Lightweight Improvement of a Road Defect Detection Model Based on YOLOv8

Qiong Peng[1], Ping Zhong[1,*], Chenrui Yang[2]
[1]School of Information and Mechatronics Engineering, Hunan International Economics University, Changsha, China
[2]School of Mathematics and Statistics, Hunan First Normal University, Changsha, China
Corresponding author: Ping Zhong (e-mail: 2904069779@qq.com)

ABSTRACT Road-defect inspection based on vehicle-mounted optical imaging requires the image-analysis model to operate under limited on-board storage and computation. This study develops a lightweight target detector for the computational interpretation stage of a visible-spectrum pavement sensing system. In the actual implementation, the original YOLOv8s backbone is replaced with a ShuffleNetV2 feature extractor, while the SPPF module, multi-scale feature-fusion neck, and detection head are retained. The ShuffleNetV2 backbone uses channel splitting, depthwise convolution, pointwise convolution, and channel shuffle to reduce redundant computation. The model is trained for 200 epochs on a seven-class road-defect dataset containing transverse patches, longitudinal patches, transverse cracks, longitudinal cracks, alligator cracks, manhole covers, and potholes. Code-level reconstruction gives 7,327,393 parameters for the improved seven-class model, and the saved best checkpoint is approximately 15 MB. The best recorded validation performance is mAP@0.5 = 0.533 and mAP@0.5:0.95 = 0.340. Compared with the original YOLOv8s result reported in this study, the lightweight model reduces model complexity but also exhibits a non-negligible decrease in detection accuracy. The results therefore demonstrate an explicit accuracy-complexity trade-off rather than accuracy-preserving compression.

INDEX TERMS Road defect detection; YOLOv8; ShuffleNetV2; Model lightweighting; Optical Imaging; Object detection; Edge Sensing

I. INTRODUCTION

WITH the rapid development of urbanization, traffic roads are increasing in number, making road facilities increasingly important. However, road surfaces are prone to cracks, potholes and other damages due to vehicle loads, climate and other factors. Failure to conduct timely inspection and maintenance will lead to worsening road damage, growing capital investment and increased risk of traffic accidents [6]. Therefore, efficient pavement defect detection is a major issue in road maintenance. Traditional manual detection is slow, greatly affected by human factors and risky, failing to meet the demand for refined road management in modern society.

In recent years, computer vision and deep learning have advanced the automation of road defect detection [10]. Methods based on images captured by inspection vehicles combined with convolutional neural network object detection are widely adopted. Among these approaches, the YOLO series is extensively applied thanks to its fast speed and high accuracy [8]. However, the optimized structure of the YOLO model leads to a corresponding increase in parameters and computational complexity, which limits its application in vehicle-mounted and mobile devices [2]. Therefore, designing more lightweight models without compromising accuracy is currently a popular research topic [3]. Based on YOLOv8, this paper introduces ShuffleNetV2 to achieve model lightweighting. This method can not only deliver favorable detection performance, but also reduce model size and parameter quantity, facilitating the practical application of road inspection systems.

The main contributions of this study are as follows. First, a ShuffleNetV2 feature extractor is integrated as a complete replacement for the original YOLOv8s backbone, while the SPPF module, multi-scale neck, and detection head are retained. Second, the lightweight mechanism is analyzed in terms of channel splitting, depthwise and pointwise convolution, and channel shuffle, and the implementation details are reported using the actual stage repetitions and channel dimensions. Third, model compactness and detection performance are evaluated jointly using parameter count, checkpoint size, theoretical computation, mAP@0.5, mAP@0.5:0.95, and class-wise PR curves. Fourth, the results are interpreted in the context of vehicle-mounted optical sensing, with a clear distinction between theoretical complexity reduction and hardware-verified inference acceleration.

II. RELATED WORK

With the continuous improvement of road traffic networks, road defect detection has become a hot topic in road maintenance management and intelligent transportation. Researchers have gradually shifted from traditional image processing methods to deep learning-based approaches, and relevant models are evolving toward lightweight design [3].

Early research detected pavement cracks and potholes based on traditional image processing technologies. In this process, researchers adopted methods including edge detection, texture analysis and threshold segmentation to obtain relevant attributes in pavement images and determine damage locations. However, real road environments are affected by various factors such as light changes, shadows and vehicle occlusion. Therefore, such detection methods are highly sensitive to external environments and have poor robustness.

