DOI: 10.19884/j.1672-5220.202409001

# MLGIA: Recognition of Traffic Panel Information Based on PaddlePaddle

JI Yukai, GE Huayong*, MENG Yaqun, LI Sisi
College of Information Science and Technology, Donghua University, Shanghai 201620, China

**Abstract: To address the challenge of recognizing small target information on traffic panels, a model named MLGIA is proposed based on PaddlePaddle. MLGIA is composed of MobilenetV3 with lightweight GhostBlock (LGB) and an improved augmented feature pyramid network (IAFPN). In this model, LGB improves MobilenetV3 by optimizing the convolutional structure and employing linear transformations to extract sufficient feature maps; IAFPN enhances feature representation through pruning techniques and channel-reduction convolutions. Additionally, knowledge distillation compresses the model and improves its accuracy, while the match category information (MCI) method further optimizes the processing of the detected category information. Experimental results demonstrate that MLGIA outperforms MobilenetV3. MLGIA achieves a detection accuracy comparable to YOLOv8n, with significantly lower resource consumption. Therefore, MLGIA is a strong complement in the traffic panel information recognition domain.**

**Keywords**: convolutional neural network; object detection; feature fusion; knowledge distillation; lightweight
**CLC number**: TP183 **Document code**: A
**Article ID**: 1672-5220(2025)05-0494-09

Open Science Identity (OSID)

## 0 Introduction

With the rapid development of the intelligent new energy vehicle industry, autonomous driving has seen widespread commercial applications. Lidar and infrared sensors can effectively capture information about spatial positions and reference objects[1-2]. However, these sensors have significant limitations in obtaining visual information, such as road lane markings and traffic signs. As a result, there is growing interest in extracting traffic information from RGB (a standard color encoding method that uses varying intensities of red, green and blue light to represent a wide array of colors) images.

Thanks to advancements in deep learning, many algorithmic frameworks have been proposed for traffic sign detection[3-5], achieving increasingly high accuracy. These methods primarily focus on simple traffic signs but lack recognition of traffic panels. The complex signs on traffic panels contain both textual and road topology information. The target regions in traffic panels are also small, making the above methods unsuitable for traffic panel recognition. The text recognition framework of traffic signs proposed by Dinesh et al.[6] is an excellent inspiration for us to improve the recognition of traffic panels. SignParser proposed by Guo et al.[7] identifies road topology and the connections between regions of interest within panels. An improved YOLOv5 proposed by Khalid et al.[8] achieves impressive accuracy in panel detection under adverse weather conditions. However, recognizing traffic panels requires vehicle deployment, which demands a highly lightweight model.

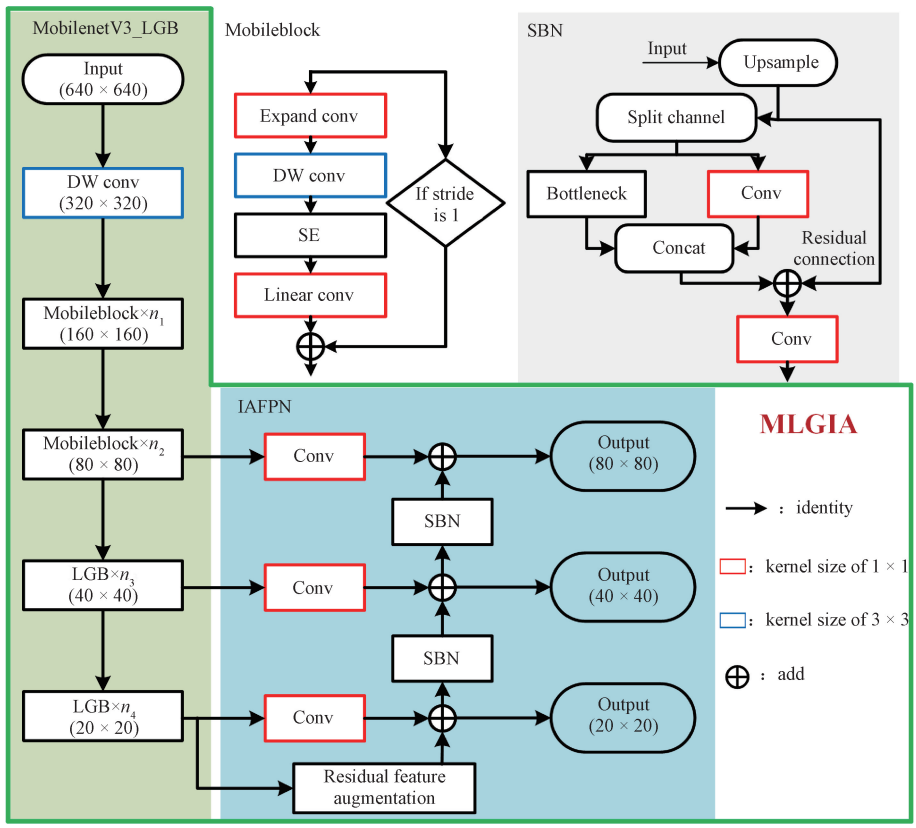
Therefore, we introduce an object detection model named MLGIA under the PaddlePaddle framework. The model employs a backbone network named MobilenetV3_LGB, which is built upon MobilenetV3[9] with an added lightweight GhostBlock (LGB). An improved augmented feature pyramid network (IAFPN) is further introduced to enhance its feature fusion capabilities. Knowledge distillation[10] is employed to improve the accuracy of the lightweight model, and the match category information (MCI) method is used to process the obtained information. Through these innovations, we aim to comprehensively detect various information on traffic panels with high accuracy while reducing model computational complexity and parameters, enhancing lightweight performance, and providing necessary support for developing intelligent transportation systems.

