Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.060304

ARTICLE

An Efficient Instance Segmentation Based on Layer Aggregation and Lightweight Convolution

Hui Jin[1,2,*], Shuaiqi Xu[1], Chengyi Duan[1], Ruixue He[1] and Ji Zhang[1]

[1]College of Mechanical Engineering, Chongqing University of Technology, Chongqing, 400054, China
[2]Robot and Intelligent Manufacturing Technology, Key Laboratory of Chongqing Education Commission of China, Chongqing, 400054, China

*Corresponding Author: Hui Jin. Email: jinhui@cqut.edu.cn

Received: 29 October 2024; Accepted: 16 January 2025; Published: 26 March 2025

ABSTRACT: Instance segmentation is crucial in various domains, such as autonomous driving and robotics. However, there is scope for improvement in the detection speed of instance-segmentation algorithms for edge devices. Therefore, it is essential to enhance detection speed while maintaining high accuracy. In this study, we propose you only look once-layer fusion (YOLO-LF), a lightweight instance segmentation method specifically designed to optimize the speed of instance segmentation for autonomous driving applications. Based on the You Only Look Once version 8 nano (YOLOv8n) framework, we introduce a lightweight convolutional module and design a lightweight layer aggregation module called Reparameterization convolution and Partial convolution Efficient Layer Aggregation Networks (RPELAN). This module effectively reduces the impact of redundant information generated by traditional convolutional stacking on the network size and detection speed while enhancing the capability to process feature information. We experimentally verified that our generalized one-stage detection network lightweight method based on Grouped Spatial Convolution (GSconv) enhances the detection speed while maintaining accuracy across various state-of-the-art (SOTA) networks. Our experiments conducted on the publicly available Cityscapes dataset demonstrated that YOLO-LF maintained the same accuracy as yolov8n (mAP@0.5 = 37.9%), the model volume decreased by 14.3% from 3.259 to 2.804 M, and the Frames Per Second (FPS) increased by 14.48% from 57.47 to 65.79 compared with YOLOv8n, thereby demonstrating its potential for real-time instance segmentation on edge devices.

KEYWORDS: Automatic driving; convolution; deep learning; real-time instance segmentation

1 Introduction

Automated vehicle driving is an application of Internet of Things (IoT) technology, and a fundamental requirement for automated driving is a comprehensive understanding of the surrounding environment, including complex traffic scenes, commonly referred to as outdoor scene understanding [1]. Outdoor scene understanding is a promising topic in computer vision; thus, deep-learning-based object detection algorithms and instance segmentation algorithms can be employed to understand outdoor scenes in autonomous driving. The research results derived from these algorithms have been extensively utilized in developing various autonomous driving applications [2].

According to the network structure and detection principle, deep-learning-based object detection algorithms can be broadly classified into two categories: two-stage and one-stage object detection methods. The former initially generates candidate frames and then accurately locates these candidates through

subsequent filtering using classification and positional regression networks. Representative examples of this approach include the Region-based Convolutional Neural Networks (R-CNN) [3], Faster R-CNN [4], and Cascade R-CNN [5]. In contrast, the latter directly predicts the location and class of objects from frames by defining dense a priori or anchor frames on images and employing classification and regression networks, has the advantages of simplicity and speed, and is more suitable for real-time detection field. The YOLO family of algorithms [6–9] and single-shot detector (SSD) [10] are representative methods in this category. Both of them have more applications in transportation, but SSD has the defects of easy to confuse background in scene understanding and misdetection in some complex scenes [11], YOLO is better than SSD in this aspect, and the combination with segmentation is more suitable for applications in outdoor scene understanding.

Instance segmentation is a further step in object detection algorithms, which can be classified into two categories: two-stage instance segmentation and one-stage instance segmentation, based on the object detection network. In 2017, Kaiming et al. proposed Mask R-CNN [12], a top-down two-stage instance segmentation algorithm that builds upon the object detection network Faster R-CNN. It introduces a novel mask prediction branch to achieve instance segmentation using a simple approach and capitalizes on advancements in the object detection field. By replacing the Faster R-CNN network in Mask R-CNN with an improved detector, stable enhancements in the performance of instance segmentation can be achieved. The Cascade Mask R-CNN of Zhaowei Cai et al. is also based on this thinking for network design [13]. In 2019, significant advancements were made in the onestage instance segmentation algorithms field with the introduction of YOLACT by D. Bolya et al. [14]. This algorithm builds upon RetinaNet, an object detection network. It utilizes operations such as channel-weighted summation, convolution, or clustering to predict a set of instances using shared 1/4 or 1/8 global features. The series of articles, BlendMask [15], EmbedMask [16], and Condinst [17], are all based on the framework proposed by YOLACT. In 2020, Wang et al. proposed SOLOv1 [18], which eliminates the bbox branch of the object detection network, leaving only the classification branch and the mask branch, and experimentally verifies the feasibility of improving the network performance by replacing the backbone and the head based on the idea of YOLO series. Xing Shen et al. proposed a new mask method, DCT-Mask [19], using the discrete cosine transform (DCT) to encode high-resolution binary mesh masks as tight vectors. In 2022, Tao Zhang et al. proposed a contour-based instance segmentation method, E2EC [20], which employs a learnable contour initialization architecture and a multi-directionally aligned label sampling scheme, demonstrating state-of-the-art performance. Haoliang Liu et al. (2023) proposed the YOLO-based contour regression network YOLO-CORE [21]. In 2024, Kang et al. proposed an ASF-YOLO based on attention-scale sequence fusion and YOLO framework [22]. The two-stage instance segmentation algorithm not only inherits the disadvantage of slow detection speed of two-stage object detection, which conflicts with real-time, but also introduces the problem of information loss on spatial features of large objects, especially in the edge part, which leads to poor edge prediction of large objects, especially those with complex contours. The one-stage instance segmentation algorithm has a higher detection speed than the two-stage instance segmentation algorithm, and at the same time, there is sufficient research on the accuracy, and real-time instance segmentation becomes possible.

