https://doi.org/10.31891/2219-9365-2024-80-9
UDC 622

ILCHUK Mykhailo
Lviv Polytechnic National University
e-mail: mykhailo.s.ilchuk@lpnu.ua

A NOVELTY METHOD FOR PCB DEFECT DETECTION ON YOLOv8 BASIS

To tackle the challenges in PCB defect detection, I've developed a new algorithm built on an improved YOLOv8 framework. This approach is aimed at boosting detection accuracy while reducing model complexity, making it well-suited for detecting smaller targets and functioning effectively in environments with limited resources. The algorithm starts by introducing a refined neck network structure, which cuts down on the number of model parameters and computational demands, improving how efficiently resources are used. Additionally, the inclusion of ShuffleAttention and a BiFPN structure strengthens the model's ability to fuse features at multiple scales, significantly enhancing its performance with smaller targets.

On top of that, I've replaced the commonly used CIoU loss function with a WIoU loss function, which makes the model more accurate and robust in its detection capabilities. In tests, this enhanced model achieved impressive results, with mAP50 and mAP90-95 scores reaching 93.4% and 48.3%, respectively. What's more, model parameters, GFLOPs, and weight size were reduced by 33%, 12%, and 32%, respectively, bringing them down to 1.882M, 7.0, and 4.3M. This makes the solution not only highly efficient and accurate but also lightweight—perfect for use in constrained environments like mobile devices and embedded systems.

Keywords: BiFPN, Printed circuit board defect detection, ShuffleAttention, WIoU, YOLOv8.

ІЛЬЧУК Михайло
Національний університет «Львівська політехніка»

НОВИЙ МЕТОД ВИЯВЛЕННЯ ДЕФЕКТІВ ДРУКОВАНОЇ ПЛАТИ НА ОСНОВІ YOLOv8

Щоб вирішити проблеми з виявленням дефектів друкованої плати, я розробив новий алгоритм, побудований на вдосконаленій структурі YOLOv8. Цей підхід спрямований на підвищення точності виявлення, одночасно зменшуючи складність моделі, що робить його добре придатним для виявлення менших цілей і ефективного функціонування в середовищах з обмеженими ресурсами. Алгоритм починається з удосконаленої структури мережі шиї, яка скорочує кількість параметрів моделі та обчислювальних вимог, покращуючи ефективність використання ресурсів. Крім того, включення ShuffleAttention і структури BiFPN посилює здатність моделі об'єднувати функції в різних масштабах, значно підвищуючи її продуктивність з меншими цілями.

Крім того, я замінив поширену функцію втрати CIoU на функцію втрати WIoU, що робить модель точнішою та надійнішою у її можливостях виявлення. Під час випробувань ця вдосконалена модель досягла вражаючих результатів: показники mAP50 і mAP90-95 досягли 93,4% і 48,3% відповідно. Більше того, параметри моделі, GFLOP та розмір ваги були зменшені на 33%, 12% та 32% відповідно, зменшивши їх до 1,882M, 7,0 та 4,3M. Це робить рішення не тільки високоефективним і точним, але й легким — ідеальним для використання в обмежених середовищах, таких як мобільні пристрої та вбудовані системи.

Ключові слова: BiFPN, виявлення дефектів друкованої плати, ShuffleAttention, WIoU, YOLOv8.

INTRODUCTION

In today's electronics manufacturing, printed circuit boards (PCBs) are essential as the backbone of electronic devices. However, because of their complex production processes and the impact of external factors and operational errors, PCBs often develop defects such as short circuits, missing holes, or burrs. These defects can compromise performance or even lead to device failure, making accurate and efficient PCB defect detection vital. Recent advances in deep learning have significantly boosted image processing and computer vision, providing new avenues for detecting defects in PCBs.

Currently, there are two main types of target detection algorithms. The first is two-stage detection algorithms, like Faster RCNN, Cascade RCNN, and Mask RCNN. One recent development uses an enhanced Faster RCNN to detect PCB defects, improving the detection of small flaws [1]. Another innovation, the tiny defect detection network (TDD-Net), also based on Faster RCNN, achieved an impressive mAP of 98.0% for PCB defect detection [2]. The second type is single-stage detection algorithms, which include models like YOLO and SSD. A notable example is a method based on YOLOv3, designed to detect solder joint defects on PCB plug-ins, incorporating attention mechanisms to improve accuracy to 96.69% [3]. Similarly, an enhanced YOLOv4 model has been developed for PCB surface defect detection, achieving 98.6% accuracy while reducing model parameters and computational complexity [4]. Another contribution is PCB-YOLO, a surface defect detection algorithm built on YOLOv5, which uses the Swin Transformer and K-means++ to balance performance and computational efficiency, reaching a mAP of 95.97% at 92.5 FPS [5]. Additionally, a PCB defect detection model based on YOLOv7 was proposed, utilizing FasterNet with the CBAM attention mechanism to address challenges of speed and accuracy in defect detection. [6]

Despite these advancements, many of these models require substantial computational resources, making them impractical for lightweight and mobile devices. To overcome this, the focus of this study is to refine the YOLOv8 model for PCB defect detection. The aim is to create lightweight models that reduce size and computational demands while maintaining strong detection performance, ensuring they can be deployed efficiently in resource-limited settings like mobile devices. The key contributions of this work include:

✓ Enhancing the neck network of the original YOLOv8 to improve detection accuracy while reducing the number of parameters.

