Date of publication xxxx xx, xxxx, date of current version xxxx xx, xxxx.

Digital Object Identifier xxxx/ACCESS.xxxx.xxxx

CheapNet: Ultra-Lightweight Real-time Network for Semantic Segmentation and Defect Detection

DAITAO WANG[1], WENJING YU[2], Hongjun Qiu[3], Ruyang Xiao[4], and Jing Yin[5]

[1]Software Engineering Institute of Guangzhou, Guangzhou, 510900, China (wangdaitao7@gmail.com)
[2]Software Engineering Institute of Guangzhou, Guangzhou, 510900, China (ywj@mail.seig.edu.cn)
[3]Guangzhou Civil Aviation College, Guangzhou 510403, China (qiuhongjun@gcac.edu.cn)
[4]Shanghai University of Electric Power, ShangHai 200090, China (xry1334675@163.com)
[5]University of Macau, China (MC36559@umac.mo)

Corresponding author: Wenjing Yu (e-mail: ywj@mail.seig.edu.cn).

ABSTRACT In response to the computational resource restraints in some industrial scenarios,this paper proposes an ultra-lightweight semantic segmentation network with high-performance and resource-efficient computing,CheapNet, which efficiently achieves defect localization and segmentation in device with low computing resource. This network is suitable for defect detection in stable industrial detection environment. This project designs two basic units, Encoding Block and Decoding Block, based on the idea of depthwise convolution with a constant number of channels. CheapNet is built by stacking multiple basic units and combining them with skip connections. Furthermore, the Two-Pass algorithm in computer graphics is employed to calculate the connected-component information of the image, which can obtain the segmentation mask, position coordinates, and corresponding pixel area of the defect region. CheapNet was validated and tested using 10 sub-datasets of DAGM. The experiments show that CheapNet can achieve a good balance between computing resource, inference speed, segmentation, and localization performance. Specifically, without any preprocessing, postprocessing, or image augmentation, CheapNet achieved 87.31% mIoU on the DAGM dataset and speed of 871.7 FPS on single NVIDIA V100 GPU with computational complexity of 0.026 GFLOPs and parameters of 0.000277M. CheapNet can achieve 94.12% mIoU when its computational complexity expand to 13.4 GFLOPs. CheapNet can achieve 79.85% mAP on 252.3 FPS after using Two-Pass algorithm. The cost-effectiveness of CheapNet is much higher than other segmentation and localization algorithms with comparable performance.CheapNet is suitable for industrial defect detection scenarios with computing resource limitation. It can effectively promote the application of deep learning algorithms in the field of industrial quality inspection. The code and pre-trained models are available at https://github.com/hanknewbird/CheapNet.

INDEX TERMS Ultra-Lightweight Network,Real-time Processing,Semantic Segmentation,Defect Detection,Convolutional Neural Network

I. INTRODUCTION

Defect detection is a common task in industrial quality inspection. For example, There are defects such as broken yarns, holes, and pilling in the production of textiles and clothing. There are defects such as pores, cracks, and inclusions in the metal processing industry. Traditional product inspection is mainly carried out manually, which results in high costs, low efficiency, and human errors, among other issues.

In recent years, with the rapid development of artificial intelligence and deep learning, automated detection with artificial intelligence have achieved many good results in the field of industrial quality inspection [1], [2]. In most industrial scenarios, there are high requirements for the inference speed, accuracy, and reliability of intelligent models applied in automated detection technologies. Many previous works [3], [4] have achieved satisfactory results in various benchmark tests [5], [6]. However, many industrial scenarios are constrained by computational resources and data transmission capabilities and need to meet the requirements of real-time and low-cost deployment. Many algorithmic models are not suitable for application in such practical work scenarios, and it is necessary to redesign and develop algorithm that can meet both the detection quality and the resource-efficient comput-

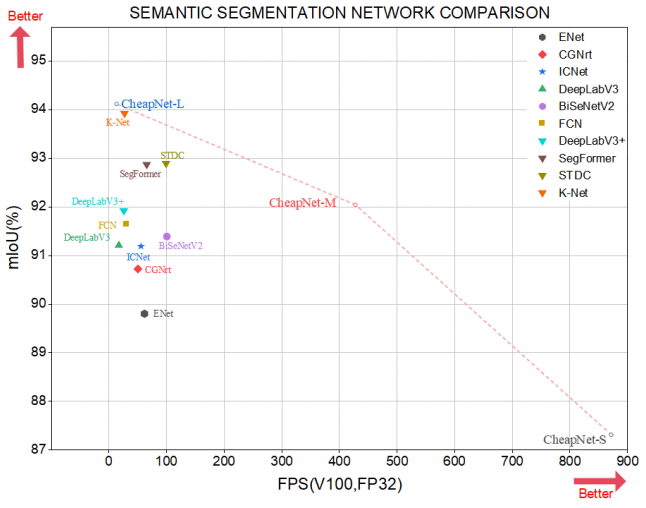

FIGURE 1. Comparsion with other semantic segmentation network.CheapNet achieve state-of-the-arts performance.

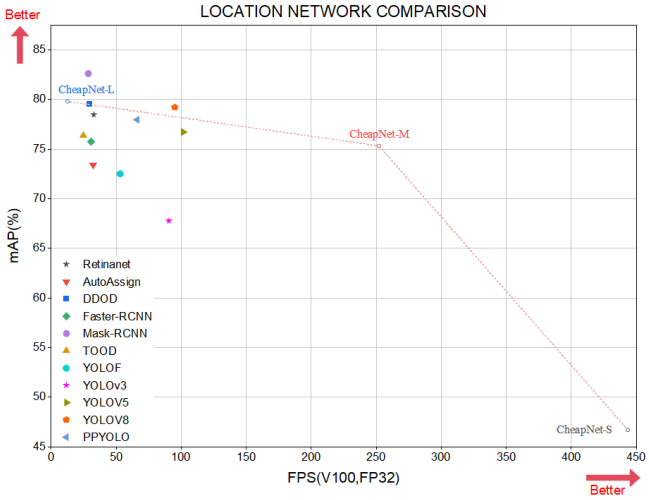

FIGURE 2. Comparsion with other object detection network.CheapNet achieve state-of-the-arts performance.

ing requirements. To address this issue, this paper proposes to construct an ultra-lightweight semantic segmentation network, CheapNet, to better adapt to industrial scenarios.

