Journal of Building Engineering 94 (2024) 109821

Contents lists available at ScienceDirect

Journal of Building Engineering

journal homepage: www.elsevier.com/locate/jobe

Lightweight network for millimeter-level concrete crack detection with dense feature connection and dual attention

Xiao Ma [a], Yang Li [a,*], Zijiang Yang [a], Shaoqi Li [b], Yancheng Li [c,**]

[a] Institute of Automation, Qilu University of Technology (Shandong Academy of Sciences), Jinan, Shandong, 250014, China
[b] Centre for Innovative Structures, College of Civil Engineering, Nanjing Tech University, Nanjing, 211816, China
[c] School of Civil and Environmental Engineering, University of Technology Sydney, Ultimo, NSW, 2007, Australia

ARTICLE INFO

Keywords:
Crack detection
Lightweight model
Efficient convolution
Dense feature enhancement connection

ABSTRACT

Development of lightweight deep learning crack detection method is essential for the future deployment of mobile device-based structure inspection. The primary challenge involves the analysis and extraction of features from narrow cracks, typically 3–6 pixels wide, which are often obscured by noise such as water stains and shadows. The lightweight model should also maintain high accuracy while ensuring low computational complexity and a minimal number of parameters. To this end, this paper proposes YOLO v5-DE (Dense Feature Enhancement Connection, Efficient and Fast Convolution), a lightweight network based on the YOLO v5 architecture tailored to address these challenges, and constructs crack datasets captured at different heights to investigate the impact of different shooting distances on network performance. The network utilizes efficient convolutions and dense feature connections, with strategic reuse of filtered features from shallow layers, to significantly enhance the model's fine-grained feature information and gradient flow. The experimental results demonstrate that YOLO v5-DE achieves a detection accuracy of 96% for cracks in concrete structures. Compared to the improved YOLO v5 with EfficientViT as the backbone network, YOLO v5-DE achieves 4.7% increase on accuracy while requiring fewer computational resources, with only 1.4 million parameters and 3.6 Giga Floating point Operations Per Second (GFLOPS). Additionally, YOLO v5-DE reduces the inference time to 3.38 ms and increases the frame rate to 295.8 FPS. Moreover, the proposed lightweight network exhibits better detection performance when facing complex backgrounds and real-world environments.

1. Introduction

The idea of using Unmanned Aerial Vehicle (UAV) for crack detection has brought great potential in revolutionizing current infrastructure inspection industry and transforming from labor-intensive to intelligent and automatize approach [1,2]. A central aspect of the systems is the development of highly efficient and accurate lightweight crack detection networks feasible to be implemented in mobile devices which was constrained by the capacity of the on-board graphical processing unit and quality of the images taken in-situ [3]. Deep learning crack detection algorithm employs multi-layer neural networks to discern complex patterns from extensive data [4], thereby automating abstract data feature extraction, and hence has the prospects in meeting above requirements by carefully tailoring

* Corresponding author.
** Corresponding author.
E-mail addresses: liyang@sdas.org (Y. Li), yancheng.li@uts.edu.au (Y. Li).

https://doi.org/10.1016/j.jobe.2024.109821
Received 19 March 2024; Received in revised form 27 May 2024; Accepted 3 June 2024
Available online 8 June 2024

and optimizing its architectures.

YOLO [5–8], as the most popular object detection network, revolutionizes object detection by framing it as a regression problem and processing the entire image through a single neural network [9]. This enables rapid determination of bounding box positions and their corresponding categories [10]. With its fast detection speed and mean Average Precision (mAP) twice that of other real-time networks [11], YOLO is widely acclaimed in computer vision and is extensively used in industries like manufacturing, animal husbandry, and transportation for tasks such as individual identification and classification [12].

However, deploying the YOLO network on mobile devices, especially platforms like drones, imposes strict demands on computational resources [13], especially for the scenario when real-time detection is required. In this scenario, the computational requirements of the YOLO network are likely to far exceed the performance limitations of mobile devices. Therefore, the YOLO model's size and complexity should be greatly reduced, otherwise it would impose limitations on its deployment in resource-constrained environments. Researchers have been exploring the integration of YOLO with lightweight neural networks (such as Shufflenet [14], Mobilenet [15], Ghostnet [16]) for lightweight deployment [17]. These lightweight networks typically exhibit parameter counts below 4 million, GFLOPS not exceeding 8.5, and achieve FPS rates surpassing 245. However, these lightweight models still have the following challenges: Firstly, limited computational capacity makes networks susceptible to overfitting; Secondly, despite maintaining small model sizes and fast inference speeds, they struggle to achieve high performance and accuracy; Thirdly, the reduction in computational load and complexity may result in a lack of feature representation capability within the networks. In addition, a significant challenge in crack detection is the prevalent issue of sample imbalance in crack images, where the number of background pixels is vastly outnumbered by crack pixels [18]. This imbalance substantially increases the difficulty for networks to extract effective features from these images and significantly hinders model convergence [19]. Moreover, the inevitable presence of various forms of noise in crack images adds to the complexity of model processing. Therefore, despite the obvious advantages of lightweight models in terms of processing speed and computational efficiency, they still face numerous challenges and limitations in the specific application of crack detection.

To address the challenges in lightweight crack detection, an improved lightweight crack detection network based on YOLO v5 is proposed. The major contributions of this research are as follows:

- A Feature Enhancement Connection method is innovatively proposed to enhance the contribution of feature information for better performance. This method, as an extension and optimization of the traditional Dense Connection approach [20], capitalizes the iterative filtering and utilization of abundant low-dimensional information, thereby significantly improving the effective use of limited feature information.
- Two novel convolution modules, the Efficient and Fast Convolution (EFConv) module and the Efficient Adaptive Fusion Convolution (EAFConv) module, are developed for efficient feature extraction. The EFConv module is dedicated to efficient feature extraction within a shortened inference time, while the EAFConv module focuses on the effective extraction and fusion of features from various branches.

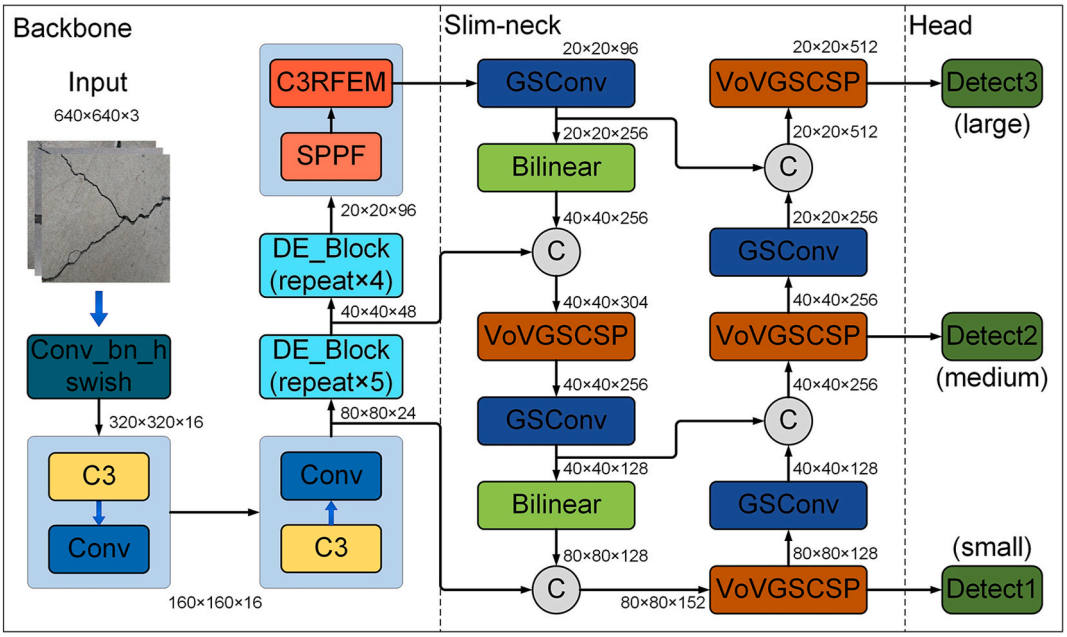


Fig. 1. Architecture of YOLO v5-DE network.

- A lightweight feature extraction module named DE is proposed to enhance feature reuse and facilitate gradient flow within the model. The network can minimize the gradient disappearance and network degradation and enhance the contribution of feature information.
- The YOLO v5-DE network is constructed, integrating the DE module and utilizing the scale-aware receptive field enhancement module [21] with multi-scale feature fusion capabilities, along with the Slim-Neck structure [22]. These designs collectively meet the requirements for lightweight deployment in concrete crack detection, ensuring rapid detection speed and high accuracy, with reduced model parameters and low computational complexity.

The structure of this article is organized as follows: Section II details the proposed connection method, convolution modules, and feature extraction modules, alongside the core architecture of YOLO v5-DE. Section III discusses the crack dataset, initialization of the algorithm, and evaluation metrics. Section IV provides a detailed analysis of the internal structure of the network. Section V introduces the experimental procedures, along with analysis and validation. Section VI provides a systematic summary and conclusion of the work conducted in this paper.

2. Proposed network

2.1. Overall network architecture

In this paper, a lightweight object detection network named YOLO v5-DE is proposed based on the YOLO v5 network architecture as detailed in Fig. 1. This network is designed to achieve high-precision detection while meeting the requirements for deployment on mobile devices. In Table 1, detailed explanations of the parameters for the modules utilized in the network are provided. This includes the input feature dimensions, convolutional kernel sizes, and stride sizes.

