2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

Faster-YOLOv6: A Lightweight Target Detection Algorithm Optimized for Edge Computing

Xiaojun Li
Novasky Technology Company Limited by Shares
Changsha, China

Xin He *
Novasky Technology Company Limited by Shares
Changsha, China
* Corresponding author: hexin@novasky.cn

Yuhui Deng
Novasky Technology Company Limited by Shares
Changsha, China

Peng Luo
Novasky Technology Company Limited by Shares
Changsha, China

***Abstract*—With the rapid advancement of edge computing, deploying target detection algorithms on resource-constrained devices has become a key challenge in computer vision. To address this, we propose Faster-YOLOv6, a lightweight target detection algorithm based on YOLOv6. First, we introduce a Partially Reparameterized Convolutional Module (PRepConv), an enhancement of the partially convolutional module (PConv) using reparameterization. This lightweight module improves feature extraction while reducing computational cost. Next, we design Faster RepBlock, a lightweight feature extraction module that enhances cross-channel information fusion and feature interaction, optimizing the network's efficiency. We replace the original RepBlock in YOLOv6 with Faster RepBlock, reducing redundant computations and memory access. Our experiments on a custom dataset deployed on a Jetson Nano show that Faster-YOLOv6 achieves a 40% faster inference speed compared to the baseline model, with no loss in accuracy. The improved model outperforms other mainstream detection algorithms in terms of speed and efficiency for edge computing.**

Keywords: Edge Computing; YOLOv6; Lightweight Algorithm

I. INTRODUCTION

With the rapid development of IoT technology, the You Only Look Once (YOLO) algorithm has become widely adopted in computer vision due to its high accuracy, fast detection speed, and flexible architecture. Over multiple iterations, YOLO has evolved into its current version, YOLOv11[1]. However, with increased accuracy comes a significant rise in model size, posing challenges for deployment on edge devices constrained by limited computing power, memory, and energy efficiency.

To overcome these challenges, researchers have introduced a variety of lightweight optimization methods aimed at reducing computational resource consumption while maintaining detection accuracy. These advancements enable lightweight YOLO algorithms to be deployed on embedded edge devices, ensuring high real-time performance and enhanced security. In recent years, lightweight network architectures such as MobileNet V1[2], V2[3], and V3[4], ShuffleNet V1[5] and V2[6], GhostNet[7], and FasterNet[8] have provided new avenues for model optimization. For instance, Sun et al. (2023) proposed an efficient pear detection method[9] based on YOLOv5[10], replacing standard convolution with the Shuffle module, introducing the CBAM[11] attention mechanism, and designing a weighted confidence loss function. This approach improved small-target detection, achieving a 97.6% mAP and compressing the model size by 39.4%. Similarly, [12] integrated FasterNet with YOLOv8s[13], reducing model parameters by 24% while improving accuracy. [14] enhanced YOLOv7[15] using Group Convolution, ShuffleNetV2, and Vision Transformer[16], which significantly decreased parameter count and memory usage, improving the real-time detection capabilities of edge devices. On another front, Liu et al. proposed a lightweight apple detection algorithm, Faster-YOLO-AP[17], based on YOLOv8. They developed the PDWFasterNet module, leveraging partial deep convolution to simplify the network, reduce computational load, and minimize model parameters. Meanwhile, [20] introduced a lightweight multi-scale citrus detection model built upon YOLOv7. Their approach incorporated a small-target detection layer, replaced standard convolution with GhostConv, added CBAM modules after each ELAN block, and utilized four reparameterized convolution modules (RepConv) to achieve a single-head network reparameterization. This model achieved a remarkable 97.29% mAP in citrus detection tasks. Despite these advancements, several limitations remain. Attention mechanisms improve detection accuracy but often increase computational overhead. Lightweight backbone networks can enhance inference speed but risk compromising accuracy. While pruning and knowledge distillation methods accelerate inference, they often involve complex parameter tuning and produce uncertain outcomes. Moreover, practical validation of these methods on edge devices, such as Jetson platforms, is still lacking in current research.

