A-Net: A Lightweight Real-time Segmentation Network for Industrial Surface Defect Detection

Biao Chen[a], Tongzhi Niu[a], Wenyong Yu*, Ruoqi Zhang, Zhenrong Wang, Bin Li

***Abstract*—This paper presents a novel lightweight convolutional neural network, A-Net, designed for semantic segmentation of industrial surface defects. In response to the limitations of existing backbone networks, A-Net implements a unique A-shaped architecture, facilitating efficient feature extraction and fusion. Feature extraction operates on low-level detail and high-level semantic information separately, eliminating the need for additional branches and maximizing the use of detailed information. Feature fusion is achieved through layer-by-layer up-sampling, minimizing memory usage by limiting the number of skip connections. The proposed A-Net achieves competitive performation against classic large models (A-Net achieves IoU of 66.27% on NEU-seg and 80.97% on DAGM-seg with only 0.39M parameters and 0.44G floating point operations per second (FLOPs), while Unet achieves IoU of 66.48% on NEU-seg and 81.27% on DAGM-seg with 31.39M parameters and 42.75G FLOPs). Besides, our method shows extremely fast inference speed on edge device without GPU because of its low FLOPs. Our work contributes to the development of effective and efficient defect segmentation networks, suitable for real-world industrial applications with limited resources.**

***Index Terms*—Surface defect detection, Lightweight neural network, Real-time neural network, Neural network architecture.**

I. Introduction

In recent years, defect segmentation has emerged as a pivotal topic in industrial surface defect detection, aimed at accurately locating and sizing defects [1], [2]. The flourishing field of semantic segmentation has significantly enhanced defect segmentation performance through various breakthroughs, such as FCN [3], SegNet [4], U-Net [5], and PGA-Net [6], among others. This has led to a growing demand for low-latency edge deployment with limited computing power. Consequently, there has been a rising interest in developing defect segmentation networks that strike a balance between effectiveness and efficiency, making this research area highly valuable to explore.

To address these requirements, numerous researchers have proposed the design of low-latency, high-efficiency CNN models that maintain satisfactory segmentation accuracy. We will discuss segmentation network design from two perspectives: the backbone networks and the lightweight approaches.

Regarding the backbone networks, there are three prevalent approaches for devising efficient segmentation methods, as illustrated in figure 2 (a-c). 1) Encoder-decoder backbone: this method employs top-down and skip connections to recover high-resolution feature representation in the decoder part (e.g., FCN [3], SegNet [4], U-Net [5], and PGA-Net [6]). However, the numerous connections can hinder low memory access costs. 2) Pyramid pooling backbone: this approach uses a pyramid pooling module (or dilation convolution [7]) to capture multi-scale contextual information, which is then concatenated and fed into the segmentation networks (e.g., PSPNet [8], DeepLab [9]). Unfortunately, the pyramid pooling backbone results in high computational complexity and memory footprint. 3) Bilateral segmentation backbone: this strategy adopts a multi-path framework to combine low-level details with high-level semantics (e.g., BiSegNet [10], BiSegNet V2 [11], STDC [12]). While these methods reduce computational complexity, adding an additional path to obtain low-level features can be time-consuming, and the auxiliary path is not utilized optimally.

To achieve both effectiveness and efficiency, based on both backbone networks, existing methods primarily employ two approaches to accelerate the model: 1) Input restriction and channel pruning, which increase the inference speed but may sacrifice spatial details around boundaries and small objects (e.g., ENet [13], ICNet [14]). 2) Well-designed convolution blocks, such as non-bottleneck ResNet block (ERFNet [15]),

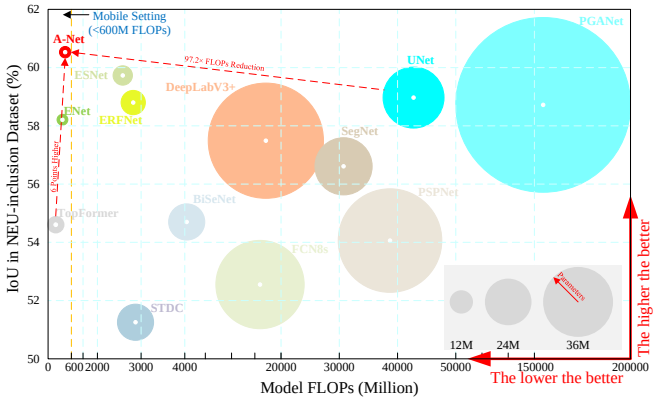


Fig. 1. Comparisons with generic and lightweight semantic segmentation network in terms of IoU performance, model Flops and parameters on NEU-inclusion dataset. Due to the large difference in the number of parameters between the lightweight segmentation network and the universal segmentation network, we enlarge the circle representing A-Net and ENet by a factor of 15, ESNet, ERFNet and Topformer by a factor of 6, and BiSeNet and STDC by a factor of 1.5 to make the picture more beautiful. The proposed A-Net is superior than all models shown in this figure, while using much fewer Flops and parameters. Best viewed in color.

The authors are from Huazhong University of Science and Technology, Wuhan 430000, China (e-mail: u202010899@hust.edu.cn; tzniu@hust.edu.cn; ywy@hust.edu.cn; m202271390@hust.edu.cn; zora$_w$ang@hust.edu.cn; libin999@hust.edu.cn).
*Correspondingauthor
[a]Equalcontribution

convolution and pooling parallel block (ENet [13]), and others. We will adopt these lightweight blocks and investigate how to improve defect segmentation performance with limited computation. 3) Multi-branch framework to combine spatial details and context information, as discussed in the Bilateral segmentation backbone. Nonetheless, we will explore constructing lighter-weight branches and fully leveraging spatial detail information.

Additionally, we analyze the challenges of designing a lightweight model for industrial surface defects. Firstly, the number of industrial defect images is limited, posing a challenge for lightweight networks with constrained feature extraction capabilities. Due to a small number of parameters, lightweight models' performance may be significantly reduced or even fail to converge effectively without sufficient training on extensive data. Secondly, defects exhibit variations in size and irregular outlines. Prior segmentation methods have tackled these challenges by adopting large-scale convolutions or dilation convolutions to expand the receptive field and pyramid feature fusion structures to simultaneously focus on information of different scales. However, both approaches entail high parameter quantity and computational complexity. Finally, the difference between the defect area and the normal area is not always apparent. This can be addressed by adopting multiple skip connections for fine feature recovery or increasing the number of auxiliary training branches dedicated to border segmentation. However, both methods come with high memory access costs.

To overcome these challenges, we propose a lightweight network called A-Net, which performs well on various surface defect datasets while remaining exceptionally lightweight.

Initially, we designed a series of lightweight convolution blocks comprising: 1) Feature extraction blocks, which include a light block and a wide block corresponding to 3x3 and 5x5 receptive fields, respectively. 2) Up-sampling and down-sampling blocks, composed of 2x2 convolutional layers with a stride of 2 and deconvolutional layers, respectively. 3) Concatenation blocks. Within these blocks, we employed depthwise convolution, dropout layers, and residual connection structures to prevent overfitting, gradient vanishing, and gradient explosion issues, thus creating a lightweight network model adaptable to small datasets.