In crack images, the number of non-crack pixels far exceeds that of crack pixels, with a substantial amount of irrelevant background information impeding the accurate identification of cracks. Therefore, to preserve the critical crack feature is the key to accurately identify the structural defect. Here, innovative use of the Conv_bn_hswish module on the backbone network effectively preserves crucial features of cracks by integrating convolution, normalization, and activation functions. Then it is followed by low-level feature extraction through the C3 module and Conv module. In the DE module, emphasis is on the reusing of features in early layers. At the end of the backbone network, the C3RFEM (Receptive Field Enhancement Module) [13] is employed to fuse multi-scale feature information. Due to the extensive adoption of depth-wise separable convolutions within the GSConv [14] and VoVGSCSP [14] modules in the Slim-neck component, a minimal parameter count suffices for downsampling and shallow feature fusion operations on the feature information from the backbone section. Following this, Slim-neck processes deliver three feature maps of varying dimensions to the detector in the head component for prediction. With the increase of the receptive field as depth increases, the network extracts more global and higher-level semantic features, hence the feature maps fed into the detector possess a sufficient number of channels.

2.2. Lightweight feature extraction module

Due to the high proportion of background pixels in crack images, the effective extraction of features is can be challenging [23], resulting in difficulty in capturing sufficient crack-related features and subsequently compromise its predictive capability. To address this issue and better utilize feature information in images with imbalanced samples, a DE module is proposed, featuring two distinct structural designs with different stride lengths. As illustrated in Fig. 2(a), the configuration with a stride of 1 epitomizes the fundamental concept of the DE module. In this configuration, the inner branch is responsible for feature extraction. While the incorporation of an outer branch at earlier layers, along with the concatenation operation, facilitates the efficient reutilization of low-dimensional feature information, establishing direct interlayer connections and allowing early layers to receive direct supervision. This proposed DE utilizes both low-dimensional and high-dimensional features, overcoming the limitation of most convolutional neural networks, i.e., heavily relying on the highest-dimensional features. As illustrated in Fig. 2(b), for the structure with a stride of 2, an additional

Table 1
Detailed configuration table of each module in the YOLO v5-DE network.

Layers	Module/step	Input size	Kernel size	Layers	Module/step	Input size	Kernel size
1	Conv_bn_hswish/2	640 × 640 × 3	3 × 3 × 16	17	GSConv	20 × 20 × 96	1 × 1 × 256
2	C3/1	320 × 320 × 16	3 × 3 × 16	18	Upsample	20 × 20 × 256	–
3	Conv/2	320 × 320 × 16	3 × 3 × 16	19	Concat	–	–
4	C3/1	160 × 160 × 16	3 × 3 × 16	20	VoVGSCSP	40 × 40 × 304	1 × 1 × 256
5	Conv/2	160 × 160 × 24	3 × 3 × 24	21	GSConv	40 × 40 × 256	1 × 1 × 128
6	DE_Block/1	80 × 80 × 24	3 × 3 × 24	22	Upsample	40 × 40 × 128	–
7	DE_Block/2	80 × 80 × 24	5 × 5 × 40	23	Concat	–	–
8	DE_Block/1	40 × 40 × 40	5 × 5 × 40	24	VoVGSCSP	80 × 80 × 152	1 × 1 × 128
9	DE_Block/1	40 × 40 × 40	5 × 5 × 40	25	GSConv	80 × 80 × 128	3 × 3 × 128
10	DE_Block/1	40 × 40 × 40	5 × 5 × 48	26	Concat	–	–
11	DE_Block/1	40 × 40 × 48	5 × 5 × 48	27	VoVGSCSP	40 × 40 × 256	1 × 1 × 256
12	DE_Block/2	40 × 40 × 48	5 × 5 × 96	28	GSConv	40 × 40 × 256	3 × 3 × 256
13	DE_Block/1	20 × 20 × 96	5 × 5 × 96	29	Concat	–	–
14	DE_Block/1	20 × 20 × 96	5 × 5 × 96	30	VoVGSCSP	20 × 20 × 512	1 × 1 × 512
15	SPPF	20 × 20 × 96	–	31	Detect	–	–
16	C3RFEM	20 × 20 × 96	–	32			

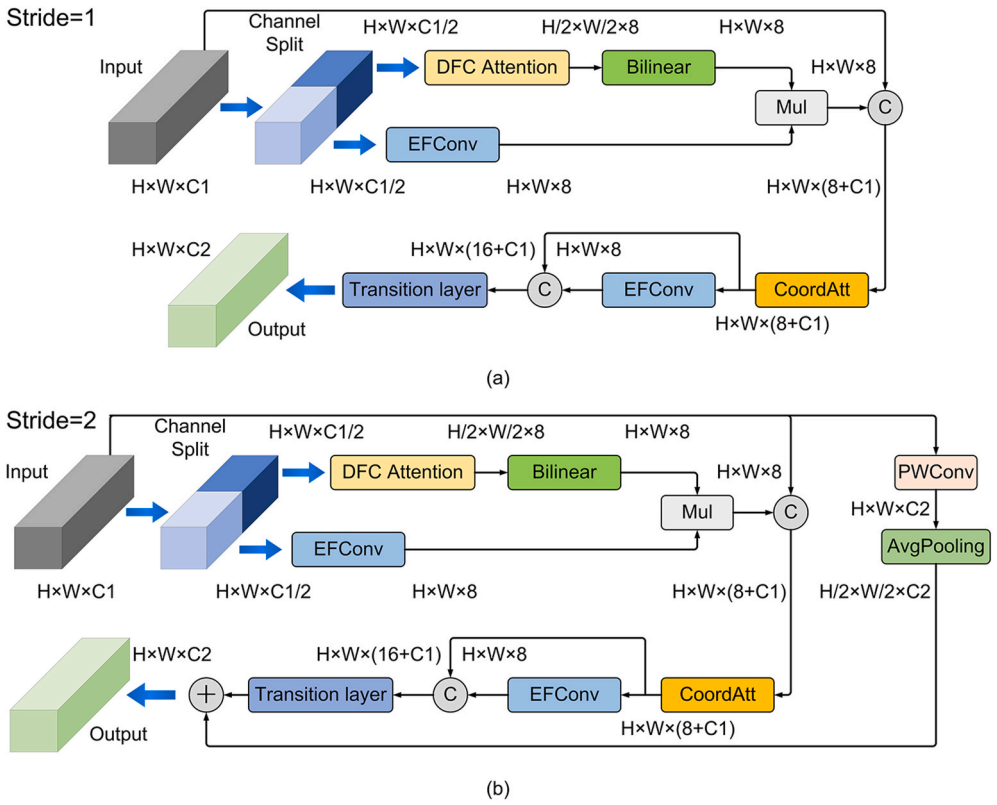


Fig. 2. Architecture of the DE module.

outermost shortcut connection [24] is proposed to mitigate the network degradation caused by increased layer depth and to incorporate low-dimensional feature information.

During the feature extraction phase, the input feature map is divided into two parts along the channel dimension through a Channel Split operation, and is connected with EFConv and DFC Attention [25] for respective processing. EFConv is tasked with extracting high-level feature information, while DFC Attention captures long-range spatial dependencies. The aggregated results of both are then concatenated with the input low-level features across channels, maximizing the utilization of feature information. However, the concatenation operation, while expanding the feature map to some extent, may also lead to partial loss of orientation and position information.

To enhance the model's perception of these two types of information, the feature map is fed into CoordAtt [26] to generate coordinate-aware feature maps. Subsequently, a reconcatenation of high-level and low-level features across channels is performed, and the output feature map size is adjusted by the Transition Layer [20], further optimizing the model capability in feature processing.

2.3. Dense feature enhancement connection

When integrating shallow-layer feature information in image fusion, it is inevitable to use convolution operations. However, this increases network parameters, and as the layers deepen, gradient vanishing issue is likely to occur. In order to mitigate gradient

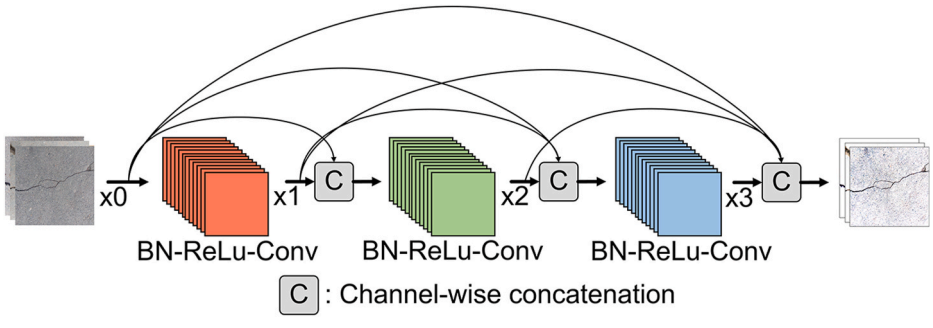


Fig. 3. Architecture of dense connection.

vanishing and fully reuse features, this paper proposes feature enhancement connections inspired by the dense connection structure [20]. As shown in Fig. 3, dense connections facilitate feature reuse by establishing cross-channel concatenation between all preceding and subsequent layers. While dense connections have the capability to mitigate gradient vanishing, the utilization of cross-channel concatenation operations ensures that the feature map sizes are consistent across different layers before concatenation. Through this mechanism, the outputs of preceding layers consistently filter out considerable irrelevant background information after passing through the attention mechanism. Moreover, the earlier layers undergo multiple rounds of attention mechanisms, thereby ensuring the retention and utilization of a significant amount of low-level information, continuously integrating with deeper-level information. To further conserve computational resources, YOLO v5-DE employs only one attention module in its feature enhancement connection.