✓ Incorporating the BiFPN, which improves multi-scale feature integration, enabling better identification of small target defects [7].

✓ Adding ShuffleAttention (SA) to the neck network, enhancing the model’s ability to detect small targets and reducing both false and missed detections [8].

✓ Replacing the CIoU loss function with WIoU to improve anchor box quality, reduce detection difficulty for small targets, and boost accuracy for small defect features [9].

ENHANCEMENTS TO THE YOLOV8 NETWORK ARCHITECTURE

Enhanced YOLOv8 Network Architecture

YOLOv8 is one of the latest advancements in the YOLO series, and while its overall architecture remains similar to YOLOv5, it has introduced several significant improvements that enhance object detection. One of the key updates lies in its backbone network, which still uses the CSP approach but replaces the C3 module from YOLOv5 with the more efficient C2f module. This adjustment, along with changes in the number of channels across different scale models, greatly boosts the model's overall performance.

Moreover, the SPPF module from YOLOv5 continues to play a role, ensuring familiarity for those transitioning from the previous version. However, the neck network has been slightly modified. While it maintains the PAN architecture, it removes the CBS1×1 convolution structure in the up-sampling phase of PAN-FPN, which was present in YOLOv5. The head network has undergone even more changes, shifting from a coupling head to a decoupling head, which separates classification tasks from detection. YOLOv8 also moves from an Anchor-Based system to an Anchor-Free design, further improving its flexibility.[12]

For loss calculation, YOLOv8 uses the TaskAlignedAssigner strategy and introduces the DFLoss function. These enhancements make YOLOv8 highly adaptable and effective across a wide range of applications, from intelligent security and autonomous driving to object tracking and industrial quality inspection. Its optimized performance ensures it remains a powerful tool for various computer vision tasks.[13]

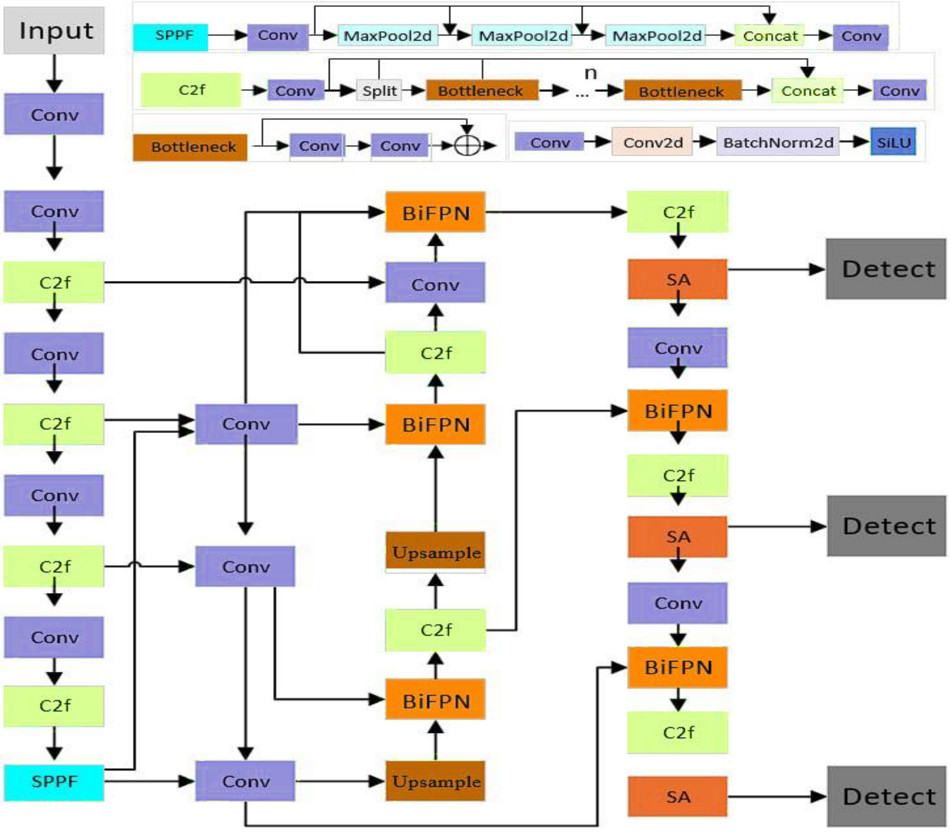


Fig. 1. Enhanced YOLOv8 structure scheme

This article takes the YOLOv8 network model a step further by introducing several key enhancements, as shown in Figure 1. First, an improved neck network structure is implemented to lower the model's parameter count and computational demands. Additionally, the SA (ShuffleAttention) mechanism and BiFPN structure are incorporated into the neck network, significantly improving the model's ability to fuse multi-scale features. To further boost performance, the CIoU loss function is replaced with the WIoU loss function, which speeds up model convergence and enhances detection accuracy.