With the development of deep learning, object detection methods applied on roads are mainly based on convolutional neural networks. Although two-stage object detection algorithms such as Faster R-CNN achieve high accuracy, they feature high computational complexity and thus low processing speed [5]. Single-stage object detection algorithms such as SSD and YOLO series are widely applied in road damage detection, intelligent traffic monitoring and environmental perception for autonomous driving due to their high speed. With the development of the YOLO algorithm, its detection accuracy and speed have been greatly improved [7].

However, deep learning models generally suffer from drawbacks such as large parameter quantities and heavy computational load, which to some extent poses difficulties for their deployment on vehicle-mounted devices and embedded platforms [2]. Therefore, in recent years, researchers have begun to study lightweight network architectures such as MobileNet, ShuffleNet and GhostNet. These models reduce computational complexity and boost running speed by optimizing network structures.

To meet the requirements for accuracy and efficiency of automated road defect detection, lightweight models are researched and optimized based on deep learning detection methods. The main work consists of four parts. The first part establishes a road defect dataset for training in YOLO format, covering seven types of defects including longitudinal cracks, transverse cracks, potholes, manhole covers, road cavities, longitudinal repaired surfaces and transverse repaired surfaces, facilitating subsequent experiments. The second part adopts the high-performance YOLOv8 as the basic model for road defect detection [7]. Realize automatic detection and localization of defects through training. Secondly, the lightweight network module ShuffleNetV2 is adopted to replace the ResNeXt module in the backbone network of YOLOv8, which achieves model lightweighting. On the premise of ensuring detection performance, it reduces model parameters and volume and improves computing speed. Finally, comparative tests are conducted on the original model and the improved model. Comparative analysis is carried out in terms of parameter quantity, model volume and detection accuracy to verify the feasibility and practicality of the lightweight improvement [4].

III. AN IMPROVED YOLOV8 MODEL BASED ON SHUFFLENETV2

A. APPROACH TO IMPROVEMENT

YOLOv8 is a popular object detection network with excellent performance in detection accuracy, training stability and inference speed. When applied to road damage detection, it can accurately identify various targets such as cracks, potholes, repair areas and manhole covers. Experimental results show that the original YOLOv8 model achieves favorable results on the dataset adopted in this paper, proving its strong capability in feature learning and object detection [11]. For distinct types of pavement diseases with obvious differences and vastly varying sizes, YOLOv8 achieves high-precision detection results through multi-scale feature fusion and an efficient detection head, which is why we adopt it as the backbone network.

Although YOLOv8 achieves favorable detection performance, it still has certain drawbacks in practical applications. Road defect detection is not merely a laboratory research topic; its practical application in real-world road environments is also indispensable. In particular, when such a detection system is deployed on vehicle-mounted inspection devices, mobile inspection terminals and embedded edge devices, the system is required to deliver high detection accuracy while completing detection tasks rapidly. These devices generally suffer from insufficient hardware resources, featuring limited processor computing power, video memory and storage space. A large or structurally complex model will reduce inference speed, consume more memory and raise deployment costs, rendering the entire detection system inoperable [13].

It can be seen from the network structure of the original YOLOv8 that the backbone network is responsible for extracting image features and is also the part with the largest computational load in the entire network. The backbone network conducts numerous convolution operations and feature encoding on input images, converting low-level texture features into high-level semantic features layer by layer. While this endows the model with strong feature representation capability, it also brings substantial parameter overhead. In road crack detection tasks, input images usually contain abundant road texture information, whereas crack targets are slender with irregular boundaries and small sizes. Accurate acquisition of such information requires the model to perform in-depth learning, which increases the workload of feature extraction. Accordingly, deploying the original YOLOv8 directly on devices with limited hardware resources struggles to achieve high precision while maintaining satisfactory operating speed.

The method proposed in this paper does not directly replace the entire YOLOv8 with ShuffleNetV2. Instead,

while retaining the basic structure of YOLOv8, some convolution operations in its backbone network are selectively substituted with lightweight ShuffleNetV2 modules. The original YOLOv8 backbone network consists of multiple convolutional layers and C2f modules, which help enhance feature quality yet bring massive parameters and computational costs. To streamline the network structure without altering the overall framework of YOLOv8, partial convolution operations are replaced by lightweight ShuffleNetV2 modules. This transforms cumbersome original convolution operations into concise and efficient calculations, effectively reducing the computational load in the front part of the model.

Tests verify the effectiveness of this method. Compared with the original YOLOv8 model, the optimized model has its parameter count reduced to 7,327,393 and its file size cut down to approximately 15 megabytes, which demonstrates that lightweighting can effectively shrink the model size [12]. From an engineering perspective, such model compression can substantially lower the memory and computing resources required for deployment. Although the lightweight model suffers a slight drop in mean average precision, the overall performance remains within an acceptable range and meets the basic requirements for road defect detection. This also indicates that the optimization is not a simple reduction of parameters; instead, lightweight processing is implemented while maintaining detection performance.