Subsequently, we proposed an A-shaped network structure, depicted in Figure 2(d), consisting of two primary components: feature extraction and feature fusion. In the feature extraction stage, inspired by the bilateral segmentation backbone, A-Net separately extracts low-level detailed information and high-level semantic information. A-Net obtains detailed information using a down-sampling block and single-layered feature extraction blocks while extracting semantic information by stacking additional feature extraction blocks. Unlike other methods, A-Net does not rely on additional branches. Instead, it fully utilizes detailed information and distinguishes between low-level detailed and high-level semantic information within the feature extraction block. In the feature fusion stage, we employ layer-by-layer up-sampling by a factor of 2 to preserve the network's ability to discern subtle features. The up-sampling operation encompasses up-sample blocks and stacked feature extraction blocks. Moreover, to minimize memory consumption, we restrict the number of skip connections to a single layer, facilitating the fusion of detailed and semantic information. The network's output is obtained through the segmentation head structure.

In summary, our main contributions are as follows:

1) We propose a novel network backbone, dubbed A-Net, which extracts information at different levels in stages during the down-sampling stage and facilitates the aggregation of information at various levels through skip connections in the up-sampling stage. The network is named A-Net due to its similarity in shape to the capital letter A.

2) We designed lightweight feature extraction blocks suitable for industrial defect detection. These blocks enhance the receptive field, capture rich contextual information, and effectively prevent severe overfitting on small datasets while minimizing computational costs.

3) our architecture achieves remarkable results on different datasets (NEU-seg, DAGM-seg). More specifically, it demonstrates competitive performance against classic large models such as U-Net (with 31.39M parameters and 42.75G FLOPs) on NEU-seg and DAGM-seg, requiring only 0.39M parameters and 0.44G FLOPs.

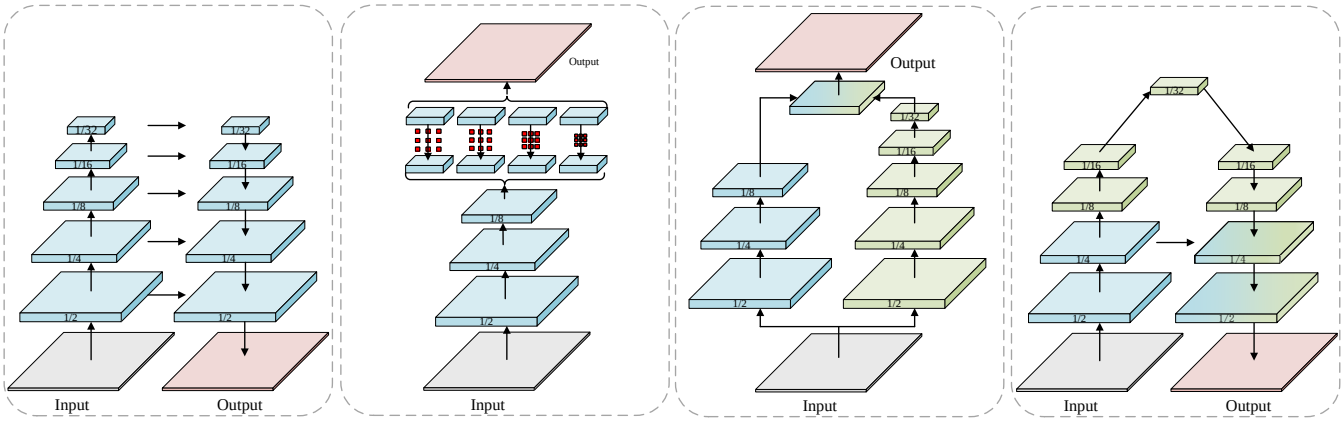
II. Related Work

In recent years, notable progress has been made in the realm of industrial surface defect segmentation. This section centers its examination on three primary categories of methodologies that are particularly germane to our work, specifically generic semantic segmentation, lightweight architectures and real-time semantic segmentation techniques, as well as industrial surface defect segmentation.

A. Generic Semantic Segmentation

With the introduction of the full convolutional network (FCN [3]), methods based on this framework have continuously pushed the state-of-the-art performance on various benchmarks. Currently, the mainstream FCN [3] structures are encoder-decoder structures, as depicted in Figure 2(a). The down-sampling stage captures information of different scales in the input image, while the up-sampling stage recovers the feature map resolution and maps it into semantic segmentation output. To enhance the performance of the encoder-decoder structure, most high-performing semantic segmentation networks employ a horizontal connection structure. For example, U-Net [5] uses a concatenate operation to connect feature maps with the same resolution in the encoder and decoder and then aggregates information in different channels through convolution operation. SegNet [4] uses a method to save maximum pooled coordinates to guide up-sampling. RefineNet [16] performs up-sampling of the encoder's feature map using multipath refinement. DFN [17] employs a channel attention module to merge the backbone network and recover details.

In addition, DeepLab [9] adopts cavity convolution of different sizes at the decoder stage to upsample the feature map obtained from the encoder stage to the same resolution and aggregate to fuse feature information of different scales, which



(a) Encoder-Decoder Backbone (b) Pyramid Pooling Backbone (c) Bilateral Segmentation Backbone (d) A-shaped Lightweight Backbone

Fig. 2. Different backbones.

shows in Figure 2(b). HRNet [18] utilizes multiple branches to maintain high resolution for higher precision segmentation.

Recently, in order to pursue higher performance, some researchers have introduced transformer [19] in the field of natural language processing (NLP) into visual tasks. The original representative of vision transformer is the ViT [20] model for image classification proposed by Dosovitskiy et al. Its basic idea is to divide the image into several patches and simultaneously input it into the network and convert it into a sequence for operation, so that the perception field can be expanded into the whole image. It improves the ability of the network to extract the overall features of the image, and finally builds a network model suitable for visual tasks. Swim-transformer [21] module is proposed on the basis of ViT to further optimize the attention mechanism. After that, Zheng et al. proposed the first VIT-based image segmentation representative model SERT [22], which realized end-to-end image segmentation by adding PUP and MLA upsampling modules. Cao et al. proposed Swim-Unet [23] for image segmentation task and replaced the convolutional layer in unet with swim transformer block to further improve the performance.

However, these architectures predominantly rely on operations with a high number of parameters and computational overhead. Consequently, the majority of such networks are characterized by a considerable size and low inference speed.

B. Light-weight and Real-time Semantic Segmentation

With the advancement of deep learning, numerous large-scale network models have been proposed. However, due to their high parameter count and computational overhead, it has become challenging to meet the stringent requirements of real-world applications that demand prompt response times. Consequently, researchers have recently shifted their focus towards neural network algorithms that exhibit lightweight and real-time characteristics. Among these, ENet [13] stands out as the pioneer work that emphasizes convolutional neural network efficiency. This network adopts an encoder-decoder structure, employs maximum pooling coordinates to guide upsampling, and achieves an extremely high reasoning speed. Similarly, ICNet [14] leverages image concatenation strategy to accelerate the network's reasoning speed. ERFNet [15] incorporates residual connections and factorized convolutions to ensure accuracy while improving efficiency. ESNet [24] employs the decomposition of convolutional units and other lightweight convolutional operations to construct a symmetrical structure real-time semantic segmentation network. Finally, DFANet [25] utilizes feature repetition to decrease network complexity while preserving feature expression.

Despite the ability of above-mentioned networks to achieve a lightweight network structure or real-time inference speed, the aforementioned lightweight or real-time semantic segmentation networks often entail a trade-off between performance and the segmentation capability of small-scale features. This is due to their inability to effectively attend to both low-level details and high-level semantic information simultaneously [11].

To address the aforementioned challenges, BiSeNetV2 [11] introduces a Bilateral Segmentation Backbone as illustrated in Figure 2 (c). This architecture incorporates both detailed and semantic branches during the sampling phase to enable the simultaneous extraction of corresponding information, which is subsequently aggregated and directly upsampled to the output resolution. Despite BiSeNetV2's ability to perform real-time semantic segmentation via GPU-accelerated computing while simultaneously extracting details and semantic information, the network's parameter count and computational requirements remain significant due to the existence of two subsampling branches. Thus, its deployment on industrial edge devices without GPU-accelerated computing is not viable.

