than the other four algorithms. With a parameter count of 13.35×10^6 , YOLOv5s-7.0-FCC improves the detection accuracy to 79.5%. Compared to the original algorithm, although the parameter count of YOLOv5s-7.0-FCC increases almost twice, the detection speed improves from 101.1 f/s to 109.4 f/s, and the *mAP* increases by 6.4%. Overall, YOLOv5s-7.0-FCC enables real-time detection functionality in engineering applications and exhibits significantly superior performance compared to other mainstream object detection algorithms.

4.2. Comparison of YOLOv5s-7.0-FCC and YOLOv5s-7.0 Detection Results

The comparison between the detection results of YOLOv5s-7.0 and YOLOv5s-7.0-FCC is shown in Figure 10. The figure includes six types of defects and their corresponding confidence scores. From the figure, it is evident that YOLOv5s-7.0-FCC yields more detection boxes, allowing for the detection of targets with less distinct features. Additionally, the confidence scores are generally higher compared to the original algorithm.

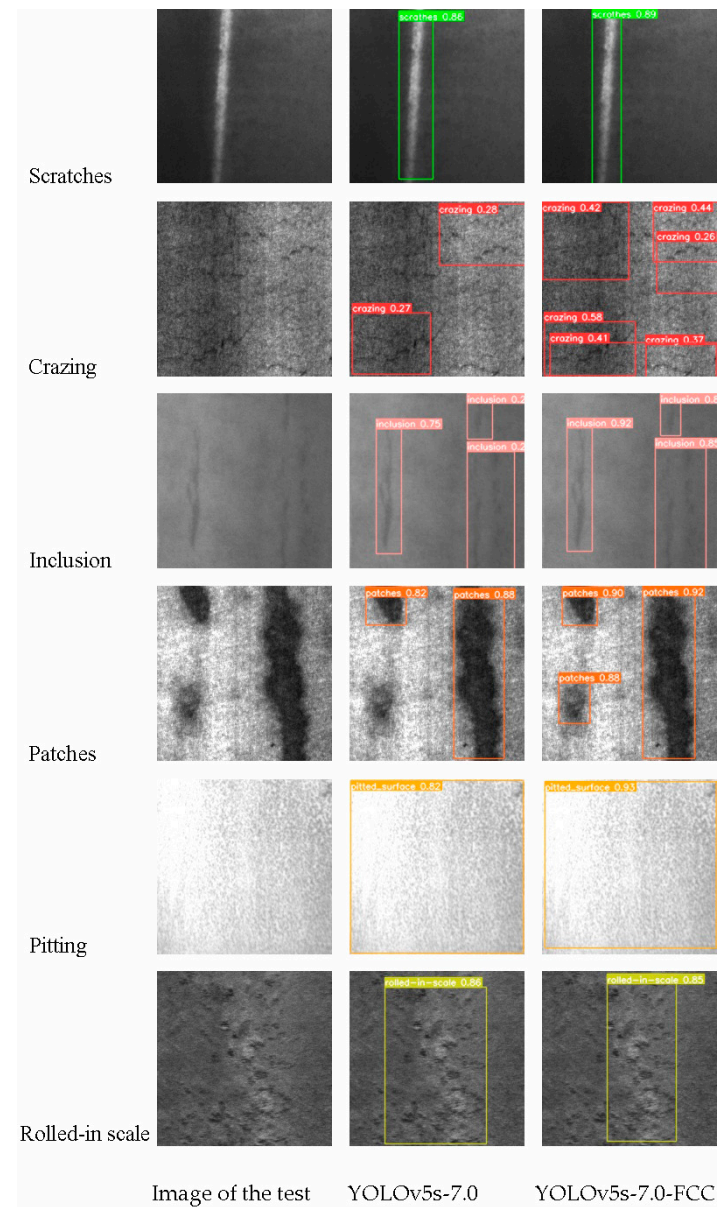


Figure 10. Comparison of YOLOv5s-7.0-FCC and YOLOv5s-7.0 detection results.

5. Conclusions

During the surface defect detection process in steel strips, it is often challenging to detect subtle features and small defect targets, resulting in a low detection accuracy. To address these issues effectively, this paper proposes an improved algorithm called YOLOv5s-7.0-FCC to meet the demands of engineering practices. YOLOv5s-7.0-FCC is an algorithm that enhances semantics and optimizes the feature extraction. It introduces a lightweight convolutional kernel operator, C3F, into the original algorithm to reduce the redundant computations. C3F focuses more on the center position, which is better suited for feature extraction. The algorithm enriches the contextual information by incorporating

a CAM module to enhance feature representation. Additionally, it integrates two CARAFE upsampling operators, experimenting with various upsampling kernels and receptive field combinations to increase the feature extraction capabilities. The experimental results demonstrate that YOLOv5s-7.0-FCC outperforms YOLOv5s-7.0 in terms of the detection speed, exhibiting a more reasonable structural distribution that is better suited for machine learning. Moreover, YOLOv5s-7.0-FCC achieves a 6.4% increase in the detection accuracy, effectively reducing false negatives and false positives in steel strip defect detection, resulting in a significant overall performance improvement.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/electronics12214422/s1>, Data File S1: Code for the YOLOv5s-7.0-FCC model.

Author Contributions: Conceptualization, C.W.; methodology, T.X.; software, H.J.; validation, Y.Q.; formal analysis, J.Y.; investigation, Y.K.; resources, X.C.; data curation, J.Y.; writing—original draft preparation, J.Y.; writing—review and editing, J.Y.; visualization, T.X.; supervision, Y.Q.; project administration, H.J.; funding acquisition, C.W. All authors have read and agreed to the published version of the manuscript.