The ShuffleAttention mechanism addresses the challenges of attention mechanisms by combining spatial and channel attention. There are two main types of attention mechanisms: spatial attention and channel attention. CBAM (Convolutional Block Attention Module) integrates both, capturing dependencies between channels and spatial pixel-level relationships, which improves accuracy but at the cost of increased computational complexity. The ShuffleAttention mechanism, however, resolves this issue by splitting the spatial and channel attention into separate blocks and running them in parallel. This method effectively enhances accuracy without adding much computational overhead, making it a more efficient option for complex tasks.[14]

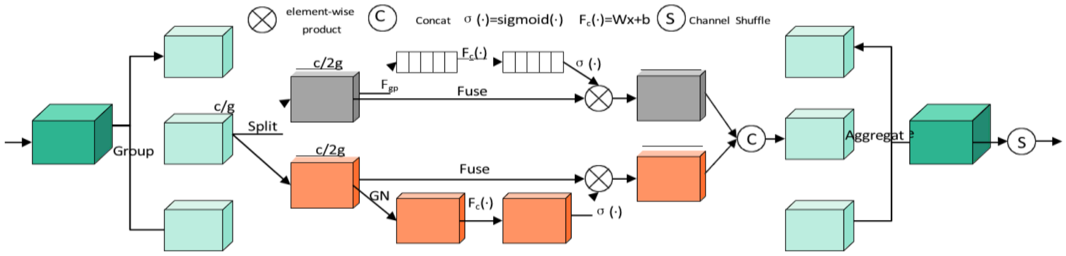
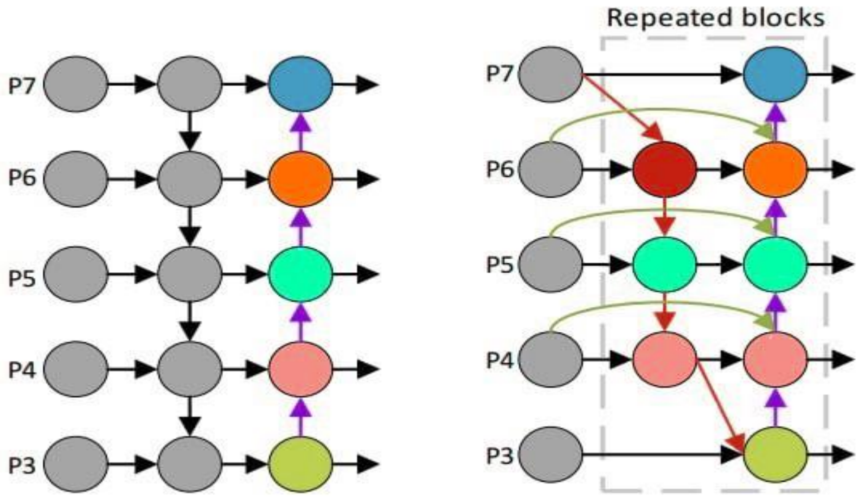


Fig. 2. Structural scheme of the SA attention mechanism

As illustrated in Figure 2, the ShuffleAttention (SA) mechanism begins by dividing the input features into multiple groups, referred to as "g groups." Each group is partially processed using a channel attention mechanism, highlighted in the gray sections of the figure, which functions similarly to the Squeeze-and-Excitation (SE) attention mechanism [15]. Meanwhile, another portion of the groups undergoes spatial attention processing, shown in the yellow sections, where group normalization (GN) is applied. These two components are then merged based on the number of channels, allowing for the fusion of information. Finally, the g groups are shuffled to produce the final output feature map.

In the realm of PCB defect detection, traditional methods often fall short when handling small defect features, which are difficult to capture and emphasize. To tackle this issue, this article introduces the SA attention mechanism, which aims to improve detection accuracy without adding extra computational burden.

Regarding the BiFPN structure, the PANet module [16], used in the neck of the original YOLOv8 model, integrates multi-scale features through a top-down information flow, as demonstrated in Figure 3. However, this approach is computationally expensive. PANet's complexity is notably higher than that of more lightweight alternatives, leading to slower training and inference times. Additionally, the top-down flow may result in losing fine details, particularly for smaller targets, which weakens the model's feature extraction capability. Furthermore, PANet often requires the storage and processing of multiple feature maps, especially when dealing with high-resolution images, making it less practical for environments with limited computational resources.



PANet **(b) BiFPN**

Fig. 3. Schemes of PANet and BiFPN structures

This article swaps out the PANet module in favor of the BiFPN module, as illustrated in part b of Figure 3. The BiFPN structure offers a two-way information flow, allowing features to move freely between different levels. This bidirectional approach improves the ability to capture multi-scale features, which is particularly useful for detecting both small and large targets. Additionally, BiFPN is lightweight and consumes fewer computing resources. This makes it ideal for object detection tasks in lightweight models or mobile devices, where speed and efficiency are crucial. Another benefit of BiFPN's bidirectional flow is that it helps gradients propagate more smoothly back to the underlying feature maps, which facilitates training deeper and more complex models. This smooth gradient flow speeds up model convergence and enhances training stability.

