

Article

Automated Book Defect Detection for Quality Inspection in China's Publishing Logistics Using LME-YOLO

Weixiang Lin ^{1,2,†}, Qin Xu ^{2,*,†}, Wencheng Hong ¹, Lingyi Liang ¹, Tao Wang ¹, Xiaoping Wu ¹ and Liangquan Jia ¹ 

¹ School of Information Engineering, Huzhou Normal University, Huzhou 313000, China; 2025388134@stu.huznu.edu.cn (W.L.); 2025388009@zjhu.edu.cn (W.H.); 2025388131@stu.zjhu.edu.cn (L.L.); 2022388330@stu.zjhu.edu.cn (T.W.); wuxipu@gmail.com (X.W.); 02426@zjhu.edu.cn (L.J.)

² School of Economics and Management, Huzhou Normal University, Huzhou 313000, China

* Correspondence: 02449@huznu.edu.cn

† These authors contributed equally to this work.

Abstract

China's publishing industry continues to expand across production, warehousing, circulation, and export. Book quality inspection affects product grading, logistics sorting, customer satisfaction, and the international image of Chinese publishing. However, these defects are often small, irregular, and low-contrast, and their visual similarity to paper textures and printed content complicates automated classification and localization. To address this problem, this study proposes Local-detail Modelling and Edge-enhancement YOLO (LME-YOLO), a book defect detection model based on YOLO11n for quality inspection in China's publishing logistics. On a dataset of 632 book images, LME-YOLO achieved a Precision of 0.829, Recall of 0.777, mAP50 of 0.818, and mAP50–95 of 0.465. The model contains 2.64 M parameters and requires 6.6 GFLOPs. Among the detectors compared in this study, LME-YOLO achieved the highest mAP50 and mAP50–95. Compared with YOLO11n, its mAP50 and mAP50–95 increased by 0.015 and 0.006, respectively, with increases of 0.05 M parameters and 0.2 GFLOPs. Compared with YOLOv8, its mAP50 and mAP50–95 increased by 0.017 and 0.005, respectively, while the parameter count and computational cost decreased by 0.37 M and 1.6 GFLOPs.

Keywords: book defect detection; publishing logistics; quality inspection; object detection; edge-enhanced fusion

1. Introduction

In China's publishing industry, books are both cultural products and physical goods that undergo production, warehousing, sorting, distribution, export, and return-handling processes. Since China formally implemented the publishing “going global” strategy in 2003 [1], programmes such as the China Book International programme and the Silk Road Book Project [2] have promoted the international circulation of Chinese publications. In this context, the physical quality of books directly affects product grading, logistics sorting, customer satisfaction, and the international image of Chinese publishing. Appearance defects such as Ink Stain, Folded Corner, Tear, and Hole may occur during printing, storage, transportation, or circulation and therefore require inspection throughout the publishing-logistics workflow. At present, book appearance inspection still relies mainly on manual visual checking. Conventional manual visual inspection is susceptible to inspector experience, attention, and fatigue, resulting in subjective and variable outcomes [3,4]. Manual visual inspection is constrained by production and sorting throughput requirements, and



Academic Editor: Guanqiu Qi

Received: 14 July 2026

Revised: 18 August 2026

Accepted: 18 August 2026

Published: 21 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

its results may vary among inspectors [5]. Sampling inspection cannot cover all books, while repeated manual handling may cause secondary contamination or physical damage [6]. Machine-vision and deep-learning methods learn texture, edge, and shape features from book images and have been applied to detecting small and irregular defects under variations in paper texture, printed content, illumination, and imaging conditions.

Previous studies have shown the feasibility of machine vision and object detection for book-related and printed-material defect inspection. Wang et al. applied machine vision to damage recognition in ancient paper books and built a detection system based on image acquisition and damage-region analysis [7]. Hong and Jeong used a YOLO-based framework for automatic condition examination of used books, modelling book appearance states as detectable targets [8]. Tomas et al. combined YOLOv8 with MIRNet low-light image enhancement to recognize multiple types of book appearance defects, showing that acquisition conditions and image enhancement can affect recognition stability [9]. In the broader field of surface defect detection, Bhatt et al. showed that deep learning can reduce dependence on handcrafted features and manual inspection rules [10]. Ren et al. reviewed the development of machine-vision defect detection and emphasized that defect morphology, imaging conditions, and inspection scenarios affect recognition performance [5]. Liu et al. further surveyed real-time surface-defect inspection methods and evaluated detection accuracy together with inference efficiency as deployment-related criteria [11]. However, existing work still pays limited attention to book defect detection as a logistics quality inspection problem. For printed-material defects, Peng et al. focused on detection under uneven illumination, indicating that illumination variation and material surface changes increase inspection difficulty [12]. Liu et al. proposed a YOLO-based printing-defect detector and evaluated its parameter count and computational cost in relation to deployment requirements [13]. In contrast, small-defect recognition, weak-boundary localization, and robustness to paper texture interference in book production, warehousing, and sorting environments remain insufficiently studied.