From Fig. 4, it can be concluded that the input and output formulas are as follows:

$$X_i = h_i(X_0, X_1, \cdots, X_{i-1}), i \in 1, 2, 3, 4 \tag{1}$$

Where, $h_i, i \in 1, 2, 3, 4$ is a combination function, which generally includes BN (Batch Normalization) [27], ReLu [28], Pooling [29] and Convolution operations.

2.4. Adaptive fusion convolution

As known, reduction on the computational complexity of models often results in the loss of significant feature information during downsampling, which imposes challenge for lightweight networks to balance between computational resources and network performance. Here, a module combining the strengths of depthwise separable convolution [30] and the non-parametric attention mechanism SimAM [31] is proposed as EAFConv to address this issue.

As illustrated in Fig. 5, the input feature map initially undergoes Pointwise Convolution (PWConv [30]) for cross-channel information fusion, followed by Depthwise Convolution (DWConv [30]) to extract deep-level information. Subsequently, the convolved feature maps are concatenated for feature reuse. Following this, the SimAM attention mechanism directs the network focus towards crack regions in the image and facilitates the fusion of results from both branches. Finally, fine-grained information extraction is conducted through PWConv and DWConv. Given the substantial depth of the network, batch normalization operations are introduced to expedite network convergence. Finally, the number of output channels is adjusted through depthwise separable convolution, enhancing the overall performance of the model while maintaining computational efficiency.

Fig. 6 illustrates the processing procedures of these two types of convolutions. In terms of computational efficiency, the parameter count of depthwise separable convolutions, denoted as $K \times K \times C_1 + C_1 \times C_2$, is significantly reduced compared to the parameter count $K \times K \times C_1 \times C_2$ of standard convolutions, as the former first applies DWConv separately to each channel and then concatenates all convolution results through PWConv.

Moreover, in contrast to well-known attention mechanisms such as SE [32], CBAM [33], and GC [34], SimAM derives three-dimensional attention weights within the network layers without adding extra computational load, effectively optimizing the neurons within the network layers. This enhancement further improves the model accuracy in object detection and the model capability in feature processing.

2.5. Efficient and fast convolution

When deploying models on actual mobile devices, the importance of inference time cannot be overstated. While depthwise separable convolutions effectively reduce the model FLOPS, their utilization in standard convolutions leads to an increase in inference time. To effectively shorten the inference time on mobile devices, the EFConv convolutional module is proposed, as depicted in Fig. 7. This module adopts a streamlined and efficient design, comprising only three standard convolutions. Similar to the concept of EAFConv, EFConv initially conducts concatenation of deep-level and shallow-level feature maps using two standard convolutions. Here, the attention mechanism is removed from the EFConv due to the significant increase in inference time associated with excessive usage in the network. Instead, 1×1 standard convolutions are employed to achieve full connections between channels, reducing overall parameter and effectively integrating feature information.

3. Methodology

This section comprehensively describes three datasets utilized in our experiment: Crack Dataset 2218, Crack Dataset 10,000, and the Crack-Fly Dataset. Additionally, the section delineates the configuration of the experimental environment and the performance evaluation metrics adopted for the network.

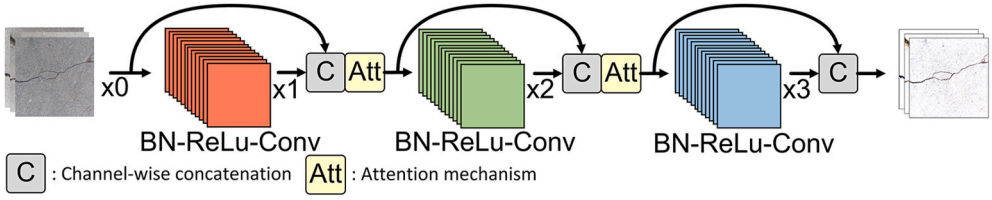


Fig. 4. Architecture of dense feature enhancement connection.

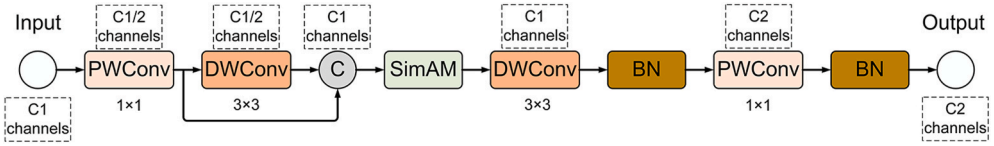


Fig. 5. Architecture of EAFConv module.

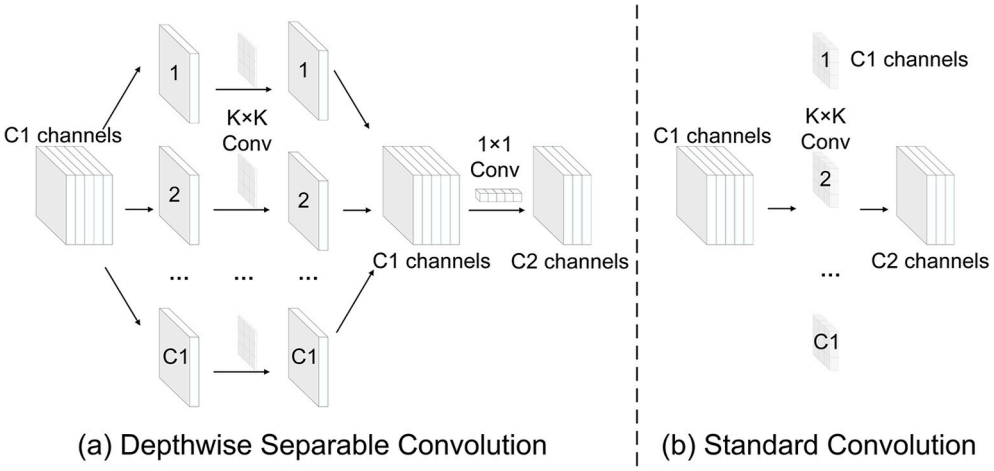


Fig. 6. Contrast between depthwise separable convolution and standard convolution.

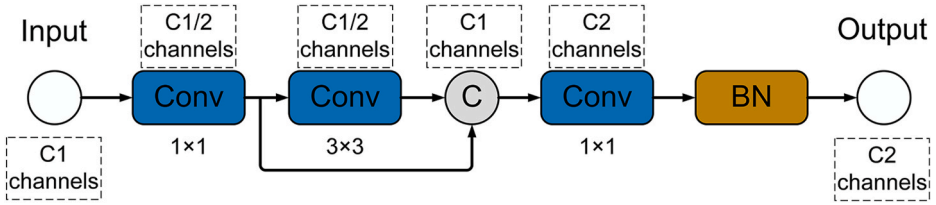


Fig. 7. Architecture of EFConv module.

3.1. Dataset description

3.1.1. Crack 2218 dataset

The Crack 2218 dataset encompasses a diverse range of concrete structures, including roads, bridges, and walls. Comprising a total of 2218 images, these photographs were captured on the Wohushan Reservoir Bridge in Jinan, Shandong Province using an iPhone 11 camera and a UAV (DJI Mavic 3). Additionally, it includes numerous noise-bearing images that reflect real-world environments, such as shadows, traffic markings, and expansion joints. The dataset is methodically segmented into training, validation, and test sets in a 7:2:1 ratio aiming to better investigate the detection performance of YOLO v5-DE in real-world scenarios. The dataset images have a resolution of 1024 × 1024 pixels and are systematically divided into training, validation, and test sets in a 7:2:1 ratio. The cracks included in the dataset have widths ranging from 1 mm to 4 mm.

3.1.2. Crack 10,000 dataset

The Crack 10,000 dataset include a selection of images from the publicly available CrackForest dataset along with approximately initial 2000 crack images captured using an iPhone 11 camera, including diverse scenes such as asphalt roads, concrete roads, and walls. The image size was standardized to 3042 × 4032. Through preprocessing techniques such as cropping, rotating, and flipping, the dataset was expanded to 10,000 images. These were subsequently divided into training and validation sets in a 3:1 ratio. Additionally, based on shape characteristics, cracks were classified into five categories: branch, diagonal, horizontal, vertical, and craze, with each category comprising 1500 samples. This was undertaken to assess the detection performance of YOLO v5-DE across different application environments and its ability to handle multi-class detection tasks. The cracks in the images have actual widths ranging from 1 mm to 4.6 mm, with which the goal is to evaluate the performance of YOLO v5-DE in handling multi-class detection tasks.

3.1.3. Crack-Fly dataset

This dataset contains images captured by UAV (DJI Mavic 3). To investigate the detection performance of YOLO v5-DE for

millimeter-level cracks in real world engineering applications, the lens aperture was set to 2.8f and the lens focal length was set to 35 mm. A total of 1210 images of cracks on concrete surfaces were captured by a UAV at different distances (120, 150, 180, 200, 220 and 250 cm) with a resolution of 5280 × 3956 pixels for training, validation, and testing purposes. The cracks captured in the images have true widths ranging from 0.8 mm to 4 mm. The intension of using this database is to study the relationship between the crack pixel width and different capture heights, thus enabling a better assessment of the network's performance.

3.2. Implementation details and network initialization

The YOLO v5-DE network is developed based on the YOLO v5 framework and implemented within the PyTorch framework. Training is conducted on a Windows 10 PC system, equipped with an Intel Xeon W-2255 CPU and an NVIDIA RTX 3090 24 GB GPU. Weight updates of the network are carried out using the SGD (Stochastic Gradient Descent) optimizer. CIoU [35] is used to evaluate the similarity between predicted and ground truth bounding boxes. The initial learning rate is set at 0.01, with an SGD momentum of 0.937 and a weight decay of 0.0005. Additionally, the network undergoes three warm-up rounds with an initial momentum of 0.8. Each training phase encompasses 200 epochs, with a batch size of 64 for each epoch.