## 1 Methods

We refer to the modified network as MobileNetV3_LGB. It is divided into five feature layers with different receptive fields, as shown in Fig. 1. The first layer is a depthwise separable convolution (DW conv). The second and third layers use Mobileblock, while the fourth and fifth layers employ LGB. In this section, we focus on explaining the principles and implementation details of the algorithm, including the implementation of MobilenetV3_LGB for feature extraction, the fusion of feature information by using IAFPN, the parameter

Received date: 2024-09-08
Foundation item: National Natural Science Foundation of China (No. 62372100)
* Correspondence should be addressed to GE Huayong, email: gehuayong@dhu.edu.cn
Citation: JI Y K, GE H Y, MENG Y Q, et al. MLGIA: recognition of traffic panel information based on PaddlePaddle[J]. *Journal of Donghua University (English Edition)*, 2025, 42(5): 494-502.

design and knowledge distillation training of MLGIA, and the use of MCI to obtain traffic panel information after achieving object detection. The complete MLGIA network with the detailed structural information primarily consists of Mobilenet_LGB and IAFPN, highlighted in Fig. 1 with the green box. Mobileblock and squeeze-and-bottleneck (SBN) are the essential modules within the model.




$n_i$ ($i$=1, 2,⋯, 4)—the number of repetitions of each module, corresponding to the model's parameter design; conv—convolution; concat—concatenation; SE—squeeze-and-excitation.

Fig. 1 Framework of MLGIA network structure

## 1.1 MobilenetV3_LGB feature extraction network

### 1.1.1 Lightweight improved GhostConv[11] (Gconv)

Gconv begins by using a convolution operation to obtain feature maps for some channels. Then, a linear transformation is applied to these feature maps to generate the remaining channels. Finally, two sets of feature maps are concatenated to complete the convolution. This structure extracts semantic information from similar feature maps and enhances feature extraction capability.

For mobile devices, a lightweight design is a crucial criterion for convolutional neural networks. To further reduce the network's computational load and improve feature extraction efficiency, we improved the activation function used in Gconv. The original Gconv utilizes the SiLU[12] function, defined as

$$F_{\mathrm{SiLU}}(x) = x \cdot \sigma(x), \tag{1}$$

where $x$ is the input; $\sigma(x) = (1 + \mathrm{e}^{-x})^{-1}$. However, the SiLU function has a relatively high computational cost and is unsuitable for resource-constrained scenarios. Inspired by the H-swish activation function of MobilenetV3[9], Gconv is expressed as

$$F_{\mathrm{G}}(x) = \frac{1}{6} x \cdot \min(\max(0,\ x + 3),\ 6). \tag{2}$$

We define an adaptive H-swish linear unit (AL_HSLU) as

$$F_{\mathrm{A}}(x) = \frac{1}{6} x \cdot \max\left(0,\ \min\left(0,\ -\frac{2x}{\lg R_{\mathrm{L}}} + 3\right),\ 6\right), \tag{3}$$

where $R_{\mathrm{L}}$ is the learning rate of the network, which adjusts with the number of training epochs. The definition and gradient of AL_HSLU are shown in Fig. 2.