Through in-depth research [3], [7] on the current lightweight semantic segmentation networks, we found that most algorithms adopt the convolutional layer construction idea of VGG [8], that is, the feature map size is halved and the number of channels is doubled after each convolutional layer. This construction method exponentially increases the number of feature map channels in the encoding stage, resulting in a sharp increase in the network's parameters. To overcome this drawback, we use depthwise convolution and depthwise transposed convolution to construct two parts, namely the Encoding Block and Decoding Block. By stacking these two basic units, we build a 10-layer autoencoder and used four skip connections to fuse low-dimensional global information with high-dimensional local information, and finally output a binary image that distinguishes defects from background, which we called CheapNet. It is worth noting that, unlike the traditional convolutional layer construction method [8],the feature map size of CheapNet is halved, while keeping the number of channels unchanged as the network deepening. It means that the parameters of each convolutional layer is constant and will not lead to a sharp increase in the algorithm's parameters as the network deepens. Thanks to this design concept, CheapNet can exponentially reduce the computational complexity of the network without reducing accuracy.

This paper also attempts to use the Two-Pass algorithm as postprocessing to calculate the connected-component information of the segmentation binary image output by CheapNet. Base on the powerful dimensionality reduction ability of CheapNet, which reduces complex original images to binary images. The Two-Pass algorithm can transform the segmentation problem into a localization problem, allowing us to obtain the position coordinates of each defect part and the corresponding defect pixel area. The overall process is shown in Figure 5. The results of using CheapNet for segmentation and localization tasks are shown in Figure 6. Figures 1 and 2 show its advantages over other semantic segmentation and object detection networks.

We have validated the effectiveness of CheapNet on 10 different sub-datasets of textile materials in DAGM [9] with different textures, lighting conditions, and anomaly types. With a grayscale image input of 512×512, CheapNet achieved 871.7 FPS and 87.32% mIoU with only computational complexity of 0.026 GFLOPs and parameters of 0.000277M on single NVIDIA V100 GPU. When CheapNet is expanded to 13G, it can achieve up to 94.12% mIoU. When combined with the Two-Pass algorithm, it can achieve 252.3 FPS and 79.85% mAP without increasing the model parameters. Its cost-effectiveness is much higher than the other networks with comparable performance, which can effectively promote the application of intelligent detection algorithms in industrial quality inspection scenarios.

Our main contributions are summarized as follows:

- We propose an ultra-lightweight semantic segmentation network, CheapNet, for industrial quality inspection field. CheapNet can be extended with the Two-Pass algorithm to achieve effective and fast defect detection. Under computational complexity of 0.026 GFLOPs,it can achieve 87.31% mIoU with inference speed of 871.7 FPS.
- Unlike the construction idea of most semantic segmentation networks, CheapNet reduces the size of its feature map by half after each convolutional layer, while keeping the number of channels constant. This change greatly reduces the parameters of the network.
- CheapNet has demonstrated its effectiveness on 10 challenging datasets. Its cost-effectiveness is much higher than the other existing algorithms with comparable segmentation and localization performance.

- The experiment shows that using deep learning algorithms for preprocessing and combining them with traditional algorithms can greatly reduce the computational load while achieving comparable performance to mainstream algorithms. This approach provides a new path for lightweight model exploration.

II. RELATED WORK

Semantic Segmentation Due to the powerful feature representation ability of CNN, semantic segmentation has made significant progress in recent years. FCN [10] pioneered the end-to-end training of dense semantic prediction and semantic segmentation in a fully convolutional manner, and many frameworks have been improved on this basis. For example, DeepLab [11] and PSPNet [12] use dilated convolution to construct the ASPP module, which can better describe multi-scale contextual information. OCNet [13] introduces object context information and a new loss function.CCNet [14] has been improved to calculate the correlation between pixels and all pixels in rows and columns. However, these efforts are all aiming at achieving high-quality segmentation indicators without specifically considering the inference speed and algorithm deployment cost of segmentation.

Lightweight Network Lightweight semantic segmentation models need to achieve a good balance between accuracy, model parameters, and computing resource. To speed up inference, related work on model compression and acceleration includes parameter pruning and quantization [15]–[18], low-rank factorization [19]–[23], transferred/compact convolutional filters [24]–[26], and knowledge distillation [27]–[29], but they all reduce the accuracy of the model. To promote the deployment and inference of semantic segmentation models, ENet [30] parallelizes pooling and convolution operations, reduces the input image size, and finally decomposes large convolution kernels into multiple small ones. ICNet [31] uses a Heavy CNN to process low-resolution images and obtain rough prediction feature maps. And then, it proposes a strategy of cascaded feature fusion units and cascaded label guidance and introduces medium and high resolution feature maps. BiSeNetV1 [32] handles spatial details and category semantics of images separately. DFANet [7] designs a feature reuse method that combines multi-level contexts into encoded features. CGNet [33] proposes learning the local features and contextual features of the surrounding environment and adjusts the weight of feature maps using depthwise convolution. BiSeNetv2 [34] reduces channel capacity and adopts a fast downsampling strategy, to make the network smaller. SegFormer [35] introduces a hierarchical structure in Transformer to extract information of different scales. STDC [36] mitigates the time-consuming of spatial encoding by eliminating structural redundancy. K-Net [3] consistently segments instances and semantic categories using a set of learnable kernels. However, most of them adopt the network construction of VGG [8], where the feature map size of each layer is half of the preceding layer , and the number of channels doubles. This leads to exponential growth in the parameters and computational complexity as the network deepening.

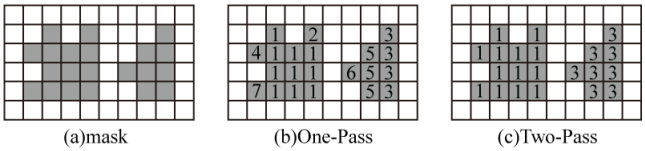

FIGURE 3. The Two-Pass algorithm scans the image(a) twice:During the first pass(b),each non-zero pixel is assigned a numerical label.During the second pass(c),different labels belonging to the same neighborhood are merged, so that all pixels within the same neighborhood have the same label in the end.