With the advent of the IoT, automotive chips have transitioned from microcontroller unit (MCU) to system-on-chip (SoC) heterogeneous chips, and the deployment of instance segmentation algorithms on edge devices has reached a level of maturity. Consequently, there is an increasing demand for the real-time deployment of instance segmentation algorithms on edge devices. Running the instance segmentation algorithm in real time on edge devices can be implemented in two parts, the lightweighting of the algorithm and the accelerated deployment of the lightweighted algorithm on edge devices. Thus, a lightweight network model based on YOLOv8n is proposed. The main contributions of this study are as follows:

(a) We introduce a novel lightweight convolution technology and design a lightweight layer aggregation module called RPELAN. Based on this, we propose a lightweight instance segmentation method based on yolov8n. Through experimental verification, we propose a generalized one-stage detection network lightweight method based on GSconv. In addition, our research results provide theoretical support for the full implementation of embedded platforms.

(b) By analyzing the last layer (the largest number of channels) of the one-stage backbone network, the use of traditional convolution will lead to the increase of redundant information. We propose a lightweight method of one-stage detection network based on GSconv. This characteristic offers an opportunity to explore a generalized lightweight method for one-stage detection networks.

(c) This GSconv-based lightweight method for one-stage detection networks replacing traditional convolution is validated on multiple one-stage State of the arts (SOTA) networks using the publicly available datasets Cityscapes and VOC2012, and this GSconv-based lightweight method for one-stage detection networks has generalizability.

(d) By integrating the design concepts of the C2f and ELAN modules and incorporating a lightweight convolution module, we propose a novel lightweight layer aggregation module called RPELAN. The RPELAN module enhances the gradient circulation ability through Repconv, merges multilevel features, and eliminates the redundancy caused by traditional convolutional stacking using Pconv. This approach making enabling realtime instance segmentation in edge devices possible.

(e) We deployed our algorithm on an edge device four-wheeled open source intelligent cart (ANS-OMOVE-I) and experimentally verified the performance improvement of the proposed algorithm over the original one.

2 Related Work

2.1 YOLOv8n Model

The YOLOv8 model represents a cutting-edge advancement in the field, building on the achievements of previous iterations to enhance performance and flexibility. With its emphasis on speed, accuracy, and user-friendliness, YOLOv8 emerges as an optimal solution for various tasks, including object detection, image segmentation, and pose estimation [23]. The YOLOv8 algorithm has five small and large models known as YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x.The demand for deploying instance segmentation algorithms on edge devices in autonomous driving application scenarios is high. To achieve real-time performance, the network architecture of YOLOv8n, as shown in Fig. 1, is divided into two main parts: the Backbone part extracts image features. In contrast, the Head part processes fused features and performs instance segmentation through the segment header. The YOLOv8 algorithm extensively utilizes C2f, a multi-branch stacking module corresponding to a denser residual structure. Residual networks are known for their ease of optimization, which enhances the nonlinear and representational capabilities of the network to model complex data effectively. The internal residual blocks employ jump connections to address the vanishing gradients associated with increasing depths in deep neural networks.

2.2 Lightweight Convolution

The design of lightweight convolutions aims to minimize convolutional network models' parameter count, computational complexity, and memory requirements [24]. These convolutions employ various composition techniques to reduce resource consumption, enhance computational efficiency, and maintain the model performance. in order to pursue better performance networks become deeper and deeper. Wider and wider, with a consequent significant increase in the number of parameters and computation, which is

detrimental to the deployment of algorithms on edge devices. Therefore, lightweight convolutional design is necessary.

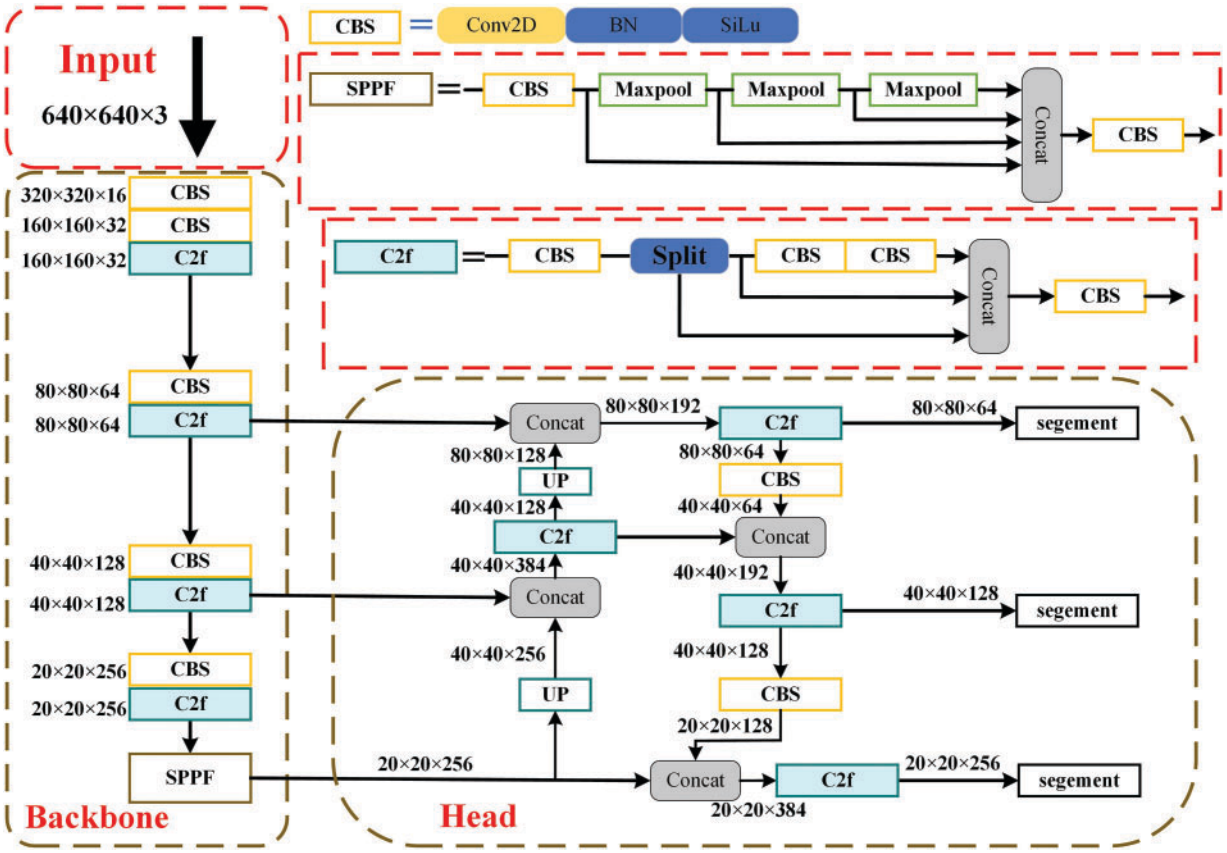

Figure 1: The network architecture of YOLOv8n

2.2.1 Repconv

Repconv is a streamlined yet robust convolutional neural network architecture comprising only 3 × 3 convolutions and Rectified Linear Unit (ReLU) activation-function stacks [25]. It employs multibranched convolutional layers during training and reparameterizes the branch parameters to the main branch during inference. This approach effectively reduces computational complexity and memory consumption, ensuring optimal learning capability during training while enhancing speed during inference.