This activation function has a simple form and low computational cost, and is suitable for lightweight models, offering a high computational efficiency and a high training speed. During the early stages of training, when network parameters are relatively random, a higher learning rate increases the derivative of the activation function at zero, thereby accelerating the activation speed and network convergence. In contrast, when the network gradually converges during the later stages of training, a lower learning rate allows for more precise extraction of high-level semantic information, helping the model improve the accuracy.

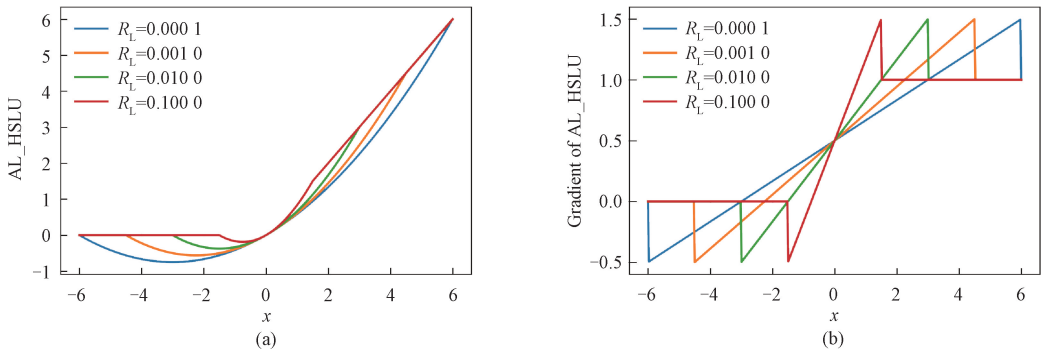



Fig. 2 Functions of AL_HSLU: (a) definition; (b) gradient

### 1.1.2 LGB enhancement of MobilenetV3

LGB is implemented by using the lightweight ghost convolution (LG conv), which enhances the complexity of the features, as shown in Fig. 3. LG conv is implemented by using AL_HSLU.

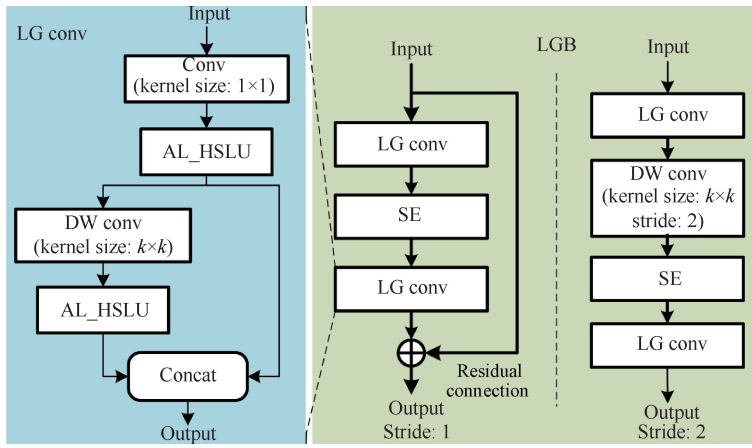



Fig. 3 Framework of LGB and LGconv

When the stride is 1, we construct the LGB with two LG conv separated by an SE[13] block and a residual connection. When the stride is 2, LGB adds a DW conv after the first LG conv to achieve downsampling. We use LGB only in the deeper layers because the number of channels is relatively small in the shallow layers of a convolutional network, and the feature maps are relatively simple. In such cases, LGB generates feature maps that do not change significantly, which prevents the extraction of sufficient feature information. In the deeper layers of the network, where more semantic information is present, LGB can capture more complex features, and its linear transformations can also reduce computational costs.

## 1.2 IAFPN feature fusion network

The IAFPN feature fusion network enhances feature fusion by introducing a residual connection to adjust and improve AugFPN[14]. AugFPN faces issues such as compressing features during layer fusion, which can result in insufficient and incomplete high-level semantic information. We improved the structure, parameters and layer connections to develop IAFPN.