To address these challenges, this paper introduces Faster-YOLOv6, a lightweight target detection algorithm optimized from YOLOv6[18] to better suit edge computing devices. First, we enhance the PConv module by employing a reparameterization technique, resulting in the proposed PRepConv, which significantly improves both feature extraction capability and hardware inference efficiency. Next, we design the Faster RepBlock, built upon PRepConv, to replace the original RepBlock module in YOLOv6. This replacement minimizes redundant computations and storage accesses, further optimizing the network for lightweight design and enhancing operational efficiency.

This work is supported by The science and technology innovation Program of Hunan Province, Project No. 2023RC3230.

979-8-3315-0976-7/25/$31.00 ©2025 IEEE

II. RELATE WORK

A. YOLOv6

As one of the most advanced target detection algorithms available today, YOLOv6 excels in maintaining real-time detection capabilities while achieving high detection accuracy. These attributes are particularly critical for industrial applications that demand both efficiency and precision. YOLOv6 introduces EfficientRep, an efficient reparameterized backbone network inspired by RepVGG[19]. During the training phase, the backbone primarily consists of RepBlocks, which are converted into RepConv stacks (Conv3×3 + ReLU) for the inference phase. This transformation leverages the high computational density and optimization of 3×3 convolutions on mainstream GPUs and CPUs, maximizing hardware utilization and significantly reducing inference latency while enhancing feature representation. Furthermore, YOLOv6 incorporates a bi-directionally spliced Neck network, the RepBi-PAN, and optimizes decoupling heads based on the detection head designs of FCOS[21] and YOLOX[22]. These improvements reduce computational costs and further lower inference latency, making YOLOv6 a highly efficient and practical choice for real-world applications.

B. Lightweight network

In recent years, the design of lightweight convolutional neural networks has gained significant attention due to the growing demand for deploying models on resource-constrained end devices. Early pioneers such as SqueezeNet[23] achieved substantial reductions in computation by replacing 3×3 convolutions with 1×1 convolutions in many layers and adjusting the input channels of large-scale convolutional layers. MobileNet V1 introduced Depthwise Separable Convolutions, enabling efficient modeling with fewer parameters and lower computational cost. ShuffleNet V2 further optimized the structural design for computation-intensive operations, based on hardware performance analysis of both ShuffleNet V1 and MobileNet V2, and demonstrated superior performance on both GPU and ARM platforms. FasterNet took a different approach by leveraging feature map redundancy and incorporating partial convolution (PConv). This method reduces computational complexity and memory access by processing only a subset of the channels, while still preserving model accuracy and spatial feature extraction capabilities. As a result, FasterNet is particularly well-suited for efficient deployment in edge computing and embedded device scenarios.

III. NETWORK ARCHITECTURE

To address the trade-off between accuracy and speed for embedded edge computing devices, we propose the Faster YOLOv6 target detection algorithm, using YOLOv6 as the baseline model. Our approach significantly enhances inference speed while minimizing accuracy loss. Specifically, we begin by reevaluating the efficiency of PConv convolution and designing a more efficient alternative, the Partially Reparameterized Convolution (PRepConv). Next, we replace PConv in the FasterNet Block with PRepConv as the core operator, optimizing the structure to introduce a lightweight and efficient feature extraction module, Faster RepBlock. Finally, we incorporate Faster RepBlock to replace the RepBlock module in YOLOv6, resulting in a lighter Backbone and Neck. The structure of the Faster YOLOv6 network is shown in Fig. 1.