Besides, topformer [26] multiscale tokens through pyramid structure, and then integrate tokens of different scales. This method reduces the number of parameters and computational complexity in transformer, and improves the inference speed of the network. However, due to the large amount of data required for its training, it is not applicable to the field of industrial defect detection.

To address these limitations, this paper proposes the A-Net structure illustrated in Figure 2 (d). A-Net employs a specially

designed feature extraction module to realize a lightweight network structure and real-time reasoning while aggregating detailed information and semantic information through a single jump connection.

C. Industrial Surface Defect Segmentation

The segmentation of industrial surface defects based on neural networks has garnered increasing attention with the development of deep learning. In recent years, full convolutional neural network-based methods for industrial surface defect segmentation have emerged continuously. For example, Wang et al. [27] proposed an FCN-based method for refining and segmenting defects in tire images by fusing multi-scale sampling layer feature maps, while Yu et al. [28] developed a multi-stage FCN method to achieve more precise defect segmentation. Moreover, MCuePush Unet [29] employs a three-channel image output of MCue module as U-Net input to improve defect segmentation performance, while FL-SegNet [30] combines the original SegNet network with a Focal loss function to segment multiple defects in tunnel lining. DeepCrack [31], based on SegNet, fuses multi-scale deep convolution features learned at hierarchical convolution stages to capture fine crack structures. Finally, PGANet [6] introduces a pyramid feature aggregation and global context attention network to achieve better defect segmentation performance. The aforementioned networks for surface defect segmentation can effectively achieve precise segmentation of specific defects. However, their network architectures are large and require high computational resources, making their deployment and real-time inference at the edge costly. In contrast, the A-Net proposed in this study employs a specially designed network architecture and feature extraction module to achieve a lightweight network structure and real-time inference while maintaining sufficient defect segmentation performance.

III. A-shaped Lightweight and Realtime Networks

A. Overview

As depicted in Figure 3, the proposed lightweight real-time industrial defect segmentation network is of A-shaped architecture, hence named A-Net. A-Net is comprised of two distinct parts, namely feature extraction and feature fusion. The feature extraction stage is composed of two stages: detail extraction and semantic extraction. These stages employ different stacking modes of down-sampling module (Down Block) and lightweight feature extraction module (Light Block and Wide Block). The aim of detail extraction is to extract low-level detailed information more effectively, whereas the goal of semantic extraction is to capture high-level semantic information more precisely. The feature fusion stage employs alternately stacked up-sampling module (Up Block) and lightweight feature extraction module (Light Block and Wide Block) to achieve refined feature recovery. Further, we aggregate low-level detailed information with high-level semantic information through a jump connection structure specially designed for this purpose. Finally, the segmentation output is obtained through the process of up-sampling, feature fusion, and seg head.

B. Motivation

To achieve a lightweight network structure capable of real-time inferencing on edge devices, it is necessary to minimize the number of parameters and computational complexity of the network. The computational complexity of the network is represented by FLOPs (Floating Point Operations).

Industrial defect images present a challenge to semantic segmentation networks due to the varying sizes and shapes of defect regions. To address this challenge, we integrate detail extraction and semantic extraction in the feature extraction stage and aggregate the extracted information via a jump connection after up-sampling. This approach enables the network to focus on information of different scales in the image simultaneously while maintaining a low parameter number and FLOPs, leading to high-precision semantic segmentation of industrial surface defects.

When the dataset size is small, deep full convolutional neural networks are susceptible to the issues of gradient disappearance and explosion, which can lead to ineffective convergence. Therefore, we designed a lightweight feature extraction convolutional operation with a residual connection structure to address these issues. Additionally, we adopted different convolutional operation block stacking modes in different feature extraction stages to further expand the receptive field of the semantic extraction stage. As a result, A-Net achieves effective extraction of low-level details and high-level semantic information with an extremely low parameter number and FLOPs.

Furthermore, to improve the performance of industrial surface defect segmentation and address the issue of indistinct boundaries between defect and non-defect regions, we have incorporated a staggered design of up-sampling and convolution operation blocks in our feature fusion stage. Nevertheless, this design imposes additional computational overhead. Hence, we have integrated lightweight feature extraction convolution operations, namely Light Block and Wide Block, in both feature extraction and feature fusion stages to mitigate the computational complexity. This approach strikes a balance between computational efficiency and segmentation accuracy, enabling our proposed network to achieve high-precision industrial surface defect segmentation.

C. Feature Extraction

This section presents a detailed description of the Down Block and two lightweight feature extraction block (Light and Wide Block). The feature extraction stage is comprised of two stages: detail extraction and semantic extraction. For the detail extraction stage, we utilize the stacking of Down Block, Light Block, and Wide Block. On the other hand, to rapidly expand the receptive field in the semantic extraction stage, we use the stacking mode of Down Block, two Light Blocks, and two Wide Blocks. The various blocks are elaborated below.

1) Down Block: To address the issue of vanishing or exploding gradients that may arise in deep neural networks, we incorporate a residual connection architecture within the Down Block. As input and output sizes vary, both branches necessitate sampling during down-sampling. For a lightweight