### 1.2.1 Channel adjustment and layer selection

AugFPN reduces the number of channels in deeper layers to match the shallower layers when fusing features, which lowers feature complexity and impacts network performance. To address this, we increased the output channels in the deeper layers to match the input channels. This adjustment enhances the network's non-linear transformation capability, ensuring effective feature extraction and increasing feature diversity.

AugFPN utilizes feature maps with downsampling factors of 4, 8, 16 and 32. To ensure adequate information transfer from deeper to shallower layers and achieve multi-scale fusion during training, we only used downsampling factors of 8, 16 and 32 in IAFPN. The feature maps with downsampling factor of 4 cannot sufficiently learn complex data patterns and features, so removing this layer helps reduce network loss.

### 1.2.2 Enhanced feature fusion

Inspired by bottleneck[15], we improve feature fusion by incorporating SBN. The bottleneck structure first reduces the input channel number to a lower dimension by using a 1 × 1 convolution, then captures spatial features with a 3 × 3 convolution, and finally expands the channel number again with another 1 × 1 convolution, followed by a residual connection for output. We commonly use this approach to enhance network performance and efficiency.

The SBN structure, as shown in Fig. 1, divides the input feature map into two parts by channels. One part uses the bottleneck layer, and the other uses a 1 × 1 convolution layer. After concatenation and residual connection, the final convolution completes the feature fusion of each layer. The cross-connection between the bottleneck layer and the 1 × 1 convolution layer reduces the computational load and facilitates information transfer and sharing between different features. The final convolution introduces non-linear transformations, enhancing the feature fusion performance. This design not only reduces network parameters and computational loads but also improves the feature extraction efficiency and addresses issues such as gradient vanishing and exploding during the training of deep neural networks.

### 1.3 MLGIA parameter design and knowledge distillation

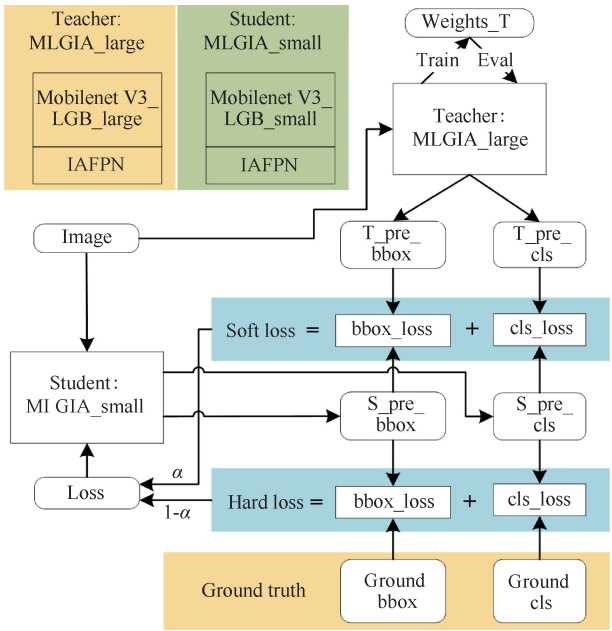
Since traffic panel recognition is deployed on vehicles, the application scenario imposes specific requirements for lightweight models with strict cost constraints for deployment. To address this challenge, inspired by MobilenetV3[9], we designed two networks, namely MobilenetV3_LGB_large and MobilenetV3_LGB_small, as shown in Table 1.