The implemented network does not replace isolated convolutional layers within the original YOLOv8s backbone. Instead, the complete original backbone is replaced by a ShuffleNetV2 feature extractor. The subsequent SPPF module, multi-scale feature-fusion neck, and three-scale detection head are retained. This architecture allows the detector to receive P3-, P4-, and P5-level feature maps generated by the lightweight backbone while preserving the downstream feature-fusion and prediction mechanisms of YOLOv8s.

The ShuffleNetV2 backbone uses a width multiplier of 1.0 and stage repetitions of [8, 4, 4]. After an initial 3×3 convolution with stride 2 and max pooling, the three main stages produce feature channels of 116, 232, and 464. In the stride-1 inverted residual unit, the input channels are divided into two branches. One branch is retained as an identity path, whereas the other branch applies 1×1 pointwise convolution, 3×3 depthwise convolution, and a second 1×1 pointwise convolution. The branch outputs are concatenated and rearranged by a two-group channel-shuffle operation. In the stride-2 unit, both branches perform downsampling before concatenation and channel shuffling.

The custom model parser treats ShuffleNetV2 as a multi-output backbone and passes its feature maps to the retained YOLOv8s neck. The final model contains 321 layers and 7,327,393 parameters for seven classes. The architecture summary reports 10.6 GFLOPs at an input resolution of 640×640.

B. LIGHTWEIGHT OPTIMIZATION STRATEGY

To quantitatively illustrate the advantages brought by lightweight design, this paper calculates the difference in computational complexity between standard convolution and depthwise separable convolution. Taking a standard convolution with a convolution kernel size of $K \times K$, input channel count of C_{in} and output channel count of C_{out} as an example, the number of parameters required for this convolution can be expressed by the following formula:

$$P_{\text{conv}} = K \times K \times C_{\text{in}} \times C_{\text{out}} \tag{1}$$

In contrast, the parameter number of depthwise separable convolution is:

$$P_{\text{ds}} = K \times K \times C_{\text{in}} + C_{\text{in}} \times C_{\text{out}} \tag{2}$$

Compared with traditional convolution operations, depthwise separable convolution splits convolution into channel-wise spatial filtering and point-wise channel fusion, which greatly reduces parameter quantity and computational complexity.

In the ShuffleNetV2 module, the input feature map X is first split along the channel dimension:

$$X = [X_1, X_2] \tag{3}$$

Among them, X_1 remains an identity mapping, and X_2 is obtained through a lightweight convolution branch. The output features can be:

$$Y = \text{Concat}(X_1, F(X_2)) \tag{4}$$

After feature concatenation, channel shuffle is applied:

$$Y' = \text{Shuffle}(Y) \tag{5}$$

This facilitates cross-channel information communication and resolves the problem of information isolation caused by branches. Therefore, the optimized network structure maintains basically the same detection performance while reducing the number of parameters and the model size.

C. IMPROVE NETWORK ARCHITECTURE

YOLOv8 consists of three parts: backbone network, neck network and detection head. The backbone network is responsible for feature extraction, accounting for most of the computational load and parameters of the entire model. It contains multiple convolutional layers and C2f modules. Convolution can extract diverse features ranging from fine edges to high-level semantics. The C2f module enhances feature representation by splitting and merging features, facilitating the learning of features of targets with different sizes. Nevertheless, excessive stacked convolutions bring heavy computational costs. In road damage detection, targets such as cracks and potholes are small and irregular in shape, and the input images also contain numerous road surface textures [9]. This requires an excellent feature extractor. However, an overly complex backbone network will increase

computational load and affect real-time operating speed. Therefore, it is essential to design lightweight models.

To reduce computational complexity and parameter quantity, lightweight ShuffleNetV2 modules are adopted to replace part of the convolutional structures in the backbone network of YOLOv8. Leveraging its features of channel splitting, depthwise separable convolution and channel shuffling, a low-cost and efficient feature representation method can be achieved. After modifying the model, the input feature maps undergo preliminary feature extraction through a standard convolutional layer before being fed into the ShuffleNetV2 module. The feature maps are split into two parts along the channel dimension. One part is delivered to subsequent layers via shortcut connection, while the other enters the convolutional branch. In this branch, standard convolution is substituted with depthwise separable convolution, which substantially cuts down model parameters and computation volume. Subsequently, features from the two branches are concatenated and processed through channel rearrangement to enable sufficient information interaction between them. This design not only lowers computational load but also ensures smooth information transmission among all channels, endowing the overall model with excellent feature representation capability.

