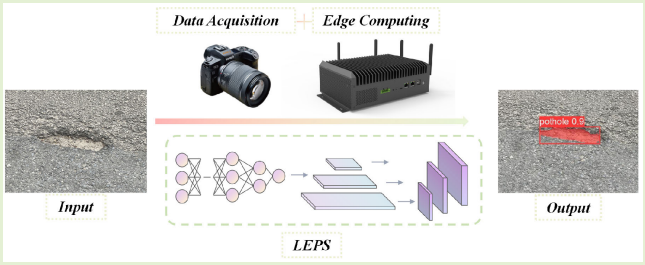
LEPS: A Lightweight and Effective Single-Stage Detector for Pothole Segmentation

Xiaoning Huang, *Student Member, IEEE*, Jintao Cheng, Qiuchi Xiang, *Student Member, IEEE*, Jun Dong, Jin Wu, *Member, IEEE*, Rui Fan, *Senior Member, IEEE*, and Xiaoyu Tang, *Member, IEEE*

***Abstract*—Currently, the problem of potholes on urban roads is becoming increasingly severe. Identifying and locating road potholes promptly has become a major challenge for urban construction. Therefore, a lightweight and effective instance segmentation method called lightweight and effective pothole segmentation (LEPS) detector was proposed for road pothole detection. To extract the image edge information and gradient information of potholes from the feature map more efficiently, a module that performs the convolutional superposition supplemented with a convolutional kernel was proposed to enhance spatial details for the backbone. LEPS has designed a novel module applied to the neck layer, improving the detection performance while reducing the parameters. To enable accurate segmentation of fine-grained features, the ProtoNet was optimized, which enables the segmentation head to generate high-quality masks for more accurate prediction. This article has fully demonstrated the effectiveness of the method through a large number of comparative experiments. This detector has excellent performance on two authoritative example datasets, URTIRI and COCO, and can successfully be applied to visual sensors to accurately detect and segment road potholes in real environments, and accurately LEPS reached 0.892 and 0.648 in terms of Mask for AP50 and AP50:95, which is improved by 4.6% and 20.6% compared with the original model. These results demonstrate its strong competitiveness when compared with other models. Comprehensively, LEPS improves detection accuracy while maintaining lightweight, which allows the model to meet the practical application requirements of edge computing devices.**



***Index Terms*— Deep learning, edge computing, instance segmentation, lightweight, pothole detection.**

Manuscript received 1 September 2023; revised 17 October 2023; accepted 25 October 2023. Date of publication 10 November 2023; date of current version 16 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62001173 and Grant 62233013, in part by the Project of Special Funds for the Cultivation of Guangdong College Students' Scientific and Technological Innovation ("Climbing Program" Special Funds) under Grant pdjh2022a0131 and Grant pdjh2023b0141, in part by the Science and Technology Commission of Shanghai Municipal under Grant 22511104500, and in part by the Fundamental Research Funds for the Central Universities. The associate editor coordinating the review of this article and approving it for publication was Prof. Markus Gardill. *(Corresponding author: Xiaoyu Tang.)*

Xiaoning Huang, Qiuchi Xiang, and Jun Dong are with the School of Data Science and Engineering, Xingzhi College, South China Normal University, Shanwei 516600, China (e-mail: 20218131003@m.scnu.edu.cn; 20218131007@m.scnu.edu.cn; 20228132044@m.scnu.edu.cn).

Jintao Cheng is with the School of Physics, South China Normal University, Guangzhou 510006, China (e-mail: chengjt_alex@outlook.com).

Jin Wu is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, China (e-mail: jin_wu_uestc@hotmail.com).

Rui Fan is with the College of Electronics and Information Engineering, the Shanghai Research Institute for Intelligent Autonomous Systems, the State Key Laboratory of Intelligent Autonomous Systems, and the Frontiers Science Center for Intelligent Autonomous Systems, Tongji University, Shanghai 201804, China (e-mail: rui.fan@ieee.org).

Xiaoyu Tang is with the School of Physics, School of Electronic and Information Engineering, South China Normal University, Guangzhou 510006, China (e-mail: tangxy@scnu.edu.cn).

Digital Object Identifier 10.1109/JSEN.2023.3330335

I. Introduction

DURING the construction and maintenance of infrastructure, it is common to encounter heavily used or deteriorating urban roads that are riddled with numerous potholes [1]. These potholes have a significant impact on the comfort and safety of vehicle passengers [2]. The traditional detection methods require professional inspectors to perform hazardous inspections, which are inefficient, costly, and time-consuming [3]. In addition, the results obtained from these methods are subjective and generalized [4]. Therefore, accurate, timely, and efficient detection of road potholes plays a crucial role in ensuring traffic safety. Therefore, intelligent road pothole detection algorithms based on computer vision (CV) are the important requirement for contemporary development [5].

Due to the continuous exploration of research relevant to CV for pothole detection tasks, numerous superior methods have emerged. Road pothole detection methods can be divided into two categories based on the CV. The traditional CV methods include edge detection, depth cameras [6], the approach based on laser scanning and CV [7], ultrasonic sensors that measure the distance between the vehicle's bottom and the road surface [8], among others. While deep learning methods

Fig. 1. Examples of the URTIRI dataset.

mainly rely on datasets and various self-learning network structures, Wang et al. [9] proposed a network of pothole 3-D point cloud segmentation and a pavement pothole motion structure (PP-SFM) to create a low-cost automatic system for reconstructing and segmenting potholes in 3-D.

As the existing convolutional frameworks have made significant progress in CV tasks in various industrial scenarios, object detection and semantic segmentation have also been applied to the web to address the issue of potholes. Rasyid et al. [10] built an Internet of Things (IoT) platform to detect potholes using Faster R-CNN. Pereira et al. [11] used the U-net [12] semantic segmentation framework to enhance the performance of pothole detection. The method based on semantic segmentation shows high accuracy at the pixel level [13].

Potholes have characteristics such as irregular contours, inconsistent scale, and subtle background differences, which pose challenges for the detection task [14]. First, the limited availability of features complicates the feature extraction process of the target. With the increase in convolutional neural networks (CNNs), there is a tendency for object feature information to decay, making it challenging to distinguish the features. In addition, the use of multilayer networks may lead to difficulties in detecting certain objects due to the weakening of feature information. At the same time, images of pavement potholes are easily influenced by the surrounding environmental factors, such as uneven road surfaces, pothole edges that resemble normal road surfaces, and other interferences that often affect detection accuracy. Fig. 1 illustrates that potholes exhibit a certain degree of variability in terms of size and shape.

For the task of pavement pothole detection, many existing algorithms are unable to simultaneously achieve both detection and segmentation of the pothole task due to limitations in scene usage. Coupled with the actual engineering of the pavement is often more complex, which can result in misdetection, omission of detection, and difficulties in achieving excellent detection results. Pothole pixels can be separated using a semantic segmentation network; however, the number and location of the potholes cannot be determined. Pothole localization can be achieved using the object detection network, but other metrics, such as the pothole area, cannot be calculated. This kind of network model using a single task cannot perfectly complete the pothole detection task. While the above methods show advantages and limitations, instance segmentation is a higher level task that combines object detection with semantic segmentation. It is capable of accurately localizing individual potholes and capturing pixel-level texture features. We address the above challenges by choosing instance segmentation, which enables us to detect object-specific locations and perform pixel-level semantic labeling.