3.3. Performance evaluation metrics

Key indicators for object detection in this paper include True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These indicators are critical for assessing the accuracy of the YOLO v5-DE network and other networks in crack detection tasks. TP and TN represent accurate predictions of positive and negative instances, respectively, while FP and FN denote inaccuracies in predicting such instances.

Precision, defined in Equation (2), quantifies the ratio of true positive instances among those identified as positive by the model. Recall, outlined in Equation (3), measures the proportion of actual positive instances correctly identified by the model. The F1 Score, representing the harmonic mean of precision and recall, is formally defined in Equation (4).

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\ Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{4}$$

mAP@.5 and mAP@.5:.95 are used to calculate the mean average precision at various Intersection over Union (IoU) thresholds, providing a detailed assessment of the model accuracy across different classes and scenarios. GFLOPS (Giga Floating point Operations Per Second) evaluates the computational complexity of the model, offering insight into its processing capabilities. Inference time is a measure of the duration required for the model to process input and generate output, crucial for real-time applications. FPS (Frames Per Second) serves as an indicator of the model processing speed, reflecting its efficiency in handling image frames.

4. Analysis of network internal structure

4.1. Growth rate optimization

In this section, the impact of growth rate on model performance metrics is explored. The growth rate refers to the number of additional channels in each layer of the DE module, including both the aggregated feature map channels and the output channels of the second EFConv. Specifically, if the number of channel of the input feature map is K, the number of channel for the i-th layer is denoted as $K + (i - 1)growth\_rate$. Optimizing the growth rate to maximize feature information extraction during channel transmission is pivotal.

As illustrated in Table 2, the experimental data reveals that setting the growth rate at 8 results in the highest performance in terms of mAP@.5, while setting the growth rate to 96 leads to the highest values in the mAP@0.5:0.95 metric. However, there exists a significant difference between these two settings: a disparity of 780,000 in the number of parameters and a gap of 3.1 GFLOPS. Additionally, as clearly demonstrated in Fig. 8, there is a noticeable discrepancy in network performance with different growth rates. It is evident that setting the growth rate to 8 results in comparatively ideal outcomes across the mAP@.5, mAP@0.5:0.95, and F1 Score metrics. Therefore, considering the requirements for lightweight model and computational efficiency, a growth rate of 8 is the more appropriate choice for the model.

Table 2
Comparative outcomes at varying growth rates.

Growth rate	mAP@.5	mAP@.5:.95	Parameters (million)	GFLOPS
8	0.96	0.719	1.4	3.6
16	0.925	0.622	1.45	3.9
32	0.952	0.711	1.56	4.1
64	0.945	0.694	1.84	5.2
96	0.956	0.733	2.18	6.7

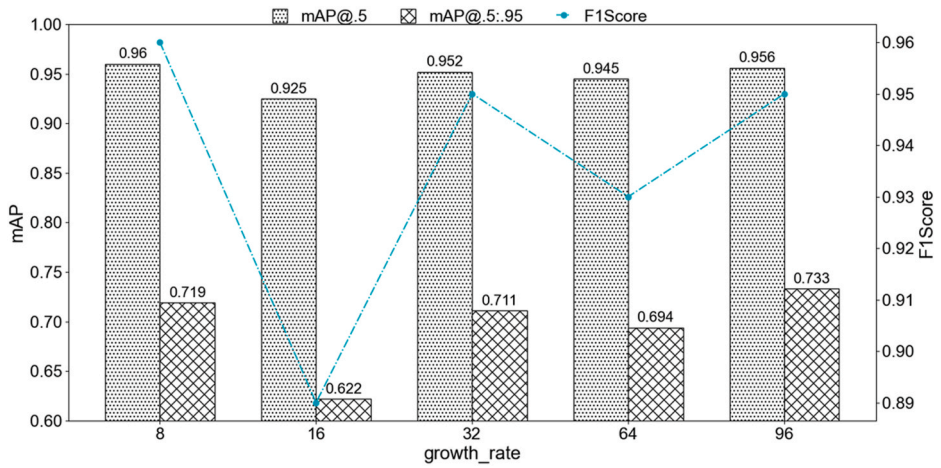


Fig. 8. Visual comparison under different growth rates.

4.2. Comparative analysis on convolution modules

In this section, the performance of convolution modules will be evaluated on the Crack 2218 dataset, including EAFConv and EFConv, as well as the newly introduced lightweight convolution module SCConv. As shown in Table 3, the network using SCConv exceeds that using EAFConv by 80,000 parameters and has a GFLOPS advantage of 0.2, failing to demonstrate lightweight advantages. In terms of model performance, networks using EFConv exhibit the best performance, especially in the mAP@.5:.95 metric, surpassing SCConv and EAFConv by 0.69 and 0.11, respectively. In terms of inference time and FPS, networks using EFConv also perform exceptionally well. Additionally, as depicted in Fig. 9, networks using EFConv have a larger area under the PR curve, indicating superior performance compared to networks using other convolutions. The experimental results indicate that, although SCConv minimizes feature redundancy through separation and reconstruction, the lightweight network requirement of setting the growth_rate to 8 limits its computational capabilities. This constraint hinders SCConv from effectively separating high-information feature maps from low-information feature maps when processing spatial and channel information, resulting in poorer performance. Additionally, while the depthwise separable convolution used in EAFConv is highly efficient, the increased memory access operations lead to slower inference speeds, which is a critical factor in evaluating performance metrics. Therefore, this study leans towards adopting networks using EFConv.

4.3. Comparative analysis on attention mechanisms

To delve into the advantages of various attention mechanisms, heat maps were utilized for intuitive demonstration of their effects, as depicted in Fig. 10. The DFC attention mechanism captures long-range dependencies in both vertical and horizontal directions through vertical and horizontal fully connected layers, thereby achieving precise localization of cracks. However, DFC attention did not adequately focus on the cracks as a whole, where circled in red. Subsequently, upon integration with the SE attention mechanism, which learns the importance of global channels to enhance useful information in feature maps, it can be observed from Fig. 10 that the attention area begins to converge, but there is still a lack of overall focus on the cracks. SE attention does not consider spatial correlations and can only enhance local crack information, as indicated in Fig. 10. In contrast, CoordAtt can capture long-range dependencies along one spatial direction and preserve precise position information along another spatial direction. This enables CoordAtt to form feature maps that are sensitive to both directional and positional crack features. As depicted in Fig. 10, the attention area significantly expands along the direction of the cracks, allowing the model to capture more relevant information. Therefore, the synergistic operation of DFC and CoordAtt attention mechanisms is more suitable for exerting the feature information filtering role within the DE module.

5. Results and discussion

5.1. Performance evaluation based on the Crack 2218 dataset

This section compares the performance of YOLO v5-DE with other lightweight networks and well-known CNN networks on the

Table 3
Model performance comparison on EFConv and EAFConv.

Networks	mAP@.5	mAP@.5:.95	Parameters (million)	GFLOPS	Inference time (ms)	FPS
v5-DE (SCConv)	0.954	0.65	1.35	3.5	4.24	226.2
v5-DE (EAFConv)	0.95	0.708	**1.27**	**3.3**	3.77	264.9
v5-DE (EFConv)	**0.96**	**0.719**	1.4	3.6	**3.38**	**295.8**

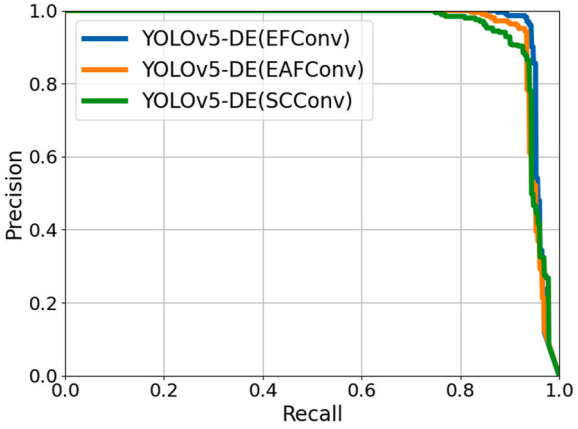


Fig. 9. Comparative analysis of Precision-Recall across different convolutions.