IV. DATASET AND EXPERIMENTAL SETUP

A. DATA SET VISUALIZATION

Visual analysis was conducted before model training to better understand the occurrence frequency and annotation status of various targets in the dataset [14]. These visual data can be used to check the occurrence frequency of different types of targets in each image, the size of detection boxes and the relationships between categories, which can provide certain guidance for subsequent model training.

First, the labels.jpg image can check the number of targets of each category in the dataset. It can be seen from the results that the sample size of each category is relatively large, yet there are noticeable disparities among different categories [6]. Crack-related objects account for a large proportion, including longitudinal cracks, transverse cracks and reticular cracks. Cracks are also the most common type of pavement distress in practical scenarios. Potholes and inspection wells are relatively few in number, yet the quantity is sufficient for model training.

Secondly, Figure 3 illustrates the relationship between different target categories and their corresponding bounding box sizes. It can be observed that targets of various types differ in width and height. For instance, crack targets are generally slender with relatively small bounding box width or height, while repair areas and manhole covers occupy a larger proportion. Accordingly, a target detection method capable of adapting well to scale variations is required to accurately identify targets of diverse sizes.

B. EXPERIMENTAL ENVIRONMENT

To verify the performance of the improved lightweight model in pavement defect detection, this paper conducts model training and evaluation under a deep learning framework. This study adopts parallel computing based on graphics processing unit-enabled unified computing devices to boost computational efficiency and accelerate matrix and convolution operations via parallel processing. This method addresses the heavy computation load of object detection models, substantially cuts training time while maintaining favorable robustness, and resolves the issues of lengthy training duration and limited iterative model updates caused by relying solely on central processing units [15].

This project adopts the PyTorch deep learning framework for training, and builds and trains models based on the mature YOLOv8 object detection framework. Leveraging the strong scalability and comprehensive training processes offered by these two frameworks, researchers can conveniently adjust network structures and add new modules, so as to obtain and apply lightweight optimized models [16]. Meanwhile, key hyperparameters including training epochs, batch size and learning rate have all been properly configured. The batch size is determined based on the graphics card video memory and model size, aiming to fully utilize hardware resources while maintaining decent training efficiency. The learning rate is fine-tuned according to prior experience and training outcomes to facilitate favorable model convergence. After each training epoch, the validation set is used to evaluate the current model performance. Changes in loss values and detection accuracy are monitored to implement corresponding adjustments, ensuring smooth training progress.

V. EXPERIMENTAL RESULTS AND ANALYSIS

A. TRAINING PROCESS ANALYSIS

During model training, we can evaluate the model's learning status by observing changes in metrics such as loss function, accuracy and recall rate. Three types of losses are the key focuses in training: bounding box loss, classification loss and distribution focal loss. They respectively reflect the model's learning performance in bounding box localization, object classification and bounding box regression.

The loss curve plotted during training shows that the initial loss is high at the early stage because the model parameters have not learned sufficient image content. As training proceeds, the model gradually learns to extract valid information, leading to a rapid drop in loss. After a period of time, all losses fluctuate slightly or even stabilize, indicating that the model parameters have nearly reached the optimal solution and the network can perform object detection effectively.

In the later stage of training, the precision and recall curves tend to flatten out, indicating that the model has basically reached a satisfactory state with the current training parameters. Throughout the training process, the model runs smoothly without severe fluctuations or divergence.

```
                 from  n    params  module                                       arguments
  0                -1  1    692204  shufflenetv2                                 []
  1                -1  1    584272  ultralytics.nn.modules.block.SPPF            [464, 512, 5]
  2                -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
  3           [-1, 3]  1         0  ultralytics.nn.modules.conv.Concat           [1]
  4                -1  1    585216  ultralytics.nn.modules.block.C2f             [744, 256, 1]
  5                -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
  6           [-1, 2]  1         0  ultralytics.nn.modules.conv.Concat           [1]
  7                -1  1    146688  ultralytics.nn.modules.block.C2f             [372, 128, 1]
  8                -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
  9           [-1, 8]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 10                -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 11                -1  1    590336  ultralytics.nn.modules.conv.Conv             [256, 256, 3, 2]
 12           [-1, 5]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 13                -1  1   1969152  ultralytics.nn.modules.block.C2f             [768, 512, 1]
 14      [11, 14, 17]  1   2118757  ultralytics.nn.modules.head.Detect           [7, [128, 256, 512]]
YOLOv8s summary: 321 layers, 7,327,393 parameters, 7,327,377 gradients, 18.6 GFLOPs
```

FIGURE 1. Improved YOLOv8-ShuffleNetV2 Network Architecture.