We propose a lightweight and effective single-stage detector for road potholes by instance segmentation to solve the above problem, which can solve the problem of feature extraction in other detection methods for potholes while also maintaining a lightweight model and minimizing feature loss. First, considering the characteristics of potholes such as irregular contour, scale inconsistency, and little background difference, we propose the embranchment aggregation and detail enhancement (EADE) module which enhances the local features and detailed features of pothole to be applied to the backbone. Then, we developed the GhostNet with context aggregation (GCA) module to replace the original feature extraction module in the neck layer and minimize the impact of the scale and feature layers on the features in this model. In addition, to increase the accuracy of segmentation, we incorporate Optimized ProtoNet into the segmentation head. The following four points summarize the main contributions of this article.

1) A plug-and-play EADE module is proposed to be applied to the backbone, which performs convolutional superposition supplemented with a convolutional kernel that enhances spatial details, and it can extend the sensory field and capture richer local information.
2) We have designed the GCA module applied to the neck layer. This module aims to minimize the impact of duplicated gradient information to enhance network learning. It reduces the parameters while further improving the detection performance of the model.
3) We have innovatively proposed the Optimized ProtoNet, enabling the detection head to generate high-quality masks and realize accurate segmentation of fine-grained features.
4) In addition to verifying the validity of the model on a large number of datasets, we also use handheld devices to capture real-world scenarios as a validation set for detection, which confirms that the proposed model meets the requirements for real-world applications of visual sensing devices.

II. Related Work

A. Pothole Detection

Because of the significance of road safety, automatic road pothole image recognition technology has long been a classic subject [15]. For better extraction of the respective features of potholes, Khumsap et al. [16] provide a feature extraction method based on area contour and Cartesian contour method based on right-angle axis features and feed them into the subsequent SVM classifier as a way of realizing the classification of pavement anomalies in pavement images. Fan et al. [17] proposed an attention aggregation (AA) framework that can be applied to U-Net or extended to multimodal

fusion networks and introduced a parallax transformation algorithm to recognize defective and nondefective regions of potholes. Zou et al. [18] provide a deep convolutional network that integrates multiscale information for the segmentation of pavement cracks and potholes, which builds a DeepCrack network based on SegNet [19] while fusing the convolutional features generated in the encoding–decoding network in pairs with the same dimensions. Zou et al. [20] develop a fully automatic method named CrackTree to detect cracks from pavement images and perfectly solve the challenging problem such as low contrast between cracks and the surrounding pavement, intensity inhomogeneity along the cracks, and possible shadows with similar intensity to the cracks. Zou et al. [20] propose in this article FoSA—F* Seed-growing Approach for automatic crack-line detection, which extends the F* algorithm in two aspects. It exploits a seed-growing strategy to remove the requirement that the start and end points should be set in advance. Moreover, it narrows the global searching space to the interested local space to improve its efficiency.

The current research status of pothole detection demonstrates that as artificial intelligence technology continues to advance, the research focus of domestic and international scholars on pothole detection has gradually shifted from digital image processing technology to deep learning and has achieved positive results.

B. Sensor Applications

To improve pothole detection, [21] developed a detection system that uses 2-D LiDAR and cameras. This system incorporates a combination of heterogeneous sensor systems and uses image processing algorithms, including noise filtering, edge extraction, and object extraction. With joint roughness and negative skewness distribution, a method for detecting pavement potholes based on a vehicle-mounted laser point cloud is proposed [22]. Silvester et al. [23] used a smartphone to collect road images, acceleration values, and gyroscope values and detect potholes using a double cross-validation mechanism.

Lightweightness is a critical characteristic in sensor applications. The traditional object detection algorithms suffer from long running times, complex feature extraction modules, high computational costs, poor recognition accuracy, and other drawbacks. On the contrary, deep-learning-based CNNs use a step-by-step extraction of image features and autonomously learn to obtain feature information, which leads to higher accuracy and smaller network size. Currently, most of the models used to implement the task of pothole detection are single-stage detection models, which have the characteristics of fast speed, low computational requirements, and minimal hardware requirements. So Yolov8n as the latest model in the YOLO series serves as the foundation for improving our model to better meet the practical application requirements of vision sensing devices.

III. Theory

In this section, our newly developed lightweight and effective pothole segmentation (LEPS) will be comprehensively introduced. Our improved method integrates computational complexity, parameters, the chosen convolution method, the structure of feature fusion, the effectiveness of model training, and other critical factors to maximize the performance of the instance segmentation algorithm.

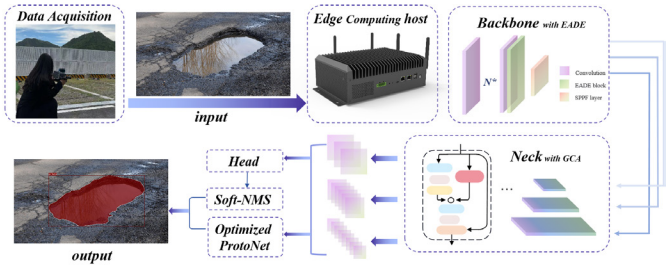


Fig. 2. Overall framework of LEPS.

A. Overall Structure

The general structure of the approach proposed in this study is shown in Fig. 2.

First, we hope that the proposed LEPS can serve as an integrated model capable of addressing various issues in the practical implementation of pavement pothole detection. This model should strike a balance between computational effort and accuracy. Therefore, we chose Yolov8n's DarkNet-53 as the backbone network, which is inspired by the design concept of the classical ResNet backbone network and incorporates the use of residual structures. By stacking the layers of the convolution module continuously, the model's performance was enhanced. Yolov8 is a new state-of-the-art (SOTA) model that offers the advantages of small scale and high detection efficiency. It builds upon the success of the previous YOLO series and demonstrates superior performance compared with networks of similar scale. Previous studies provide evidence that justifies our choice.

Second, we discovered that the standard convolution in the C2f module restricts the sensory field of the network, making it challenging to accurately detect potholes in complex scenes. Therefore, we implemented the EADE module in the backbone section. The input feature map processed by this module can capture more edge information and details related to the pothole detection task, specifically focusing on local edge features. Then, we further optimize the neck layer by inserting the GCA module. The GhostConv [24] and the context aggregation [25] are combined in a parallel structure to complement each other and improve the feature processing capability of the model.

In addition, we incorporate a multiscale segmentation head to optimize ProtoNet's feature output. This differs from the conventional approach by combining feature inputs from three different scales. This method allows for the fusion of features with varying levels of refinement. To further increase the accuracy of segmentation, we use Soft-non-maximum suppression (NMS) [26] instead of the NMS method used in previous studies, to improve the accuracy of the bounding box. Finally, the instances that survive Soft-NMS will be linearly combined with the output of Optimized ProtoNet to obtain the final

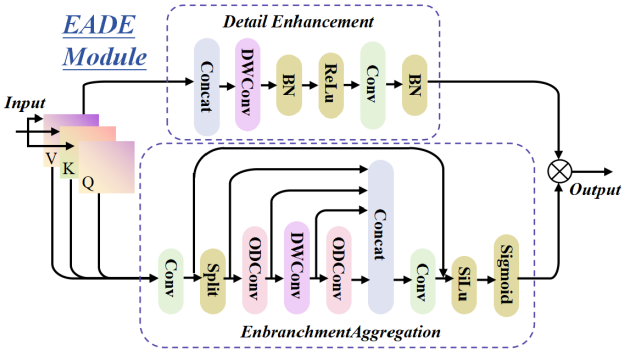


Fig. 3. Structure of EADE module.

feature maps. The aforementioned improvements enable our model to efficiently extract and combine features, while also meeting the requirement for a lightweight model with minimal computational complexity.

B. EADE Module

We marked partial improvements to the backbone structure and propose our EADE module to enhance backbone. Fig. 3 shows the network structure of bottleneck, which is designed to retain most of the effective global information at a low computational cost. Nevertheless, we present an innovative and effective module based on the notion of Seaformer [27], to address the problem of ignoring the local information in the target information. The EADE module performs multiple convolutions and gradient information splicing by separating local information from two scales. The output features of the embranchment aggregation block and detail enhancement block are fused as the final result.