Connected-Component Analysis Connected component labeling [37] is used in computer vision to analyze the connected-component of an image. Its main function is to identify and label adjacent pixels with the same value in an image. Common connected component labeling algorithms include Two-Pass [38] and Flood Fill [39]. The former is assigned values by traversing the image twice (as shown in Figure 3), while the latter can achieve Connected component labeling through recursion. Both algorithms can find the coordinates and area of the connected components. They have the advantages of fast inference speed and clever principles, but require high-quality input images.

Summery Although there have been many lightweight semantic segmentation and object detection networks, we believe that the balance between inference speed and accuracy has not been fully explored. The focus of our work in this project is to develop an ultra-lightweight segmentation and localization network that aims to achieve a good balance between inference speed and accuracy.

III. METHODS

A. OBSERVATIONS

Conventional CNN are typically composed of multiple basic units with the same structure.The current lightweight semantic segmentation networks [3], [32], [40] have their feature maps halved in size and doubled in the number of channels after each convolutional layer.According to the formula(1) and (2), the parameters and computational complexity of the network increase sharply as network deepening, the main reason is due to the number of channels after multiple multiplication.To address this issue, this paper stipulates that the number of each channel, except for the initial input and final output channel, is set to G. The basic structure of CheapNet is constructed based on this idea.

$$Params = (C_{out} \times K \times K \times C_{in}) + Bias \quad (1)$$

$$FLOPs = H_{out} \times W_{out} \times C_{out} \times K \times K \times C_{in} \quad (2)$$

In this context, the variable C represents the number of channels, K represents the size of the convolutional kernel, and H and W represent the height and width of the feature map.

B. CHEAPNET UNIT

The convolutional block is the fundamental unit of CNN, and deep learning networks are generally constructed by stacking

multiple basic units. Therefore, lightweight basic units are crucial for efficient networks. After in-depth research on depthwise separable convolution [41], we abandoned most of the structures and proposed two basic units specifically designed for ultra-lightweight semantic segmentation networks: the Encoding Block and the Decoding Block, as shown in Figure 4. Unlike most networks [8], [10], [42], the output channel of each convolutional layer in CheapNet is set to the initial value of G as mentioned before.

1) Encoding Block

The Encoding Block mainly consists of stacked depthwise convolution, Batch Normalization, and Leaky ReLU. The kernel size of the depthwise convolution is 3×3, followed by BatchNorm [43] to optimize the data distribution, and finally LeakyReLu [44] is used as the activation function.In the Encoding Block, the depthwise convolution has a stride of 2 and a padding of 1, and the output channels are all set to G. According to the formula(3), after each Encoding Block, the feature map has the same number of channels but is reduced by half in size, making it a lower-dimensional feature information. The main purpose of the entire encoder part is to extract as much feature information as possible, so as to use the extracted spatial information and global information for accurate segmentation.

$$S_{out} = \lfloor \frac{S_{in} - K + 2P}{Stride} \rfloor + 1 \tag{3}$$

In this context, the variable S_{out} represents the output size of a feature map, while S_{in} represents its input size. The variable P is used to denote the size of padding applied to the feature map.

2) Decoding Block

The Decoding Block primarily consists of depthwise transposed convolution, BatchNorm, and LeakyReLU. Specifically, the input data of the Decoding Block is a horizontal concatenation of the output from the previous layer and the corresponding output from the Encoding Block, while the input data of the Encoding Block is only the output data from the former layer. For the convolutional part, the Encoding Block is depthwise convolution. As the network deepens, the size of its feature map is halved, but the number of channels remains unchanged. This process extracts lower-dimensional features.The Decoding Block is the depthwise transposed convolution, and the size of the convolution kernel is set to 3×3, the stride is 2, and the padding and output padding are both 1.According to the formula(4), after passing through each Decoding Block, the number of channels of feature map is unchanged but the size is doubled, which is a process of restoring features while minimizing information loss.

$$S_{out} = (S_{in} - 1) \times Stride - 2P + K + P_{out} \tag{4}$$

In this context,S_{out} and S_{in} represent the output and input dimensions of feature maps,respectively.P and P_{out} represent the padding and output padding sizes,respectively.

3) Skip Connection

The fusion of global information from high-dimensional features and local information from low-dimensional features can jointly improve the accuracy of the network. Shallower information can provide more global information, while deeper features have richer local information. To improve the accuracy of the model, the input data of the Decoding Block is obtained from horizontally concatenating the output data of the previous layer and the output data of the corresponding Encoding Block with the same size. This method can improve the accuracy of the model by fusing high and low dimensional information.

C. LOCATION MODEL

Traditional machine vision algorithms have strong theoretical support and are easier to understand, but these algorithms have high requirement to the input images. In this paper, the Two-Pass algorithm is used as a postprocessing technique, which can combine the powerful feature learning ability of deep learning with the solid theoretical foundation of traditional machine learning algorithms. Specifically, the binary map output by CheapNet is used as the input to the Two-Pass algorithm. By using CheapNet to reduce the complex input image to a binary map distinguishing defects from the background, the Two-Pass algorithm can then calculate the pixel area and location information of the defects.

D. NETWORK ARCHITECTURE

Figure 5 shows the overall architecture of this project, which consists of CheapNet (Figure 5(left)) and the Two-Pass algorithm (Figure 5(right)) with defect detection capabilities.

The overall architecture of CheapNet (Figure5(left)) is similar to UNet [45], consisting of five stacked Encoding Blocks and five Decoding Blocks in the encoder and decoder parts. To better utilize high and low-dimensional feature information, we have incorporated four skip connections into CheapNet.