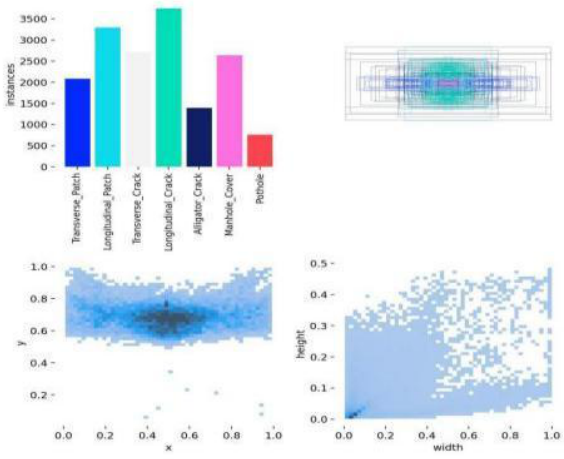

FIGURE 2. Statistical Distribution Chart of Dataset Categories.

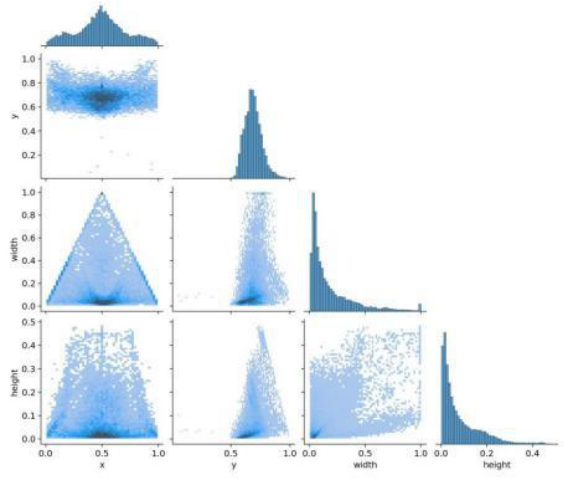

FIGURE 3. Target Size and Correlation Distribution of the Dataset.

The loss function and various evaluation indicators change steadily, the performance remains stable throughout the training process, and the convergence effect is favorable [17].

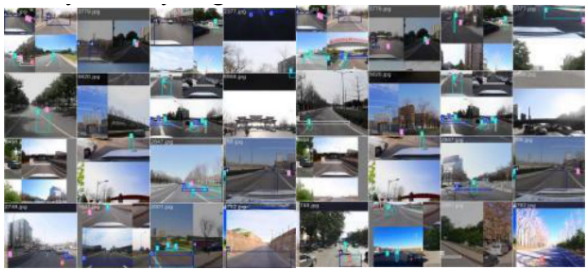

FIGURE 4. Training Sample Visualization Example.

B. PR CURVE ANALYSIS

To comprehensively evaluate the model's performance across various object detection tasks, the Precision-Recall curve is adopted here to assess detection effectiveness. As a widely used evaluation metric in object detection, the Precision-Recall curve clearly illustrates the correlation between precision and recall, and also reflects the model's detection capability under different confidence thresholds.

In the PR curve, the horizontal axis stands for recall. Generally, precision declines as recall increases. This is because when trying to retrieve as many true positive samples as possible, the likelihood of false positive occurrences rises accordingly. The overall trend of the PR curve can demonstrate the model's performance on diverse detection tasks.

It can also be observed from the precision-recall curve derived from the experimental results that the curve of this model is relatively flat and can achieve high precision under most categories, indicating that this method delivers favorable recognition performance for pavement distresses. For targets with distinct structural features such as manhole covers and patched areas, the model can attain extremely high accuracy. However, cracks are slender with irregular edges and susceptible to interference from complex pavement textures, making them difficult to detect, which leads to a significant gap between their precision and recall rates.