**Table 1** Network parameters for MLGIA

| Network | Block | Kernel | Number of expansion channels | Number of out channels | SE | Activation | Stride |
|---|---|---|---|---|---|---|---|
| MobilenetV3_LGB_large | DW conv | 3 | – | 16 | 0 | ReLU | 2 |
| | Mobileblock | 3 | 16 | 16 | 0 | ReLU | 1 |
| | Mobileblock | 3 | 64 | 24 | 0 | ReLU | 2 |
| | Mobileblock | 3 | 72 | 24 | 0 | ReLU | 1 |
| | Mobileblock | 5 | 72 | 40 | 1 | ReLU | 2 |
| | Mobileblock | 5 | 120 | 40 | 1 | ReLU | 1 |
| | Mobileblock | 5 | 120 | 40 | 1 | ReLU | 1 |
| | LGB | 3 | 240 | 80 | 0 | AL_ HSLU | 2 |
| | LGB | 3 | 200 | 80 | 0 | AL_HSLU | 1 |
| | LGB | 3 | 184 | 80 | 0 | AL_HSLU | 1 |
| | LGB | 3 | 184 | 80 | 0 | AL_HSLU | 1 |
| | LGB | 3 | 480 | 112 | 1 | AL_HSLU | 1 |
| | LGB | 3 | 672 | 112 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 672 | 160 | 1 | AL_HSLU | 2 |
| | LGB | 5 | 960 | 160 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 960 | 160 | 1 | AL_HSLU | 1 |
| MobilenetV3_LGB_small | DW conv | 3 | – | 16 | 0 | ReLU | 2 |
| | Mobileblock | 3 | 16 | 16 | 1 | ReLU | 2 |
| | Mobileblock | 3 | 72 | 24 | 0 | ReLU | 2 |
| | Mobileblock | 3 | 88 | 24 | 0 | ReLU | 1 |
| | LGB | 5 | 96 | 40 | 1 | AL_HSLU | 2 |
| | LGB | 5 | 240 | 40 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 240 | 40 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 120 | 48 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 144 | 48 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 288 | 96 | 1 | AL_HSLU | 2 |
| | LGB | 5 | 576 | 96 | 1 | AL_HSLU | 1 |
| | LGB | 5 | 576 | 96 | 1 | AL_HSLU | 1 |

We combined MobilenetV3_LGB_large with IAFPN to create MLGIA_large as the teacher model and combined MobilenetV3_LGB_small with IAFPN to create MLGIA_small as the student model for knowledge distillation training, as shown in Fig. 4. We first trained the teacher model to obtain the weight file Weights_T, then switched it to an evaluation mode to provide soft labels to the student model. During training, the student model learned not only from the true labels but also from the soft labels provided by the teacher model. We calculated the loss by merging soft loss and hard loss at a ratio of $\alpha/(1-\alpha)$, which helped the student model capture comprehensive feature information and significantly improved the model's

performance. The MLGIA_small model, trained in this way, maintains high performance while achieving model compression and acceleration, facilitating lightweight deployment.




bbox—bounding box; cls—classification; S_pre_bbox—bounding box predicted by student model; T_pre_bbox—bounding box predicted by teacher model; S_pre_cls—classification probabilities predicted by student model; T_pre_cls—classification probabilities predicted by teacher model.

Fig. 4 Framework of knowledge distillation

### 1.4 MCI

The MCI algorithm links the various regions of information obtained from the object detection system to derive the final traffic panel information. The MCI algorithm involves three main components: text information extraction, road information extraction and panel information matching. We used the text recognition module from PaddleOCR[16] to achieve accurate text extraction.

An improved thinning method is utilized in the road information extraction algorithm to extract information about road intersections. Traditional thinning algorithms rely on small rectangular boxes for judgments, which can lead to biases in global information assessment. Since intersections are composed of geometric shapes, and the Hough transform[17] can effectively capture geometric information, we improved the thinning algorithm using the Hough transform, as detailed in Algorithm 1. This improvement enhances the utilization of global information, enabling effective road information extraction.

The panel information matching algorithm determines the final traffic panel information based on the regions' coordinates and area information. In Algorithm 1, the data obtained from the road information extraction is used and combined with the positional information and words of the text regions. Each text box is matched with the corresponding road intersection to link the traffic panel information and achieve traffic panel recognition.

**Algorithm 1** Road information extraction

Input:

*Img*: road region image

*thresh*: threshold to determine lines and circles

Output:

*roadout*: road information

Begin:

1. Convert *Img* to a grayscale image *G_Img*.
2. Apply a circular filter to *G_Img* to remove edge noise; apply erosion and dilation to remove salt-and-pepper noise. The resulting image is *D_Img*.
3. Convert *D_Img* to a binary image *B_Img*.
4. Thin *B_Img* results in *T_Img*. Calculate image width $W$, height $H$, diagonal length $R$, and maximum angle $\Theta$.
5. For $(x, y)$ in $(W, H)$:
   For $(\rho, \theta)$ in $(R, \Theta)$:
   Compute *nums*: the number of points at each length $\rho$ and angle $\theta$.
6. If *nums* > *thresh*: filter out lines or circles, integrate similar data and compute the intersection points of road endpoints. Return *roadout*.