The batch normalization (BN) layer is calculated as follows: first calculate the mean μ and variance σ of all the elements in an input feature (x_i), then subtract the mean divided by the standard deviation for x_i finally utilize the learnable parameters γ and β to carry out affine transformations to obtain the final BN output:

$$\hat{x}_i = \gamma \cdot \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta = \frac{\gamma}{\sqrt{\sigma^2 + \varepsilon}} \cdot x_i + \left(\beta - \frac{\gamma \cdot \mu}{\sqrt{\sigma^2 + \varepsilon}}\right) \tag{1}$$

Can be viewed as:

$$y = wx + b \tag{2}$$

The formula for the convolutional layer is:

$$Conv\left(x\right) = W\left(x\right) + b \tag{3}$$

The convolutional layer + BN merged:

$$\begin{aligned} \hat{x}_i &= \gamma \cdot \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \\ &= \frac{\gamma}{\sqrt{\sigma^2 + \varepsilon}} \cdot \left(W\left(x\right) + b\right) + \left(\beta - \frac{\gamma \cdot \mu}{\sqrt{\sigma^2 + \varepsilon}}\right) \\ &= \frac{\gamma}{\sqrt{\sigma^2 + \varepsilon}} \cdot W\left(x\right) + \left(\frac{\gamma \cdot \left(b - \mu\right)}{\sqrt{\sigma^2 + \varepsilon}} + \beta\right) \end{aligned} \tag{4}$$

The merged Repconv still maintains the basic formula Conv(x) = W(x) + b. This enables the fusion of 3 × 3 convolution and BN (Batch Normalization) layers.

2.2.2 GSconv

Li et al. proposed GSconv, a hybrid convolution method that combines multiple techniques and offers the advantages of depth-separable convolution while preserving all the feature information. This approach is more suitable for replacing the traditional convolution blocks directly used in neural networks. The first step involved reducing the number of channels by half using a 1 × 1 convolution to retain the feature information [26]. Subsequently, a copy was made for depth-separable convolution, and the resulting output feature maps were concatenated and subjected to a shuffle operation to prevent model bias caused by the data order, as shown in Fig. 2.

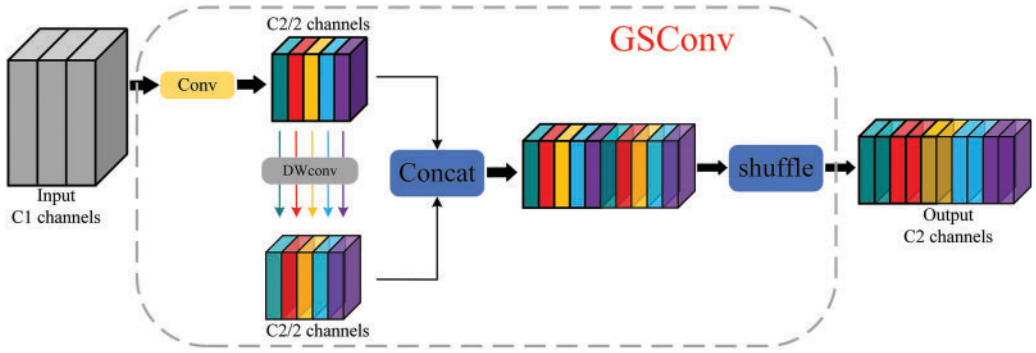

Figure 2: GSConv architecture

2.2.3 Pconv

The Pconv method proposed by Chen et al. introduced a novel partial convolution technique, as shown in Fig. 3. Numerous related studies have highlighted that the visualization of feature information from different channels in a feature map exhibits a significant level of similarity, which can be interpreted as redundant information that is neglected by traditional convolutions to ensure accurate and comprehensive processing [27]. To enhance the network learning capability, the YOLO series of detection algorithms incorporates a multi-feature information fusion module, enabling the elimination of these redundancies through Pconv for accelerated detection.

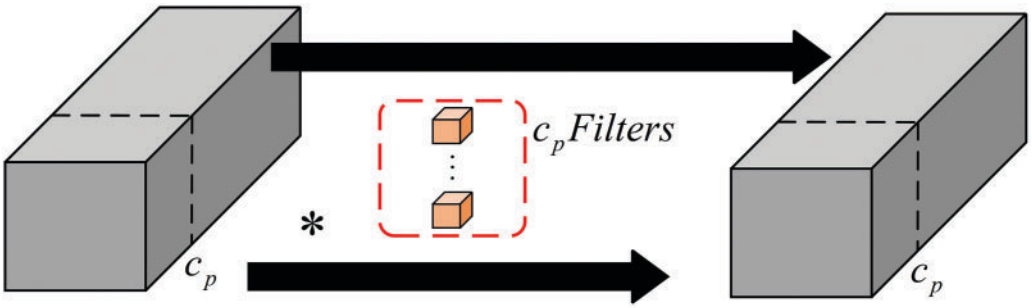

Figure 3: Pconv architecture

3 Architecture

3.1 Lightweight Network Structure

The YOLOv8 model demonstrated exceptional performance across various public computer vision datasets, whereas the YOLOv8n variant represented the most compact iteration of this model. As autonomous driving continues to advance, there is an increasing demand for real-time instance segmentation algorithms that can be deployed on edge devices, and existing instance segmentation algorithms have the potential to improve detection speed. We propose improvements to the algorithm to maintain the detection accuracy of YOLOv8n while increasing its speed. First, we introduce RPELAN as a replacement for the C2f module in the network, enabling lightweight layer aggregation and enhancing the feature-processing capability without compromising the inference speed. Second, during the investigation of the lightweight convolution module, it was observed that the final layer of the one-stage detection method backbone possessed the highest number of channels compared with the other layers in the entire network. This excessive redundancy can be effectively reduced by employing lightweight convolutions in this particular layer, thereby facilitating the overall network optimization. Based on the experimental results, we present the YOLO-LF network architecture by incorporating the RPELAN module (Table 1 and Fig. 4).