As shown in Fig. 3, the input $X \in R^{C\times H\times W}$ is fed into both the embranchment aggregation block and detail enhancement block for processing. The output features from these blocks are then fused together to produce the final output.

1) Embranchment Aggregation Block: In the embranchment aggregation block, the feature maps will be processed by a 1×1 convolution module. As shown in (1), this module first performs 2-D convolution on the input feature maps, then performs batch normalization at the BN layer, and finally is processed by the SiLu activation function. After the Split operation, the feature map is divided into two parts with an equal number of channels, and then the two segmented feature maps, x_1 and x_2, which contain different local information, are processed differently. The feature map represented by x_1 is directly used as the basis for the later concatenation operation. On the other hand, the feature map represented by x_2 is input into a stacking filter consisting of three 3×3 convolution modules, two omnidimensional dynamic convolution (ODConv [28]) and one depthwise separable convolution (DWConv [29]). Both ODConv and DWconv can enhance the adaptability of features, allowing the module to achieve the same feature extraction effect as ordinary convolution, reduce the calculation amount, and improve the calculation efficiency at the same time, which is very suitable for lightweight model design in this article. Each 3×3 convolution module produces a portion of the feature map, one of which is directly used as the foundation for the subsequent cascade operation, while the other of which is supplied into the next 3×3 convolution module. Similarly, x_3 and x_4 will be convolved in the same way as x_2

$$F_1(X, C) = \sigma_1(\rho(f_{\text{conv}}(X, C))) \tag{1}$$

$$F_2(X) = \rho(\sigma_1(f_{\text{conv}}(X))) \tag{2}$$

$$F_3(X) = X\left(\prod_{t=1}^{n} \alpha_t w_t\right). \tag{3}$$

In the given formula, the convolution module is denoted by $F_1(X, C)$, while $F_2(X)$ is applied for DWConv and $F_3(X)$ is applied for ODConv. $\rho(X)$ represents batch normalization, which can accelerate the convergence rate. $\sigma_1(X)$ represents the Silu activation function, and $f_{\text{conv}}(X, C)$ represents the 2-D convolution operation. C denotes the output channel number, X denotes the feature map to be processed, w_t denotes the tth convolution kernel consisting of filters, and α_t denotes the attention scalar

$$\begin{cases} x_1 = \text{Split}(F_1(X), 2) \\ x_2 = F_3(F_2(F_3(\text{Split}(F_1(X), 2)))) \\ x_3 = F_2(F_3(\text{Split}(F_1(X), 2))) \\ x_4 = F_3(\text{Split}(F_1(X), 2)). \end{cases} \tag{4}$$

$\text{Split}(X, 2)$ indicates splitting the feature map. Here, X represents the feature map that will be split, and 2 represents the number of output channels, denoted as $C = 2$.

Four output feature maps are channel-spliced to obtain a feature map that contains more bias and gradient information. The final channel number of the module is obtained through the last 1×1 convolution module. A shortcut is used to determine whether to retain the residual information or not. The feature maps activated by the Silu activation function will be used as the final output of the embranchment aggregation block process

$$y = \begin{cases} \sigma_1(F_1(\text{cat}(x_1, x_2, x_3, x_4))) \\ \text{shortcut} = \text{Flase} \\ \sigma_1(F_1(\text{cat}(x_1, x_2, x_3, x_4)) + F_1(X)) \\ \text{shortcut} = \text{True} \end{cases} \tag{5}$$

where cat indicates the operation of concatenating the number of channels in the feature map, while y represents the final feature map output from the embranchment aggregation block.

2) Detail Enhancement Block: q, k, and v are linear projections of $X \in R^{C\times H\times W}$, which means $q = W_q X$, $k = W_k X$, and $v = W_v X$, where W_q, $W_k \in R^{C_{qk}\times C}$, and $W_v \in R^{C_v\times C}$. The detail enhancement block for parallel processing concatenates q, k, and v along the dimension and then puts them through a 3×3 depthwise convolution module and batch normalization. Using a 3×3 convolution, local detail enhancement can be assisted by aggregating q, k, and v. Then, the feature map is passed through a linear projection with activation and batch normalization to generate detail enhancement weights. Finally, the feature map processed by detail enhancement and the feature

map processed by embranchment aggregation are fused to output the final result

$$z = \rho(F_1(\sigma_2(\rho(F_2(\text{cat}(q, k, v)))))) \tag{6}$$

$$Y = \text{Mul}(\sigma_2(y), z). \tag{7}$$

In the given formula, $\sigma_2(X)$ represents the ReLU activation function, and $\text{Mul}(X, Y)$ denotes the multiplication operation. z represents the final feature map that is generated from the detail enhancement block, y represents the output from the embranchment aggregation block, while Y represents the output of the EADE module.

The EADE module can extract additional local feature information and gradient information from the input feature mapping. This module then provides more effective feature mapping information for subsequent feature fusion and head prediction. The feature map x_1 retains the original channel feature information, while x_2 can obtain a larger receptive field than x_1 after three rounds of 3×3 convolutional processing. The output feature map of the combined feature map contains richer local information on gradient flow. Meanwhile, the auxiliary convolution kernel, which enhances spatial details, also accomplishes good cooperation, effectively extracting global information while preserving local details.

C. GCA Module

To enhance the implementation of the application on mobile devices, we refer to the lightweight CNN with faster inference speed. To achieve faster convergence, we optimize the neck layer by incorporating the GhostNet with context aggregation block (CA-Block), which is constructed based on the fully connected layer. The module from GhostNet can not only make our upgraded algorithms easier to execute quickly on standard hardware but also capture the dependencies between remote pixels. CA-Block is a generic building block for multihead context aggregation; it can use remote interactions, similar to a transformer, while still taking advantage of the inductive bias of local convolution operations.

As shown in Fig. 4, we initially use a 1×1 convolution for input features to change channels, then replace the 1×1 convolution with a Split operation to divide the features, and increase the network receptivity field by stacking the GCA module multiple times and incorporating more jump connections. We obtain a more complex gradient stream structure by reducing the number of references, enabling the extraction of more comprehensive multiscale pothole features.

Fig. 4 shows that we use two GhostConv modules [represented by $f_{\text{Ghost}}(X)$] for the inverted residual bottleneck. The first GhostConv module generates extended features and increases the number of channels. Afterward, it performs batch normalization [represented by $\rho(X)$] and is finally processed by the SiLu activation function [represented by $\sigma_2(X)$]. This module performs a more cost-effective linear variation operation on the feature map compared with normal convolution. However, we observe that partial features generated from cheap operations may somewhat compromise their expressiveness and capacity. So, we add the CA-Block [represented by $f_{\text{CA-Block}}(X)$] to the parallel structure of this GhostConv module to enhance the functionality of the extension, strengthen

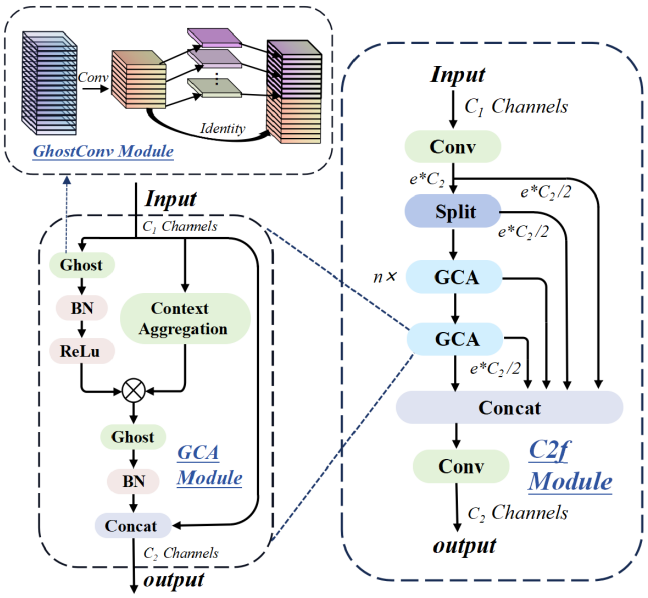


Fig. 4. Structure of GCA module.