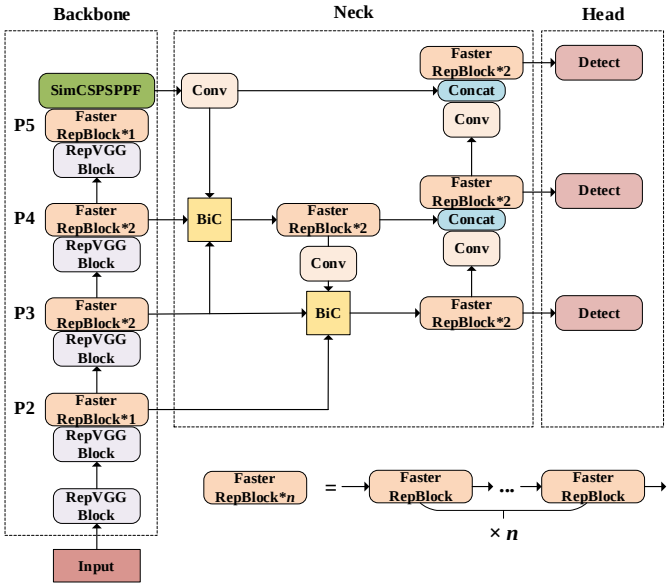

Figure 1. Faster YOLOv6 Network Architecture

A. PRepConv

PConv extracts features by applying standard 3×3 convolution to a subset of the input channels, leaving the remaining channels unchanged, as illustrated in Fig. 2(a). This approach effectively reduces computational and memory access costs while maintaining overall feature representation. However, directly applying 3×3 convolution to certain channels has inherent limitations. To address this, we propose an improved module called the Partially Reparameterized Convolution (PRepConv).

As illustrated in Fig. 2, Figure (b) presents the structure of PRepConv, while Figures (c) and (d) depict the workflow of the RepVGG module during the training and inference phases, respectively. In the training phase, PRepConv incorporates the RepVGG module, which performs convolution operations on a subset of the input channels via three branches: BN, Conv1×1 + BN, and Conv3×3 + BN. The outputs of these branches are then concatenated with the unprocessed channels. Unlike PConv, which applies convolution to only 1/4 of the input channels, PRepConv optimizes this process by convolving 1/2 of the input channels, striking a better balance between model accuracy and inference speed. During the inference phase, the three branches are merged into a single RepConv using a reparameterization technique. This fusion significantly enhances inference efficiency and improves feature expression capabilities, achieving inference speeds comparable to those of PConv.

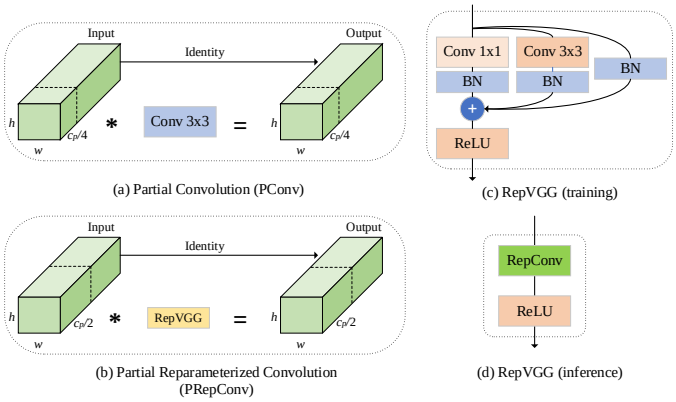

Figure 2. (a) Schematic Diagram of the PConv Structure; (b) Schematic Diagram of the PRepConv Structure; (c) Training Structure of the RepVGG Module; (d) Inference Structure of the RepVGG Module.

B. *Faster RepBlock*

Relying solely on PRepConv makes it challenging to fully exploit the effective features across different channels. To address this limitation, we propose the Faster RepBlock module, which incorporates PRepConv as its core operator. PRepConv first performs feature extraction on 1/2 of the input channels and concatenates the processed output with the remaining unprocessed channels, as shown in Fig. 3.

Building on this, the first ConvModule in Faster RepBlock expands the number of channels to twice the original size using Pointwise Convolution (PWConv), followed by Batch Normalization (BN) and ReLU activation. The second ConvModule then reduces the channels back to their original dimension. Finally, the output of the second ConvModule is merged with the input residuals of the Faster RepBlock via an addition operation. This structure facilitates efficient information fusion and interaction among different channels, significantly enhancing the model's feature representation capability.

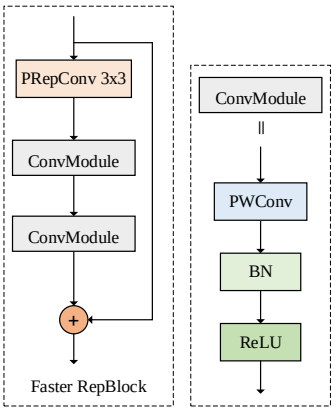

Figure 3. Faster RepBlock schematic

C. *Faster Backbone and Faster Neck*

As illustrated in Fig. 1, this paper integrates the Faster RepBlock into YOLOv6, replacing the original RepBlock in both the Backbone and Neck to construct a lighter and more efficient feature extraction and fusion network. Specifically, based on the stacked configuration (1, 2, 3, 1) of P2 to P5 RepBlocks in YOLOv6, we substitute these with 1, 2, 2, and 1 Faster RepBlock modules, respectively, in the Backbone. Similarly, in the Neck, we replace the original RepBlock modules in the P3 to P5 layers with stacked Faster RepBlock modules, configured as either all 1 or all 2 per layer.