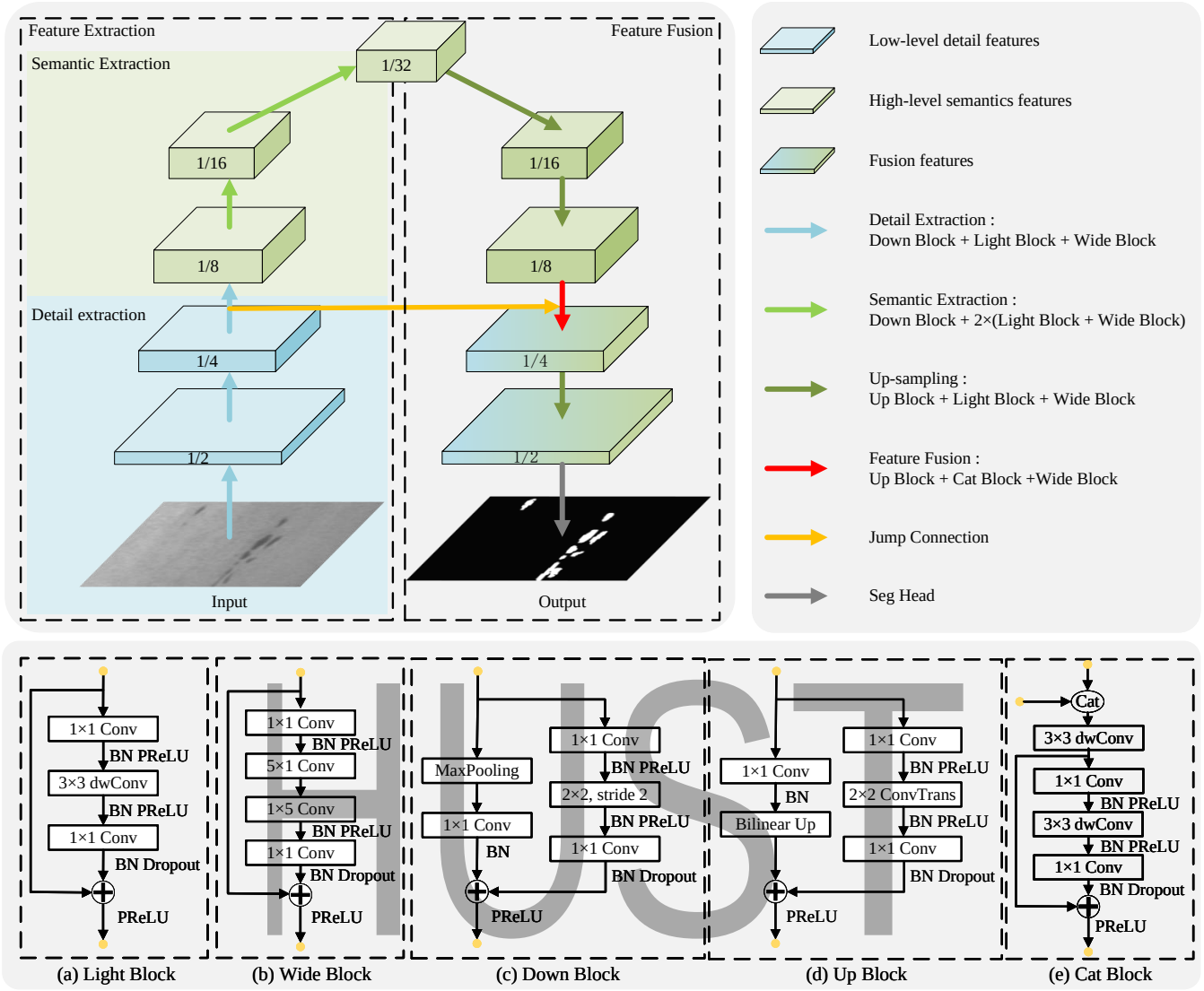


Fig. 3. The architecture of proposed network. During the Feature Extraction stage, the feature maps with darker colors correspond to higher levels of information. Conversely, in the Feature Fusion stage, feature maps with darker colors correspond to a greater degree of detailed information recovery.

design, we apply point-wise convolution to condense the primary channel, followed by a 2x2 convolution with a stride of 2 for down-sampling the feature map, and then another point-wise convolution to expand the channel count. Meanwhile, the residual channel utilizes max-pooling for down-sampling. To integrate the distinct information from both branches, we merge their sampled outputs and apply the PReLU activation function, yielding the final output of the sampling module. Figure 3 illustrates the Down Block structure, while Table 1 provides further details, including the gradual increase in input and output channels throughout down-sampling and the intermediate channel count being set to one-fourth of the output channel quantity.

2) Light Block and Wide Block: The feature extraction module is a vital element of a semantic segmentation neural network, significantly impacting its training convergence and dataset performance. However, standard convolution operations involve substantial computational demands. To maintain a lightweight structure while enabling the network to extract features from images with an extensive receptive field, we substitute traditional convolutions with lightweight convolution operations, such as depthwise separable convolution, point-wise convolution, and factorized convolution. This reduction in network parameters diminishes computational complexity. Although dilated convolution can expand the receptive field without increasing parameter count and computational complexity, its inferior computational efficiency results in a higher inference delay; thus, we exclude it from our network design.

In pursuit of network lightness, we devise two unique feature extraction modules with varying receptive field sizes. The first module, dubbed Light Block, consists of a depthwise separable convolution between two point-wise convolutions and employs a residual connection. This module, with a 3x3 receptive field, is optimized for computational efficiency. The second module, termed Wide Block, adopts factorized convolution (5x1 and 1x5) instead of the traditional 5x5 convolution,

as depicted in Figure 4, enabling a larger 5x5 receptive field. Analogous to the Light Block, the Wide Block is flanked by two point-wise convolutions and incorporates a residual connection. Figure 3 showcases the specific architectures of these feature extraction modules.

Our proposed feature extraction module exhibits a symmetric channel structure, maintaining an equal number of input and output channels. The initial point-wise convolution reduces the channel count to 1/4 of the output channels, followed by depthwise separable convolution or factorized convolution with an equal number of input and output channels to expand the receptive field. Subsequently, the latter point-wise convolution increases the channel count to achieve the desired output channel dimension. This channel design effectively mitigates the computational complexity arising from large convolution kernels. Table 1 provides more detailed channel configurations.

Using an input size of 32x112x112 and an output size of 32x112x112 as an example, with the intermediate channel count set to 1/4 of the output channel count, we compute the parameter quantity and FLOPs of the feature extraction module and compare them to those of a standard convolution operation. The specific formulas for calculating the parameter quantity and FLOPs of common convolution operations are as follows (bias is not considered):

$$Params_{Conv} = K_h \times K_w \times C_{in} \times C_{out} \quad (1)$$

$$FLOPs_{Conv} = \frac{2K_h \times K_w - 1}{g} \times C_{in} \times F_h \times F_w \times C_{out} \quad (2)$$

Where, C_{out} and C_{in} represent the number of output and input channels for the convolution, respectively. K_h and K_w denote the height and width of the convolution kernel, while F_h and F_w represent the height and width of the feature map. k corresponds to the size of the convolution kernel, and g stands for the number of convolution groups.

For depthwise separable convolution, it can be considered as a standard convolution with the number of groups g = $K_h \times K_w$, and the number of input and output channels being C_{in}. Additionally, it includes the standard 1x1 convolution. Thus, the specific formula for calculating the parameter quantity and FLOPs is as follows (excluding bias consideration):

$$Params_{dwConv} = C_{in} \times (K_h \times K_w) + C_{in} \times C_{out} \quad (3)$$

$$FLOPs_{dwConv} = (2K_h \times K_w - 1) \times F_h \times F_w \times C_{out} + C_{in} \times F_h \times F_w \times C_{out} \quad (4)$$

Upon calculating the above parameters, we observe that the 3x3 standard convolution operation contains 9.22k parameters and 115.61M FLOPs, while the Light Block only has 0.75k parameters and 9.93M FLOPs. Similarly, the 5x5 standard convolution operation has 25.6k parameters and 309.76M FLOPs, compared to the Wide Block, which only has 1.25k parameters and 16.26M FLOPs. Consequently, our designed feature extraction module significantly reduces the parameter count and FLOPs while retaining the same receptive field size as the standard convolution.

Considering feature extraction at multiple scales, our module is designed to accommodate receptive fields of 3x3 and 5x5. By utilizing various stacking configurations of feature extraction modules during different stages of down-sampling (Detail Extraction and Semantic Extraction), we can effectively control the receptive field size for each pixel in the feature map at different stages. This approach enables efficient extraction of both low-level details and high-level semantics according to our requirements.

Additionally, we employ several strategies to improve the performance of our module. In particular, we incorporate the residual connection approach, embed the Dropout layer, and implement the PReLU (Parametric Rectified Linear Unit) function for activation before combining the input and output of the feature extraction module. The PReLU activation function is expressed as follows:

$$PReLU(x_i) = \begin{cases} x_i & if x_i > 0 \\ a_i x_i & if x_i \leq 0 \end{cases} \quad (5)$$

Where, a is the parameter obtained through training.

The residual connection effectively tackles the issues of gradient explosion or vanishing gradients that can occur in deep networks, facilitating efficient convergence of the network on small datasets. Incorporating the Dropout layer within the feature extraction module also helps prevent overfitting on small datasets. Moreover, the PReLU activation function introduces increased flexibility to the network without substantially augmenting the parameter count or computational overhead, thus further optimizing the performance of the feature extraction module.

As mentioned earlier, our two lightweight feature extraction modules are capable of effectively extracting features from images at different stages of the network using specific combinations.