the semantic and positional information in the features, and contribute to the capacity for feature fusion. X_{out} indicates the enhanced feature obtained by multiplying [denoted as $\text{Mul}(X, Y)$] the feature from the GhostConv extension with the features from the CA-Block

$$X_{\text{out}} = \text{Mul}(\sigma_2(\rho(f_{\text{Ghost}}(X))), f_{\text{CA-Block}}(X)). \tag{8}$$

The second GhostConv trims down the number of output channels to match the input channels. This module captures the long-range dependencies between pixels at different spatial locations, which can reduce the number of channels while improving the model's representation. Finally, the feature map Y_{out} obtained in the previous step is combined with the residual edges. The two parts are then merged into a complete feature map using the Concat operation [represented by $\text{cat}(X, Y)$]

$$Y_{\text{out}} = \text{cat}(\rho(f_{\text{Ghost}}(X_{\text{out}})), X). \tag{9}$$

The GCA module constructed in this article can reduce the computational and parametric quantities of the network while preserving the size of the original output and the channel size. In the C2f structure built with the GCA module, the cross-stage feature fusion strategy and the truncated gradient flow technique are used to increase the variability of the learned features across different network layers, minimize the effect of redundant gradient information, and improve the learning capacity of the network. Due to the introduction of GhostConv to replace a large number of 3×3 ordinary convolutions in the original structure, the model size of the network is compressed, which allows the model to be deployed on the mobile side and makes it easier to realize the detection of pavement pothole. In comparison to the previous bottleneck, the GCA module allows for more informative details, improved recognition, and better segmentation results. It also tends to be lightweight in terms of computational cost and number of spatial parameters, enabling us to outperform similarly lightweight competitors with better performance and lower latency.

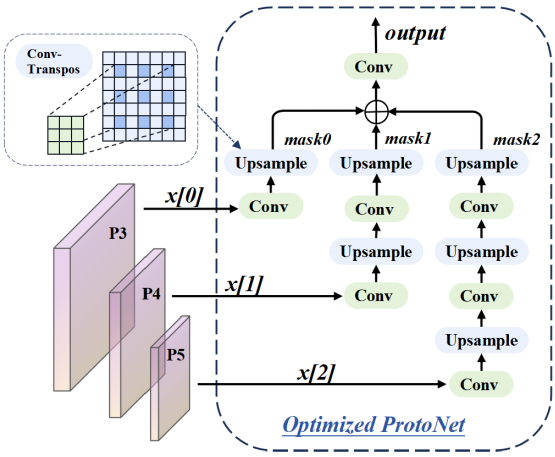


Fig. 5. Structure of Optimized ProtoNet.

D. Optimized ProtoNet

ProtoNet is a structure in which YOLACT [30] generates k sets of high-quality masks after the neck layer. Each detection head predicts k mask coefficients, bounding boxes, and classes. These correspond to k different masks that are linearly combined to assign compatible masks to each predicted object. However, the network can only predict the shapes of k sets of masks in an image without including positional information. Therefore, to maintain the fast detection speed of the segmentation task and preserve the single-stage detection style, this article proposes an additional mask generation header and a set of mask coefficient predictions in the segmentation header module. This approach is inspired by YOLACT, which is named Optimized ProtoNet.

Fig. 5 shows the Optimized ProtoNet, The original ProtoNet uses only the P3 layer as input, which is a very convenient and lightweight design, but the quality of the mask generated only from the P3 feature layer needs to be further improved. Our improved Optimized ProtoNet takes P3, P4, and P5 as inputs, which can significantly enhance the feature richness. Each branch, as shown in Fig. 5, undergoes multiple upsampling operations of convolutional layers and feature fusion. It fuses the three inputs by summing them by channel, and after unifying them to the same number of channels, we use the convolutional module for feature fusion as the final output. Among the adjustments in the upsampling operation, we analyze several common methods. Interpolation and depooling techniques struggle to provide additional meaningful information to the feature mapping. Pixel shuffling, on the other hand, requires an increase in the number of channels, resulting in significant computational costs. Therefore, we opt to perform upsampling using the inverse convolution operation

$$F_{\text{upsample}}(X) = F_2\left(F_{\text{ConvTranspose}}(X)\right) \tag{10}$$

$$P_{\text{out}} = F_1\left(\sum_{i=1}^{n}\left(F_{\text{upsample}}\left(F_1(P_i)^i\right)\right)\right). \tag{11}$$

As we introduced in Section III-B, the convolution module is denoted by F_1, P_i represents the input of P3, P4, and P5, respectively, and P_{out} means the final output. The Optimized ProtoNet has finer and higher quality feature extraction, which contributes to the quality of the final mask generation, but the unavoidable aspect is the slight increase in parameters and FLOPs.

Input : $\mathcal{B} = \{b_1, .., b_N\}, \mathcal{S} = \{s_1, .., s_N\}, N_t$
$\mathcal{B}$ is the list of initial detection boxes
$\mathcal{S}$ contains corresponding detection scores
N_t is the NMS threshold
begin
  $\mathcal{D} \leftarrow \{\}$
  while $\mathcal{B} \neq empty$ **do**
    $m \leftarrow$ argmax $\mathcal{S}$
    $\mathcal{M} \leftarrow b_m$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{M}; \mathcal{B} \leftarrow \mathcal{B} - \mathcal{M}$
    for b_i *in* $\mathcal{B}$ **do**
      if $iou(\mathcal{M}, b_i) \geq N_t$ **then**
        $\mathcal{B} \leftarrow \mathcal{B} - b_i; \mathcal{S} \leftarrow \mathcal{S} - s_i$
      end NMS
      $s_i \leftarrow s_i f(iou(\mathcal{M}, b_i))$ Soft-NMS
    end
  end
  return $\mathcal{D}, \mathcal{S}$
end

Fig. 6. In the diagram of Soft-NMS, the red pseudocode represents the original NMS and the green pseudocode represents Soft-NMS.

E. Soft-NMS

NMS can eliminate highly redundant Bboxes and is a significant part of the target detection pipeline. Due to the irregular shape of potholes, we use the Soft-NMS in place of the original NMS to avoid the low-quality bounding box that affects the computation process, as shown in Fig. 6. The original NMS algorithm categorizes the detection frames based on their scores, selects the detection frame with the highest scoring score, suppresses all other detection frames that significantly overlap with it, and applies this process recursively to the remaining frames. Instead of directly removing the frames that have an IOU greater than a threshold with the highest scoring frame, Soft-NMS reduces their confidence levels so that more frames are retained, avoiding masking to some extent. This is only one of the more subtle improvement points in the article, but it can provide better results, and its effectiveness can be found through subsequent experiments.

IV. Experiment

A. Preparation of Experiment

1) Datasets:

1) The UDTIRI dataset [31] offers various formats for the road pothole detection task. The instance segmentation task dataset contains images of potholes captured under various lighting and road conditions. UDTIRI contains potholes of various sizes, ranging from the smallest pothole, which accounts for 0.3% of the image area, to the largest pothole, which accounts for 92.0%. This indicates that the pothole data can accurately represent the pothole situation in most real-life scenes.
2) The COCO dataset is a large dataset that has been widely used in the field of CV for tasks such as image

captioning, object detection, and semantic segmentation. The dataset includes more than 330 000 photographs and 1.5 million instances of objects, with 91 categories for stuff and 80 different categories for items.