End.

## 2 Experiments

Starting with the experimental environment configuration, the operating system used in this paper is Ubuntu 18.04, with an Intel® Xeon® Gold 6330 CPU, NVIDIA GeForce RTX 3090 24 G GPU, and CUDA and CuDNN versions of 11.2 and 8.4.1, respectively. The deep learning framework used is PaddlePaddle 2.5.1.

The evaluation metrics used in this study include computational complexity, number of parameters (Params), $mAP_{50}$, and mAP@[.5:.95]. The computational complexity of the model is measured in giga floating-point operations per second (GFLOPS). Params represents the model's storage and memory requirements. $mAP_{50}$ indicates the mean average precision (mAP) at an intersection over the union (IoU) threshold of 50%, assessing the detection accuracy at a fixed threshold. mAP@[.5:.95] represents the average of the mAP values across multiple IoU thresholds (from 0.50 to 0.95 with a step size of 0.05), providing a more comprehensive evaluation of the model's detection performance. These metrics reflect the model's efficiency and effectiveness.

The dataset used in this experiment is supported by the National Natural Science Foundation of China (NSFC) and is the Traffic Panel Database (TPD) from China. The TPD includes 1 525 traffic images with

13 657 labels covering various traffic panels. These images are collected under different conditions, such as varying weather, lighting, and other environments. We categorized and annotated the images from the TPD dataset into nine categories based on road intersections, text, and directional information: text region, road sign, forward north, forward south, forward east, forward west, forward, left turn, and right turn.

The hyperparameters for the experiment were set as follows to achieve better training results. The model input size was 640 × 640 pixels, with an initial learning rate of 0.010 0. The Adam optimizer was used to adjust the model parameters. Due to the limited dataset, the model was pre-trained for 150 epochs on the COCO dataset[18], achieving a mAP@[.5:.95] of 34.2%. Subsequently, the TPD dataset was split into training, validation, and test sets at 8∶1∶1, and the model was trained for 300 epochs.

## 3 Results and Analyses

The effectiveness of the proposed improvements to the MobilenetV3 network was validated through a series of ablation experiments. These experiments evaluated the improvements of LGB, IAFPN, and knowledge distillation based on MobilenetV3_large and MobilenetV3_small networks. The results were then compared with the proposed MLGIA network, as presented in Table 2.

**Table 2** Ablation experiment results

| Network | Computational complexity/GFLOPS | Params/$10^6$ | $mAP_{50}$/% | mAP@[.5:.95]/% |
|---|---|---|---|---|
| MobilenetV3_large | 1.89 | 2.85 | 89.1 | 52.5 |
| MobilenetV3_large+LGB | 1.59 | 2.43 | 93.3 | 55.8 |
| MobilenetV3_large+IAFPN | 2.14 | 3.08 | 97.2 | 59.7 |
| MobilenetV3_large+LGB+IAFPN | 1.74 | 2.66 | 98.9 | 62.1 |
| MobilenetV3_small | 0.49 | 0.88 | 76.8 | 43.2 |
| MobilenetV3_small+LGB | 0.42 | 0.73 | 83.4 | 47.6 |
| MobilenetV3_small+IAFPN | 0.53 | 0.96 | 88.2 | 52.2 |
| MobilenetV3_small+LGB+IAFPN | 0.47 | 0.81 | 92.8 | 55.6 |
| MobilenetV3_small+LGB+IAFPN+KD | 0.47 | 0.81 | 96.4 | 59.3 |

Notes: MobilenetV3_small+LGB+IAFPN and MobilenetV3_large+LGB+IAFPN denote large and small models of MLGIA, respectively; MobilenetV3_small+LGB+IAFPN+KD represents MLGIA_small model trained by knowledge distillation from MLGIA_large as teacher model.

The results show that the unmodified MobilenetV3 performs poorly. Adding LGB reduces both computational complexity and parameter count and improves detection accuracy, demonstrating that LGB supports lightweight model implementation. Although IAFPN increases the computational complexity, it significantly improves detection accuracy. This is due to the efficient feature extraction by the bottleneck layers and enhanced feature representation enabled by the residual connections. Finally, knowledge distillation yielded on absolute improvements of 3.6% (from 92.8% to 96.4%) in $mAP_{50}$ and 3.7% (from 55.6% to 59.3%) in mAP@[.5:.95] for the small model.