TABLE I
THE NUMBER OF CHANNELS IN EACH STAGE OF A-NET. (THE *opr* REPRESENTS DIFFERENT OPERATIONS AT DIFFERENT STAGES, THE INPUT REPRESENTS INPUT IMAGE, THE DOWN AND THE UP REPRESENT DOWNSAMPLING MODULE AND UPSAMPLING MODULE RESPECTIVELY, THE LFEM REPRESENTS LIGHTWEIGHT FEATURE EXTRACTION MODULE (INCLUDING LIGHT BLOCK, WIDE BLOCK AND CAT BLOCK), THE C_{in} REPRESENTS THE NUMBER OF INPUT CHANNELS, THE C_m REPRESENTS THE NUMBER OF INTERMEDIATE CHANNELS, THE C_{out} REPRESENTS THE NUMBER OF OUTPUT CHANNELS, AND THE OUTPUT SIZE REPRESENTS THE RESOLUTION OF OUTPUT FEATURE GRAPH OF EACH MODULE.)

Stage	Downsampling				Upsampling				Output Size
	opr	C_{in}	C_m	C_{out}	*opr*	C_{in}	C_m	C_{out}	
S_0	Input			3	Seg Head	32	16	1	224×224
S_1	Down	3	8	32	LFEM	32	8	32	112×112
	LFEM	32	8	32	Up	64	8	32	112×112
S_2	Down	32	16	64	LFEM	64	16	64	64×64
	LFEM	64	16	64	Up	128	16	64	64×64
S_3	Down	64	32	128	LFEM	128	32	128	32×32
	LFEM	128	32	128	Up	128	32	128	32×32
S_4	Down	128	32	128	LFEM	128	32	128	16×16
	LFEM	128	32	128	Up	128	32	128	16×16
S_5	Down	128	32	128					8×8
	LFEM	128	32	128					8×8

D. Feature Fusion

During the feature fusion stage, we utilize a stacking configuration consisting of up blocks, light blocks (or cat blocks), and wide blocks in an interleaved manner to accomplish fine feature recovery and feature fusion. In the final up-sampling step, we develop a simple Seg Head to map the up-sampled features to segmentation output.

1) Up Block: Two prevalent methods for up-sampling are interpolation up-sampling and deconvolution. To address the issues of vanishing or exploding gradients in deep networks, we adopt the residual connection structure in our up block, as detailed in section 3.3.1. This approach involves creating two branches using deconvolution and bilinear up-sampling operations, and implementing channel compression through point-wise convolution before deconvolution. By expanding the number of channels after deconvolution, a lightweight sampling module is constructed. The outputs of the two branches are then summed and activated by PReLU. During the up-sampling process, the number of output channels gradually decreases, with the number of intermediate channels set at 1/4 of the output channel count. Table 1 presents the channel settings.

Moreover, in the up-sampling process, we merge low-level details with high-level semantics after up-sampling through a jump connection at 1/4 size of input, as it is the boundary between the detail extraction stage and the semantic extraction stage. First, we concatenate the feature map obtained from up-sampling high-level semantics with the details extracted during the detail extraction stage. Subsequently, we utilize lightweight depthwise separable convolution to compress the channel count and integrate the spatial information across different channels. This combined feature map is then input into the Light Block for further feature fusion and extraction operations. Figure 3 illustrates the specific architecture of this process.

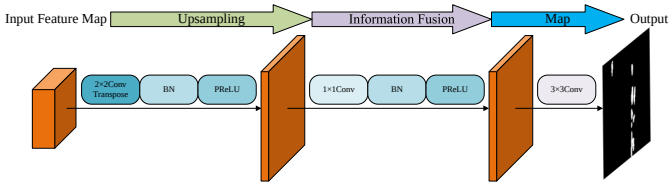


Fig. 4. Seg head architecture.

2) Seg Head: In the final up-sampling stage of our network, we have designed a straightforward segmentation head. This segmentation head consists of a deconvolution layer, a point-wise convolution layer, and a 3x3 standard convolution layer, as depicted in Figure 5. The deconvolution layer is responsible for up-sampling the feature map, initially half the size of the input image, while simultaneously reducing the number of channels. The point-wise convolution layer serves to integrate spatial information from various channels of the up-sampled feature map. Lastly, the 3x3 standard convolution layer maps the feature map into the desired segmentation output, thereby completing the entire network computation process.

E. Loss Function and training

To further enhance network performance, the loss function formula used in the training process is as follows:

$$L_{weight}(p_d, g_d)=L_{dice}(p_d, g_d) + 0.5 \times L_{bce}(p_d, g_d) \quad (6)$$

where $p_d \in \mathbb{R}^{H\times W}$ denotes the predicted pixel and $g_d \in \mathbb{R}^{H\times W}$ denotes the corresponding pixel of ground-truth. Additionally, L_{bce} represents the binary cross-entropy loss, while L_{dice} represents the dice loss, which is given as follows:

$$L_{dice}(p_d, g_d) = 1 - \frac{2\sum_i^{H\times W} p_d^i g_d^i + \varepsilon}{\sum_i^{H\times W} (p_d^i)^2 + \sum_i^{H\times W} (g_d^i)^2 + \varepsilon} \quad (7)$$

Moreover, we have not employed a complex training method to train A-Net. Instead, we have utilized a simple gradient descent method to train A-Net without incorporating any auxiliary training strategies.

IV. Experiments

In this section, we begin by introducing the industrial surface defect dataset, our experimental setup, and the evaluation metrics employed. Next, we carry out ablative experiments to examine the impact of our designed components on network performance. We then perform a comparative analysis of the performance and network structure lightness of our proposed method relative to other state-of-the-art algorithms on different datasets. Lastly, we assess the computational efficiency of our proposed lightweight networks on the CPU platform, followed by a comprehensive analysis and comparison of the results.

A. Datasets, Settings, and Evaluation Metrics

1) Datasets: In this article, we have selected two distinct surface defect datasets, namely the NEU-DET defect dataset and the DAGM defect dataset, to substantiate and evaluate the applicability and generality of our proposed method.

NEU-Seg Dataset: The NEU dataset is a standard dataset collected by [32] to address the problem of automatic recognition for hot-rolled steel strips. The dataset includes six types of strip steel plates, comprising patch, crazing, pitted-surface, inclusion, scratches, and rolled-in scale, with each surface defect containing 300 images. The original resolution of the images in the dataset is 200×200, and all have corresponding defect type labels. We selected three surface defects (inclusion, patches, and scratches) for pixel-level marking. We then adjusted their resolution to 224×224 and divided them into training sets and test sets, containing 250 and 50 images, respectively, to enable their application to our industrial defect image segmentation.

DAGM-Seg Dataset: The DAGM dataset [33] is manually generated and contains multiple types of industrial surface defect images with an original resolution of 512x512. We chose categories 7 through 10, encompassing a total of 4 datasets, and then divided them into training sets and test sets, containing 250 and 50 images, respectively.

2) Settings: **Training**: To ensure fairness, all models are trained from scratch. We employ the stochastic gradient descent (SGD) algorithm with a learning rate of 0.0003 and a momentum of 0.9 to train all models. For the NEU-Seg datasets, we adopt a batch size of 16, while for the DAGM-Seg datasets, we use a batch size of 4. The weight decay is set at 0.0001. Moreover, we train all models for 2000 iterations for all the datasets.

Data augmentation: Images are randomly rotated by 90° and randomly flipped during training to expand the training set and prevent severe overfitting.

Evaluation: When testing network performance, we employ the simplest and fastest method, which involves directly loading the test data to assess the performance of every model after training.

Setup: We conduct experiments using PyTorch 1.9.0, and all models are evaluated on a single NVIDIA GeForce GTX 1080Ti with CUDA 11.7, CUDNN 8.5, and TensorRT 8.5.3.