In the encoder part, the number of channel of the feature map does not change after each convolu tional layer, but the feature map size is reduced by half. In the decoder part, the

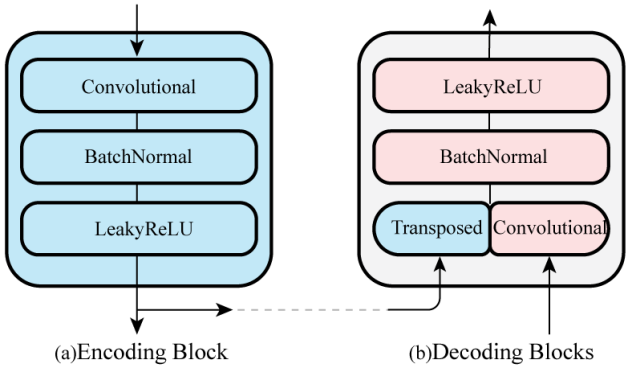

FIGURE 4. Encoding Block(a) and Decoding Block(b) are the basic building blocks of CheapNet.

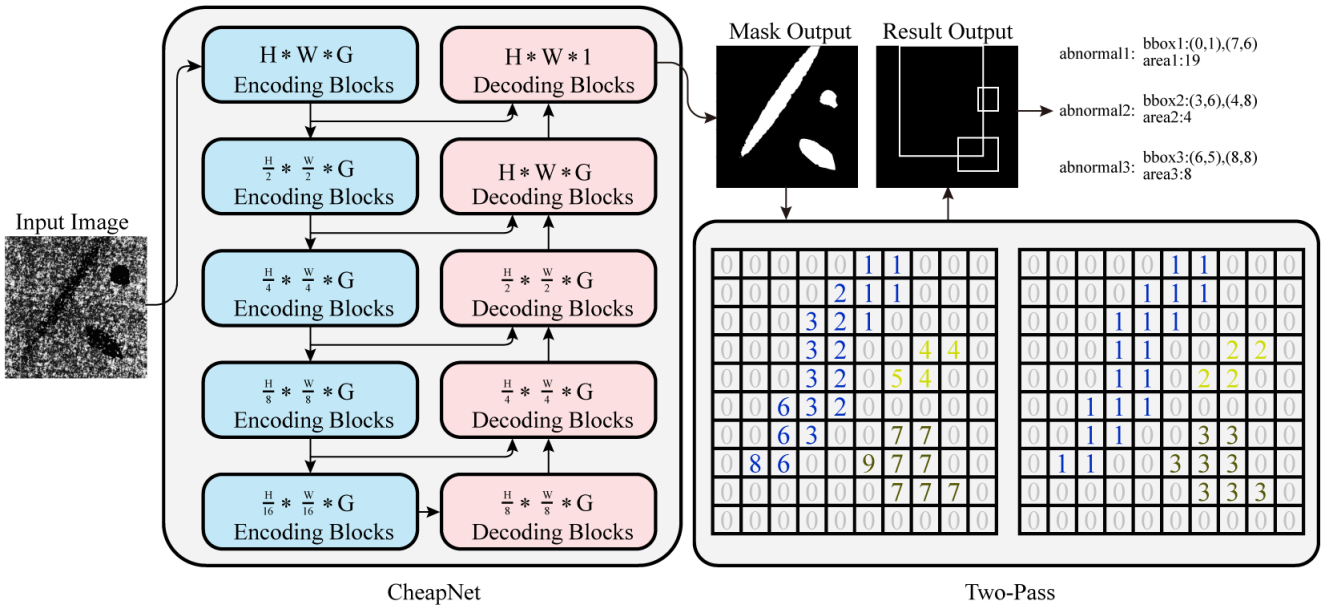

FIGURE 5. The complete network architecture diagram,where the left side is CheapNet and the right side is the Two-Pass module,the combination of which achieves semantic segmentation and defect detection.G refers to the number of channels.

Module	Layer	Input Size	Output Size
-	Input Image	$1 \times 512 \times 512$	-
CheapNet	Encoding Block1	$1 \times 512 \times 512$	$G \times 512 \times 512$
	Encoding Block2	$G \times 512 \times 512$	$G \times 256 \times 256$
	Encoding Block3	$G \times 256 \times 256$	$G \times 128 \times 128$
	Encoding Block4	$G \times 128 \times 128$	$G \times 64 \times 64$
	Encoding Block5	$G \times 64 \times 64$	$G \times 32 \times 32$
	Decoding Block1	$G \times 32 \times 32$	$G \times 64 \times 64$
	Decoding Block2	$2G \times 64 \times 64$	$G \times 128 \times 128$
	Decoding Block3	$2G \times 128 \times 128$	$G \times 256 \times 256$
	Decoding Block4	$2G \times 256 \times 256$	$G \times 512 \times 512$
	Decoding Block5	$2G \times 512 \times 512$	$1 \times 512 \times 512$
-	Mask Output	-	$1 \times 512 \times 512$
Expand	Two-Pass	$1 \times 512 \times 512$	$N \times 5$
-	Result Output	-	$N \times 5$

TABLE 1. Table of CheapNet components,where G represents the number of channels and N represents the number of detected defects.

input data comes from the horizontal concatenation of the previous Decoding Block and its corresponding Encoding Block (for example, the input data of Decoding Block 2 is the horizontal concatenation of Decoding Block 1 and Encoding Block 4, while the input data of Decoding Block 3 is the horizontal concatenation of Decoding Block 2 and Encoding Block 3). The number of channels in the feature map remains unchanged, but the size is doubled. Finally, transforming the feature map through a Sigmoid function [46]and defining outputs greater than 0.5 as defects and the rest as background.Therefore,The macro form of the entire CheapNet is to transform the input image to a segmentation binary image of the same size by the way of dimension reduction.

Parsing The extension part of the defect connected-component information by Two-Pass algorithm is shown in Figure 5(right), where the binary segmentation image generated by the CheapNet is passed to the Two-Pass algorithm. After the image is traversed twice and its pixel values are assigned individually, the connected-component information of the image calculated to the position information and pixel area size of the defect part. Table 1 shows the more detailed composition structure parameters, where G represents the number of channel, N represents the number of detected defects, and each defect is represented by a tuple, namely $[x_1, y_1, x_2, y_2, area]$, which respectively represent the upper left corner coordinates, lower right corner coordinates, and pixel area of the abnormal part.