These enhancements significantly improve the efficiency of both the Backbone and Neck modules by reducing computational complexity and boosting inference speed. At the same time, they effectively maintain the network's feature extraction and fusion capabilities, ensuring robust performance in real-time applications.

IV. EXPERIMENT

A. *Setups*

This paper presents an algorithm designed for edge device deployment in industrial scenarios. We collected and annotated a dataset of over 18,000 images from security environments like prisons, airports, and other surveillance areas. The dataset was split into training, validation, and test sets (8:1:1 ratio) and includes four categories: people, cars, cats, and birds. The model was trained on a high-performance deep learning server and deployed on NVIDIA Jetson Nano devices for inference using the TensorRT framework, with models converted to *.engine format and run at FP16 precision.

B. *Ablation Study and Analysis*

To validate the performance of Faster-YOLOv6, the lightweight target detection algorithm proposed in this paper, we deployed the models on an NVIDIA Jetson Nano device, converting them all to FP16 precision. Table I compares the performance of the YOLOv6n baseline model with two improved models. The experiments evaluate the performance of each model based on detection accuracy (mAP@0.5), number of parameters (Params), and inference speed (FPS).

TABLE I. STYLES ABLATION STUDY

Model	mAP@0.5/%	Params/MB	FPS
YOLOv6n	75.6	15.48	25
Faster-YOLOv6n-1	74.8	7.47	37
Faster-YOLOv6n-2	75.6	9.60	35

As shown in Table I, to strike a balance between lightweighting and detection performance, the Faster-YOLOv6n-1 model introduced in this paper replaces the Backbone and Neck RepBlocks with Faster RepBlock and stacks a single Faster RepBlock module in the Neck. Compared to the baseline model, the detection accuracy of Faster-YOLOv6n-1 only decreases by 0.8%, the inference speed improves to 37 FPS (a 48% increase), and the model's parameter size is reduced to 7.47 MB, a 51.7% reduction. This model achieves a balance of lightweight optimization and high detection accuracy, making it especially suitable for scenarios requiring efficient processing. The further optimized Faster-YOLOv6n-2 model stacks two Faster RepBlock modules in the Neck of YOLOv6. It achieves 75.6% of the baseline model's detection accuracy while increasing the inference speed to 35 FPS (a 40% improvement), and reducing the model size to 9.60 MB, a 38% decrease compared to YOLOv6n. Although the speedup is slightly lower than that of Faster-YOLOv6n-1, this

model strikes an excellent balance between accuracy and efficiency, making it ideal for resource-constrained scenarios that still demand high detection accuracy.

C. Comparison and Analysis of Mainstream Algorithms

TABLE II. COMPARISON WITH STATE-OF-THE-ART ALGORITHMS

Model	mAP@0.5/%	FPS
YOLOv5n	74	33
YOLOv6n	75 .6	25
YOLOv7-tiny	76.5	19
YOLOv8n	76.7	22
Faster-YOLOv6n-2	75.6	**35**

To verify the superiority of the algorithm, Faster-YOLOv6n-2 is compared with several mainstream models, including YOLOv5n, YOLOv6n, YOLOv7-tiny, and YOLOv8n. The results are presented in Table II. On the NVIDIA Jetson Nano device, Faster-YOLOv6n-2 achieves a 1.6% higher mAP and 6% faster inference speed compared to YOLOv5n. When compared to YOLOv7-tiny, although the mAP is slightly lower by 0.9%, the inference speed is 84% faster, addressing the real-time detection demands that YOLOv7-tiny struggles to meet, with its 19 FPS. Additionally, compared to YOLOv8n, Faster-YOLOv6n-2 shows only a 1.1% lower mAP, but with a 59% higher frame rate. In summary, the improved algorithm proposed in this paper demonstrates outstanding accuracy and inference speed in edge computing environments, offering significant advantages in overall performance.