3) Evaluation Metrics: In order to evaluate the model performance and complexity more comprehensively, we use the Intersection over Union (IoU) index of segmentation results to assess the model performance and the number of model parameters and FLOPs to evaluate model complexity and computational consumption. The IoU is represented as a percentage, with higher IoU values indicating better model performance. The calculation formula is as follows:

$$IoU = \frac{TP}{TP + FN + FP} \tag{8}$$

True Positives (TP) refers to positive predictions that match the ground truth. False Negatives (FN) represent negative predictions that do not match the ground truth. False Positives (FP) denote positive predictions that do not match the ground truth.

Additionally, the number of model parameters is the sum of the number of parameters for all operations in the model, and its unit is typically expressed in megaParams (M). The calculation formula for the number of parameters of a single convolution operation is shown in Equation (1). The fewer the number of model parameters, the lower the model complexity. The model FLOPs is the sum of FLOPs of all operations in the model, with the unit generally being gigaFLOPS (G). The calculation formula for FLOPs of a single convolution operation is shown in Equation (2). The lower the FLOPs of the model, the lower the computational consumption. Therefore, a lightweight model requires that the number of network parameters and FLOPs be maintained at a low level.

B. Ablative Experiments

In this section, a comprehensive analysis of the lightweight nature and feature extraction capability of the proposed Light Block and Wide Block architectures is conducted by replacing them with 3×3 convolution and 5×5 convolution layers. Subsequently, the jump connection aggregation structure and the final split header structure are incorporated into the network architecture in a step-by-step manner. By systematically examining the network's performance with varying degrees of ablation and conducting a thorough evaluation of the number of network parameters and FLOPs, the efficacy and lightweight advantages of the proposed components are effectively demonstrated. The outcomes of the ablation experiments are presented in Table 2.

TABLE II
ABLATIVE EXPERIMENTS ON THE NEU-SEG DATASET. THE NUMBERS UNDER NEU-INCLUSION, NEU-PATCHES, AND NEU-SCRATCHES REPRESENTS IOU (%) OF MODELS ON CORRESPONDING DATASET.

Light Block	Wide Block	Jump Connection	Seg Head	NEU-inclusion	NEU-patches	NEU-scratches	Parameters(M)	FLOPs(G)
				47.13	75.70	55.44	4.41	3.12
√				46.43	76.08	56.10	3.36	2.39
	√			52.71	76.08	55.79	1.43	1.02
√	√			51.28	77.65	54.94	0.38	0.28
√	√	√		56.77	77.04	59.30	0.39	0.31
√	√	√	√	**60.53**	**78.76**	**59.51**	0.39	0.44

The results presented in Table 2 demonstrate that the proposed A-Net backbone yields commendable segmentation performance and maintains a low parameter count and FLOPs simultaneously, even in the absence of specifically designed lightweight feature extraction modules, jump connections, and segmentation headers. Upon incorporating the proposed lightweight feature extraction structure, the model's parameter count and FLOPs are reduced by more than 90%, compared to the ordinary convolutional model, with a slight increase in performance. This outcome validates the effectiveness of the lightweight feature extraction module proposed in this study, which uses special convolutions, dropout, and residual connection rationally to adapt the network to different industrial surface defect detection datasets. Upon adopting the jump connection structure, the model's performance is significantly improved while only adding a few parameters and FLOPs.

Following the integration of the Seg Head structure proposed in this study, the network's performance on the NEU-inclusion dataset is notably enhanced, while a slight performance improvement is observed on other datasets. This observation substantiates the efficacy of the Seg Head structure proposed in enhancing the model's generalization ability across various datasets. Finally, from the perspective of model lightness, the A-Net model structure's parameter count determined in this study is only 0.39M, and FLOPs are only 0.44G, thereby satisfying the deployment requirements of edge devices (FLOPs lower than 0.6G).

C. Comparative Experiments

We ended up choosing six classical segmentation networks (FCN [3], SegNet [4], PSPNet [8], DeeplabV3+ [34], RefineNet [16], and U-Net [5]), a network designed for industrial image segmentation (PGA-Net [6]), and seven light networks (BiSeNet [10], BiSeNetV2 [11], STDC [35], ERFNet [15], ESNet [24], ENet [24], and Topformer [26]) that performs well in natural images as the baseline network to compare with our network.

TABLE III
PERFORMANCE OF DIFFERENT METHODS AND OUR METHOD ON THE NEU-SEG DATASET. THE NUMBERS UNDER NEU-INCLUSION, NEU-PATCHES, AND NEU-SCRATCHES REPRESENTS IOU (%) OF MODELS ON CORRESPONDING DATASET.

methods	NEU-inclusion	NEU-patches	NEU-scratches	Parameters(M)	GFLOPs
classical models					
FCN	52.55	78.78	55.39	45.47	16.00
SegNet	56.61	79.84	58.05	29.44	30.73
PSPNet	54.06	79.80	57.78	53.32	38.71
DeepLabV3+	57.49	78.34	55.07	59.34	16.97
RefineNet	58.72	**79.91**	60.02	80.22	161.30
UNet	58.97	79.55	**60.92**	31.39	42.75
PGANet	31.12	66.89	24.91	51.41	315.69
light models					
BiSeNet	54.70	79.13	56.71	12.40	4.14
BiSeNetV2	11.23	57.83	20.16	4.95	1.91
STDC	50.43	76.84	54.74	12.04	2.97
ERFNet	58.80	77.77	33.61	2.08	2.82
ESNet	59.73	78.84	58.71	1.66	2.58
ENet	58.21	78.57	59.43	**0.35**	0.37
TopFormer	54.56	76.13	53.24	3.00	**0.24**
A-Net	**60.53**	78.76	59.51	0.39	0.44

1) NEU-Seg Dataset: Table 3 presents the performance of each baseline network and the A-Net proposed in this paper on the NEU-Seg dataset.

The analysis of various segmentation network performances in the table reveals that larger models generally achieve higher IoU scores than smaller models. In comparison with larger models, the A-Net proposed in this paper achieves the highest IoU on the NEU-inclusion dataset and is only 1.15% away from the highest IoU on the NEU-patches dataset. However, the performance of A-Net on the scratches dataset ranks third among all methods in the table. These results demonstrate that the A-Net proposed in this paper exhibits excellent performance on the industrial surface defect dataset.

In terms of network lightweightness, the A-Net proposed in this paper achieves a remarkable advantage over large models concerning the number of parameters and FLOPs. Specifically, A-Net's parameter quantity is only 1.32% of SegNet, the network with the minimum parameters among the large models, and its FLOPs are only 2.75% of the FLOPs of FCN, the network with the lowest FLOPs among the large networks. Compared to small models, A-Net's number of parameters and FLOPs are only slightly higher than those of ENet and lower than other small models. Furthermore, it is evident that the segmentation performance of A-Net surpasses that of other small models. The A-Net architecture successfully achieves the design goal of a lightweight network structure, thereby attaining the best precision-lightweightness balance on the NEU-Seg dataset.

Figure 6 displays the visual segmentation outputs of each comparative network on the NEU-Seg dataset. The results demonstrate that A-Net not only accomplishes efficient defect segmentation but also exhibits noteworthy proficiency in detecting defects of diverse scales. Furthermore, A-Net manifests impressive boundary segmentation capabilities. These accomplishments can primarily be attributed to the network backbone and the lightweight feature extraction module devised by the authors. This module comprises phased feature extraction stages and feature fusion stages, which enables the network to effectively extract and resolve features of varying scales.