3) In this article, we used a handheld device to capture photographs of a dataset for experimental validation. This dataset is referred to as the pothole validation set. The road section we photographed is an asphalt road with a high volume of traffic, especially heavy trucks. Over the years, due to both usage and exposure to the natural environment, the road has deteriorated significantly. It is riddled with numerous potholes, cracks, patches, and other signs of pavement deterioration. The device is equipped with an industrial edge computing unit. The CPU processor is an Intel i5-1240p with 16 GB of memory. The sensor is equipped with two industrial cameras that can collect real-time data from pits and quickly process it using a CPU.

2) Experimental Parameter: We validated and analyzed the proposed method, and the configurations are as follows: For the fairer experimental validation, we used Intel Core i9-11900K, Ubuntu20.04 system, and NVIDIA RTX 3060 graphics card, based on CUDA version 11.4 and PyTorch framework to build the proposed modeling approach, and we scaled the training and validation image size of all the models to 640 × 640. In experiments, we tuned the training parameters of the LEPS. The batch size was set to 4, the momentum was 0.937, the momentum decay coefficient was 0.0005, the initial learning rate was 0.01, and the number of iterations for all the datasets was 300. The camera has 1.3 million pixels and a resolution of 1280 × 1024.

3) Evaluation Criteria: We use $AP_{0.5}$ and $AP_{0.5}^{0.95}$ to evaluate the performance of the algorithm, as the detection model performance under different IoU thresholds is considered, which allows for a more comprehensive evaluation of the model's detection and localization capabilities. In terms of measuring model size, we also introduce FLOPs and parameters. FLOPs is a measure of the computational complexity of a model and can be used to compare the computational overhead of different models. And FPS was added to evaluate the inference time for better highlight our lightweightness.

B. Ablation Experiment

To validate the effectiveness of the new approach described in this research, we conducted specific ablation experiments in this study. The experimental findings are shown in Table I.

In this study, we designed ten control groups to conduct ablation experiments and compared the performance of the EADE module (represented by A), the GCA module (represented by B), Optimized ProtoNet (represented by C), and Soft-NMS individually. We also evaluated the performance of a combination of the three modules using Yolov8n as the baseline model. When we integrated the GCA module into the neck structure, we observed a reduction in model parameters and computational complexity. This suggests that the model achieved a 3% increase in mAP while also becoming more lightweight. Soft-NMS enhancement worked as expected, resulting in a slight improvement in the model's performance. The introduction of the EADE module and Optimized ProtoNet modules increases the number of parameters by only 0.1–0.3 M and FLOPs by only 1–2 G, yet results in a consistent improvement in mAP. The Optimized ProtoNet, designed for instance segmentation, not only improves mask accuracy but also greatly enhances bounding box detection. We infer that this improvement occurs because the prediction of instance masks relies on the cropping operation performed based on the associated bounding box. The computation of the loss function for both is correlated, so the Optimized ProtoNet improves the efficiency of mask detection while also positively impacting bounding box detection. Experimental groups 5–8 represent the experiments in which combinations of the EADE module, GCA module, and Optimized

TABLE I
ABLATION STUDIES OF THREE IMPLEMENTS

No	Model	FLOPs (G)	Para (M)	Mask		BBox	
				$AP_{0.5}$	$AP_{0.5}^{0.95}$	$AP_{0.5}$	$AP_{0.5}^{0.95}$
1	Baesline(Yolov8n)	11.2	3.0	0.849	0.547	0.853	0.537
2	8n+A	12.1	3.1	0.870	0.576	0.875	0.582
3	8n+B	11.2	2.9	0.879	0.644	0.879	0.632
4	8n+C	13.7	3.6	0.878	0.644	0.880	0.640
5	8n+NMS	12.1	3.2	0.854	0.564	0.856	0.569
6	8n+A+B	11.6	2.8	0.879	0.620	0.883	0.624
7	8n+B+C	12.9	3.2	0.878	0.617	0.874	0.614
8	8n+A+C	12.3	3.2	0.875	0.599	0.878	0.629
9	8n+A+B+C	13.3	3.3	0.880	**0.645**	0.886	0.645
10	8n+NMS+A+B+C	13.3	3.3	0.885	0.640	0.892	0.648
	LEPS(ours)	13.3	3.2	**0.885**	0.640	**0.892**	**0.648**

TABLE II
COMPARISON OF DIFFERENT CONFIGURATIONS FOR GCA MODULE

No	Model	FLOPs (G)	Para (M)	Mask		BBox	
				$AP_{0.5}$	$AP_{0.5}^{0.95}$	$AP_{0.5}$	$AP_{0.5}^{0.95}$
1	R=0	11.5	2.8	0.881	0.621	0.885	0.626
2	DFL (r=2)	11.7	2.9	0.873	0.619	0.870	0.624
3	DFL (r=8)	11.7	3.0	0.879	0.636	0.876	0.632
4	DFL (r=4)	11.7	3.0	0.884	0.636	0.885	0.641
5	Scconv[32](r=2)	11.6	2.8	**0.886**	0.625	0.863	0.618
6	Scconv(r=4)	11.6	2.8	0.877	0.623	0.874	0.620
7	Scconv(r=8)	11.6	2.8	0.870	0.630	0.873	0.631
8	CA-Block(r=4)	11.6	2.8	0.885	0.640	0.892	0.648
9	CA-Block(r=8)(ours)	11.6	2.8	0.885	**0.640**	**0.892**	**0.648**

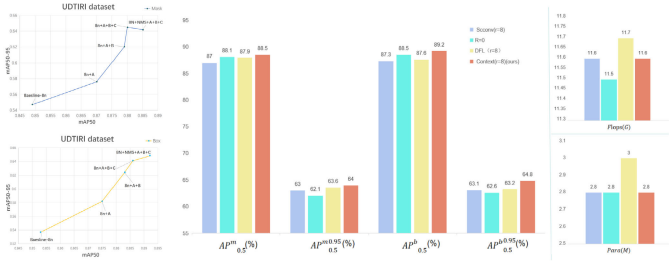


Fig. 7. Left: performance of four modules: EADE (A), GCA (B), Optimized ProtoNet (C), and Soft-NMS. These modules improve the model on the URTIRI dataset from both box and mask perspectives. Right: comparison of accuracy and parameter counts for different configurations of the GCA module in the ablation experiments. It can be visualized that CA-Block has the advantage in terms of accuracy and parameters.