E. COMPARISON WITH SIMILAR WORKS

The CheapNet structure is concise and different from the depthwise separable convolution used in Xception [41] and MobileNet [47], [48]. CheapNet mainly uses 3×3 depthwise convolution to construct a network which is similiar to structure of UNet, which can be integrated with the Two-Pass algorithm to parsing of defect information. Most importantly, unlike the convolutional idea of VGG [8], whose feature map size is reduced by half and the number of channel is doubled after each convolutional layer, except for the initial input and final output. The output channel of CheapNet is always G was initially set, which is to make the network as lightweight and efficient as possible.

Table 2 shows the comparison of parameters when constructing CheapNet using standard convolution and depthwise convolution. G represents the output channel of each layer. If standard convolution is used, the parameters can be calculated by formula (5) as $99G^2+38G+3$. But if depthwise

Layer	Kernel	Channels		Bias	BN	Params	
		Input	Output			Standard	Depthwise
Encoding Block1	3	1	G	G	2G	$12G$	$9+3G$
Encoding Block2	3	G	G	G	2G	$(9G+3)G$	$12G$
Encoding Block3	3	G	G	G	2G	$(9G+3)G$	$12G$
Encoding Block4	3	G	G	G	2G	$(9G+3)G$	$12G$
Encoding Block5	3	G	G	G	2G	$(9G+3)G$	$12G$
Decoding Block1	3	G	G	G	2G	$(9G+3)G$	$12G$
Decoding Block2	3	2G	G	G	2G	$(18G+3)G$	$21G$
Decoding Block3	3	2G	G	G	2G	$(18G+3)G$	$21G$
Decoding Block4	3	2G	G	G	2G	$(18G+3)G$	$21G$
Decoding Block5	1	2G	1	1	2	$2G+3$	$2G+3$
sum	-	-	-	-	-	$99G^2+38G+3$	$128G+12$

TABLE 2. Table comparing the parameter counts of CheapNet constructed with standard convolution and depthwise convolution,where G is the number of channels.

convolution is used, the parameters of the network is only $128G+12$, which is only about $\frac{1.29}{G}$ of the standard convolution. As the number of channel G increases, the lightweight transformation effect of depthwise convolution on CheapNet becomes more significant. When the number of channel G is set to 1024, the parameters will be reduced about 791 times.

$$params_s = C_{out} \times C_{in} \times k \times k \tag{5}$$

$$params_g = \frac{C_{out}}{G} \times \frac{C_{in}}{G} \times k \times k \times G \tag{6}$$

In this context, $paras_s$ and $paras_g$ represent the parameters of standard and grouped convolutions, C_{out} and C_{in} represent the number of output and input channels respectively, and G represents the number of groups.

IV. EXPERIMENTS

A. DATASETS

DAGM:The dataset consists of 10 comprehensive sub-datasets used for industrial optical inspection, including 10 synthetic textile datasets (referred to as DAGM1∼10 below). These datasets have large differences in texture, lighting intensity, and defect types. Since the number of images in each sub-dataset is not uniform, in order to train the model fairly, we adjusted the number of images in some sub-datasets. The final dataset used is composed of 2100 grayscale images with a size of 512×512 pixels and rough annotations, which are divided into a training set and a validation set in a ratio of 8:2.

B. TRAINING DETAILS

All training processes of this work were run on 8 × NVIDIA V100 GPU. During the inference stage, we only used single V100 GPU for prediction.In this project,a series of experiments are conducted on the PyTorch [49] framework, including ablation study and model comparisons.Unless otherwise specified, the BCE loss [50] is used as the loss function, Adam [51] is used as the optimizer, the epoch is set to 120, the initial learning rate is set to 0.01,decay rate is set 0.1, and the MultiStepLR is adopted as learning rate strategy, with milestones at $[60, 90]$.No data augmentation was used in the training phase.In the obtained model, we only compared the best mIoU model in the verification process.In order to quantitative evaluation, mIoU is used to define the segmentation performance,and FLOPs is used to computational complexity.FPS is used to measure the running speed.

C. ABLATION STUDY

In order to evaluate the effectiveness of depthwise convolution in the lightweight design of CheapNet, this project specifies 10 different numbers of channels from 2 to 1024, and constructs the structure of CheapNet by standard convolution and depthwise convolution,respectively.The detailed data shown in Table 3.

Paramters and Computational complexity. By comparing the computational complexity and parameters of CheapNet constructed in different ways, it is evident that depthwise convolution has significant lightweighting effect on CheapNet. Its computational complexity and parameters is only 0.176%∼65.3% and 0.135%∼58.3% of standard convolution, respectively. In other words, compared to build CheapNet by standard convolution, depthwise convolution can reduce computational complexity and parameters by 0.5∼565.5 and 0.7∼739.2 times, respectively.

Inference Time. Inference speed is an important indicator of the model. CheapNet constructed using depthwise convolution achieves very fast inference speed, with a maximum of 871.7 FPS.It is significantly higher than other semantic segmentation networks with comparable performance.

mIoU. According to the experimental results, the mIoU metric of CheapNet constructed by standard convolution is always better than that constructed by depthwise convolution. But we found that the former's mIoU does not increase with the increase of the number of channels. It reaches the highest point when the number of channel is 4, then slowly decreases, and then continues to rise. We will investigate the reasons for this phenomenon in further research. The latter has better fitting ability with the increase of the number of channels, leading to an improvement in mIoU. Compared with the former, we believe that depthwise convolution makes CheapNet sparser, which is equivalent to a regularization process on