TABLE IV
PERFORMANCE OF DIFFERENT METHODS AND OUR METHOD ON THE DAGM-SEG DATASET. THE NUMBERS UNDER DAGM-CLASS7, DAGM-CLASS8, DAGM-CLASS9, AND DAGM-CLASS10 REPRESENT IOU (%) OF MODELS ON CORRESPONDING DATASET, WHILE "-" REPRESENTS THAT THE MODEL CANNOT CONVERGE EFFECTIVELY ON THE CORRESPONDING DATASET.

methods	DAGM-class7	DAGM-class8	DAGM-class9	DAGM-class10	FLOPs (G)
classical models					
FCN	79.16	43.54	74.89	52.70	83.61
SegNet	80.62	69.40	86.75	73.04	160.52
PSPNet	81.21	71.00	86.99	72.34	201.35
DeepLabV3+	81.09	70.73	88.22	73.89	88.63
RefineNet	81.06	70.70	87.93	73.73	842.74
UNet	82.74	**76.16**	**88.37**	**77.79**	223.34
PGANet	81.07	61.76	82.55	65.65	1649.33
light models					
BiSeNet	80.64	-	-	-	21.62
BiSeNetV2	62.82	-	-	-	9.96
STDC	78.93	48.84	84.68	59.15	15.54
ERFNet	49.87	-	-	-	14.74
ESNet	82.02	74.51	88.12	75.36	13.48
ENet	79.47	-	-	-	1.94
TopFormer	80.73	68.37	85.76	72.17	**1.22**
A-Net	**82.86**	75.99	88.03	77.01	2.30

2) DAGM-Seg Dataset: Table 4 presents the performance of each baseline network, as well as the A-Net model proposed in this study, on the DAGM-Seg dataset.

The convergence performance of the comparison networks indicates that A-Net exhibits commendable convergence performance like large models and can effectively converge on the DAGM-Seg dataset even with a limited number of images (i.e., 250 images). In contrast, most of the smaller comparative models are unable to converge effectively on DAGM-class8, DAGM-class9, and DAGM-class10. Hence, the A-Net model surpasses its smaller counterparts by demonstrating superior convergence capabilities for datasets of smaller magnitudes.

Analysis of the performance of various split networks, as presented in the table, reveals that larger models generally achieve higher IoU values than smaller models. However, among the models compared, the A-Net proposed in this study outperforms all others by achieving the highest IoU on the DAGM-class7 dataset. Additionally, A-Net's performance on the DAGM-class8 and DAGM-class10 datasets ranks second among all models in the table, trailing only the U-Net in the larger model category. Despite this, the IoU attained by A-Net on the DAGM-class9 dataset is only 0.34% lower than the highest IoU recorded. These results attest to the exceptional performance of the A-Net model on industrial surface defect datasets.

Considering the extremely low parameter quantity and FLOPs of A-Net, the proposed A-Net segmentation network achieves the best precision-lightweight tradeoff on the DAGM-Seg dataset. This demonstrates the effectiveness of the A-Net model in addressing the challenges posed by industrial surface defect segmentation tasks while maintaining a lightweight architecture suitable for deployment on edge devices.

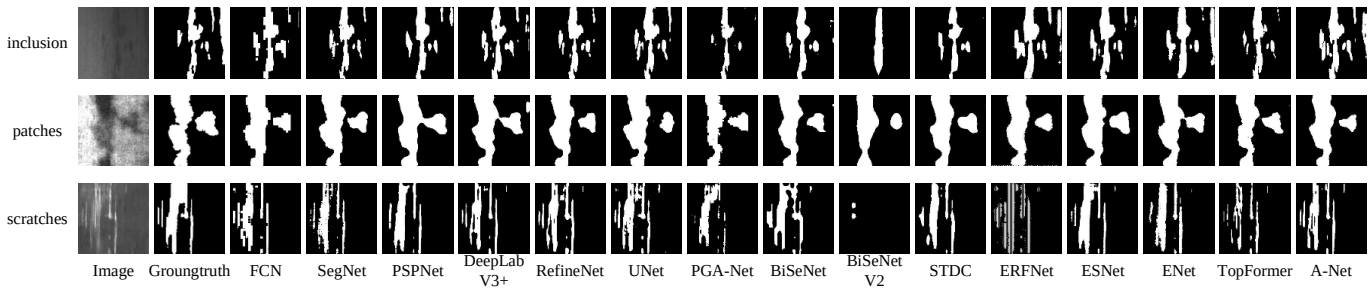


Fig. 5. The visual display of the results of every network on the NEU-Seg Dataset.

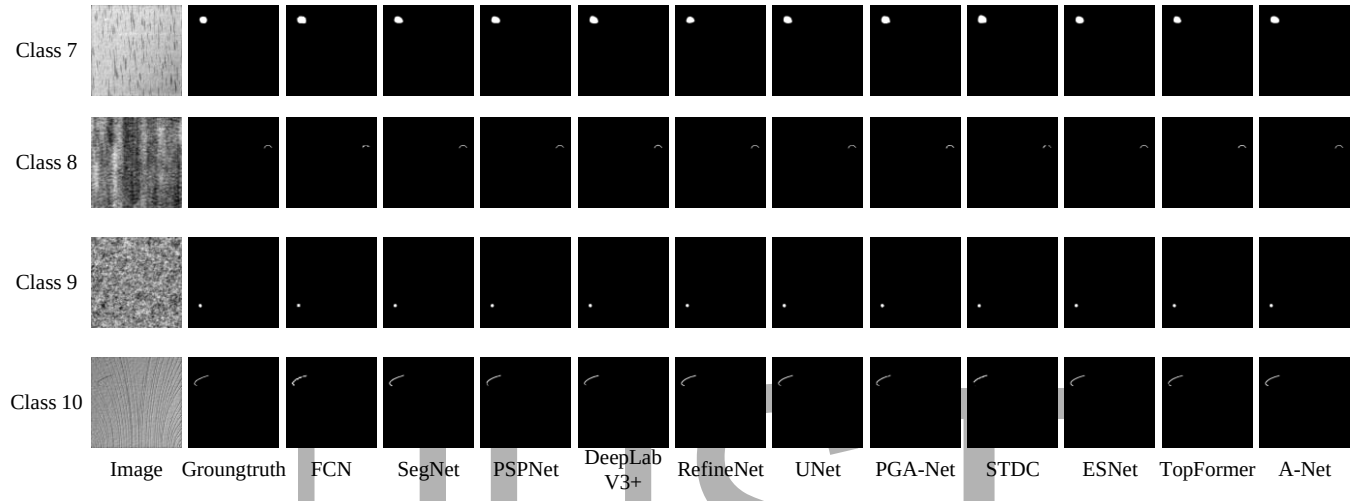


Fig. 6. The visual display of the results of every network on the DAGM-Seg Dataset.

Figure 7 displays the visual segmentation outputs of each comparative network on the DAGM-Seg dataset, except for networks that can not converge effectively.

D. Inference Speed Test on Edge Devices

To better simulate model deployment at the industrial edge and explore the inference speed of the model without GPU acceleration, we use the Benchmark Python Tool in OpenVINO [36] to test the inference speed of BiSeNet, BiSeNetV2, STDC, ERFNet, ESNet, TopFormer, ENet, and A-Net on two edge devices which are CPU-based platforms (Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz in windows and Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz in Ubuntu18.04). The test is set as follows: The input image size is 3 x 224 x 224, the batch size is 1, and test epoch number is 5000.

The results obtained from the test, as shown in Figure 7, demonstrate that the A-Net model proposed in this study outperforms other models in terms of inference speed on both Windows and Linux systems. The slower inference speed on Windows systems can be attributed to the greater number of irrelevant processes competing for system resources.