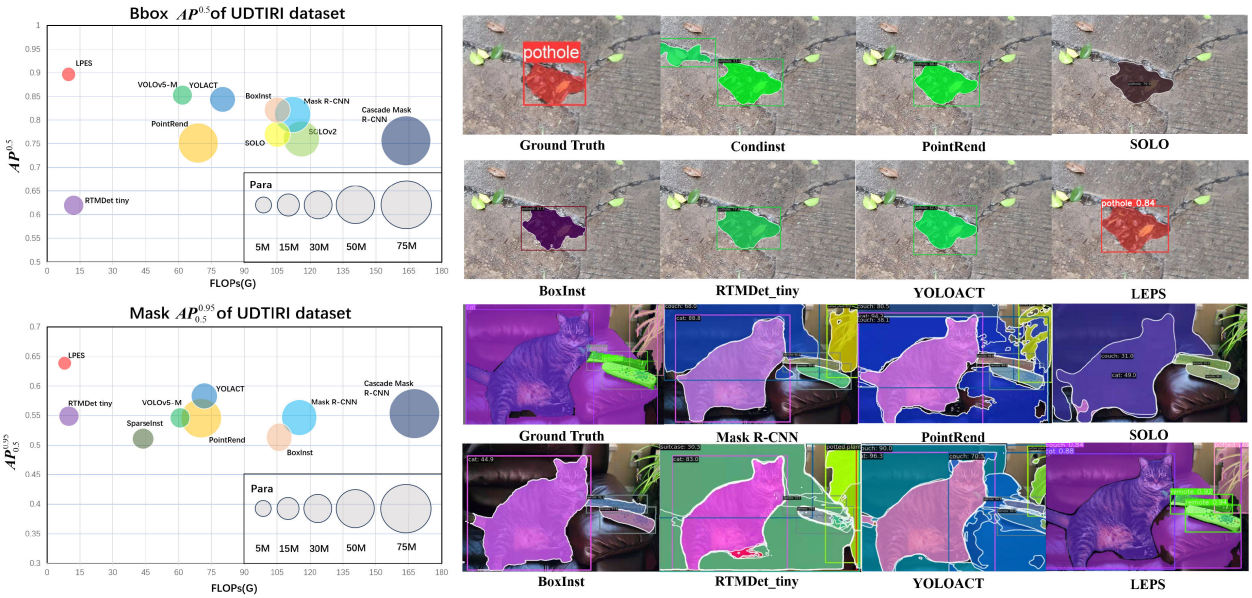


Fig. 8. Left: performance of various models on the UDTIRI dataset across three dimensions: accuracy, computational complexity, and parameters. Two graphs depict the impact on $AP^{0.5}$ (B) and $AP^{0.95}_{0.5}$ (M). Right: detection result of different models (including ground truth) on the UDTIRI and COCO datasets.

ProtoNet are applied to the baseline. The detection results show that the combination of any two or three improvement points is superior to using only a single improvement point. This observation confirms the complementary nature of the aforementioned improvements and their synergistic effects. Experimental group 10 represents our final model. It can be observed that after incorporating all our improvements into the baseline model, we achieved a 0.3 M reduction in the number of parameters and a 2.1 G increase in FLOPs. As a result, we obtained improvements of 9.3% in $AP^{0.95}_{0.5}$(BBox), 3.6% in $AP_{0.5}$(BBox), 11.1% in $AP^{0.95}_{0.5}$(Mask), and 3.9% in $AP_{0.5}$(Mask).

We report the results of experiments using various modules as GhostConv parallel structures during the GCA module design process in Table II, and the data visualization analysis diagram of the ablation experiment can be viewed in Fig. 7. It has been found that the difference in the number of parameters and FLOPs for these nine groups of experiments is almost negligible, but the impact on accuracy is significant. Among them, experimental group 1 used the GhostNetv1 version. Experimental groups 2–4 used the GhostNetv2 version. Experimental groups 5–7 were inserted with the Scconv module; and experimental groups 8 and 9 had the CA-Block module inserted, which performed the best. The ratio is denoted by r in the table, which represents the channel drop multiplier of the first fully connected channel and affects the effectiveness of the module to some extent. We conducted a comparative experiment using various ratio values to further demonstrate the effectiveness of the fusion method. For DFL and Scconv, this value has a significant impact and varies significantly between them, but not for CA-Block, and it can be seen that the change in the number of channels does not play a substantial role in the CA-Block. We have selected the most efficient modules from experiment 9 to act as the foundational elements of the final GCA module. This selection ensured the optimal balance between effectiveness and computational requirements.

TABLE III
COMPARISON WITH OTHER MODELS ON THE UDTIRI DATASET

Methods	Year	FLOPs (G)	Para (M)	FPS (f/s)	Mask		BBox	
					$AP_{0.5}$	$AP^{0.95}_{0.5}$	$AP_{0.5}$	$AP^{0.95}_{0.5}$
Yolov8n(Baseline)	2023	11.2	3.0	84.7	0.849	0.547	0.854	0.537
Mask R-CNN[33]	2017	112	44.4	142.1	0.781	0.543	0.775	0.591
Cascade Mask R-CNN[34]	2018	168.6	77.3	87.0	0.754	0.568	0.776	0.566
YOLACT [35]	2019	81.5	34.7	115.7	0.816	0.582	0.853	0.596
SOLO[36]	2020	112	36.3	135.0	-	-	0.754	0.600
SOLOV2[37]	2020	119	46.6	138.8	-	-	0.752	0.562
Yolov5-M[38]	2020	70.2	21.7	61.2	0.847	0.544	0.821	0.550
PointRend[39]	2020	70	60.2	143.2	0.741	0.532	0.733	0.554
BoxInst[40]	2021	108	35.1	93.1	0.767	0.501	0.816	0.512
RTMDet_tiny[41]	2022	12.0	5.6	**246.2**	0.746	0.562	0.611	0.485
SparseInst [42]	2022	44.3	8.7	590.4	0.785	0.514	-	-
Co-Detr[43]	2023	267.7	64.13	136.25	-	-	0.859	0.590
PatchDCT[44]	2023	14.1	-	-	0.754	0.452	0.728	0.408
ConvNeXt V2[45]	2023	242	47.67	24.9	0.882	0.601	0.889	0.572
LEPS(ours)	2023	13.3	3.2	144.9	**0.885**	**0.640**	**0.892**	**0.648**

C. Comparative Experiment

To demonstrate the excellent performance of LEPS, we design the comparative experiment on the UDTIRI dataset and the public dataset COCO. Table III shows comparison of the detection performance of our model with other models on the UDTIRI dataset and Table IV shows comparison of the COCO dataset. We compared it with many other models and achieved competitive results. The chosen models are either classical or SOTA models. For convenience and reproducibility, we conducted experiments using the MMDetection framework. The models, experimental parameters, and pretrained weights were obtained from the official MMDetection repository. LEPS demonstrates significant superiority on the UDTIRI dataset, particularly in terms of mAP values. The scores for Mask

TABLE IV
COMPARISON WITH OTHER MODELS ON THE COCO DATASET

Methods	Year	BBox		Mask		FLOPs	Para
		$AP_{0.5}$	$AP_{0.5}^{0.95}$	$AP_{0.5}$	$AP_{0.5}^{0.95}$	(G)	(M)
EffcientDet-D0[46]	2020	52.2	33.8	-	-	2.5	3.9
ResNet18[47]	2020	54.0	34.0	51.0	31.2	-	31.2
Yolov4-T[48]	2021	42.1	24.9	-	-	6.9	6.1
Yolox-T[49]	2021	50.3	32.8	-	-	6.5	5.1
Yolov5s[50]	2021	56.8	37.4	-	-	16.5	7.2
Mask-RCN(1x)[51]	2021	58.9	40.4	55.5	36.1	-	-
PVT-tiny[52]	2021	**59.2**	36.7	56.7	35.1	-	32.9
Yolov6m[53]	2022	56.6	40.3	-	-	36.7	15.0
MF-294M[54]	2022	56.6	36.6	-	-	-	6.5
Yolov7-T[55]	2022	52.8	35.2	-	-	5.8	6.2
LEPS(ours)	2023	57.4	**42.4**	**57.6**	**37.9**	13.9	**3.3**

TABLE V
EXPERIMENT ON THE POTHOLE VALIDATION SET

Model	FLOPs	Para	Mask		BBox	
	(G)	(M)	$AP_{0.5}$	$AP_{0.5}^{0.95}$	$AP_{0.5}$	$AP_{0.5}^{0.95}$
Yolov5s	25.7	7.4	0.716	0.371	0.712	0.426
LEPS	13.3	3.2	**0.787**	**0.419**	**0.768**	**0.427**
Comparison	-12.4	-4.2	+0.071	+0.048	+0.056	+0.001

and BBox are 0.640 and 0.648, which are 17% and 20.7% higher than those of the original model. Through the FPS, we can compare the inference time of each model; the table presents the values we obtained by doing the comparison test on the same device, and our LEPS performance is just lower than the SOTA model RTMDet_tiny and SparseInst, which fully demonstrates the advantages we show in lightweight. When comparing parameters and computational complexity, we observe that other models have computational complexity and parameter counts that are nearly dozens of times greater than those of LEPS. Despite this, our proposed model still maintains high performance on all the four indexes and outperforms most models. We present the results of eight models for comparison in Fig. 8.