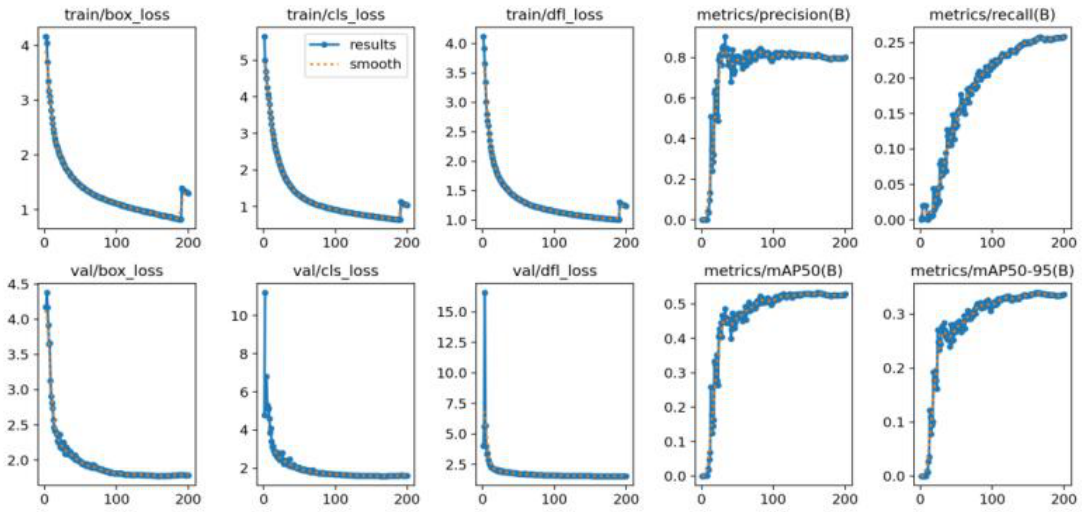

FIGURE 5. Model Training Process Curve.

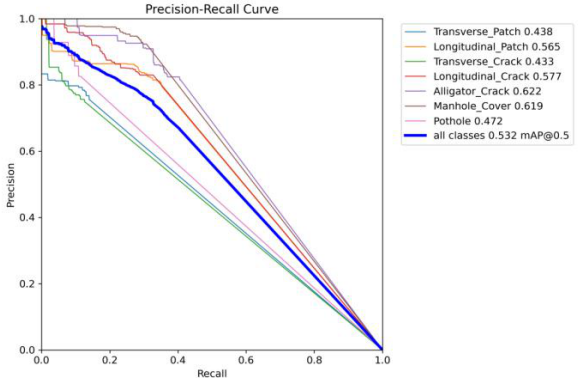

FIGURE 6. PR curve.

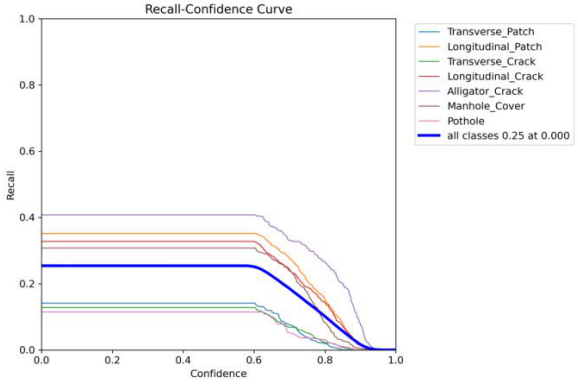

FIGURE 8. Recall curve.

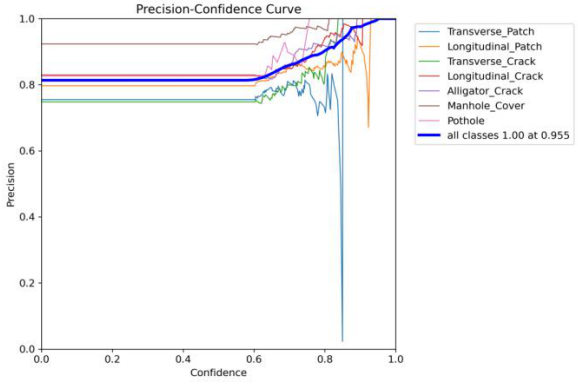

FIGURE 7. Precision curve.

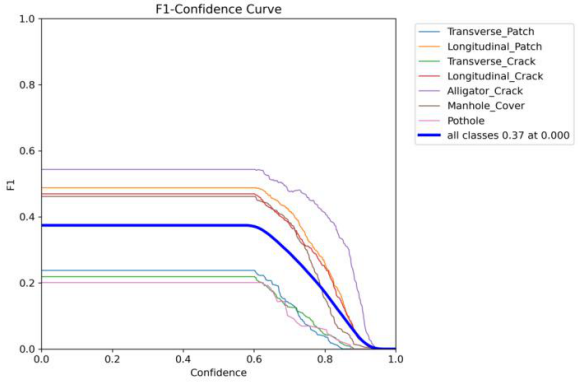

FIGURE 9. F1-Score curve.