Table 1: Interlayer information in the YOLO-LF backbone

Input	Operator	Out
640 × 640 × 3	CBS	16
320 × 320 × 16	CBS	32
160 × 160 × 32	RPELAN	32
160 × 160 × 32	CBS	64
80 × 80 × 64	RPELAN	64
80 × 80 × 64	CBS	128
40 × 40 × 128	RPELAN	128
40 × 40 × 128	CBS	256
20 × 20 × 256	RPELAN	256
20 × 20 × 256	SPPF (with GSconv) CBS + Maxpool + Concat	512
20 × 20 × 512	GBS	256

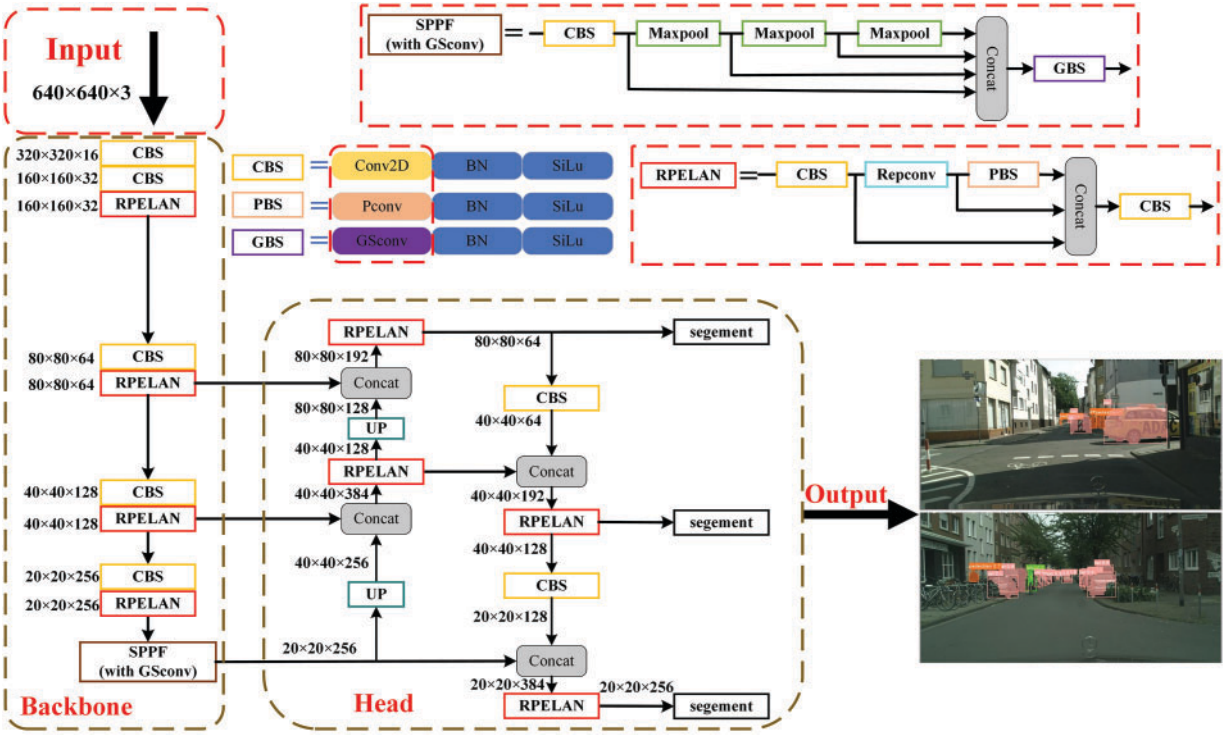

Figure 4: YOLO-LF network architecture

The inputs to each network layer are presented in Table 1 and Fig. 4. Notably, the one-stage detection algorithm incorporates the highest number of channels in the final layer of the backbone, enabling a straightforward and effective lightweight approach by employing lightweight convolutions instead of traditional ones. This strategy preserves essential features, eliminates the redundancy associated with conventional convolutional stacking, and significantly accelerates network performance. Furthermore, an application study of lightweight convolutions showed that GSconv, a hybrid convolution, is more suitable for replacing independently used traditional convolutions within the network architecture [28]. Consequently, we propose a generalized method for lightweight one-stage detection networks based on GSconv.

3.2 Lightweight Network Structure

The YOLOv8n network extensively utilizes the C2f module, which employs a Split operation to divide the features into two parts. One part of the features remains unprocessed, while the other part undergoes processing through multiple BottleNeck layers. Subsequently, all branches are concatenated and integrated using 1 × 1 convolution to in- corporate channel information [23]. By regulating the shortest and longest gradient paths, the network demonstrates efficient learning and convergence capabilities. Moreover, the ELAN module built upon this design strategy exhibited remarkable proficiency in preserving network accuracy during large-scale deployment, irrespective of the gradient pathlengths or computational block stacks [29]. As shown in Fig. 5a,b, both the C2f and ELAN modules use gradient circulation branching to achieve hierarchical information circulation.