V. CONCLUSION

To achieve more efficient target detection on resource-constrained edge devices, this paper proposes the Faster-YOLOv6n algorithm, an improved version of YOLOv6, aimed at significantly enhancing inference speed with minimal accuracy loss. Specifically, the paper optimizes PConv using reparameterization techniques and introduces a more efficient partially reparameterized convolution module, PRepConv. Building on this, the Faster RepBlock module is designed as the core operator to enable more efficient information fusion and interaction between channels. By replacing the RepBlock in YOLOv6 with Faster RepBlock, a lighter backbone and neck structure is constructed, reducing model complexity while retaining feature extraction and fusion capabilities. Experimental results show that on the NVIDIA Jetson Nano device, Faster-YOLOv6n achieves a 40% improvement in inference speed over the baseline model with no loss in accuracy. Compared to mainstream algorithms, this model demonstrates the fastest inference speed while maintaining high detection accuracy, showcasing its superior performance.

REFERENCES

[1] Khanam, Rahima, and Muhammad Hussain. "YOLOv11: An overview of the key architectural enhancements." arXiv preprint arXiv:2410.17725 (2024).

[2] Howard A G. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arxiv preprint arxiv:1704.04861, 2017.

[3] Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.

[4] Howard A, Sandler M, Chu G, et al. Searching for mobilenetv3[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 1314-1324.

[5] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6848-6856.

[6] Ma N, Zhang X, Zheng H T, et al. Shufflenet v2: Practical guidelines for efficient cnn architecture design[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 116-131.

[7] Han K, Wang Y, Tian Q, et al. Ghostnet: More features from cheap operations[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 1580-1589.

[8] Chen J, Kao S, He H, et al. Run, don't walk: chasing higher FLOPS for faster neural networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023: 12021-12031.

[9] Sun H, Wang B, Xue J. YOLO-P: An efficient method for pear fast detection in complex orchard picking environment[J]. Frontiers in plant science, 2023, 13: 1089454.

[10] Glenn.: ultralytics/ YOLOv5:v5.0 (2020). https://github.com /ultralytics/ YOLOv5

[11] Woo S, Park J, Lee J Y, et al. Cbam: Convolutional block attention module[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 3-19.

[12] Guo A, Sun K, Zhang Z. A lightweight YOLOv8 integrating FasterNet for real-time underwater object detection[J]. Journal of Real-Time Image Processing, 2024, 21(2): 49.

[13] Glenn.: ultralytics/ultralytics: v8.0.136 (2023). https://github.com/ ultralytics/ultralytics

[14] Gong W. Lightweight Object Detection: A Study Based on YOLOv7 Integrated with ShuffleNetv2 and Vision Transformer[J]. arxiv preprint arxiv:2403.01736, 2024.

[15] Wang C Y, Bochkovskiy A, Liao H Y M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors[C] //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023: 7464-7475.

[16] Dosovitskiy A. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arxiv preprint arxiv:2010.11929, 2020.

[17] Liu Z, Abeyrathna R M R D, Sampurno R M, et al. Faster-YOLO-AP: A lightweight apple detection algorithm based on improved YOLOv8 with a new efficient PDWConv in orchard[J]. Computers and Electronics in Agriculture, 2024, 223: 109118.

[18] Li C, Li L, Geng Y, et al. Yolov6 v3. 0: A full-scale reloading[J]. arxiv preprint arxiv:2301.05586, 2023.

[19] Ding X, Zhang X, Ma N, et al. Repvgg: Making vgg-style convnets great again[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 13733-13742.

[20] Chen J, Liu H, Zhang Y, et al. A multiscale lightweight and efficient model based on YOLOv7: Applied to citrus orchard[J]. Plants, 2022, 11(23): 3260.

[21] Tian, Zhi, et al. "FCOS: A simple and strong anchor-free object detector." IEEE transactions on pattern analysis and machine intelligence 44.4 (2020): 1922-1933.

[22] Ge, Z. "Yolox: Exceeding yolo series in 2021." arXiv preprint arXiv:2107.08430 (2021).

[23] Koonce, Brett, and Brett Koonce. "SqueezeNet." Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization (2021): 73-85.