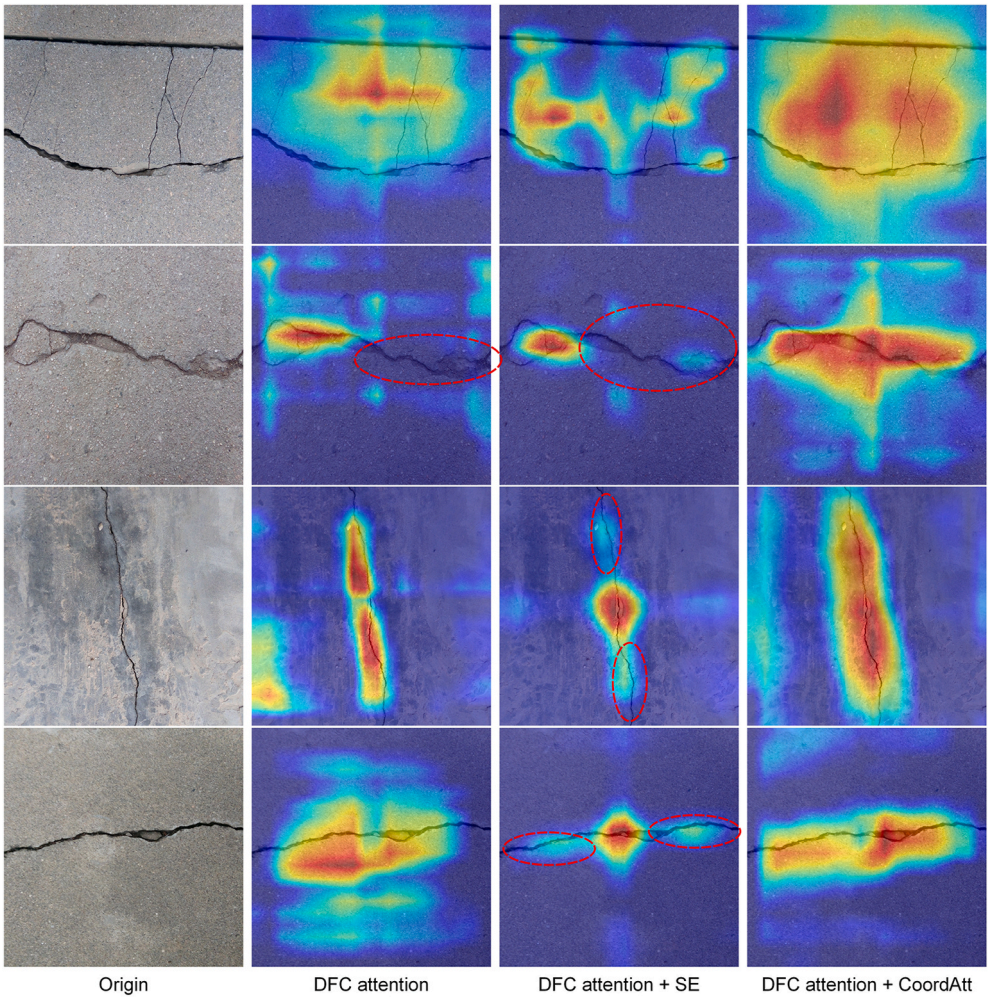


Fig. 10. Heatmap comparison of different attention mechanisms in YOLO v5-DE.

Crack 2218 dataset, including EfficientDet-D0 [36], CenterNet [37] (ResNet50 [24]), FasterNet, EfficientViT, GhostNet, YOLO v5s, and YOLO v7tiny. As shown in Table 4, YOLO v5-DE demonstrates an optimal balance between performance and computational cost. Among the eight networks, it ranks first in the mAP@.5 metric and closely follows YOLO v5s in the mAP@.5:.95 metric. In term of

Table 4
Performances of various lightweight network on Crack 2218 dataset (Bold number indicating best performance).

Model	mAP@.5	mAP@.5:.95	GFLOPS	Parameters (million)	Weight (M)
EfficientDet-D0	0.328	/	5.2	3.87	13.81
CenterNet (ResNet50)	0.732	/	70.2	32.65	108.72
YOLO v5-FasterNet	0.891	0.57	7.1	3.18	6.8
YOLO v5-EfficientViT	0.913	0.535	5.9	3.02	6.9
YOLO v5-GhostNet	0.952	**0.731**	8.3	3.69	7.9
YOLO v5s	0.955	0.72	16.0	7.22	14.1
YOLO v7tiny	0.946	0.682	13.0	6.0	12.0
Ours	**0.96**	0.719	**3.6**	**1.4**	**3.3**

computational cost, it has lowest number of parameters, GFLOPS and weight, i.e., 1.4 million, 3.6 and 3.3 M, respectively. This balance is attributed to the lightweight architecture proposed in this paper and the feature enhancement processing adopted for crack detection targets.

Figs. 11 and 12 present a more intuitive performance comparison of several networks. In both figures, YOLO v5-DE is positioned in the top-left corner, indicating its superior performance in key metrics such as mAP@.5 and F1 Score, while maintaining the lowest parameter count and computational complexity. In the task of crack detection, compared to mainstream lightweight CNN networks such as GhostNet and lightweight networks with transformer such as EfficientViT, YOLO v5-DE not only further compresses model size but also exhibits excellent performance.

Fig. 13 shows comparison of the selected networks on complex images, such as those with various disturbances. When dealing with images featuring shadow cover and wooden stick interference, only YOLO v5-DE is capable of accurately identifying the cracks within. Although CenterNet, FasterNet, and YOLO v5s can produce detection results with low confidence level and is unable to clearly define the edges of cracks. In images with handwritten mark (upper part of the image), YOLO v5-DE also performs well with the highest confidence level among the eight networks. Conversely, lightweight networks like FasterNet and GhostNet exhibited misclassification issues. This is because GhostNet's overall design is based on the assumption of feature information smoothness, which leads to information loss when dealing with disturbance possessing pixel resemblance to the crack. For FasterNet, the issue lies in its proposed partial convolution, which only extracts features from a subset of channels, resulting in insufficient learning capacity and misclassification problems. Furthermore, experiments conducted on crack images with poor lighting highlight a common challenge among most networks: unclear crack boundary detection. This issue stems from the similarity in features between background pixels in shaded environments and crack pixels. Notably, the predicted box of YOLO v5-DE conforms most closely to the edges of cracks compared to those of several other networks.

Overall, in terms of actual crack detection performance, YOLO v5-DE exhibits stability that other lightweight networks are lack of, demonstrating good resistance to environmental noise interference. This can be attributed to the ability of DE module in enhancing the contribution of low-dimensional feature information in the network, as well as the effective fusion of multi-scale fine-grained information by the C3RFEM module. These results also confirm that YOLO v5-DE meets the performance requirements of lightweight target detection networks.

5.2. Performance evaluation based on the Crack 10,000 dataset

In this section, the experimental scope has been expanded to include multi-category detection and application in asphalt pavement

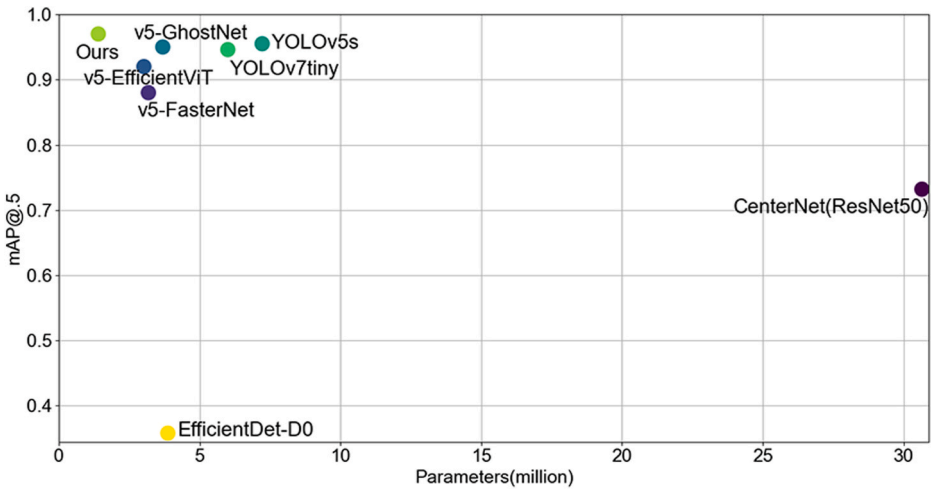


Fig. 11. Performances of different lightweight networks on mAP@.5 and parameters.

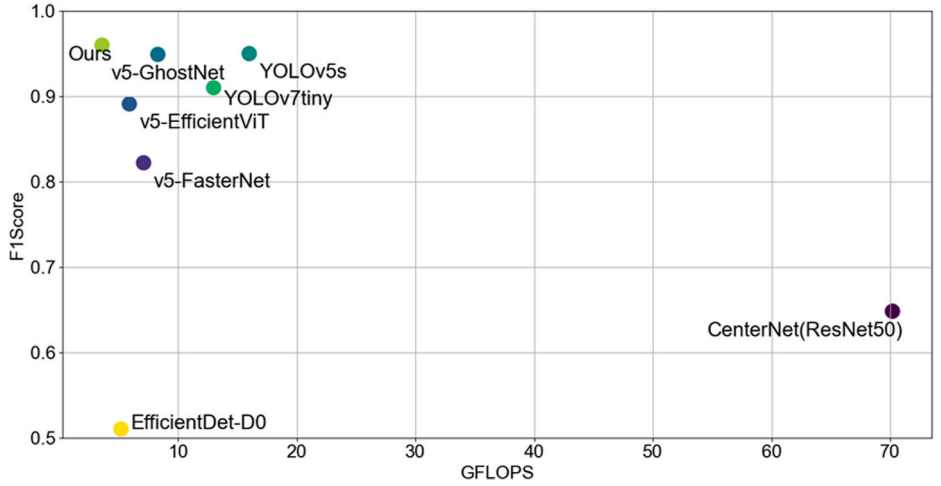


Fig. 12. Performances of different lightweight networks on F1 score and GFLOPS.