C. CONFUSION MATRIX ANALYSIS

To conduct a more detailed study on the model's recognition capability for different categories, this paper adopts the confusion matrix for statistical analysis. The confusion matrix can clearly demonstrate the model's classification performance across all categories. By comparing the discrepancies between ground-truth categories and predicted categories, we can evaluate the model's recognition accuracy for each target as well as its existing defects.

In the confusion matrix, the values on the diagonal represent the number or proportion of correctly classified samples, while the off-diagonal values stand for the number of misclassified samples between different categories. The larger the values on the diagonal, the better the classification performance of the corresponding category. As can be seen from the figure above, this model achieves excellent classification performance for most categories.

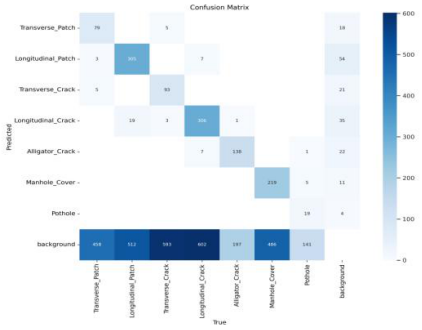

FIGURE 10. Model Confusion Matrix.

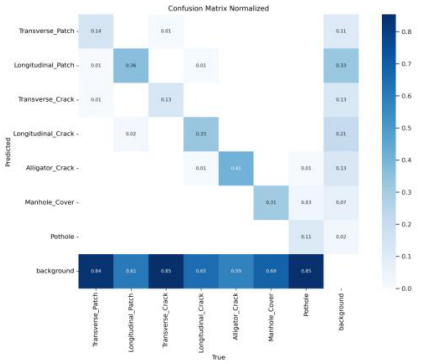

FIGURE 11. Normalized Confusion Matrix.

D. VISUALIZATION OF TEST RESULTS

To intuitively compare the detection performance of the two methods, this paper uses a set of test images to present the detection results of the original YOLOv8 model and the optimized lightweight YOLOv8 model. Prediction bounding boxes and categories are marked with labels, and the detection performance of the two models under various scenarios is analyzed. The original YOLOv8 model can effectively detect various road surface defects such as cracks and manhole covers. It achieves high accuracy and confidence level when identifying objects with distinct features, and performs well in multi-target detection tasks, which demonstrates its strong capability in feature learning and object detection.

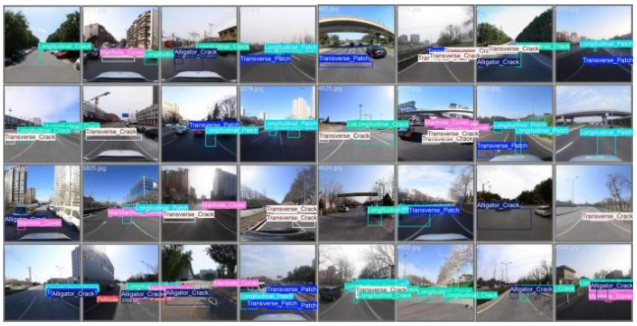

FIGURE 12. Example of Ground Truth Annotations for the Test Set.

FIGURE 13. Example of Model Detection Results.

By comparing the detection results of the original model and the optimized model, it can be seen that lightweight optimization reduces model parameters to a certain extent without impairing the overall detection performance.

E. MODEL PERFORMANCE COMPARISON

To verify the effectiveness of the lightweight optimization scheme proposed in this paper, a comparative study is conducted between the original YOLOv8 network and the improved model integrated with ShuffleNetV2. The comparison mainly covers the number of model parameters, model size and mean average precision of detection. These indicators can reflect the trade-off degree between detection performance and computational speed of the model.

TABLE 1. Model Performance Comparison

Model	Parameter quantity	Model size	mAP
YOLOv8	11128293	22MB	0.663
YOLOv8+ShuffleNetV2	7321657	15MB	0.532

It can be seen from the experiment that the original YOLOv8 model has 11,128,293 parameters with a model file size of about 22MB, and achieves an mAP of 0.663 on this dataset. This model achieves favorable detection performance. However, its complex network structure and massive number of parameters consume substantial resources during operation. On vehicle-mounted detection devices and embedded platforms with limited hardware resources, an oversized model may slow down system operation and even trigger system crashes.