Convolution	Channel	mIoU(%)	Paramters	mAcc(%)	512*512 - NVIDIA V100 32G		
					GFLOPs	fps	ms
Standard	2	93.92	**0.000475M**	96.98	**0.040448000**	**458.6**	**2.18**
Depthwise		87.31	**0.000277M**	91.73	**0.026421248**	**871.7**	**1.15**
Standard	4	**94.80**	0.001739M	96.68	0.135954432	381.0	2.62
Depthwise		88.19	0.000551M	96.29	0.051793920	837.3	1.19
Standard	8	94.06	0.006643M	97.00	0.495288320	296.8	3.37
Depthwise		90.76	0.001099M	94.01	0.102539264	733.4	1.36
Standard	16	93.98	0.025955M	96.33	1.887240192	222.3	4.50
Depthwise		91.62	0.002195M	97.05	0.204029952	703.0	1.42
Standard	32	93.78	0.102595M	**98.11**	7.364280320	141.4	7.07
Depthwise		92.04	0.004387M	**97.75**	0.407011328	427.8	2.34
Standard	64	93.71	0.407939M	96.48	29.090906112	117.6	8.50
Depthwise		92.84	0.008771M	96.59	0.812974080	234.4	4.27
Standard	128	93.97	1.626883M	96.33	115.634339840	33.6	29.76
Depthwise		93.19	0.017539M	95.43	1.624899584	188.6	8.43
Standard	256	94.48	6.497795M	96.60	461.081935872	10.7	93.71
Depthwise		93.24	0.035075M	95.37	3.248750592	56.8	17.62
Standard	512	94.50	25.9717515M	96.76	1841.420042240	2.9	341.02
Depthwise		93.31	0.070147M	95.36	6.496452608	27.5	36.35
Standard	1024	-	103.847939M	-	7359.867912192	-	-
Depthwise		**94.12**	0.140291M	95.39	12.991856641	13.4	74.81

TABLE 3. A more detailed comparison of the performance of constructing CheapNet using standard convolution and depthwise convolution.

each layer of the network. This is why the mIoU of the latter continues to improve with the increase of the number of channels.

BaseLine. For comparison with other networks, we define networks with numbers of channels is 2, 32, and 1024 as the small, medium, and large versions of CheapNet, respectively. These versions were named CheapNet-S, CheapNet-M, and CheapNet-L, respectively.

V. COMPARISON ON OTHER FRAMEWORKS

Tables 4 and 5 show the performance comparison of CheapNet with some mainstream models in semantic segmentation and object detection. Figures 1 and 2 provide visual comparisons. All comparative models are implemented based on mmlab's [61]–[63] baseline. Figure 6 illustrates the segmentation and localization results of CheapNet on 10 sub-datasets of DAGM. By comparing the ground truth labels with the corresponding predicted results in the figure, it can be observed that CheapNet can accurately segment and locate defects, and has good generalization performance on different sub-datasets. This indicates that CheapNet can be used as an effective method for image segmentation and localization in various practical applications.

A. SEGMENTATION

Table 4 reports the comparison of CheapNet with some mainstream semantic segmentation models in terms of mIoU, parameters, mAcc, FLOPs, and FPS. According to the experimental results in the table, CheapNet outperforms other methods significantly, including lightweight networks such as ENet [30] and CGNet [33], as well as large networks such as DeepLabV3+ [40] and K-Net [3].

CheapNet-S achieves 87.31% mIoU with only 0.026 GFLOPs, and its inference speed is as fast as 871.7 FPS. This is two orders of magnitude lower in FLOPs than ENet and CGNet which are current minimum parameters models. Moreover, its inference speed is 13 and 16 times faster than ENet and CGNet, respectively.

CheapNet-M is the most balanced network in terms of all metrics in this work. Compared with other state-of-the-art lightweight networks with comparable accuracy, such as SegFormer [35] and STDC [36], CheapNet-M achieves a faster speed (427.8FPS), smaller parameters (0.004387M) and higher performance (97.75% mAcc).

CheapNet-L is the network with the largest computational complexity in this work. Despite this, it is still two orders of magnitude smaller than other large networks. However, its mIoU reaches 94.12%, which is even higher than the state-of-the-art semantic segmentation network K-Net by 0.2% mIoU. This confirms the effectiveness of CheapNet.

B. LOCATION

In this project, Two-Pass algorithm is integrated to extend the function of defect information analysis, which includes the location information of defects. Table 5 reports the comparison of the extended CheapNet with some mainstream object detection networks.

According to Table 5, CheapNet-S has lower mAP, which is due to its insufficient accuracy in the segmentation part. However, CheapNet-M and CheapNet-L perform excel-

model	mIoU(%)⇑	Paramters	mAcc(%)	512*512 - NVIDIA V100 32G		
				GFLOPs	fps	ms
ENet [30]	89.80	**0.37M**	92.26	4.36	62.3	16.05
CGNet [33]	90.72	0.49M	93.63	**3.41**	50.8	19.69
ICNet [31]	91.19	47.82M	93.13	15.39	56.2	17.79
DeepLabV3 [11]	91.21	68.10M	93.19	269.64	17.8	56.18
BiSeNetV2 [34]	91.39	14.76M	93.84	12.30	**100.8**	**9.92**
FCN [10]	91.64	49.48M	94.00	197.69	30.8	32.47
DeepLabV3+ [40]	91.92	43.58M	93.86	176.22	26.5	37.74
SegFormer [35]	92.87	3.72M	94.71	6.36	65.9	15.17
STDC [36]	92.89	8.57M	**97.12**	8.46	99.9	10.01
K-Net [3]	**93.92**	62.17M	96.23	181.69	27.3	36.63
CheapNet-S	87.31	**0.000277M**	91.73	**0.026421248**	**871.7**	**1.15**
CheapNet-M	92.04	0.004387M	**97.75**	0.407011328	427.8	2.34
CheapNet-L	**94.12**	0.140291M	95.39	12.991856641	13.4	74.81

TABLE 4. Table comparing CheapNet with mainstream semantic segmentation networks.

model	mAP(%)⇑	Paramters	512*512 - NVIDIA V100 32G		
			GFLOPs	fps	ms
YOLOv3 [52]	67.73	**3.67M**	**4.21**	90.8	11.0
YOLOF [53]	72.49	42.06M	25.13	53.3	18.8
AutoAssign [54]	73.41	35.97M	50.60	32.4	30.9
Faster-RCNN [55]	75.74	41.12M	63.25	31.0	32.2
TOOD [56]	76.33	31.79M	46.19	25.0	40.0
YOLOV5 [57]	76.70	7.02M	5.07	**101.5**	**9.9**
PPYOLO [58]	77.95	7.61M	5.17	66.4	15.1
Retinanet [59]	78.45	36.10M	52.28	33.1	30.2
YOLOV8 [42]	79.20	11.17M	7.48	95.4	10.5
DDOD [4]	79.51	31.97M	45.55	30.1	33.3
Mask-RCNN [60]	**82.60**	43.75M	114.73	28.8	34.7
CheapNet-S	46.72	**0.000277M**	**0.026421248**	**443.4**	**2.31**
CheapNet-M	75.33	0.004387M	0.407011328	252.3	3.96
CheapNet-L	**79.85**	0.140291M	12.991856641	12.8	78.41

TABLE 5. Table comparing CheapNet with mainstream object detection networks.