Table IV shows the results of comparing our model with other models on the COCO dataset. We compare our model with recent excellent single-stage detectors and some lightweight detectors. The results show that LEPS has the fewest parameters and outperforms other transformer-based detection models. In terms of accuracy, the $AP_{0.5}^{0.95}$ of LEPS outperforms other detection models of the same size in both Bbox and Mask. The data in Table IV show that our LEPS performs slightly lower than PVT-Tiny and Mask-RCN in terms of $AP_{0.5}$(BBox) only. As a detector for instance segmentation, LEPS performs optimally among all the models in terms of $AP_{0.5}$(Mask). Overall, LEPS is the best detector for instance segmentation among lightweight detectors, considering parameters, computational complexity, and detection accuracy.

We selected the best performing models on the UDTIRI dataset for comparison, and Table V presents the experimental results of potholes on pavements, which were photographed using a handheld device. It can be observed that both the models exhibit errors in detecting pothole features, particularly when the shape varies from different angles. However, when compared with Yolov5, LEPS demonstrates higher performance in terms of $AP_{0.5}$(Mask), $AP_{0.5}^{0.95}$(Mask), $AP_{0.5}$(BBox), and $AP_{0.5}^{0.95}$(BBox), with improvements of 7.1%, 0.48%, 5.6%, and 1%, respectively. This further validates the overall effectiveness and applicability of the module to visual sensors. Fig. 9 shows an example of the experimental results for the pothole validation set.

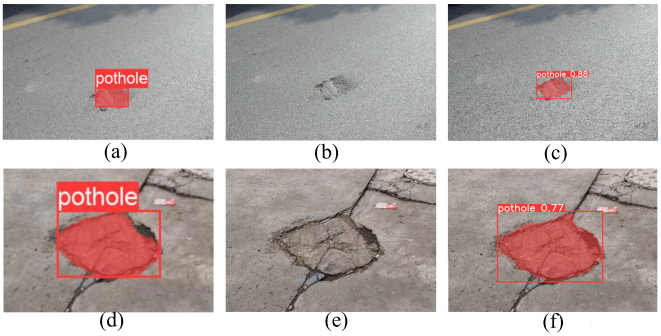


Fig. 9. Examples of pothole validation set. (a) and (d) Ground truth. (b) and (e) Before. (c) and (f) After.

V. CONCLUSION

In this article, we observe that commonly used deep learning algorithms perform poorly in the task of pothole detection. Although several SOTA methods can accomplish the task of pothole detection, there is a need to optimize the robustness and accuracy of the detection. For pothole features, we proposed an efficient lightweight detector for instance segmentation, which significantly improves the overall detection accuracy of both the pothole features and high-pixel features while maintaining a lightweight design. In ablation experiments, our proposed EADE module, GCA module, and Optimized ProtoNet performed well in the pothole detection task, confirming the effectiveness of the algorithmic enhancement. Extensive comparative experiments demonstrate that our pothole detection network achieves high accuracy in identifying potholes and also exhibits advantages in terms of lightweight design. Notably, we have applied our algorithm to a handheld shooting device and achieved promising experimental results. This indicates that it is feasible to deploy our model to the sensor. However, there is still room for improvement in terms of lightweightness and robustness. In the future, we will continue conducting further research based on the model, while also considering the practical value of applying this direction in real life. This will enable us to streamline the model and make it easier to implement.

ACKNOWLEDGMENT

In the process of experimental design and verification, the authors obtained help from the Robotics and IntelliSense Innovation Laboratory, Xingzhi College, South China Normal University, Shanwei, China, and they are grateful for the kindness.

REFERENCES

[1] R. Fan, U. Ozgunalp, Y. Wang, M. Liu, and I. Pitas, "Rethinking road surface 3-D reconstruction and pothole detection: From perspective transformation to disparity map segmentation," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 5799–5808, Jul. 2022.

[2] N. Ma et al., "Computer vision for road imaging and pothole detection: A state-of-the-art review of systems and algorithms," *Transp. Saf. Environ.*, vol. 4, no. 4, Nov. 2022, Art. no. tdac026.

[3] R. Fan et al., *Autonomous Driving Perception*. Cham, Switzerland: Springer, 2023.

[4] R. Fan, U. Ozgunalp, B. Hosking, M. Liu, and I. Pitas, "Pothole detection based on disparity transformation and road surface modeling," *IEEE Trans. Image Process.*, vol. 29, pp. 897–908, 2020.

[5] R. Fan, X. Ai, and N. Dahnoun, "Road surface 3D reconstruction based on dense subpixel disparity map estimation," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 3025–3035, Jun. 2018.

[6] K. Vigneshwar and B. H. Kumar, "Detection and counting of pothole using image processing techniques," in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res. (ICCIC)*, Dec. 2016, pp. 1–4.

[7] H. Wu et al., "Road pothole extraction and safety evaluation by integration of point cloud and images derived from mobile mapping sensors," *Adv. Eng. Informat.*, vol. 42, Oct. 2019, Art. no. 100936.

[8] R. Madli, S. Hebbar, P. Pattar, and V. Golla, "Automatic detection and notification of potholes and humps on roads to aid drivers," *IEEE Sensors J.*, vol. 15, no. 8, pp. 4313–4318, Aug. 2015.

[9] N. Wang et al., "3D reconstruction and segmentation system for pavement potholes based on improved structure-from-motion (SFM) and deep learning," *Construct. Building Mater.*, vol. 398, Sep. 2023, Art. no. 132499.

[10] A. Rasyid et al., "Pothole visual detection using machine learning method integrated with Internet of Thing video streaming platform," in *Proc. Int. Electron. Symp. (IES)*, Sep. 2019, pp. 672–675.

[11] V. Pereira, S. Tamura, S. Hayamizu, and H. Fukai, "Semantic segmentation of paved road and pothole image using U-Net architecture," in *Proc. Int. Conf. Adv. Inform., Concepts, Theory Appl. (ICAICTA)*, Sep. 2019, pp. 1–4.

[12] 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*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham, Switzerland: Springer, 2015, pp. 234–241.

[13] R. Fan, H. Wang, Y. Wang, M. Liu, and I. Pitas, "Graph attention layer evolves semantic segmentation for road pothole detection: A benchmark and algorithms," *IEEE Trans. Image Process.*, vol. 30, pp. 8144–8154, 2021.

[14] R. Fan and M. Liu, "Road damage detection based on unsupervised disparity map segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4906–4911, Nov. 2020.

[15] J. Li, Y. Zhang, P. Yun, G. Zhou, Q. Chen, and R. Fan, "RoadFormer: Duplex transformer for RGB-normal semantic road scene parsing," 2023, *arXiv:2309.10356*.

[16] P. Khumsap, N. Phisanbut, P. Watanapongse, and P. Piamsa-Nga, "Novel feature extractions for reflection, alligator cracks and potholes road surface classification," in *Proc. 22nd Int. Comput. Sci. Eng. Conf. (ICSEC)*, Nov. 2018, pp. 1–4.