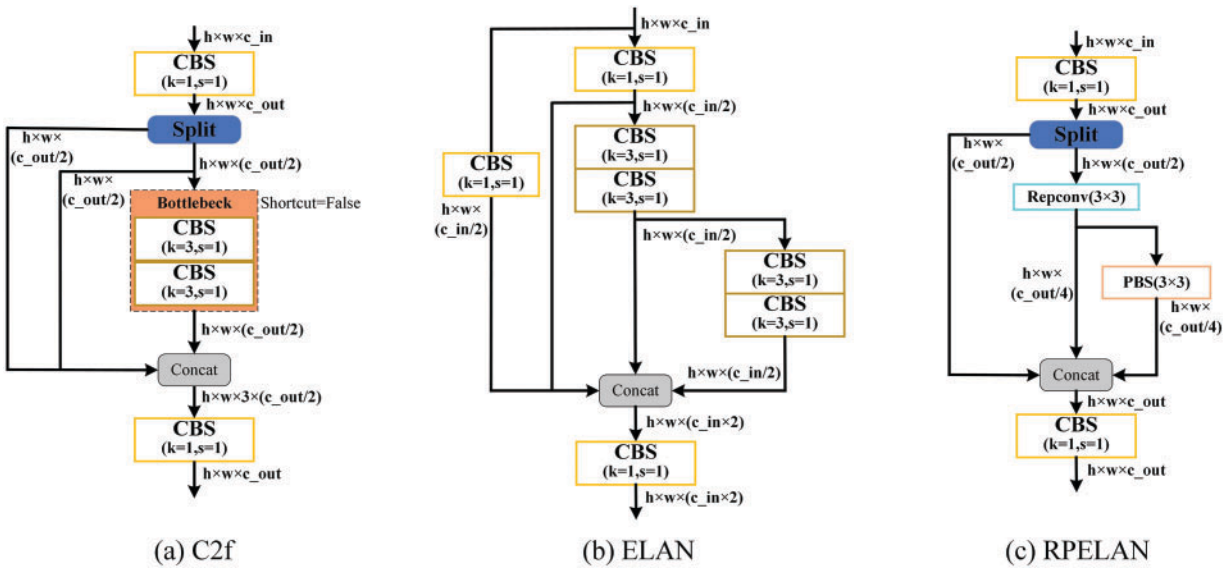

Figure 5: Lightweight Layer Aggregation Network RPELAN.RPELAN incorporates the Repconv and Pconv lightweight convolution modules based on ELAN's gradient flow branching and C2f architecture. This design effectively eliminates the redundancy generated by traditional convolution while enhancing the feature extraction capability through gradient flow branching

This study proposes the RPELAN, a lightweight layer aggregation network, as illustrated in Fig. 5c. The ELAN module employs conventional convolutions extensively, whereas the ELAN's gradient flow branch is designed as a gradient path to prevent excessive stacking of conventional convolutions. To eliminate redundancy caused by such stacking, PConv replaces the conventional convolutions on the gradient flow branch. Meanwhile, to compensate for the performance degradation resulting from discarding the residual block, RepConv processes a subset of features split by Split, thereby enhancing feature extraction capability and gradient propagation. Repconv enables the parameter fusion of specific structural components as a model re-referencing technique, consolidating multiple computational modules into one during the inference stage, thus achieving an improved inference speed without sacrificing accuracy. The RPELAN network exhibits superior efficiency and agility; therefore, in this study, we substituted YOLOv8n's C2f module with RPELAN.

4 Experimental Results

The timeliness of the proposed lightweight instance segmentation method is validated and analyzed in conjunction with the experimental results in this section. First, we investigated the impact on model performance by replacing traditional convolution with GSconv at a fixed position for different one-stage object detection and instance segmentation algorithms. Second, we compare our approach with the state-of-the-ar model and provide experimental results and analysis. Finally, the effectiveness of the improved method was verified through ablation experiments.

4.1 Datasets

The Cityscapes dataset [30] is a high-quality dataset widely utilized in computer vision, primarily for the semantic segmentation of urban street-view images and object detection tasks. This dataset was collaboratively developed by multiple esteemed universities and research institutes in Germany to promote advancements in autonomous driving and transportation systems.

The Cityscapes dataset comprises 5000 high-resolution Street View images collected from 50 diverse cities, with a split of 2975 for training, 500 for validation and 1525 for testing. Professionals meticulously annotated each image at the pixel level to encompass detailed information across 30 categories; however, only 19 categories were used for evaluation purposes. These annotations cover crucial elements relevant in real-world traffic scenarios, such as roads, sidewalks, buildings, traffic signs, vehicles, and pedestrians. Because of its exceptional image quality (1024 × 2048 pixels) and comprehensive annotation scheme, this dataset is widely acknowledged as a benchmark for semantic segmentation and scene-understanding tasks.

The PASCAL VOC 2012 dataset [31] is a pivotal benchmark in computer vision, encompassing crucial tasks such as object detection, image segmentation, and classification. This dataset was graciously provided by the Pattern Analysis, Statistical Modelling and Computational Learning (PASCAL) Challenge organizing team with the primary objective of fostering advancement and assessment of cutting-edge computer vision algorithms.

The VOC 2012 dataset comprises 20 object categories encompassing humans, animals, vehicles, and everyday objects that represent common scenarios in real-world applications. It is partitioned into training, validation, and test sets. The training and validation sets comprised 11,530 labeled images, and the number of images in the test set remained undisclosed for challenge evaluation purposes. Each image was accompanied by a detailed bounding box and pixel-level segmentation annotations to facilitate diverse research tasks. The dataset's meticulous annotation quality and diversity rendered it an essential benchmark for evaluating object detection algorithms.

4.2 Training Details of Model

The experiments were conducted in a Python 3.8, PyTorch 1.13.1, CUDA 11.7, Torchvision 0.14.1 environment with an Intel Xeon W2223, 16 G DDR4 2666 mhz RAM, DELL Precision Tower 5820 graphic workstation, and a graphics card with NVIDIA GeForce RTX 2080ti with 11 G RAM, depicted in Table 2.

Table 2: Details of the training platform

Equipment	Name
Hardware enviroment	CPU: Intel Xeon W2223; Memory:16 G DDR4 2666 mhz RAM; GPU: NVIDIA GeForce RTX 2080ti 11 G
Software enviroment	Windows10; Pycharm2023.1.1
Network framework	PyTorch 1.13.12; CUDA 11.7; Torchvision 0.14.1
Programming language	Python 3.8

Transfer learning is employed in the network training process to ensure stable and efficient training and enhance the overall training outcomes. Given the potential challenges associated with expensive or limited availability of training data, it is imperative to develop high-performance learners by leveraging readily accessible data from diverse domains. This approach is commonly referred to as migration learning [32]. The weights utilized for migration learning are commonly referred to as pretraining weights. This approach effectively mitigates the cost of data labeling and training time while enhancing the model performance and generalization, rendering it a practical and scalable solution for real-world applications. Training a neural network typically requires multiple iterations to optimize the model parameters, where the batch size denotes the number of training samples employed in a single iteration.