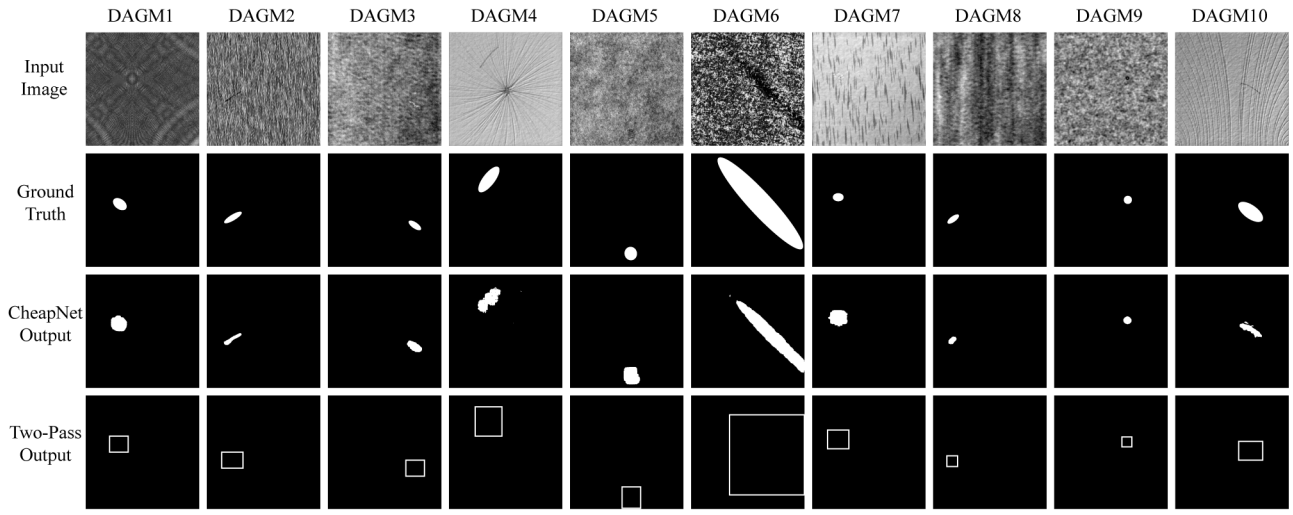

FIGURE 6. Output result maps of CheapNet-S for different datasets at different stages.

lently. CheapNet-M has the same level of mIoU as Faster-RCNN [55], its speed is more fast Faster-RCNN (252.3FPS). So CheapNet-M has better cost-performance. CheapNet-L reaches 79.85% mAP, which is the same level of YoloV8 [42] and DDOD [4].

VI. CONCLUSION AND FUTURE WORK

In this paper,we propose an idea of constant number of channels. Base on this idea, we have developed a new ultra-lightweight semantic segmentation network - CheapNet. This algorithm is designed specifically for industrial quality inspection scenarios and is combined with the Two-Pass algorithm to obtain the position coordinates and the area of the defect part.Our main goal is to make the network as lightweight as possible and implement segmentation and localization functions.This work has made significant progress towards this goal. CheapNet has a smaller size and faster speed, and its cost-performance is far higher than the other algorithms with segmentation and localization functions.Although the main goal of this work is to enable the network to run on computational resource restricted equipment, we found that CheapNet also performs very well on high performance GPU. If deployed in data centers, CheapNet will be more fast and more efficient.It can complete larger scale quality inspection tasks, which will save more costs for enterprises.

Although CheapNet has advantages such as fast inference speed and small model size, our work still has some limitations. One of them is that CheapNet's training method is supervised learning, which requires a large number of images containing defects as the dataset. However, it is difficult to collect a large amount of abnormal data in actual industrial scenarios. In the future, we plan to explore the use of semi-supervised or unsupervised learning methods to solve this limitation, reduce the dependence on abnormal images. We also plan to use Generative Adversarial Network (GAN) to generate abnormal images as datasets, to further reduce the cost of model deployment in industrial scenarios. Overall, we believe that CheapNet will have significant potential for industrial quality inspection scenarios with stable environment, and we look forward to this algorithm can be widely used in this field.

VII. ACKNOWLEDGMENTS

The research is partially supported by Special projects in key fields of Colleges and universities of Guangdong Province in 2021 (No: 2021ZDZX1065) and Intelligent Analysis Digital Visualization Platform For Petroleum Explo ration(No:PDJH2022b0658).