The original neck network is usually fine-tuned for specific tasks or datasets, but this can limit performance when applied across different domains or datasets. In YOLOv8, the original neck network design often adds extra computational complexity, requiring additional processing of feature maps from the backbone network. This extra processing can slow down model performance, especially in environments with limited resources, like embedded systems or mobile devices. Additionally, the original design may struggle to handle very small or very large objects, losing important details or failing to capture the overall target. This paper aims to refine and optimize the neck network, specifically focusing on improving small target defect detection in PCBs.

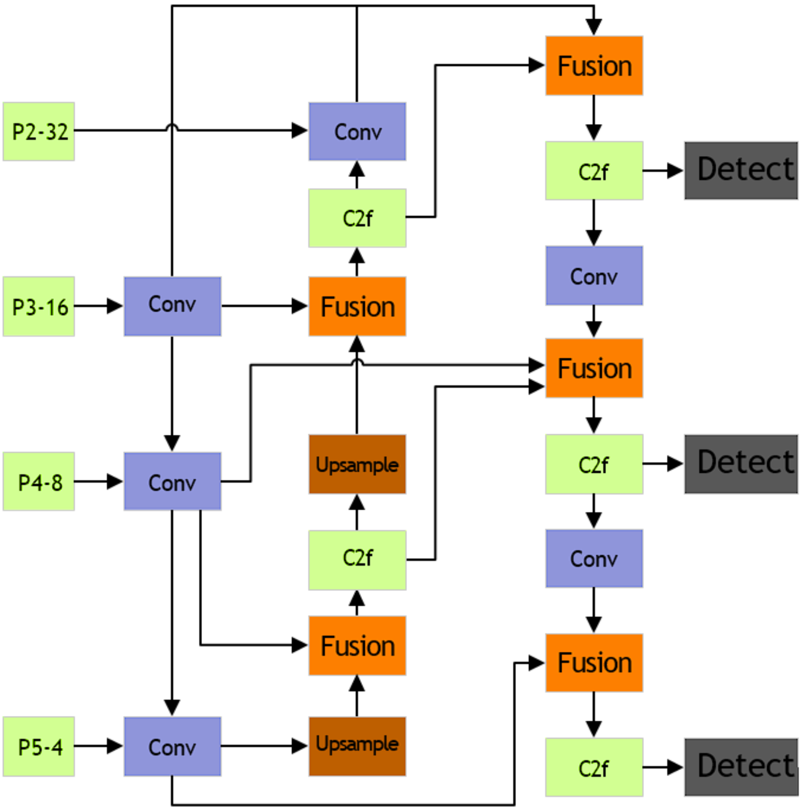


Fig. 4. Enhanced scheme of the neck network structure

As shown in Figure 4, three convolutional layers are initially added after the backbone network, and the number of channels in the neck network is fixed at 256. This adjustment helps the network prioritize the quality of feature fusion over sheer quantity, while also lowering memory usage and computational overhead. Additionally, the P2 section of the backbone is integrated into the neck network's feature fusion process, as it contains detailed information that enhances the detection of smaller targets. By incorporating features from P2, the model improves the precision of bounding box localization, leading to more accurate target identification. To further boost performance, three additional layers—Conv, Fusion, and C2f—are introduced into the neck network. The Fusion layer includes parameters such as Weight, Adaptive, Concat, and BiFPN, with the BiFPN fusion method being implemented as noted in reference [17].

The original YOLOv8 model uses the CIoU (Complete Intersection over Union) loss function to evaluate the quality of bounding box predictions in object detection tasks. CIoU extends the traditional IoU (Intersection over Union) by providing a more accurate assessment of the overlap between predicted and true bounding boxes. IoU calculates the ratio of the intersection to the union of two bounding boxes, but it has limitations, especially when the height or width of a box approaches zero, leading to poor gradient calculation. CIoU addresses these shortcomings by considering factors like the center point distance, width, and height of the bounding boxes, resulting in more reliable loss gradients and better model training.

$$\mathrm{L_{CIOU}} = 1 - \mathrm{IOU} + \frac{\mathrm{p}^2\left(\mathrm{b},\mathrm{b}^{\{gt\}}\right)}{c^2} + \alpha \mathrm{v} \tag{1}$$

In this formula:

IoU represents the traditional ratio of intersection to union between two bounding boxes.

p^2 refers to the square of the Euclidean distance between the predicted bounding box and the center point of the actual bounding box.

c^2 is a normalization factor for the center point distance, calculated as the square of the diagonal of the actual bounding box.

α is a weight parameter that helps balance the effect of the center point distance.

ν measures the difference in width and height between the actual and predicted bounding boxes.

While CIoU improves the detection of small targets, it still struggles with instability when dealing with targets that have extremely small or unusual aspect ratios. To tackle this, WIoU introduces a dynamic, non-monotonic focusing mechanism to better assess the quality of the anchor box. It offers a gradient gain allocation strategy that strengthens the algorithm’s ability to accurately position the target. WIoU also builds a two-layer attention mechanism, improving the model's generalization. The first version of WIoU (WIoUv1) is defined:

$$\mathrm{L_{WIOUv1}} = \mathrm{R_{WIOU}} \mathrm{L_{IOU}} \tag{2}$$

For the formula:

$$\mathrm{R_{WIOU}} = \exp\left(\frac{\left(\mathrm{x}-\mathrm{x_{gt}}\right)^2 + \left(\mathrm{y}-\mathrm{y_{gt}}\right)^2}{\left(\mathrm{W_g^2} + \mathrm{H_g^2}\right)^*}\right) \tag{3}$$

$$\mathrm{L_{IOU}} = 1 - \mathrm{IOU} \tag{4}$$

Explenation for the formula:

H_g^2 ,W_g^2 is the height and width of the minimum anchor block used to create gradients that prevent convergence.