The RGB image inputs to the network were resized to 640 × 640 pixels. The network weights were initialized using pre-trained weights from the COCO2017 dataset, and a batch size of 16 was set. The training process continued until there was no significant improvement in the model performance for more than 50 epochs, with a maximum of 300 epochs. An Adam W optimizer was employed. All the algorithms and ablation experiments in this study were conducted under identical experimental conditions and training parameters to ensure a fair comparison.

4.3 Evaluation Indicators of Model

The following evaluation metrics were used: parameters, FLOPs, mean average precision (mAP), detection speed, and model size.

Precision is the proportion of samples classified as positive categories that are truly positive:

$$precision = \frac{TP}{TP + FP} \times 100\% \tag{5}$$

The proportion of samples in which the recall is truly a positive category is correctly categorized as follows:

$$recall = \frac{TP}{TP + FN} \times 100\% \tag{6}$$

The mean average precision (mAP) is calculated as follows:

$$AP = \int_0^1 P(R)\, dR \tag{7}$$

$$mAP = \frac{\sum_{C}^{i=1} AP_i}{C} \tag{8}$$

The detection speed was quantified in (FPS). In the given equation, TP represents true positive cases (the number of samples correctly predicted by the model as positive categories), FP represents false positive cases (the number of samples incorrectly predicted by the model as positive categories), FN represents false negative cases (the number of samples incorrectly predicted by the model as negative categories), C denotes the total number of categories, and AP denotes average precision, where P signifies precision, R signifies recall, and $P(R)$ indicates the level of recall achieved at a specific level of precision [31].

4.4 Experiment Results and Analysis

4.4.1 Experimental Results of a Generalized Lightweight Method Based on Gsconv

Experiments were conducted to investigate the generalization of GSconv when applied to networks, as presented in Table 3. The number of input channels in the final layer of the one-stage detection algorithm backbone, as described in Section 3.1, attains its maximum among all layers, where the utilization of GSconv effectively reduces network complexity without altering its structure. The segmentation effects are validated on the Cityscapes dataset, and the experimental results demonstrate slight fluctuations in mAP@0.5, which remain within the ±0.3% range of network training error. However, there was a significant improvement in the frames per second (FPS). Object detection effects were evaluated on the VOC2012 dataset, revealing enhanced accuracy and a notable increase in FPS. Therefore, incorporating GSconv into the final layer of the one-stage detection algorithm backbone exhibits remarkable potential for enhancing detection speed while maintaining accuracy. Empirical evidence from various state-of-the-art algorithms

further confirms the versatility of this lightweight approach based on GSconv. This plays a crucial role in real-time instance segmentation for autonomous driving and aligns with the desired outcome of lightweight one-stage detection algorithms.

Table 3: Results of a generalized one-stage detection network lightweighting method in multiple networks

Models	mAP(box)@0.5	mAP(mask)@0.5	Inference time (ms)	FPS	FLOPs/G
Validation of a GSconv-based generalized one-stage detection network light weighting approach using the Cityscapes dataset					
YOLOv8n	41%	37.9%	17.4	57.47	12.0
YOLOv8n with GSconv	41.1% (+0.1%)	37.6% (−0.3%)	16.3	61.35 (+6.75%)	11.9
YOLOv5n	36.7%	32.8%	21.8	45.87	6.8
YOLOv5n with GSconv	36.8% (+0.1%)	32.7% (−0.1%)	20.9	47.85 (+4.32%)	6.7
YOLOv11	41.8%	37.8%	19.3	51.81	10.1
YOLOv11 with GSconv	41.7% (−0.1%)	37.7% (−0.1%)	18.4	54.35 (+4.9%)	10.0
YOLOv5s	42.5%	38.8%	29.1	34.36	25.9
YOLOv5s with GSconv	42.2% (−0.3%)	38.6% (−0.2%)	25.8	38.76 (+12.8%)	25.7
Validation of a GSconv-based generalized one-stage detection network lightweighting approach using the VOC2012 datase					
YOLOv7tiny	65.2%	–	15.6	64.10	13.2
YOLOv7tiny with GSconv	66.2% (+1.0%)	–	14.9	67.11 (+4.696%)	13.1

4.4.2 Comparison with Other Algorithms

The effectiveness of the acceleration achieved by utilizing the reduced redundancy lightweight layer aggregation network RPELAN and the lightweight convolutional module, compared with YOLOv8n, YOLOv11, MDet-ins-tiny and YOLOv5s, was verified by training a model on the Cityscapes dataset. Subsequently, the algorithm's performance was evaluated, and the results are presented in Table 4 and Fig. 6.

As shown in Table 4, the proposed method maintained a high mAP value while achieving a smaller model size and faster detection speed than the other algorithms. Specifically, compared to the YOLOv11, the parameter and mAP value are basically the same, but the FPS of our method is significantly higher, and compared toYOLOv8n, our method improved the FPS by 14.48%, reduced the parameters by 14.02%, and decreased the FLOPs by 13.33%. Although YOLOv5s and RTMDet-ins-tiny have a slightly higher mAP value in the experiment, this comes at the cost of an increase in the number of parameters by more than 100% of our method and a reduction in detection speed of more than 50%. Moreover, as demonstrated in Fig. 6, our approach exhibits greater stability and efficiency across different features during detection with lower misrecognition ratesowing to its lightweight layer aggregation network, which reduces redundancy and employs GSconv convolutional module acceleration.

Table 4: Results of different networks on the Cityscapes dataset

Models	mAP(mask)@0.5	Inference time (ms)	FPS	Parameter/M	FLOPs/G
YOLOv8n	37.9%	17.4	57.47	3.259	12.0
YOLOv11	37.8% (−0.1%)	19.3	51.81 (−9.85%)	2.759 (−15.34%)	10.1 (−15.83%)
RTMDet-ins-tiny	38.5% (+0.6%)	47.4	21.1 (−63.29%)	5.616 (+72.32%)	11.877 (−1.02%)
YOLOv5s	38.8% (+0.9%)	29.1	34.36 (−40.21%)	7.412 (+127.43%)	25.9 (+115.83%)

(Continued)

Table 4 (continued)

Models	mAP(mask)@0.5	Inference time (ms)	FPS	Parameter/M	FLOPs/G
YOLO-LF	37.9% (+0%)	15.2	65.79 (+14.48%)	2.802 (−14.02%)	10.4 (−13.33%)