REFERENCES

[1] H. Y. Ngan, G. K. Pang, and N. H. Yung, "Automated fabric defect detection—a review," *Image and Vision Computing*, vol. 29, no. 7, pp. 442–458, 2011.
[2] A. Kumar, "Computer-vision-based fabric defect detection: A survey," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 1, pp. 348–363, 2008.
[3] W. Zhang, J. Pang, K. Chen, and C. C. Loy, "K-Net: Towards unified image segmentation," in *NeurIPS*, 2021.
[4] Z. Chen, C. Yang, Q. Li, F. Zhao, Z.-J. Zha, and F. Wu, "Disentangle your dense object detector," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 4939–4948, 2021.
[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
[6] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," *http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html*, 2012.
[7] H. Li, P. Xiong, H. Fan, and J. Sun, "Dfanet: Deep feature aggregation for real-time semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9522–9531, 2019.
[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[9] F. A. H. . Matthias Wieler, Tobias Hahn, "Weakly supervised learning for indus-trial optical inspection." https://hci.iwr.uni-heidelberg.de/content/weakly-supervised-learning-industrial-optical-inspection.
[10] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 640–651, 2017.
[11] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.
[12] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890, 2017.
[13] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "Ocnet: Object context network for scene parsing," *arXiv preprint arXiv:1809.00916*, 2018.
[14] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 603–612, 2019.
[15] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *arXiv preprint arXiv:1412.6115*, 2014.
[16] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4820–4828, 2016.
[17] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on cpus," in *Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011*, 2011.
[18] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *International conference on machine learning*, pp. 1737–1746, PMLR, 2015.
[19] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2754–2761, 2013.
[20] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," *Advances in neural information processing systems*, vol. 27, 2014.
[21] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," *arXiv preprint arXiv:1405.3866*, 2014.
[22] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned cp-decomposition," *arXiv preprint arXiv:1412.6553*, 2014.
[23] C. Tai, T. Xiao, Y. Zhang, X. Wang, *et al.*, "Convolutional neural networks with low-rank regularization," *arXiv preprint arXiv:1511.06067*, 2015.
[24] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*, pp. 2990–2999, PMLR, 2016.
[25] S. Zhai, Y. Cheng, Z. M. Zhang, and W. Lu, "Doubly convolutional neural networks," *Advances in neural information processing systems*, vol. 29, 2016.
[26] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *international conference on machine learning*, pp. 2217–2225, PMLR, 2016.
[27] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[28] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.
[29] J. Ba and R. Caruana, "Do deep nets really need to be deep," *Advances in neural information processing systems*, vol. 27, 2014.
[30] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.
[31] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 405–420, 2018.
[32] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 325–341, 2018.
[33] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, "Cgnet: A light-weight context guided network for semantic segmentation," *IEEE Transactions on Image Processing*, vol. 30, pp. 1169–1179, 2020.
[34] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *International Journal of Computer Vision*, pp. 1–18, 2021.
[35] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *arXiv preprint arXiv:2105.15203*, 2021.
[36] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking bisenet for real-time semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9716–9725, 2021.
[37] wiki, "Connected component labeling." https://en.wikipedia.org/wiki/Connected-component_labeling.
[38] G. Shapiro, L.; Stockman, *computer vision*. Prentice Hall, 2002.
[39] A. R. Smith, "Tint fill," in *Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, pp. 276–283, 1979.
[40] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018.
[41] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.
[42] YOLOv8, "Yolov8." https://github.com/ultralytics/ultralytics.
[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, pp. 448–456, 2015.
[44] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, p. 3, 2013.
[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.
[46] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Neurocomputing*, vol. 1, no. 4, pp. 541–551, 1986.
[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 484–492, 2017.
[48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.
[49] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *NIPS-W*, 2017.
[50] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2014.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[52] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.
[53] Q. Chen, Y. Wang, T. Yang, X. Zhang, J. Cheng, and J. Sun, "You only look one-level feature," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.
[54] B. Zhu, J. Wang, Z. Jiang, F. Zong, S. Liu, Z. Li, and J. Sun, "Autoassign: Differentiable label assignment for dense object detection," *arXiv preprint arXiv:2007.03496*, 2020.
[55] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jun 2017.
[56] C. Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in *ICCV*, 2021.
[57] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, Y. Kwon, K. Michael, and J. Fang, "ultralytics/yolov5: v6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations," *Zenodo. org*, 2022.
[58] S. Xu, X. Wang, W. Lv, Q. Chang, C. Cui, K. Deng, G. Wang, Q. Dang, S. Wei, Y. Du, and B. Lai, "Pp-yoloe: An evolved version of yolo," *ArXiv*, vol. abs/2203.16250, 2022.
[59] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017.
[60] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
[61] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, Y. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Mmcv: Openmmlab's open-source computer vision toolbox," *arXiv preprint arXiv:2007.13171*, 2020.
[62] Z. Dai, Y. Chen, S. Liu, D. Zhang, J. Yi, S. Pu, J. Jia, H. Li, G. Li, Y. Zhou, *et al.*, "Openmmlab semantic segmentation toolbox and benchmark," *arXiv preprint arXiv:2012.15712*, 2020.
[63] K. Chen, J. Wang, S. Li, W. Wu, T. Zhang, B. Zhou, and X. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7122–7131, 2019.

DAITAO WANG graduated from Guangzhou Software Institute in China with a B.S. in Network Engineering in 2023. He has a strong interest in computer science and programming, and is actively involved in research projects and competitions at the school.

During his college years, he excelled and won several provincial and national level competition honors and was supported by several scholarships. He has also shown outstanding ability in colleges, participating in several research projects and presenting several papers at international academic conferences. Although as an undergraduate student, his research and contributions in the field of computer science are still limited. However, his desire to expand his field of knowledge and to keep learning makes him a promising researcher and innovator. He hopes to contribute more to the development of the field of computer science through in-depth study and participation in research projects.

WENJING YU received the B.S. degree in computer science and technology from Northwestern Polytechnical University, Xi'an, China, in 2005 and the M.S. degree in computer application technology from Dalian University, Dalian, China in 2008.

From 2008 to 2018, she was an assistant professor and a lecturer with the Network technology Department, Software Engineering Institute of Guangzhou. Since 2019, she has been an associate professor with the Network technology Department, Software Engineering Institute of Guangzhou. Her research interests include artificial intelligence, machine vision, graphics and image processing, optimization algorithms, etc. She is a senior member of China Computer Society.

HONGJUN QIU received the B.S. degree in mechanical engineering and automation from Northwestern Polytechnical University, Xi'an, China, in 2000 and the M.S. and Ph.D. degree in aeronautics and astronautics manufacturing engineering from Northwestern Polytechnical University, Xi'an, China, in 2006.

From 2007 to 2010, he was a lecturer with the Mechatronics Engineering Department, Shantou University. Since 2010, he has been an associate professor with the Aircraft Maintenance Engineering Department, Guangzhou Civil Aviation College. His research interests include aircraft manufacturing, aircraft maintenance, intelligent manufacturing technology.

RUYANG XIAO received the B.S. degree in Network Engineering from Guangzhou Software Institute, Guangzhou, China, in 2023.He will study for a master's degree in artificial intelligence in Shanghai Electric Power University,shanghai,China,In 2023.

During his college years, he won several provincial and national level competition honors.During graduate school, his research focused on private computing and artificial intelligence security.

JING YIN received her Bachelor's degree in Network Engineering from Software Engineering Institution of Guangzhou in 2023, and has been studying at the University of Macau for her Master of Science in Data Science since 2023. Her main research direction is Analytics in Teaching and Learning, and she is committed to the application of big data analysis in the field of education.