WIoUv3 is the dynamic non-monotone focusing coefficient r on the basis of WIoUv1. Its formula is:

$$L_{WIOUv3} = rL_{WIOUv1} \tag{5}$$

$$r = \frac{\beta}{\delta\alpha^{\beta-\delta}} \tag{6}$$

$$\beta = \frac{L_{IOU}^{*}}{L_{IOU}} \in [0, +\infty) \tag{7}$$

In this formula:

$\mathrm{L_{IoU}^{*}}$ defines the separation of $\mathrm{L_{IoU}}$ from the calculation graph.

$\mathrm{L_{IoU}^{avg}}$ it is a moving average value of momentum m.

β is the outlier.

α and δ are hyperparameters which control the gradient amplification r.

RESULTS OF THE EXPERIMENT AND ANALYSIS

The PCB defect dataset used in this study comes from the Open Databases and personal experiments. It consists of images, covering six types of defects: missing hole, mouse bite, open circuit, short, spur, and spurious copper. The dataset was split into training and validation sets using an 8:2 ratio, with online data augmentation applied during the training process. Figure 5 provides close-up images of the various defects, with each image displaying 3 to 5 defects.

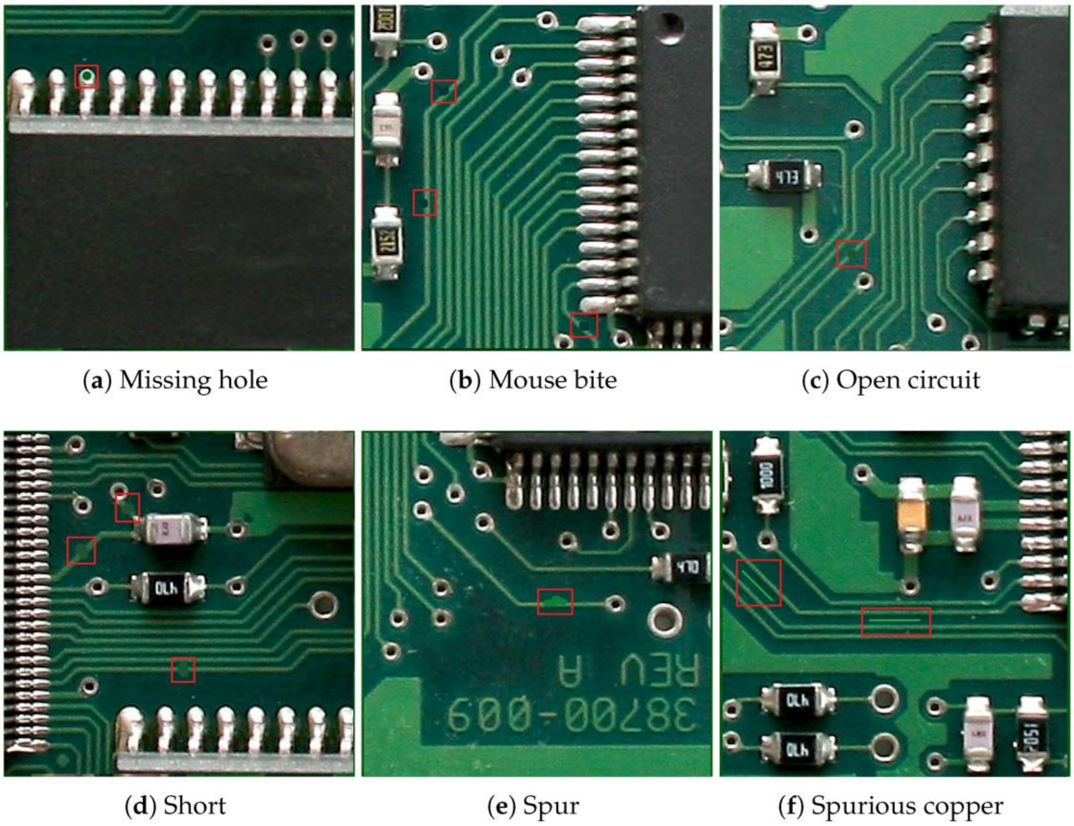

(**a**) Missing hole (**b**) Mouse bite (**c**) Open circuit
(**d**) Short (**e**) Spur (**f**) Spurious copper

Fig. 5. An example of a circuit board defect

The experimental hardware setup for this study includes an NVIDIA GeForce RTX 3080 with 12GB of video memory. The software environment is configured with Python 3.8.16, PyTorch 2.0.0, and CUDA 11.8. Training is performed using the PyTorch framework with the SGD optimizer. The hyperparameters are set as follows: a batch size of 32, 200 training epochs, an initial learning rate of 0.01, and no pre-trained weights are used during training.