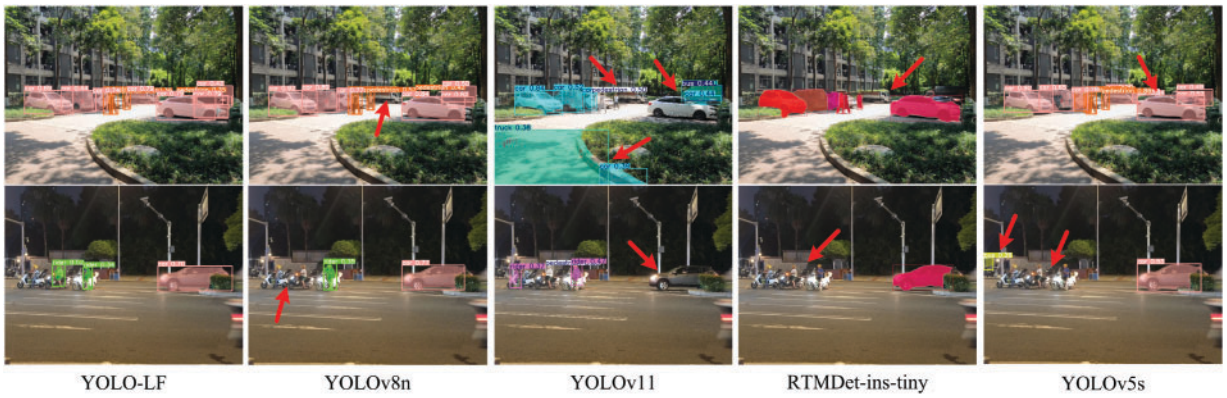

Figure 6: The experimental results of the comparison of different instance segmentation algorithm

4.4.3 Ablation Experiment

This section discusses the effects of the improved method on the network model. The corresponding data are listed in Table 5. Four experiments were conducted by incorporating different modules to evaluate the network's performance using metrics such as mAP, GFLOPs, parameters, and FPS. To reduce network complexity, GSconv was employed for lighting. After acceleration with GSconv (referred to as GS), RPELAN was included in the network to eliminate redundant information and ensure the complete fusion of feature information across stages. To ensure experimental fairness, official pre-training weights were utilized for each experiment, with a fixed number of epochs set at 300.

Table 5: Effects of various designs module on instance segmentation

Models	mAP(mask)@0.5	FPS	FLOPs/G	Parameter/M
YOLOv8n	37.9%	57.47	12.0	3.259
YOLOv8n+GS	37.6%	61.35 (+6.75%)	11.9	3.397
YOLOv8n+RPELAN	38.4% (+0.5%)	62.89 (+9.43%)	11.2 (−6.67%)	3.007 (−7.73%)
YOLOv8n+GS+RPELAN	37.9%	65.79 (+14.48%)	10.4 (−13.33%)	2.802 (−14.03%)

As presented in Table 5, when replacing the traditional convolution in the backbone with GSconv, the network exhibited a significant enhancement in detection speed, resulting in a 6.75% increase in frames per second (FPS) while maintaining the same mAP(mask)@0.5. Moreover, replacing C2f with RPELAN resulted in a 0.5% improvement in mAP(mask)@0.5 and an impressive 9.43% improvement in FPS. Ultimately, these modifications led to a remarkable overall improvement of 14.48% in FPS and a 14.03% reduction in the parameter count.

4.4.4 Edge Device Deployment Experiment

O-MOVE four-wheeled mobile cart is Chongqing Anisen Intelligent Technology Co., Ltd.'s independent development of teaching and competition equipment. O-MOVE uses McNamee wheel four-wheel drive mode, can complete the left and right translation, *in situ* rotation, and diagonal movement, in the narrow space moving and through the performance is excellent. The bottom driver is based on the STM32F103 multi-channel motor driver board. The main controller is an Intel NUC kit, running Ubuntu 16.04 with ROS Kinetic. Additionally, reserved interfaces for ultrasonic sensors and Bluetooth are included. The system has an Intel RealSense D415 depth camera, offering robust and stable performance. We deployed the proposed YOLO-LF and the original algorithm YOLOv8n to the O-MOVE four-wheeled mobile cart and tested them respectively, and the specific results are shown in Table 6. Our method improves the detection speed by 8.43% while keeping the map largely unchanged on the edge device. It further validates the feasibility of our proposed lightweight approach for deployment on edge devices.

Table 6: Experimental results on edge devices (intel NUC)

Models	mAP(mask)@0.5	Inference time (ms)	FPS	Parameter/M
YOLOv8n	38.2%	60.1	16.6	3.259
YOLO-LF	38.1%	55.7	18.0	2.802

5 Conclusion and Future Work

A lightweight instance segmentation method was proposed in this study. Building upon the YOLOv8n network model, a series of measures were implemented to enhance the speed of instance segmentation while maintaining accuracy. A lightweight layer aggregation module called RPELAN was devised to eliminate redundancy and effectively learn the channel and spatial features at different scales. Additionally, a general one-stage detection network lightweight approach based on GSconv was introduced by incorporating the concept of lightweight convolution modules. The versatility of this method was validated through experiments conducted on both Cityscapes and VOC2012 datasets. The results obtained from experiments on the Cityscapes dataset demonstrate that our proposed lightweight instance segmentation method significantly improves segmentation speed without compromising accuracy. Furthermore, it enables image archiving and location recording for various segmentation classes in autonomous driving applications, with potential deployment on real-time edge devices. In future research, it will be essential to investigate acceleration techniques for deploying lightweight algorithms on embedded platforms while considering practical application scenarios to gather more effective data for enhancing the overall instance segmentation categories.

Acknowledgement: None.

Funding Statement: This work was supported by Science and Technology Research Youth Project of Chongqing Municipal Education Commission (No. KJQN202301104), Cooperative Project between universities in Chongqing and Affiliated Institutes of Chinese Academy of Sciences (No. HZ2021011), Chongqing Municipal Science and Technology Commission Technology Innovation and Application Development Special Project (No. 2022TIAD-KPX0040), Action Plan for Quality Development of Chongqing University of Technology Graduate Education (Grant No. gzlcx20242014).

Author Contributions: Study conception and design: Hui Jin, Chengyi Duan; data collection: Shuaiqi Xu; analysis and interpretation of results: Shuaiqi Xu, Ji Zhang; draft manuscript preparation: Shuaiqi Xu, Ruixue He. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available upon reasonable request from the authors.