The optimized models from the above training were compared with high-performance object detection networks. The experimental results are shown in Table 3. MLGIA_large demonstrates significantly higher $mAP_{50}$ and mAP@[.5:.95] compared to MobilenetV3 and GhostNet networks. Among other high-performing networks, MLGIA performs slightly worse than PPYOLOE+_s in detection accuracy but outperforms YOLOv5n. MLGIA_small_KD shows slightly lower detection accuracy than YOLOv8n. However, our network vastly outperforms other networks in terms of lightweight design. Thus, MLGIA_small_KD is an excellent lightweight network for traffic panel object detection.

**Table 3** Comparative experiment results

| Network | Computational complexity/GFLOPS | Params/$10^6$ | $mAP_{50}$/% | mAP@[.5:.95]/% |
|---|---|---|---|---|
| MobilenetV3_large | 1.89 | 2.85 | 89.1 | 52.5 |
| MobilenetV3_small | 0.49 | 0.88 | 76.8 | 43.2 |
| GhostNet | 5.54 | 4.26 | 92.0 | 57.4 |
| YOLOv5n | 2.11 | 1.79 | 95.0 | 54.2 |
| YOLOv8n | 4.07 | 3.01 | 98.4 | 62.3 |
| PPYOLOE+_s[19] | 8.17 | 7.68 | 99.1 | 65.8 |
| MLGIA_large | 1.74 | 2.66 | 98.9 | 62.1 |
| MLGIA_small_KD | 0.47 | 0.81 | 96.4 | 59.3 |

Notes: MLGIA_large is an abbreviation for MobilenetV3_large+LGB+IAFPN; MLGIA_small_KD is an abbreviation for MobilenetV3_large+LGB+IAFPN+KD.

The MLGIA_small_KD model detects objects on traffic panel images from various environments to validate visualizations of experimental results, as shown in Fig. 5. In Fig. 5, TR refers to the text region, and RS refers to the road sign. The numbers following them denote the corresponding confidence scores. It demonstrates that MLGIA performs well in daylight environments. However, dark and rain-fog conditions require higher image quality due to the impact of underexposure and water droplet obstructions.

After completing object detection, the MCI method was used to recognize traffic panel information and obtain complete information. The results in Fig. 6 indicate that with the MCI method, MLGIA performs well in extracting traffic panel information, thereby providing reliable data for vehicle navigation. However, MLGIA's performance in text recognition remains limited under occlusions, primarily due to its inability to leverage contextual cues (e.g., word association) to infer adjacent text.

(a) (b) (c)

Fig. 5 Visualization of object detection by MLGIA_small_KD model in different environments: (a) daylight; (b) dark; (c) rain-fog

(a) (b) (c)

Fig. 6 Visualization of traffic panel information recognition by MLGIA with MCI method in different environments: (a) daylight; (b) dark; (c) rain-fog

## 4 Conclusions

The recognition of traffic panel information has significant practical value. This paper proposes MLGIA, a model specifically designed for detecting small targets in traffic panels, based on the PaddlePaddle framework. MLGIA improves the backbone feature extraction and neck feature fusion components, and utilizes the MCI method to achieve comprehensive traffic panel information recognition. The algorithm makes the network more lightweight and improves the extraction of the directional and road layout information, thereby refining the panel information extraction capabilities. Experimental comparisons show that MLGIA achieves high average detection accuracy. Overall, MLGIA exhibits more robust performance, more effective information extraction, and a more straightforward structure. Future work will focus on enhancing MLGIA's contextual reasoning capabilities to overcome its current limitations in handling occlusions.

## References


[1] LI Y, IBANEZ-GUZMAN J. Lidar for autonomous driving: the principles, challenges, and trends for automotive lidar and perception systems[J]. *IEEE Signal Processing Magazine*, 2020, 37(4): 50-61.

[2] DAI X R, YUAN X, WEI X Y. TIRNet: object detection in thermal infrared images for autonomous driving[J]. *Applied Intelligence*, 2021, 51(3): 1244-1261.

[3] TABASSUM A, VAIDEHI K. Traffic sign recognition for an intelligent vehicle: a review[J]. *Turkish Online Journal of Qualitative Inquiry*, 2021, 12(6): 6209-6215.