The evaluation metrics used in this study include precision (P), recall (R), F1 score, mean average precision (mAP), parameter count (Params), Giga floating-point operations per second (GFLOPs), and model weight (Weights). The corresponding formulas are as follows:

$$P = \frac{TP}{(TP+FN)} \tag{8}$$

$$R = \frac{TP}{TP+FN} \tag{9}$$

$$F1 = \frac{2\times(P\times R)}{(P+R)} \tag{10}$$

$$mAP = \frac{1}{n}\sum_{i=1}^{n}\int_0^1 P(R)d(R) \tag{11}$$

Explenation for the formulas:

✓ True Positives (TP) refer to the number of instances where the model correctly predicts positive examples.

✓ False Positives (FP) occur when the model mistakenly classifies negative examples as positive.

✓ False Negatives (FN) represent positive examples that the model incorrectly labels as negative.

✓ Precision reflects how accurately the model identifies positive examples from its predictions.

✓ Recall indicates the model’s effectiveness in identifying all the true positive samples within the dataset.

✓ The F1 Score provides a balanced measure by combining both precision and recall into a single metric.

✓ Mean Average Precision (mAP) evaluates the model’s performance across various categories by averaging individual precision scores.

✓ Parameter Count refers to the total number of trainable parameters in the model, with higher counts usually indicating a more complex architecture.

✓ GFLOPs (Giga Floating-Point Operations Per Second) quantify the number of billions of operations with floating-point the model performs per each second, which gives insight into its computational requirements and speed.

✓ Model Weight refers to the size of the model’s stored parameters, impacting storage needs and deployment costs, especially important for applications on devices with limited resources, such as mobile platforms.

RESULT ANALYSIS

This article evaluates the effectiveness of replacing the CIoU loss function with the WIoU loss function in the original YOLOv8 model by comparing it to other well-known loss functions, such as DIoU [18], EIoU [19], GIoU [20], SIoU [21], and FocalIoU [22]. The comparative findings are summarized in Table 1.

Table 1.

The experiment of comparing the loss function

Loss function	**mAP50(%)**	**mAP90-95(%)**	**P(%)**	**R(%)**	**F1(%)**
CIoU	90.5	46.3	91.7	85.5	88.0
DIoU	91.6	47.4	93.8	86.6	91.0
EIoU	90.5	47.6	92.9	84.7	89.0
GIoU	89.8	45.7	90.5	85.5	88.0
SIoU	90.3	47.2	90.7	85.6	88.0
FocalIoU	91.9	46.8	92.8	86.8	91.0
WIoU	92.8	49.2	91.5	89.7	92.0

The results clearly show that WIoU offers significant advantages across several critical performance metrics. Notably, it achieves the highest mean Average Precision (mAP), leading to more precise detection of defective targets. Furthermore, WIoU outperforms the other loss functions in both precision and recall, highlighting its enhanced ability to predict true positives and capture more actual positives. The F1 score for WIoU also demonstrates its superior performance in the PCB defect detection task discussed in the paper.

The line diagram in Figure 6 illustrates the effect of different IOU calculation methods on the YOLOv8 model's performance. Specifically, WIoU exhibits clear benefits in terms of val/box_loss, showing a lower loss value compared to other functions. This indicates that WIoU is more effective at matching object boundary boxes.

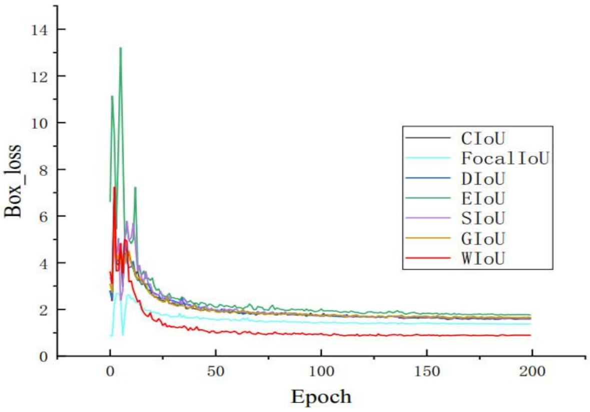


Fig. 5. Experimental results of various loss functions

To validate the effectiveness of the improvements made to the YOLOv8 network structure, ablation experiments were conducted on several enhancement points under identical conditions. The findings are summarized in Table 2. Introduction of the improved neck network: The results showed a notable increase in accuracy compared to the original YOLOv8n, while the number of parameters, GFLOPs, and model weight decreased. Addition of the BiFPN structure: This led to an improvement in mAP, with minimal changes in the number of parameters and GFLOPs. Incorporation of the SA attention mechanism: This further boosted mAP without significantly affecting the parameter count or GFLOPs. Replacement of CIoU with WIoU loss function: This substitution enhanced mAP while maintaining a similar number of parameters and GFLOPs. Combination of the improved neck network with BiFPN: This configuration resulted in fewer parameters, GFLOPs, and a smaller weight file size compared to the original YOLOv8n, Improvement 1, and Improvement 2, although the mAP was slightly lower than that of Improvement 1. Addition of the SA attention mechanism to the neck network with BiFPN: This retained the number of parameters, GFLOPs, and weight file size, while achieving a higher mAP than the previous configurations. Finally, the algorithm integrates all four improvements into the original YOLOv8n. The results show that mAP50 increased by 3.4%, while the number of parameters, GFLOPs, and weight file size decreased by 33%, 12%, and 32%, accordingly.