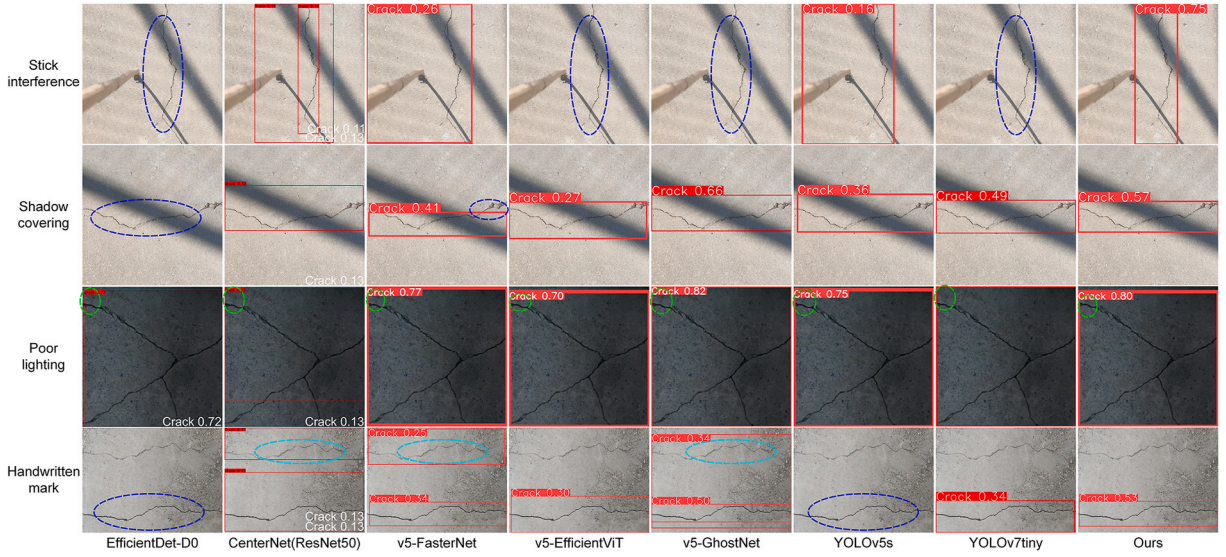


Fig. 13. Performance comparison of different networks in detecting representative cracks (misdetected objects are indicated by light blue circles, undetected objects are denoted by dark blue circles, and incomplete detections are represented by green circles).

surfaces. Experiments were conducted using the Crack 10,000 dataset to evaluate the performance of chosen networks. The comprehensive results of these evaluations are presented in Table 5. Relative to the baseline network YOLO v5s, the YOLO v5-DE, configured with a growth rate of 64, demonstrates an increase of 1.3% in mAP@.5 and 4.2% in mAP@.5:.95, respectively. The proposed network also has the least GFLOPS, number of parameters and model weights.

Fig. 14 compares the performances of various lightweight networks using the Crack 10,000 database. In the Crack 10,000 dataset,

Table 5
Performances of various lightweight network on Crack 10,000 dataset (growth_rate = 64).

Model	mAP@.5	mAP@.5:.95	GFLOPS	Parameters (million)	Weight (M)
EfficientDet-D0	0.549	/	5.2	3.87	13.81
CenterNet (ResNet50)	0.795	/	70.2	32.65	108.72
YOLO v5-FasterNet	0.752	0.495	7.1	3.18	6.8
YOLO v5-EfficientViT	0.762	0.528	5.9	3.02	6.9
YOLO v5-GhostNet	0.792	0.607	8.3	3.69	7.9
YOLO v5s	0.789	0.573	16.0	7.22	14.1
Ours	**0.802**	**0.615**	**5.2**	**1.85**	**4.2**

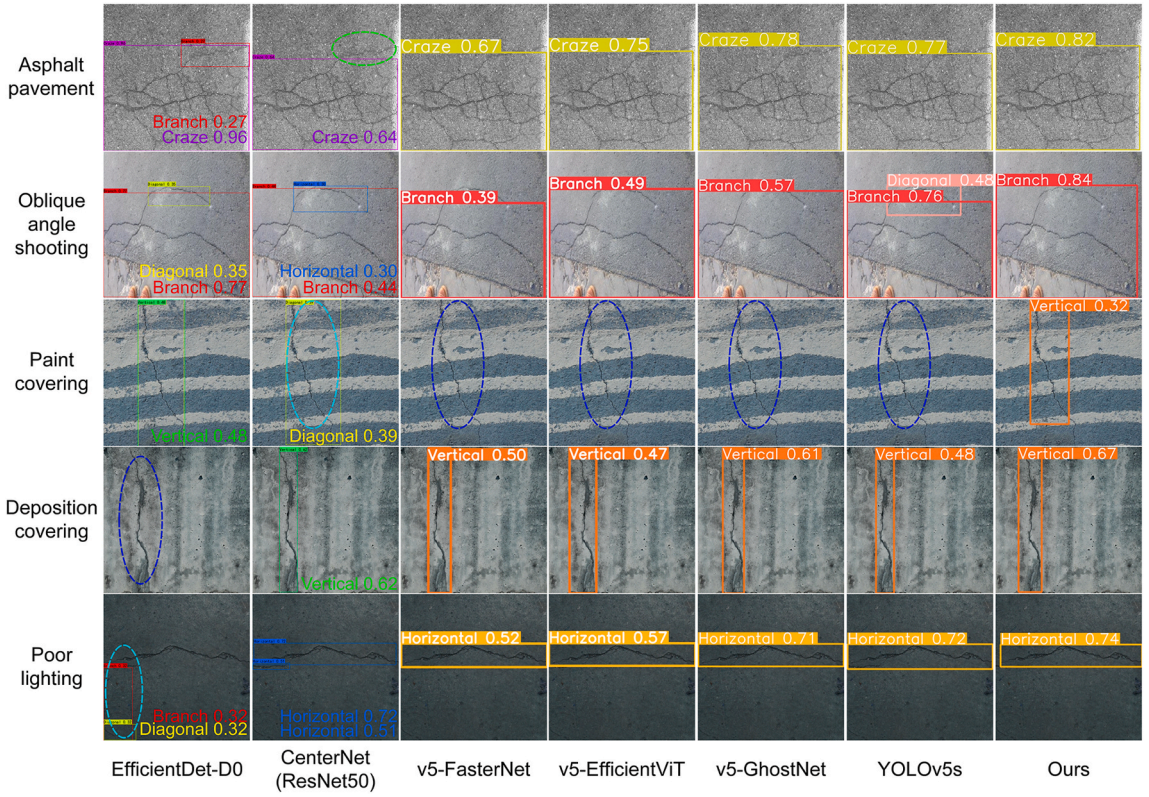


Fig. 14. Networks performance comparison in multi-class crack detection (misclassifications are indicated by light blue circles, undetected objects are denoted by dark blue circles, and incomplete detections are represented by green circles).

images captured from an oblique angle, images with paint cover, images with sediment interference and images with poor lighting are selected to examine the network performance. When detecting images captured from an oblique angle, EfficientDet, CenterNet, and YOLO v5s mistakenly identified two cracks due to the loss of local pixel information caused by the tilted angle. In contrast, only YOLO v5-DE demonstrated ideal recognition performance under the mentioned types of noise, while other networks encountered issues such as misclassification and failure. When faced with sediment interference, all networks except EfficientDet exhibited good resistance to

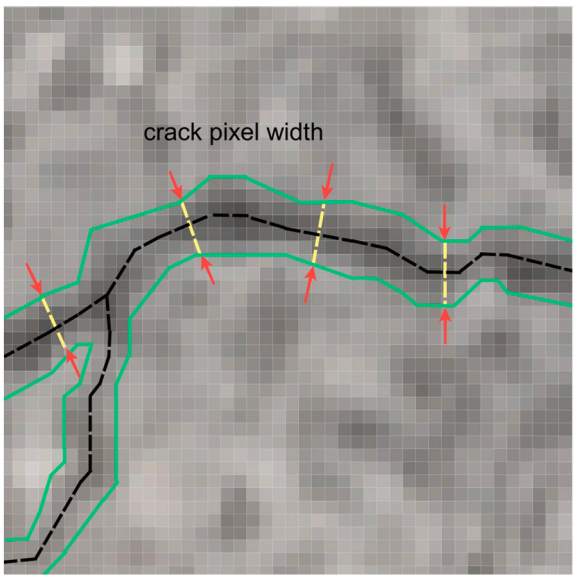


Fig. 15. Crack width measurement schematic.

interference.It is worth noting that EfficientDet frequently misclassified during the experiment, which can be attributed to the computational limitations of its D0 version. This limitation is due to insufficient extraction of multi-scale information before inputting into the weighted bidirectional feature pyramid network, resulting in inadequate network feature representation. With less computational requirement, YOLO v5-DE mitigates these shortcomings by employing the DE module for global low-dimensional information supplementation. Additionally, it conducts multi-scale information fusion and channel shuffle operations, partially compensating for computational deficiencies. The experimental results demonstrate that YOLO v5-DE exhibits excellent detection performance and outperforms other networks when conducting classification tasks in complex scenes with various types of noise.

5.3. Millimeter-level crack detection based on the Crack-Fly dataset

In this section, we investigated the impact of different shooting distances on network performance using the Crack-Fly dataset, which contains crack images captured at various distances. Specifically, we randomly selected ten images as testing set taken at distances of 120 cm, 180 cm, and 220 cm to represent the real-world scenarios of low, medium, and high shooting heights. The remaining images were divided into training and validation sets in a 9:1 ratio. Additionally, transfer learning was employed to fine-tune YOLO v5-DE, pre-training the model on the Crack 2218 dataset and further refining it on the Crack-Fly dataset.

To accurately assess the performance of the network at each shooting distance, separate experiments were conducted for the three distances, with crack widths ranging from 0.8 to 4 mm. The original high resolution images are adjusted to be 640 × 640 pixels considering the computational resource constraints in practice and the requirement for inference speed. However, this adjustment undoubtedly imposes great challenge to the network to delineate crack boundaries, as the compressed images contain blurriness and have less proportion of crack pixels.

As shown in Fig. 15, the pixel width of cracks in the image was measured using the orthogonal skeleton method. The solid green line represents the edge of the crack, the black dashed line indicates the skeleton of the crack, and the yellow dashed line represents the width of the crack. According to the measurement results, at a height of 1.2 m, a crack with a real-world width of 1 mm corresponds to a width of 2.6 pixels in an image with a resolution of 5280 × 3956 pixels. At a height of 1.8 m, this relationship is 1 mm = 2.1 pixel, while at a height of 2.2 m, it is only 1 mm = 1.7 pixel.