[4] TIAN Y, GELERNTER J, WANG X, et al. Traffic sign detection using a multi-scale recurrent attention network[J]. *IEEE Transactions on Intelligent Transportation Systems*, 2019, 20(12): 4466-4475.

[5] HAQUE W A, AREFIN S, SHIHAVUDDIN A S M, et al. DeepThin: a novel lightweight CNN architecture for traffic sign recognition without GPU requirements[J]. *Expert Systems with Applications*, 2021, 168: 114481.

[6] DINESH N, MATHI S. Optical character recognition-based signboard detection[C]// Disruptive Technologies for Big Data and Cloud Applications. Singapore: Springer Nature Singapore, 2022: 447-455.

[7] GUO Y F, FENG W, YIN F, et al. SignParser: an end-to-end framework for traffic sign understanding[J]. *International Journal of Computer Vision*, 2024, 132(3): 805-821.

[8] KHALID S, SHAH J H, SHARIF M, et al. A robust intelligent system for text-based traffic signs detection and recognition in challenging weather conditions[J]. *IEEE Access*, 2024, 12: 78261-78274.

[9] HOWARD A, SANDLER M, CHEN B, et al. Searching for MobileNetV3[C]//2019 IEEE/CVF International Conference on Computer Vision (ICCV). New York: IEEE, 2019: 1314-1324.

[10] Hinton G, VINYALS O, DEAN J. Distilling the knowledge in a neural network[EB/OL]. (2015-3-9)[2024-08-16]. https://arxiv.org/abs/1503.02531.

[11] HAN K, WANG Y H, TIAN Q, et al. GhostNet: more features from cheap operations[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New York: IEEE, 2020: 1577-1586.

[12] ELFWING S, UCHIBE E, DOYA K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning[J]. *Neural Networks*, 2018, 107: 3-11.

[13] HU J, SHEN L, ALBANIE S, et al. Squeeze-and-excitation networks[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020, 42(8): 2011-2023.

[14] GUO C X, FAN B, ZHANG Q, et al. AugFPN: improving multi-scale feature learning for object detection[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New York: IEEE, 2020: 12592-12601.

[15] HE K M, ZHANG X Y, REN S Q, et al. Deep residual learning for image recognition[C]// 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). New York: IEEE, 2016: 770-778.

[16] DU Y N, LI C X, GUO R Y, et al. PP-OCR: a practical ultra lightweight OCR system[EB/OL]. (2020-10-15)[2024-08-16]. https://arxiv.org/abs/2009.09941.

[17] BALLARD D H. Generalizing the Hough transform to detect arbitrary shapes[J]. *Pattern Recognition*, 1981, 13(2): 111-122.

[18] LIN T Y, MAIRE M, BELONGIE S, et al. Microsoft COCO: common objects in context[C]//Computer Vision-ECCV 2014. Cham: Springer International Publishing, 2014: 740-755.

[19] XU S L, WANG X X, LV W Y, et al. PP-YOLOE: an evolved version of YOLO[EB/OL]. (2022-12-12)[2024-08-16]. https://arxiv.org/abs/2203.16250.

# MLGIA：基于 PaddlePaddle 的交通面板信息识别

吉宇凯，葛华勇*，孟亚群，李思思
东华大学 信息科学与技术学院，上海 201620

**摘　要：**为应对交通标志牌小目标信息识别的挑战，基于 PaddlePaddle 框架提出了 MLGIA 模型。MLGIA 由轻量级 GhostBlock（lightweight GhostBlock，LGB）改进的 MobilenetV3 与增强型特征金字塔网络（improved augmented feature pyramid network，IAFPN）构建。在该模型中，LGB 通过优化卷积结构并结合线性变换以充分提取特征图，从而实现对 MobilenetV3 的改进；IAFPN 通过剪枝与通道缩减卷积技术增强了对图形特征的表达。此外，知识蒸馏技术被运用于实现模型压缩和精度提升；信息类别匹配（match category information，MCI）方法进一步优化检测到的类别信息的处理流程。实验结果表明，MLGIA 性能优于 MobilenetV3，其检测精度与 YOLOv8n 相当，资源消耗更少。因此，MLGIA 模型是对交通标志牌信息识别领域的一个重要补充。

**关键词：**卷积神经网络；目标检测；特征融合；知识蒸馏；轻量级