Ethics Approval: Not applicable. This study did not involve human or animal subjects.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Janai J, Güney F, Behl A, Geiger A. Computer vision for autonomous vehicles: problems, datasets and state of the art. Foundat Tren Comput Graph Vis. 2020;12(1–3):1–308. doi:10.1561/0600000079.
2. Fang S, Zhang B, Hu J. Improved mask R-CNN multi-target detection and segmentation for autonomous driving in complex scenes. Sensors. 2023;23(8):3853. doi:10.3390/s23083853.
3. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2014 Sep; Columbus, OH, USA: IEEE. p. 580–7.
4. Girshick R. Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision; 2015 Dec; Santiago, Chile: IEEE. p. 1440–8.
5. Cai Z, Vasconcelos N. Cascade R-CNN: delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun; Salt Lake City, UT, USA: IEEE. p. 6154–62.
6. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun; Las Vegas, NV, USA: IEEE. p. 779–88.
7. Redmon J, Farhadi A. YOLOv3: an incremental improvement. arXiv:1804.02767. 2018.
8. Li C, Li L, Jiang H, Weng K, Geng Y, Li L, et al. YOLOv6: a single-stage object detection framework for industrial applications. arXiv:2209.02976. 2022.
9. Wang CY, Bochkovskiy A, Liao H-YM. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2023 Jun; Vancouver, BC, Canada: IEEE. p. 7464–75.
10. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, et al. SSD: single shot multibox detector. In: Computer Vision—ECCV 2016: 14th European Conference; 2016 Oct 11–14; Amsterdam, The Netherlands; 2016. p. 21–37.
11. Chen ZC, Guo HQ, Yang J. Fast vehicle detection algorithm in traffic scene based on improved SSD. Measurement. 2022;201(7):111655. doi:10.1016/j.measurement.2022.111655.
12. He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision; 2017 Oct; Venice, Italy: IEEE. p. 2961–9.
13. Cai Z, Vasconcelos N. Cascade R-CNN: high quality object detection and instance segmentation. IEEE Trans Pattern Anal Mach Intell. 2019;43(5):1483–98. doi:10.1109/TPAMI.2019.2956516.
14. Bolya D, Zhou C, Xiao F, Lee YJ. YOLACT: real-time instance segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision; 2019 Oct; Seoul, Republic of Korea: IEEE. p. 9157–66.
15. Chen H, Sun K, Tian Z, Shen C, Huang Y, Yan Y. BlendMask: top-down meets bottom-up for instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun; Seattle, WA, USA: IEEE. p. 8573–81.
16. Ying H, Huang Z, Liu S, Shao T, Zhou K. EmbedMask: embedding coupling for one-stage instance segmentation. arXiv:1912.01954. 2019.
17. Tian Z, Shen C, Chen H. Conditional convolutions for instance segmentation. In: Computer Vision—ECCV 2020: 16th European Conference; 2020 Aug 23-28; Glasgow, UK; 2020. p. 282–98.
18. Wang X, Kong T, Shen C, Jiang Y, Li L. SOLO: segmenting objects by locations. In: Computer Vision—ECCV 2020: 16th European Conference; 2020 Aug 23–28; Glasgow, UK; 2020. p. 649–65.

19. Shen X, Yang J, Wei C, Deng B, Huang J, Hua X-S, et al. DCT-Mask: discrete cosine transform mask representation for instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021 Jun; Nashvile, TN, USA: IEEE. p. 8720–9.
20. Zhang T, Wei S, Ji S. E2EC: an end-to-end contour-based method for high-quality high-speed instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022 Jun; New Orleans, LA, USA: IEEE. p. 4443–52.
21. Liu H, Xiong W, Zhang Y. YOLO-CORE: contour regression for efficient instance segmentation. Mach Intell Res. 2023;20(5):716–28. doi:10.1007/s11633-022-1379-3.
22. Kang M, Ting C-M, Ting FF, Phan RC-W. ASF-YOLO: a novel YOLO model with attentional scale sequence fusion for cell instance segmentation. Image Vis Comput. 2024;147(4):105057. doi:10.1016/j.imavis.2024.105057.
23. Jocher G, Chaurasia A, Qiu J. OLOv8 [Online]. Version 8.0.0. License: AGPL-3.0; 2023 [cited 2025 Jan 15]. Available from: https://github.com/ultralytics/ultralytics.
24. Yan Z, Chen S, Wang Y, Huanl W. Review of research on lightweight convolutional neural networks. In: 5th Information Technology and Mechatronics Engineering Conference (ITOEC); 2020 Jun; Chongqing, China: IEEE. p. 1713–20.
25. Ding X, Zhang X, Ma N, Han J, Ding G, Sun J. RepVGG: making VGG-style ConvNets great again. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021 Jun; Nashville, TN, USA: IEEE. p. 13733–42.
26. Li H, Li J, Wei H, Liu Z, Zhan Z, Ren Q. Slim neck by GSConv: a better design paradigm of detector architectures for autonomous vehicles. arXiv:2206.02424. 2022.
27. Chen J, Kao SH, He H, Zhuo W, Wen S, Lee C-H, et al. Run, don't walk: chasing higher FLOPS for faster neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2023 Jun; Vancouver, BC, Canada: IEEE. p. 12021–31.
28. Jin H, Duan C, Lu S, Ding J, Xu S, Zhang J. A more efficient method for wire rope surface defect detection based on fusing cross-stage features and lightweight convolution modules. Meas Sci Technol. 2024 Jun;35(9):095406. doi:10.1088/1361-6501/ad4e56.
29. Wang CY, Liao H-Y, Yeh I-H. Designing network design strategies through gradient path analysis. arXiv:2211.04800. 2022.
30. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, et al. The Cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun; Las Vegas, NV, USA: IEEE. p. 3213–23.
31. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A. The pascal visual object classes (VOC) challenge. Int J Comput Vis. 2010;88(2):303–38. doi:10.1007/s11263-009-0275-4.
32. Wimmer P, Mehnert J, Condurache AP. Dimensionality reduced training by pruning and freezing parts of a deep neural network: a survey. Artif Intell Rev. 2023;56(12):14257–95. doi:10.1007/s10462-023-10489-1.