Based on the above analysis, this study addresses the demand for automated book appearance quality inspection in China's publishing logistics. Using the nano variant of You Only Look Once version 11 (YOLO11n) [14] as the baseline framework, we propose an improved model termed Local-detail Modelling and Edge-enhancement YOLO (LME-YOLO) for four common defect categories: Ink Stain, Folded Corner, Tear, and Hole. Inspired by the "See Large, Focus Small" design of LSNet [15] and the edge-aware contour modelling motivation of MEMO [16], we construct a fine-grained feature enhancement module termed C3k2_LSNet and design MEMFuseBlock as a task-specific edge-enhanced fusion module. The proposed design strengthens page-context modelling and local defect detail representation, while the edge-information compensation mechanism improves localization for weak-boundary defects such as Tear and Hole. The model targets automated quality inspection in publishing logistics and is evaluated using detection accuracy, parameter count, and computational cost.

The main contributions of this study are as follows:

1. This study develops LME-YOLO, a lightweight book defect detection model based on YOLO11n. Considering that book defects are often small and local details can be weakened during feature extraction, C3k2_LSNet is constructed in part of the C3k2 structure to enhance local detail representation and contextual modelling.
2. MEMFuseBlock is introduced as an edge-enhanced fusion module. It extracts, filters, and supplements high-frequency edge residual information in features, thereby strengthening the model's perception of weak-boundary defects such as Tear, Hole, Ink Stain, and Folded Corner.

3. A custom-built book defect detection dataset is constructed, covering four common defect categories: Ink Stain, Folded Corner, Tear, and Hole. Comparative experiments, ablation experiments, and loss-function experiments verify the balance between detection accuracy and model complexity for logistics-oriented quality inspection.

The remainder of this paper is organized as follows. Section 2 introduces the construction of the book defect dataset, the overall architecture of LME-YOLO, and the C3k2_LSNNet and MEMFuseBlock modules. Section 3 describes the experimental setup and evaluation metrics and presents the comparative, ablation, loss-function, and visualization results. Section 4 discusses model performance, computational requirements, deployment conditions, and study limitations. Section 5 concludes the paper and outlines future work.

2. Materials and Methods

2.1. Dataset Construction

Experiments were conducted on a custom-built book defect detection dataset. The images were collected from real book inspection scenes and covered page appearance defects generated during use, transportation, storage, and circulation. The dataset contains 632 images, including 456 training images, 115 validation images, and 61 test images, corresponding approximately to a (7:2:1) split. The three subsets were kept mutually independent during partitioning. Images from the same book, identical physical background, or similar consecutive acquisition scene were not assigned across the training, validation, and test subsets, thereby reducing the risk that visually similar samples could inflate the generalization assessment.

The defects were divided into four categories according to their observable damage patterns. Ink Stain refers to an unintended local dark or colored region whose texture differs from the surrounding paper or printed content; its annotation encloses the visible contaminated region. Folded Corner refers to a triangular or polygonal page region displaced by folding, usually accompanied by a crease, shadow, or change in page orientation; the folded region and its principal crease are enclosed together. Tear denotes a physical discontinuity of the paper that commonly appears as an irregular, narrow, jagged, or low-contrast damaged edge; the annotation covers the continuous visible damaged portion while excluding unrelated page texture. Hole denotes an enclosed area of missing paper material, often accompanied by a clear surrounding contour or visible background through the page; the annotation encloses the missing-material region and its boundary. These category definitions distinguish local texture contamination, geometric deformation, open-edge damage, and enclosed structural loss, respectively. Figure 1 presents representative examples of the four categories.

2.2. Data Annotation and Preprocessing

After manual screening, each defect was annotated with a bounding box recording its category and spatial location. For irregular targets such as Tear and Hole, the box was defined to cover the principal visible defect region without including a large area of normal paper background. The annotations were then reviewed for incorrect categories, missed instances, duplicate boxes, and localization offsets. This annotation format is consistent with the class-prediction and bounding-box-regression outputs of the detection head.

Before entering the network, each image was resized to 1280×1280 pixels, converted to floating-point values, and normalized from $[0, 255]$ to $[0, 1]$ by division by 255. Data augmentation was performed only after dataset partitioning and was applied exclusively to the training subset. Mosaic augmentation was applied with a probability of 1.0, the translation and scaling parameters were set to 0.1 and 0.5, respectively, and horizontal flipping was applied with a probability of 0.5. The geometric operations were limited

to horizontal flipping, isotropic scaling, and constrained translation/cropping. Random rotation, shear, perspective deformation, and vertical flipping were not used because these operations could alter the geometric relationship between a page corner, its crease, and the folded region. For every applied geometric transformation, the bounding-box coordinates were transformed using the same mapping and clipped to the image boundary. An augmented instance was retained only when the principal visible defect region remained in the image and the transformed bounding box retained a valid area; instances for which cropping removed the principal defect region or produced an invalid box were excluded. Color perturbation was performed in the HSV color space using $hsv_h = 0.015$, $hsv_s = 0.7$, and $hsv_v = 0.4$.