Fig. 16 shows the detection results for images taken at distances of 1.2 m, 1.8 m and 2.2 m, with crack widths ranging from 0.8 mm to 3 mm. Despite the resolution of 5280 × 3956 pixels and the gradual increase in distance, it is still difficult to observe the cracks with the naked eye. At a distance of 1.2 m, the network can accurately identify and delineate crack boundaries. When the distance extends to 1.8 m, although there is a decrease in confidence, the network still demonstrates satisfactory detection performance. However, at a distance of 2.2 m, there is a slight error in defining crack boundaries, yet YOLO v5-DE can still accurately identify the crack locations even under shadow interference. It can be seen that although the confidence for each crack decreases with increasing distance, the network maintains good detection performance within a shooting distance of 2.2 m and can cope with environmental noise interference.

Fig. 17 illustrates the evolve of mAP@.5 metric along epoches during training for YOLO v5s, YOLO v7tiny, and YOLO v5-DE at three different shooting distances. It is evident that regardless of the shooting distance, the YOLO v5s network exhibited the best training results, with the fastest convergence speed among the three networks. However, as indicated in Table 6, during testing, at shooting distances of 1.2 m and 1.8 m, YOLO v5s has significant overfitting, where the mAP@.5 metric during testing was lower than that during training. For instance, at a shooting distance of 1.2 m, the mAP@.5 during training could reach 0.992, whereas during testing, it was only 0.866. This phenomenon was also observed in the YOLO v7tiny network. On the other hand, due to limited computational resources and low-resolution images, feature extraction was more challenging for YOLO v5-DE, as reflected in the test results. Although the mAP@.5 metric reaches approximately 0.98 at distances of 1.2 m and 1.8 m, the gradual reduction in extractable crack pixels as the shooting distance increases to 2.2 m exposes the disadvantage of YOLO v5-DE in feature representation. Its

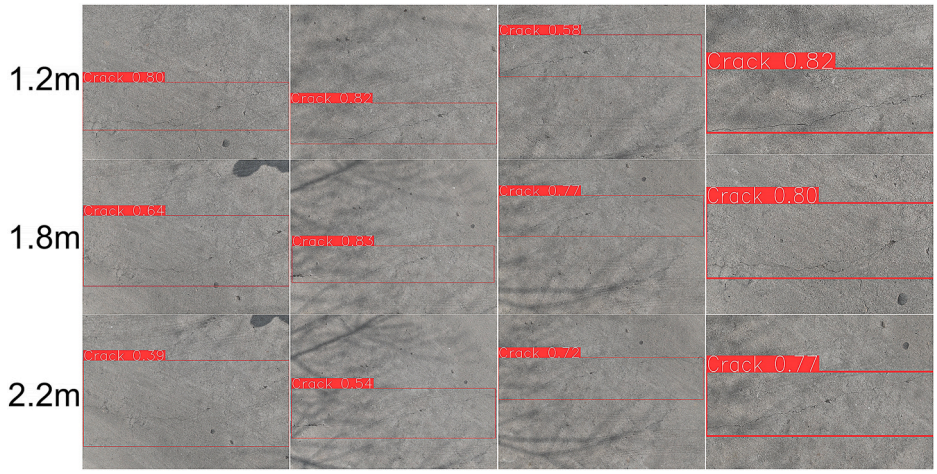
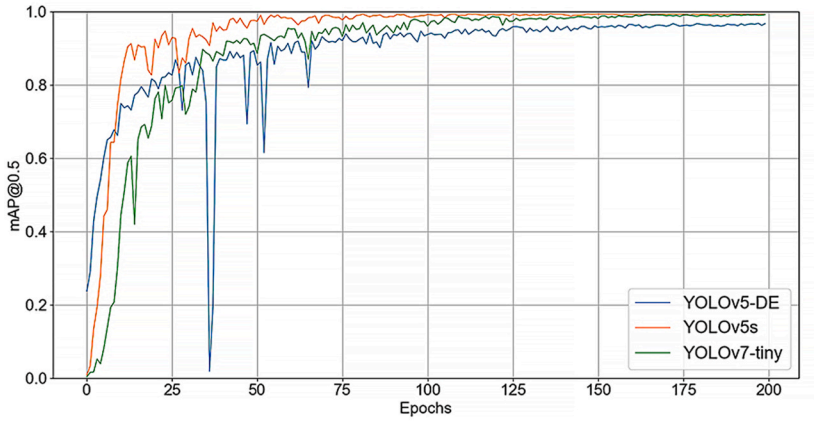
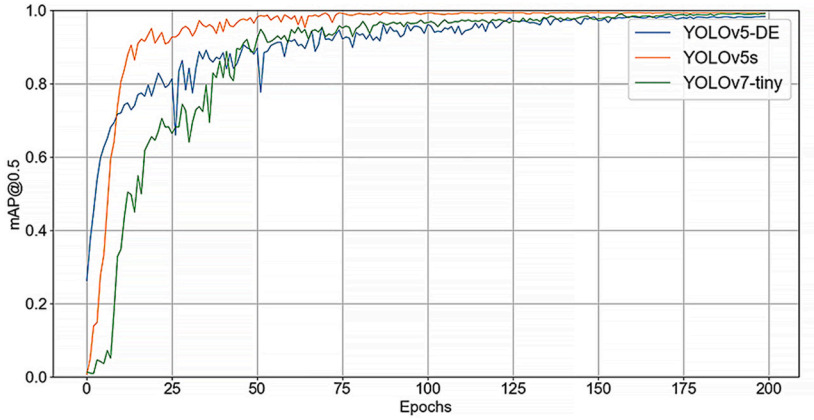


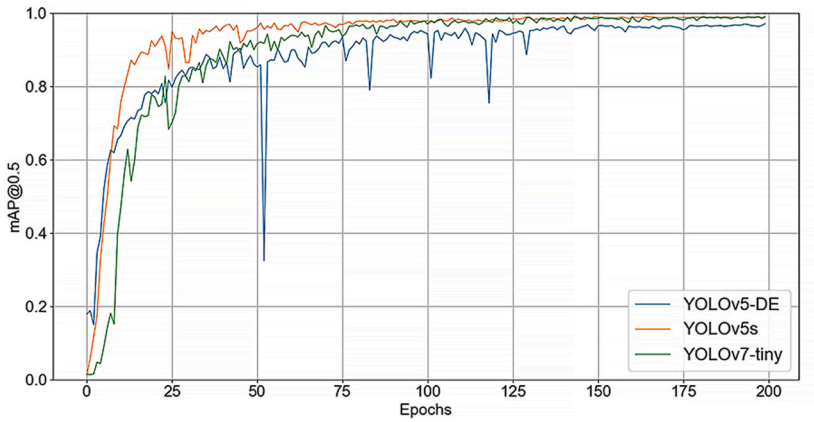
Fig. 16. Detection images captured at different shooting distances.



(a) Distance=1.2m



(b) Distance=1.8m



(c) Distance=2.2m

Fig. 17. Performance comparison of different networks on the Crack-Fly dataset.

Table 6
Performance comparison of different networks on crack image detection at different shooting distances.

Network	mAP@.5 at different shooting distance		
	1.2 m	1.8 m	2.2 m
YOLO v5s	0.866	0.921	0.989
YOLO V7tiny	0.898	0.935	0.997
YOLO v5-DE	0.978	0.989	0.966

Table 7
Ablation study results for DE, CA, RFEM, and Slim-Neck in crack detection.

Model	mAP@.5	Weight (M)	Parameter (million)	GFLOPS	Inference time (ms)	FPS
v5s	0.955	14.1	7.22	16.0	4.3	232.4
v5s + DE	0.946	7.4	3.37	8.7	4.35	230.0
v5s + DE (+CA)	0.954	7.5	3.39	8.7	4.56	219.3
v5s + DE (+CA) + Slim-Neck	0.951	3.3	1.37	3.6	3.39	296.0
v5s + DE (+CA) + Slim-Neck + RFEM	0.96	3.3	1.4	3.6	3.38	295.8

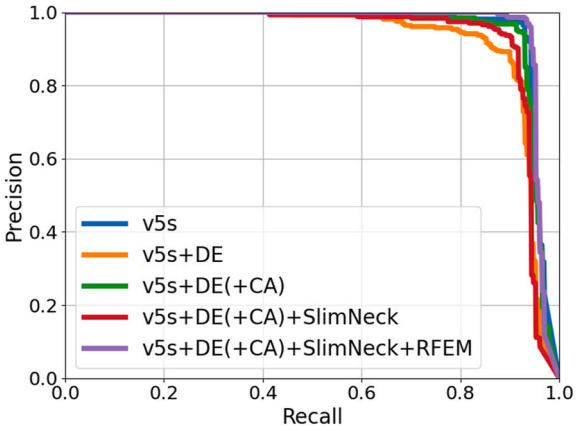


Fig. 18. Comparative analysis of Precision-Recall across different stages.

performance begins to fall below that of YOLO v5s and YOLO v7tiny as the distance increases.

Overall, while increasing shooting distances gradually decrease, YOLO v5-DE as a lightweight network, still demonstrates excellent performance in crack detection tasks and is applicable for structural inspections in practical scenarios.