Funding: The applicant for the fund is Haijuan Ju. This research was funded by Natural Science Basic Research Program of Shaanxi, China, the fund number 2023-JC-QN-0696.

Data Availability Statement: The data that support the findings of this research are openly available at https://download.csdn.net/download/qj_41264055/85490311 (accessed on 15 September 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, Z. Review of non-destructive testing methods for defect detection of ceramics. *Ceram. Int.* **2021**, *47*, 4389–4397. [[CrossRef](#)]
2. Jain, S.; Seth, G.; Paruthi, A.; Soni, U.; Kumar, G. Synthetic data augmentation for surface defect detection and classification using deep learning. *J. Intell. Manuf.* **2022**, *33*, 1007–1020. [[CrossRef](#)]
3. He, Y.; Song, K.; Meng, Q.; Yan, Y. An End-to-end Steel Surface Defect Detection Approach via Fusing Multiple Hierarchical Features. *IEEE Trans. Instrum. Meas.* **2019**, *69*, 1493–1504. [[CrossRef](#)]
4. He, D.; Xu, K.; Zhou, P. Defect detection of hot rolled steels with a new object detection framework called classification priority network. *Comput. Ind. Eng.* **2019**, *128*, 290–297. [[CrossRef](#)]
5. Zhang, M.; Yin, L. Solar Cell Surface Defect Detection Based on Improved YOLO v5. *IEEE Access* **2022**, *10*, 80804–80815. [[CrossRef](#)]
6. Li, X.; Wang, C.; Ju, H.; Li, Z. Surface defect detection model for aero-engine components based on improved YOLOv5. *Appl. Sci.* **2022**, *12*, 7235. [[CrossRef](#)]
7. Fu, G.Z.; Sun, P.Z.; Zhu, W.B.; Yang, J.X.; Cao, Y.L.; Yang, M.Y.; Cao, Y.P. A deep-learning-based approach for fast and robust steel surface defects classification. *Opt. Laser Eng.* **2019**, *121*, 397–405. [[CrossRef](#)]
8. Wang, Q.; Mao, J.; Zhai, X.; Gui, J.; Shen, W.; Liu, Y. Improvements of YoloV3 for road damage detection. *J. Phys. Conf. Ser.* **2021**, *1903*, 012008. [[CrossRef](#)]
9. Jiang, Z.; Zhao, L.; Li, S.; Jia, Y. Real-time object detection method based on improved YOLOv4-tiny. *arXiv* **2020**, arXiv:2011.04244.
10. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
11. Cai, X.; Zhou, S.; Cheng, P.; Feng, D.; Sun, H.; Ji, J. A Social Distance Monitoring Method Based on Improved YOLOv4 for Surveillance Videos. *Int. J. Pattern Recognit. Artif. Intell.* **2023**, *37*, 2354007. [[CrossRef](#)]
12. Qiu, M.; Huang, L.; Tang, B.H. ASFF-YOLOv5: Multielement Detection Method for Road Traffic in UAV Images Based on Multiscale Feature Fusion. *Remote Sens.* **2022**, *14*, 3498. [[CrossRef](#)]
13. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
14. Pinto, L.G.; Martins, W.M.; Ramos, A.C.; Pimenta, T.C. Analysis and Deployment of an OCR-SSD Deep Learning Technique for Real-Time Active Car Tracking and Positioning on a Quadrotor. In *Data Science: Theory, Algorithms, and Applications*; Springer: Singapore, 2021.
15. Berg, A.C.; Fu, C.Y.; Szegedy, C.; Anguelov, D.; Erhan, D.; Reed, S.; Liu, W. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016.
16. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]
17. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
18. Zheng, Z.; Hu, Y.; Zhang, Y.; Yang, H.; Qiao, Y.; Qu, Z.; Huang, Y. CASPPNet: A chained atrous spatial pyramid pooling network for steel defect detection. *Meas. Sci. Technol.* **2022**, *33*, 085403. [[CrossRef](#)]

19. Xue, N.; Niu, L.; Li, Z. Pedestrian Detection with modified R-FCN. In Proceedings of the UAE Graduate Students Research Conference 2021 (UAEGSRC'2021), Abu Dhabi, United Arab Emirates.
20. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C. GhostNet: More Features from Cheap Operations. *arXiv* **2019**, arXiv:1911.11907.
21. Zhang, Q.; Jiang, Z.; Lu, Q.; Han, J.N.; Zeng, Z.; Gao, S.H.; Men, A. Split to Be Slim: An Overlooked Redundancy in Vanilla Convolution. *arXiv* **2020**, arXiv:2006.12085.
22. Chen, J.; Kao, S.-h.; He, H.; Zhuo, W.; Wen, S.; Lee, C.-H.; Chan, S.-H.G. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 12021–12031.
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [[CrossRef](#)] [[PubMed](#)]
24. Loy, C.C.; Lin, D.; Wang, J.; Chen, K.; Xu, R.; Liu, Z. CARAFE: Content-Aware ReAssembly of FEatures. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
25. Cui, L.; Jiang, X.; Xu, M.; Li, W.; Lv, P.; Zhou, B. SDDNet: A fast and accurate network for surface defect detection. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–13. [[CrossRef](#)]
26. Zhu, X.; Hu, H.; Lin, S.; Dai, J. Deformable ConvNets v2: More Deformable, Better Results. *arXiv* **2018**, arXiv:1811.11168.
27. Qiao, S.; Chen, L.-C.; Yuille, A. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 10213–10224.
28. Gennari, M.; Fawcett, R.; Prisacariu, V.A. DSConv: Efficient Convolution Operator. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
29. Ding, X.; Zhang, X.; Han, J.; Ding, G. Diverse Branch Block: Building a Convolution as an Inception-like Unit. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.