VI. CONCLUSION

This study developed a lightweight road-defect target detector for vehicle-mounted optical images by replacing the original YOLOv8s backbone with a ShuffleNetV2 feature extractor while retaining the SPPF module, multi-scale feature-fusion neck, and detection head. The code-verified seven-class model contains 7,327,393 parameters, and its best checkpoint occupies approximately 15 MB. The best recorded validation performance is mAP@0.5 = 0.533 and mAP@0.5:0.95 = 0.340. Class-wise analysis shows relatively stronger performance for alligator cracks and manhole covers and weaker performance for transverse cracks and transverse patches. These findings demonstrate that the proposed architecture reduces model complexity but does not preserve the full detection accuracy of the original detector. Its scientific significance therefore lies in quantifying an architecture-specific accuracy-complexity trade-off for the image-interpretation stage of an optical road-sensing system. The current evidence supports compactness and lower theoretical computational demands.

ACKNOWLEDGEMENTS

The authors acknowledge the Hunan Provincial Department of Education Scientific Research Project (Grant: 25C1226); Hunan Provincial Association of Educational Researchers 2024 University Research Project (Grant: XJKX24B042); Ministry of Education Project on Aligning Labor Supply and Demand to Foster Employment and Talent Development (Grant: 2025072619939).

REFERENCES

[1] N. Ma, X. Zhang, H. T. Zheng, et al., "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," Proceedings of the European Conference on Computer Vision (ECCV) 2018, pp. 116–131.
[2] Y. Chen, et al., "YOLO V8 Network Lightweight Method Based on Pruning and Quantization," Journal of Advances in Applied Mathematics, 2024.
[3] Y. Cheng, et al., "YOLO V8 Network Lightweight Method Based on Pruning and Quantization," International Conference on Computer Science and Artificial Intelligence 2024.
[4] H. Gan, et al., "PKD-YOLOv8: A Collaborative Pruning and Knowledge Distillation Framework for Lightweight Rapeseed Pest Detection," Sensors, vol. 25, no. 16, Art. no. 5004, 2025.
[5] M. Kortmann, et al., "Fast road defect detection using an improved faster R-CNN," International Conference on Computer Vision and Pattern Recognition Workshops 2023.
[6] J. N. Opara, et al., "Road defect detection using deep learning: A survey," IEEE Access, vol. 11, pp. 109582–109607, 2023.
[7] H. Wang, et al., "YOLOv8-D-CBAM: Improved YOLOv8 for pavement crack detection," Sensors, vol. 24, no. 21, Art. no. 6783, 2024.
[8] F. Wan, et al., "YOLO-LRDD: A lightweight road defect detection algorithm," IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 8, pp. 8921–8932, 2023.
[9] Y. Xu, et al., "YOLOv5-PD: An improved YOLOv5 algorithm for asphalt pavement defect detection," Measurement, vol. 235, Art. no. 114785, 2024.
[10] L. Zhang, et al., "Deep Learning-Based Algorithm for Road Defect Detection," Sensors, vol. 25, no. 5, Art. no. 1287, 2025.
[11] C. Wu, M. Ye, J. Zhang, and Y. Ma, "YOLO-LWNet: A lightweight road damage object detection network for mobile terminal devices," Sensors, vol. 23, no. 6, Art. no. 3268, 2023.
[12] J. Chen, X. Yu, Q. Li, W. Wang, et al., "LAG-YOLO: Efficient road damage detector via lightweight attention ghost module," Journal of Intelligent Construction, vol. 2, no. 1, Art. no. 9180032, 2024.
[13] A. Yu, Y. Gao, Y. Xiong, W. Liu, and J. She, "Mobile-YOLO: A lightweight YOLO for road crack detection on mobile devices," Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 29, no. 6, pp. 1443–1454, 2025.
[14] P. Huang, S. Wang, J. Chen, W. Li, and X. Peng, "Lightweight model for pavement defect detection based on improved YOLOv7," Sensors, vol. 23, no. 16, Art. no. 7112, 2023.
[15] W. Wang and W. Yu, "Enhancing real-time road object detection: The RD-YOLO algorithm with higher precision and efficiency," IEEE Access, vol. 12, pp. 190876–190888, 2024.
[16] Z. Zhang, H. Zhang, and T. Zhang, "Enhanced YOLOv8-based pavement crack detection: A high-precision approach," PLoS One, vol. 20, no. 5, Art. no. e0324512, 2025.
[17] S. Zhang, Z. Bei, T. Ling, Q. Chen, and L. Zhang, "Research on high-precision recognition model for multiscene asphalt pavement distresses based on deep learning," Scientific Reports, vol. 14, Art. no. 27173, 2024.