5.4. Ablation study

To evaluate the effectiveness of the proposed network in crack detection, ablation studies were conducted on the DE module, CA (Coordinate attention), RFEM module, and the Slim-neck structure, shown in Table 7. The DE module, featuring dense feature enhancement connections, significantly reduces the weight size by 46.9%, parameter count by 53.1%, and computational complexity by 45.7% of the baseline network, while maintaining nearly the same level of accuracy. This robustly validates the superior performance of the DE feature extraction module proposed in this paper. Furthermore, the synergistic operation of these modules has led to notable improvements in inference time and FPS. Additionally, as depicted in Fig. 18, the PR (Precision-Recall) curve of YOLO v5-DE essentially overlaps with that of YOLO v5s throughout most of the curve. However, towards the end of the curve, as the threshold values increase, the confidence cannot reach the threshold, resulting in YOLO v5-DE trailing slightly behind YOLO v5s. This is due to the limited computational capacity of YOLO v5-DE compared to YOLO v5s, resulting in slight deficiencies in feature extraction for YOLO v5-DE and consequently slightly lower confidence. Nevertheless, the overall performance of the network still demonstrates a balance between lightweight design and network performance.

6. Conclusion

In this paper, a lightweight crack detection network for concrete surfaces, YOLO v5-DE, is proposed to address the challenges encountered in practical crack detection. YOLO v5-DE enhances the feature information contribution in the model by strategically reusing low-dimensional information after multiple filtrations. The efficient EFConv convolutional module, proposed in this work, not only fully extracts features but also shortens the inference time, achieving a detection accuracy of 96%, a frame rate of 295.8 FPS, and an inference time of 3.38 ms with only 1.4 million parameters, thus offers an optimal balance between lightweight design and

performance. To further evaluate the proposed model, experiments were conducted using the Crack 10,000 dataset, which includes multiple categories of cracks and asphalt pavement scenarios. The results demonstrate that YOLO v5-DE performed exceptionally well in comparative experiments, achieving an accuracy of 80.2%. Additionally, this paper delves into the impact of images taken at different shooting distances on network performance. The experiments show that with millimeter-level crack images captured at a height of 2.2 m, YOLO v5-DE maintains a precision of 96.6%. Overall, YOLO v5-DE demonstrates excellent performance in terms of detection accuracy and speed. Its lightweight feature, robustness and excellent performance place the network as a great candidate for deployment on mobile devices.

Funding

This research was supported by the Research and Demonstration Application of Key Technologies and Equipment Development for Discrete Manufacturing Industry (No. 2022JBZ02-02).

CRediT authorship contribution statement

Xiao Ma: Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation. **Yang Li:** Writing – original draft, Supervision, Project administration, Investigation, Funding acquisition. **Zijiang Yang:** Visualization, Validation, Software, Formal analysis. **Shaoqi Li:** Writing – review & editing, Visualization, Validation, Methodology. **Yancheng Li:** Writing – review & editing, Validation, Supervision, Project administration, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Z. Ding, Y. Yu, D. Tan, et al., Adaptive vision feature extractions and reinforced learning-assisted evolution for structural condition assessment, Struct. Multidiscip. Optim. 66 (2023) 209, https://doi.org/10.1007/s00158-023-03668-9.
[2] Q. Han, X. Liu, J. Xu, Detection and location of steel structure surface cracks based on unmanned aerial vehicle images, J. Build. Eng. 50 (2022) 104098, https://doi.org/10.1016/j.jobe.2022.104098.
[3] J. Hang, Y. Wu, Y. Li, T. Lai, J. Zhang, Y. Li, A deep learning semantic segmentation network with attention mechanism for concrete crack detection, Struct. Health Monit. 22 (5) (2023) 3006–3026, https://doi.org/10.1177/14759217221126170.
[4] K. Chen, G. Reichard, X. Xu, A. Akanmu, Automated crack segmentation in close-range building façade inspection images using deep learning techniques, J. Build. Eng. 43 (2021) 102913, https://doi.org/10.1016/j.jobe.2021.102913.
[5] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only Look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Jun. 2016, pp. 779–788, https://doi.org/10.48550/arXiv.1506.02640.
[6] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7263–7271, https://doi.org/10.48550/arXiv.1612.08242.
[7] J. Redmon, A. Farhadi, Yolov3: an incremental improvement, arXiv preprint arXiv:1804.02767, https://doi.org/10.48550/arXiv.1804.02767, 2018.
[8] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934, https://doi.org/10.48550/arXiv.2004.10934, 2020.
[9] S. Duan, et al., Tunnel lining crack detection model based on improved YOLOv5, Tunn. Undergr. Space Technol. 147 (May 2024) 105713, https://doi.org/10.1016/j.tust.2024.105713.
[10] G. Ye, J. Qu, J. Tao, W. Dai, Y. Mao, Q. Jin, Autonomous surface crack identification of concrete structures based on the YOLOv7 algorithm, J. Build. Eng. 73 (2023) 106688, https://doi.org/10.1016/j.jobe.2023.106688.
[11] J. Deng, Y. Lu, V.C.-S. Lee, Imaging-based crack detection on concrete surfaces using You Only Look once network, Struct. Health Monit. 20 (2) (Mar. 2021) 484–499, https://doi.org/10.1177/1475921720938486.
[12] J. Deng, Y. Lu, V.C. Lee, Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network, Comput. Aided Civ. Infrastruct. Eng. 35 (4) (Apr. 2020) 373–388, https://doi.org/10.1111/mice.12497.
[13] S. Katsigiannis, S. Seyedzadeh, A. Agapiou, N. Ramzan, Deep learning for crack detection on masonry façades using limited data and transfer learning, J. Build. Eng. (2023) 107105, https://doi.org/10.1016/j.jobe.2023.107105.
[14] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: an extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856, https://doi.org/10.48550/arXiv.1707.01083.
[15] A.G. Howard, et al., Mobilenets: efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017), https://doi.org/10.48550/arXiv.1704.04861.
[16] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, Ghostnet: more features from cheap operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1580–1589, https://doi.org/10.48550/arXiv.1911.11907.
[17] H. Hu, Z. Li, Z. He, L. Wang, S. Cao, W. Du, Road surface crack detection method based on improved YOLOv5 and vehicle-mounted images, Measurement 229 (Apr. 2024) 114443, https://doi.org/10.1016/j.measurement.2024.114443.
[18] Y. Yu, B. Samali, M. Rashidi, M. Mohammadi, T.N. Nguyen, G. Zhang, Vision-based concrete crack detection using a hybrid framework considering noise effect, J. Build. Eng. 61 (2022) 105246, https://doi.org/10.1016/j.jobe.2022.105246.
[19] Y. Wu, S. Li, J. Zhang, Y. Li, Y. Li, Y. Zhang, Dual attention transformer network for pixel-level concrete crack segmentation considering camera placement, Autom. ConStruct. 157 (2024) 105166, https://doi.org/10.1016/j.autcon.2023.105166.
[20] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708, https://doi.org/10.48550/arXiv.1608.06993.
[21] Z. Yu, H. Huang, W. Chen, Y. Su, Y. Liu, X. Wang, Yolo-facev2: a scale and occlusion aware face detector, arXiv preprint arXiv:2208.02019 (2022), https://doi.org/10.48550/arXiv.2208.02019.

[22] H. Li, J. Li, H. Wei, Z. Liu, Z. Zhan, Q. Ren, Slim-neck by GSConv: a better design paradigm of detector architectures for autonomous vehicles, arXiv preprint arXiv:2206.02424 (2022), https://doi.org/10.48550/arXiv.2206.02424.
[23] W. Wang, C. Su, G. Han, H. Zhang, A lightweight crack segmentation network based on knowledge distillation, J. Build. Eng. 76 (2023) 107200, https://doi.org/10.1016/j.jobe.2023.107200.
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778, https://doi.org/10.48550/arXiv.1512.03385.
[25] Y. Tang, K. Han, J. Guo, C. Xu, C. Xu, Y. Wang, GhostNetv2: enhance cheap operation with long-range attention, Adv. Neural Inf. Process. Syst. 35 (2022) 9969–9982, https://doi.org/10.48550/arXiv.2211.12905.
[26] Q. Hou, D. Zhou, J. Feng, Coordinate attention for efficient mobile network design. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Mar. 2021, pp. 13713–13722, https://doi.org/10.48550/arXiv.2103.02907.
[27] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, pmlr, 2015, pp. 448–456, https://doi.org/10.48550/arXiv.1502.03167.
[28] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2011, pp. 315–323.
[29] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (May 2017) 84–90, https://doi.org/10.1145/3065386.
[30] F. Chollet, Xception: deep learning with depthwise separable convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258, https://doi.org/10.48550/arXiv.1610.02357.
[31] L. Yang, R.-Y. Zhang, L. Li, X. Xie, Simam: a simple, parameter-free attention module for convolutional neural networks, in: International Conference on Machine Learning, PMLR, 2021, pp. 11863–11874.
[32] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141, https://doi.org/10.48550/arXiv.1709.01507.
[33] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, Cbam: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision, ECCV), 2018, pp. 3–19, https://doi.org/10.48550/arXiv.1807.06521.
[34] Y. Cao, J. Xu, S. Lin, F. Wei, H. Hu, Gcnet: non-local networks meet squeeze-excitation networks and beyond, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019, https://doi.org/10.48550/arXiv.1904.11492.
[35] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren, Distance-IoU loss: faster and better learning for bounding box regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 12993–13000, https://doi.org/10.48550/arXiv.1911.08287.
[36] M. Tan, R. Pang, Q. V Le, Efficientdet: Scalable and efficient object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10781–10790, https://doi.org/10.48550/arXiv.1911.09070.
[37] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, Q. Tian, Centernet: Keypoint triplets for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6569–6578, https://doi.org/10.48550/arXiv.1904.08189.