[17] R. Fan et al., "We learn better road pothole detection: From attention aggregation to adversarial domain adaptation," in *Proc. Eur. Conf. Comput. Vis. Workshops (ECCVW)*, 2020, pp. 285–300.

[18] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "DeepCrack: Learning hierarchical convolutional features for crack detection," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1498–1512, Mar. 2019.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[20] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, "CrackTree: Automatic crack detection from pavement images," *Pattern Recognit. Lett.*, vol. 33, no. 3, pp. 227–238, Feb. 2012.

[21] B.-H. Kang and S.-I. Choi, "Pothole detection system using 2D LiDAR and camera," in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2017, pp. 744–746.

[22] X. Ma, D. Yue, S. Li, D. Cai, and Y. Zhang, "Road potholes detection from MLS point clouds," *Meas. Sci. Technol.*, vol. 34, no. 9, Jun. 2023, Art. no. 095017. [Online]. Available: https://dx.doi.org/10.1088/1361-6501/acdb8d

[23] S. Silvister et al., "Deep learning approach to detect potholes in real-time using smartphone," in *Proc. IEEE Pune Sect. Int. Conf. (PuneCon)*, Dec. 2019, pp. 1–4.

[24] Y. Tang, K. Han, J. Guo, C. Xu, C. Xu, and Y. Wang, "GhostNetv2: Enhance cheap operation with long-range attention," in *Advances in Neural Information Processing Systems*, vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. Red Hook, NY, USA: Curran Associates, 2022, pp. 9969–9982.

[25] P. Gao, J. Lu, H. Li, R. Mottaghi, and A. Kembhavi, "Container: Context aggregation network," 2021, *arXiv:2106.01401*.

[26] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," 2017, *arXiv:1704.04503*.

[27] Q. Wan, Z. Huang, J. Lu, G. Yu, and L. Zhang, "SeaFormer: Squeeze-enhanced axial transformer for mobile semantic segmentation," 2023, *arXiv:2301.13156*.

[28] J. Cao et al., "DO-Conv: Depthwise over-parameterized convolutional layer," *IEEE Trans. Image Process.*, vol. 31, pp. 3726–3736, 2022.

[29] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2016, *arXiv:1610.02357*.

[30] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT: Real-time instance segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9156–9165.

[31] S. Guo et al., "UDTIRI: An open-source intelligent road inspection benchmark suite," 2023, *arXiv:2304.08842*.

[32] J. Li, Y. Wen, and L. He, "SCConv: Spatial and channel reconstruction convolution for feature redundancy," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 6153–6162.

[33] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.

[34] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High quality object detection and instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1483–1498, May 2021.

[35] D. Bolya, C. Zhou, F. Xiao, and Y. Jae Lee, "YOLACT: Real-time instance segmentation," 2019, *arXiv:1904.02689*.

[36] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "Solo: Segmenting objects by locations," in *Proc. 16th Eur. Conf. Comput. Vis. (ECCV)*. Glasgow, U.K.: Springer, Aug. 2020, pp. 649–665.

[37] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: Dynamic and fast instance segmentation," 2020, *arXiv:2003.10152*.

[38] G. Jocher, "Ultralytics/YOLOv5: v6.0—YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," Zenodo, Version V6.0, Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5563715

[39] A. Kirillov, Y. Wu, K. He, and R. Girshick, "PointRend: Image segmentation as rendering," 2019, *arXiv:1912.08193*.

[40] Z. Tian, C. Shen, X. Wang, and H. Chen, "BoxInst: High-performance instance segmentation with box annotations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5439–5448.

[41] C. Lyu et al., "RTMDet: An empirical study of designing real-time object detectors," 2022, *arXiv:2212.07784*.

[42] T. Cheng et al., "Sparse instance activation for real-time instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 4423–4432.

[43] Z. Zong, G. Song, and Y. Liu, "DETRs with collaborative hybrid assignments training," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Jun. 2023, pp. 6748–6758.

[44] Q. Wen, J. Yang, X. Yang, and K. Liang, "PatchDCT: Patch refinement for high quality instance segmentation," 2023, *arXiv:2302.02693*.

[45] S. Woo et al., "ConvNeXt v2: Co-designing and scaling ConvNets with masked autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 16133–16142.

[46] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10778–10787.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[48] C.-Y. Wang, A. Bochkovskiy, and H. M. Liao, "Scaled-YOLOv4: Scaling cross stage partial network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13024–13033.

[49] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," 2021, *arXiv:2107.08430*.

[50] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 2778–2788.

[51] L. Rossi, A. Karimi, and A. Prati, "Recursively refined R-CNN: Instance segmentation with self-RoI rebalancing," in *Computer Analysis of Images and Patterns*. Cham, Switzerland: Springer, 2021, pp. 476–486.
[52] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 548–558.
[53] C. Li et al., "YOLOv6: A single-stage object detection framework for industrial applications," 2022, *arXiv:2209.02976*.
[54] Y. Chen et al., "Mobile-former: Bridging MobileNet and transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5260–5269.
[55] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 7464–7475.

Xiaoning Huang (Student Member, IEEE) is currently pursuing the B.S. degree in data science and big data technology with the Xingzhi College, South China Normal University, Shanwei, China.

Her main research interests include computer vision and text classification.

Jintao Cheng received the bachelor's degree from the School of Physics and Telecommunications Engineering, South China Normal University, Guangzhou, China, in 2021.

His research interests include computer vision, SLAM, and deep learning.

Qiuchi Xiang (Student Member, IEEE) is currently pursuing the B.S. degree in data science and big data technology with the Xingzhi College, South China Normal University, Shanwei, China.

His main research interests include computer vision, SLAM, and deep learning.

Jun Dong is currently pursuing the bachelor's degree in Internet of Things (IoT) engineering with the School of Data Science and Engineering, South China Normal University, Shanwei, China.

His main research interests include embedded development, artificial intelligence, and target detection.

Jin Wu (Member, IEEE) was born in Zhenjiang, China, in 1994. He received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2016. He is currently pursuing the Ph.D. degree with the Robotics and Multiperception Laboratory, The Hong Kong University of Science and Technology (HKUST), Hong Kong, under the supervision of Prof. M. Liu.

He has been a Research Assistant with the Department of Electronic and Computer Engineering, HKUST, since 2018. He has coauthored more than 140 technical papers in representative journals and conference proceedings of IEEE, AIAA, and IET. He was selected as the Top 2% Scientist by Stanford University and published by Elsevier from 2019 to 2022. His research interests include robot navigation, multisensor fusion, mechatronics, and circuitization of robotic applications.

Rui Fan (Senior Member, IEEE) received the B.Eng. degree in automation from the Harbin Institute of Technology, Harbin, China, in 2015, and the Ph.D. degree in electrical and electronic engineering from the University of Bristol, Bristol, U.K., in 2018.

He was a Research Associate with The Hong Kong University of Science and Technology, Hong Kong, from 2018 to 2020. He was a Postdoctoral Scholar-Employee with the University of California at San Diego, La Jolla, CA, from 2020 to 2021. He is currently a Full Professor with the Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, China. His research interests include computer vision, deep learning, and robotics.

Dr. Fan was named in Stanford University List of Top 2% Scientists Worldwide in 2022 and 2023.

Xiaoyu Tang (Member, IEEE) received the B.S. degree from South China Normal University, Guangzhou, China, in 2003, and the M.S. degree from Sun Yat-sen University, Zhuhai, China, in 2011. He is currently pursuing the Ph.D. degree with South China Normal University.

He is serving as an Associate Professor with the School of Physics, South China Normal University. His research interests primarily revolve around machine vision, intelligent control, and the Internet of Things.

Mr. Tang is a member of the IEEE ICICSP Technical Committee.