Figure 1. Examples from the book defect detection dataset.

2.3. Overall Model Architecture

The proposed LME-YOLO consists of the YOLO11n baseline, the C3k2_LSNNet fine-grained feature enhancement module, and the MEMFuseBlock edge-enhanced fusion module. YOLO11n was selected as the baseline because its real-time detection structure provides multi-scale prediction with a relatively low parameter count and computational cost [14]. Previous studies of real-time surface inspection and lightweight printing-defect detection likewise show that an industrial inspection model must balance detection accuracy, inference efficiency, and edge-device deployment cost [11,13].

As shown in Figure 2, the input image is first downsampled progressively through convolution (Conv) and C3k2 modules. In the C3k2_LSNNet detail panel of Figure 2, the asterisk (*) following LSNNetBlock indicates that this block is repeated according to the configured repetition number. Spatial Pyramid Pooling-Fast (SPPF) expands the multi-scale contextual receptive range at the end of the backbone, and C2 Position-Sensitive Attention (C2PSA) further selects informative responses. The neck uses top-down and bottom-up paths. High-level features are upsampled by a factor of two using nearest-neighbor interpolation and concatenated with the corresponding shallower features; 3×3 convolutions with stride two then downsample the fused features before they are recombined with intermediate and deep features. This top-down and bottom-up fusion strategy follows the multi-scale feature fusion principles of FPN and EfficientDet [17,18], and is also consistent with PANet, which uses bottom-up path augmentation to shorten the propagation

path of low-level localization information [19]. On the high-resolution P3 branch and the subsequently reconstructed P4 and P5 branches, C3k2_LSNNet first enhances local detail and context, after which MEMFuseBlock supplements high-frequency boundary responses. P3, P4, and P5 correspond to input strides of approximately 8, 16, and 32 pixels, respectively, and are passed to three detection heads for four-class prediction and bounding-box regression. During training, the detection head uses a classification objective together with bounding-box regression and distribution focal losses; during inference, it outputs class confidence scores and predicted boxes.