Table 2.

The experiment of comparing the loss function

Model	New Neck	BiFPN	SA	WIoU	mAP50(%)	Params/M	GFLOPs	Weight/M
YOLOv8n					90.5	3.023	8.2	6.3
Improvement 1	√				92.9	2.063	7.4	4.5
Improvement 2		√			90.9	3.088	8.4	6.5
Improvement 3			√		91.7	3.013	8.2	6.4
Improvement 4				√	92.8	3.013	8.2	6.3
Improvement 5	√	√			91.9	1.882	7.2	4.3
Improvement 6	√	√	√		93.6	1.882	7.2	4.3
Proposed model	√	√	√	√	93.4	1.882	7.2	4.3

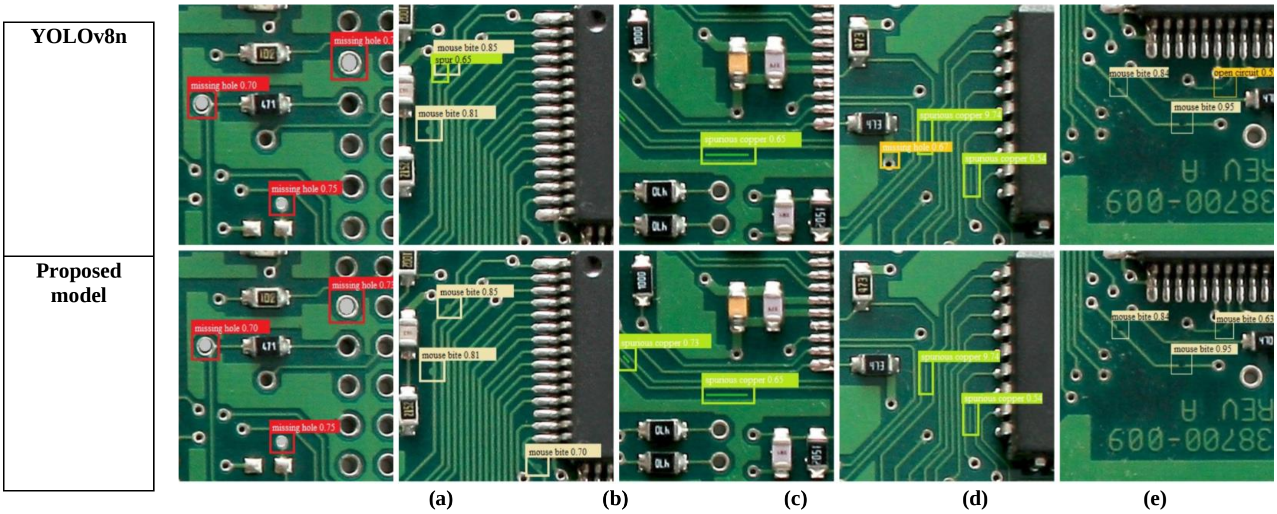
To evaluate the effectiveness of the proposed algorithm, a comparative experiment was conducted, with results presented in Table 3. The findings highlight several key advantages of the algorithm. Firstly, the proposed algorithm has a parameter count of only 1.882M, which is significantly lower than SSD's 23.270M [23], which makes it very suitable for mobile deployment and other environments with limited resources. Additionally, the algorithm's computational workload is just 7.0 GFLOPs, that is much less than the highest recorded value of 265.3 GFLOPs in other models, ensuring efficient real-time detection on mobile devices. In terms of accuracy, the algorithm achieves a strong mAP50 score of 93.4%, outperforming most other models and approaching YOLOv8s' score of 93.9%. Moreover, it performs impressively in the mAP90-95 range, reaching 48.3%, demonstrating its ability to accurately capture and detect target objects with high reliability.

Table 3.

The experiment of comparing the loss function

Model	mAP50(%)	mAP90-95(%)	Params/M	GFLOPs	Weight/M
SSD	83.6	35.8	23.270	265.3	92.8
YOLOv5n	65.8	27.2	1.758	4.3	3.6
YOLOv5s	86.9	41.9	7.134	15.6	15.1
YOLOv6s	90.2	44.8	16.135	43.7	31.9
YOLOv7-tiny	78.5	34.8	6.264	12.6	13.1
YOLOv8n	90.7	46.3	3.147	8.3	6.7
YOLOv8s	93.9	50.5	11.228	28.2	22.8
Proposed model	93.4	48.3	1.882	7.0	4.3

In conclusion, the method described in this article is an excellent choice for deployment in resource-constrained environments like mobile devices. Its lightweight design, low computational requirements, and high detection precision give it wide potential for applications in real-time object detection, built-in systems, and mobile apps. To further demonstrate the algorithm's effectiveness in real-world detection scenarios, five experimental comparison sets were carried out, as in Figure 7, featuring scaled-up images. First, the results from the initial set show that the proposed method outperforms the default YOLOv8n in detection precision, meaning it identifies defects with greater accuracy. Second, the outcomes of the II and III sets indicate that the algorithm has a lower rate of missed detections compared to YOLOv8n, suggesting it captures more potential defects and improves detection coverage. And, finally, the IV and V sets reveal that this algorithm significantly reduces the number of false detections, leading to fewer normal objects being incorrectly identified as defects, thus minimizing the risk of false alarm signals. Overall, the method presented in this article has demonstrated substantial enhancements in detection accuracy, reduced number of missed detections, and fewer false detections, making it more substantial and effective tool for defect detection.