Despite this, the proposed A-Net architecture achieves inference speeds that are several times faster than those of real-time or lightweight semantic segmentation networks, such as BiSeNet, BiSeNetV2, ERFNet, and ESNet, when running on a CPU. Additionally, A-Net approaches the inference speeds of the lightest network, ENet, on CPU, thereby establishing its superiority over competing models.

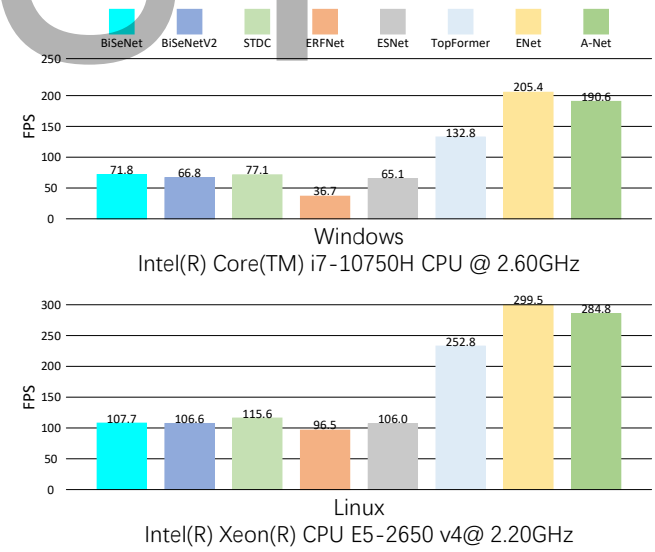


Fig. 7. Results of inference speed test on edge devices.

These results confirm the effectiveness of the A-Net model for deployment on industrial edge devices, where high-speed inference and lightweight architecture are crucial for real-

time processing and analysis of industrial surface defects. By outperforming other state-of-the-art models, A-Net proves to be a suitable solution for addressing the challenges associated with industrial edge computing.

E. Experiment Conclusion

Based on the performance and inference speed tests conducted in sections 4.2, 4.3, and 4.4, along with the analyses of parameter numbers and FLOPs, and the inference FPS(Frames Per Second) tests performed on edge devices, it is evident that the A-Net network structure proposed in this study demonstrates competitive performance on various industrial surface defect segmentation datasets when compared to larger semantic segmentation network models.

In addition, A-Net boasts an impressively low parameter count and FLOPs, while also achieving high inference speeds on CPU platforms. These attributes contribute to the lightweight nature and computational efficiency of the A-Net model, making it particularly well-suited for deployment on edge devices in industrial settings.

In conclusion, the A-Net network structure achieves an optimal balance between precision and speed compared to the other networks examined in this study. This balance makes it a promising solution for real-time detection and analysis of industrial surface defects, thereby addressing the challenges associated with industrial edge computing.

V. Conclusion

In this paper, we have presented A-Net, a lightweight and real-time network for industrial surface defect segmentation, specifically designed to address the challenges arising from limited data, varying defect sizes, irregular outlines, and subtle differences between defect and normal areas. The proposed A-shaped network structure consists of two main components, feature extraction and feature fusion, efficiently extracting low-level detail and high-level semantic information while facilitating the aggregation of information at different levels.

Through the design of lightweight convolution blocks, we have managed to prevent overfitting, gradient disappearance, and gradient explosion, making the network suitable for small datasets. Moreover, A-Net demonstrates competitive performance compared to classic large models, such as U-Net, while significantly reducing the number of parameters and computational costs and shows high inference speed without GPU acceleration.

Our work contributes to the ongoing development of effective and efficient defect segmentation networks, paving the way for real-world industrial applications with limited resources. Future research directions include further optimization of the network architecture, exploring additional lightweight approaches, and investigating the applicability of A-Net to other domains and tasks that require low-latency and computationally efficient models.

References

[1] W. Wang, C. Mi, Z. Wu, K. Lu, H. Long, B. Pan, D. Li, J. Zhang, P. Chen, and B. Wang, "A real-time steel surface defect detection approach with high accuracy," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022.

[2] H. Chen, Y. Du, Y. Fu, J. Zhu, and H. Zeng, "Dcam-net: A rapid detection network for strip steel surface defects based on deformable convolution and attention mechanism," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–12, 2023.

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[6] H. Dong, K. Song, Y. He, J. Xu, Y. Yan, and Q. Meng, "Pga-net: Pyramid feature fusion and global context attention network for automated surface defect detection," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 12, pp. 7448–7458, 2019.

[7] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6230–6239, 2016.

[9] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[10] 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)*, 2018, pp. 325–341.

[11] 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*, vol. 129, pp. 3051–3068, 2021.

[12] X. Lei, L. Lu, Z. Jiang, Z. Gong, C. Lu, J. Liang, and J. Xie, "Stdc-ma network for semantic segmentation," *IET Image Processing*, vol. 16, no. 14, pp. 3758–3767, 2022.

[13] 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.

[14] 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)*, 2018, pp. 405–420.

[15] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2017.

[16] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1925–1934.

[17] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1857–1866.

[18] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[22] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, "Rethinking semantic segmentation from a

sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6881–6890.

[23] H. Cao, Y. Wang, J. Chen, D. Jiang, X. Zhang, Q. Tian, and M. Wang, "Swin-unet: Unet-like pure transformer for medical image segmentation," in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*. Springer, 2023, pp. 205–218.

[24] Y. Wang, Q. Zhou, J. Xiong, X. Wu, and X. Jin, "Esnet: An efficient symmetric network for real-time semantic segmentation," in *Pattern Recognition and Computer Vision: Second Chinese Conference, PRCV 2019, Xi'an, China, November 8–11, 2019, Proceedings, Part II 2*. Springer, 2019, pp. 41–52.

[25] 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*, 2019, pp. 9522–9531.

[26] W. Zhang, Z. Huang, G. Luo, T. Chen, X. Wang, W. Liu, G. Yu, and C. Shen, "Topformer: Token pyramid transformer for mobile semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 083–12 093.

[27] R. Wang, Q. Guo, S. Lu, and C. Zhang, "Tire defect detection using fully convolutional network," *IEEE Access*, vol. 7, pp. 43 502–43 510, 2019.

[28] Z. Yu, X. Wu, and X. Gu, "Fully convolutional networks for surface defect inspection in industrial environment," in *Computer Vision Systems: 11th International Conference, ICVS 2017, Shenzhen, China, July 10-13, 2017, Revised Selected Papers 11*. Springer, 2017, pp. 417–426.

[29] Y. Huang, C. Qiu, and K. Yuan, "Surface defect saliency of magnetic tile," *The Visual Computer*, vol. 36, pp. 85–96, 2020.

[30] Y. Dong, J. Wang, Z. Wang, X. Zhang, Y. Gao, Q. Sui, and P. Jiang, "A deep-learning-based multiple defect detection method for tunnel lining damages," *IEEE Access*, vol. 7, pp. 182 643–182 657, 2019.

[31] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "Deepcrack: Learning hierarchical convolutional features for crack detection," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1498–1512, 2018.

[32] K. Song and Y. Yan, "A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects," *Applied Surface Science*, vol. 285, pp. 858–864, 2013.

[33] M. Wieler and T. Hahn, "Weakly supervised learning for industrial optical inspection," in *DAGM symposium in*, 2007.

[34] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[35] 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*, 2021, pp. 9716–9725.

[36] Y. Gorbachev, M. Fedorov, I. Slavutin, A. Tugarev, M. Fatekhov, and Y. Tarkan, "Openvino deep learning workbench: Comprehensive analysis and tuning of neural networks inference," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.