(a) (b) (c) (d) (e)

Fig. 7. Comparison table of Visualized Results

CONCLUSION

This is research introduces an optimized version of the YOLOv8 model tailored for detecting defects on printed circuit boards (PCBs). Key enhancements include the addition of a new neck network, which significantly reduces both the number of parameters and the model's computational load, improving its efficiency, particularly in environments with limited resources. Furthermore, the integration of the ShuffleAttention (SA) mechanism and BiFPN structure enhances the model's ability to fuse features across multiple scales, making it better suited for identifying small targets. A crucial change was the replacement of the traditional CIoU loss function with the WIoU loss function, leading to a marked improvement in detection accuracy and model reliability. The experimental outcomes affirm the effectiveness of these upgrades, with the model achieving remarkable results in PCB defect detection. Specifically, it delivered impressive metrics, with an mAP50 of 94.2% and an mAP90-95 of 49.0%. Despite these high accuracy rates, the model retains a low parameter count and minimal computational requirements, making it a strong candidate for deployment on mobile devices and built-in systems in limited environments. Comparative analysis with other detection models shows this method strikes an excellent balance between a lightweight design and high performance, suggesting significant potential for real-world applications. In conclusion, this study presents a method that is both efficient and highly precise for detecting PCB defects, offering practical benefits for industrial quality control and automated manufacturing. Future work could extend the model's applicability to other fields and further refine its productivity and potency.

References

[1] Li, J., et al. (2022). "Improving PCB Defect Detection Using Cascade R-CNN and Mask R-CNN with Contextual Information." IEEE Transactions on Industrial Electronics.

[2] Chen, L., et al. (2021). "Advanced PCB Defect Detection with TDD-Net: Enhancing Tiny Defect Recognition." IEEE Transactions on Industrial Informatics.

[3] Zhang, Y., et al. (2021). "Improving Object Detection Performance with Attention Mechanisms: A Focus on PCB Defect Detection." IEEE Transactions on Automation Science and Engineering.

[4] Zhou et al. (2023). "Lightweight YOLOv4 for Real-Time PCB Defect Detection".

[5] Zhang et al. (2023). "PCB-YOLO: A High-Performance Surface Defect Detection Algorithm Based on YOLOv5" IEEE Transactions on Industrial Informatics.

[6] Wang et al. (2024). "YOLOv7-Based PCB Defect Detection Model Enhanced by FasterNet and CBAM". Published in Pattern Recognition and Artificial Intelligence.

[7] J. Doherty et al. (2023) Bifpn-Yolo: One-Stage Object Detection Integrating Bi-Directional Feature Pyramid Networks

[8] Zhang Q L, Yang Y B. Sa-net: Shuffle attention for deep convolutional neural networks[C]//ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021: 2235-2239.

[9] Tong Z, Chen Y, Xu Z, et al. Wise-IoU: Bounding Box Regression Loss with Dynamic Focusing Mechanism[J]. arXiv preprint arXiv:2301.10051, 2023.

[10] Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[11] Bochkovskiy A, Wang C Y, Liao H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[12] S. Liu, et al. Analysis of Anchor-Based and Anchor-Free Object Detection Methods Based on Deep Learning. 2020 IEEE International Conference on Mechatronics and Automation (ICMA).

[13] Li C, Li L, Jiang H, et al. YOLOv6: A single-stage object detection framework for industrial applications[J]. arXiv preprint arXiv:2209.02976, 2022.

[14] Y. Wang, et al. An attention mechanism module with spatial perception and channel information interaction. Complex & Intelligent Systems.

[15] Hu J, Shen L, Sun G. Squeeze-and-excitation networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7132-7141.

[16] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 8759-8768.

[17] Tan, M., Pang, R., & Le, Q. V. "BiFPN: Bidirectional Feature Pyramid Network"

[18] Zheng Z, Wang P, Liu W, et al. Distance-IoU loss: Faster and better learning for bounding box regression[C]//Proceedings of the AAAI conference on artificial intelligence. 2020, 34(07): 12993-13000.

[19] Zhang Y F, Ren W, Zhang Z, et al. Focal and efficient IOU loss for accurate bounding box regression[J]. Neurocomputing, 2022, 506: 146-157.

[20] Rezatofighi H, Tsoi N, Gwak J Y, et al. Generalized intersection over union: A metric and a loss for bounding box regression[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 658-666.

[21] Gevorgyan Z. SIoU loss: More powerful learning for bounding box regression[J]. arXiv preprint arXiv:2205.12740, 2022.

[22] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.

[23] Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016: 21-37.