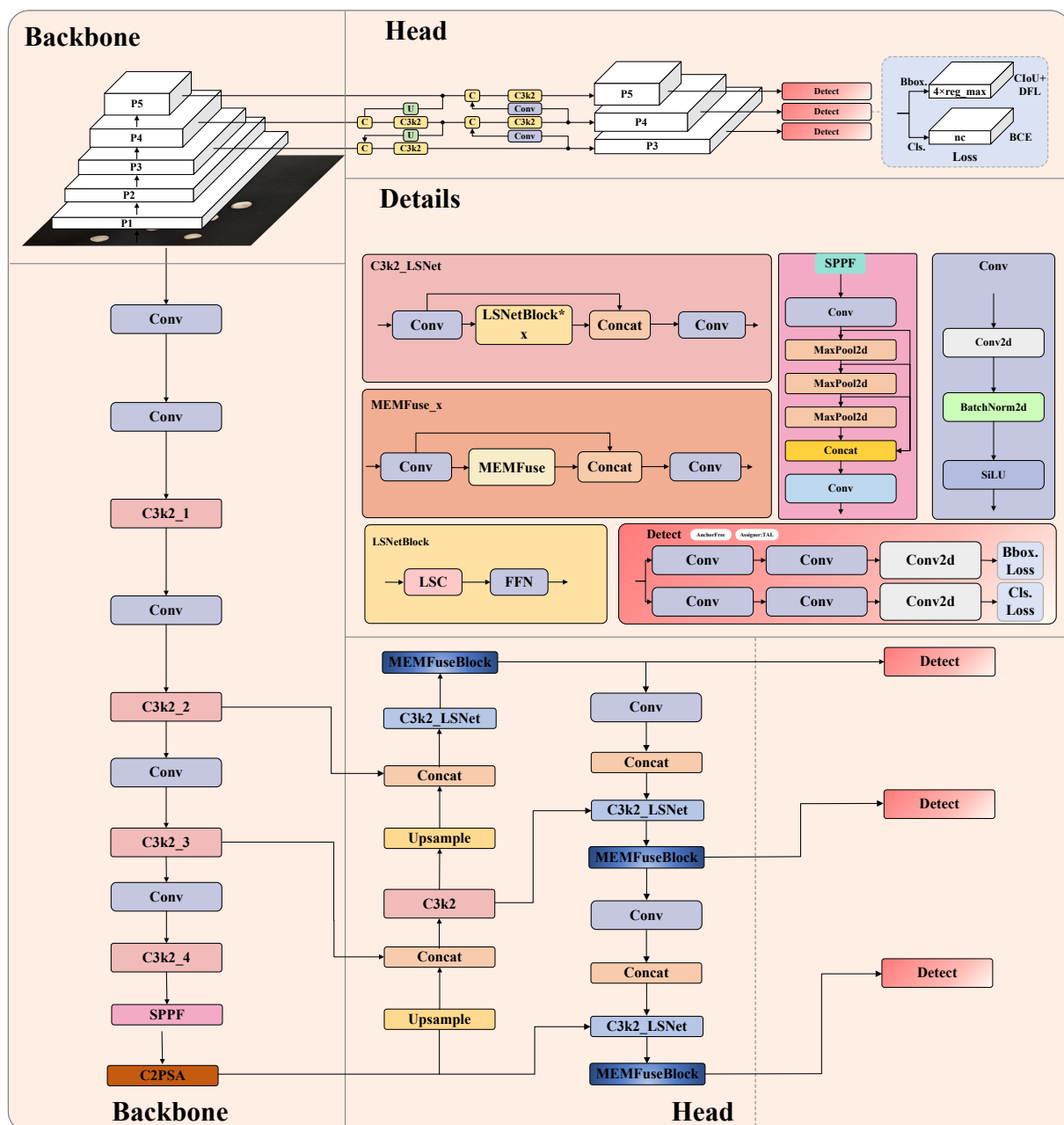


Figure 2. Structure of the proposed improved YOLO11n book defect detection model.

2.4. YOLO11n Baseline Detection Network

YOLO11n is the nano-scale lightweight version of the YOLO family. Its backbone forms hierarchical features through stride convolutions. C3k2 reuses features through branch splitting, bottleneck transformation, and channel concatenation while controlling computational cost; SPPF aggregates context at different pooling scales; and C2PSA em-

phasizes discriminative spatial and channel responses. The neck combines different feature resolutions through upsampling, concatenation, and convolutional downsampling, and the three detection heads perform classification and localization at different scales.

Logistics sorting lines require continuous image processing, while practical edge devices generally provide less memory, power, and computational capacity than the unified training platform. A large detector increases per-frame latency and deployment cost [11,13]. MobileNets demonstrated that depthwise-convolution-based architectures can provide a controllable trade-off between accuracy and computational cost for mobile and embedded vision tasks [20]. In LME-YOLO, the standard fusion units at the three YOLO11n neck scales are replaced with C3k2_LSNNet followed by MEMFuseBlock. This configuration concentrates the additional computation on local-detail refinement and boundary compensation while retaining the YOLO11n backbone.

2.5. C3k2_LSNNet Fine-Grained Feature Enhancement Module

C3k2_LSNNet retains the external branch splitting and concatenation of C3k2 but replaces its internal standard Bottleneck with LSNNetBlock. This choice follows the “See Large, Focus Small” rationale of LSNNet for broad-context modelling and local selection [15]. Given an input $X \in \mathbb{R}^{B \times C \times H \times W}$, LSNNetBlock first applies a 1×1 convolution, batch normalization, and SiLU activation for channel projection, producing F_0 . Within C3k2_LSNNet, the outer channel expansion ratio is 0.5, and each internal LSNNetBlock preserves the same input and output hidden-channel width.

LSConv consists of Large-Kernel Projection (LKP) and Selective Kernel Aggregation (SKA). LKP sequentially applies 1×1 channel compression, a 7×7 depthwise convolution, and a 1×1 projection, with batch normalization and ReLU after the convolutional stages. The use of a large depthwise kernel is further supported by RepLKNet, which showed that larger convolution kernels can expand the effective receptive field and strengthen shape modelling [21]. The projected features are then group-normalized and reshaped into spatially varying weights for each 3×3 neighborhood:

$$W = \text{LKP}_{7 \times 7}(F_0), \quad W \in \mathbb{R}^{B \times (C/G) \times 9 \times H \times W}, \quad (1)$$

In Equation (1), $G = \max(1, C/16)$ is the group number. Using the spatially varying weights defined in Equation (1), SKA applies a 3×3 unfold operation with stride one and padding one to extract the nine-neighborhood at each pixel. The neighborhood is multiplied element-wise by W , summed along the neighborhood dimension, batch-normalized, and added to the input through a residual connection, as expressed in Equation (2):

$$F_{\text{LS}} = F_0 + \text{BN} \left(\sum_{q \in \mathcal{N}_{3 \times 3}} W_q \odot F_{0,q} \right). \quad (2)$$

The Squeeze-and-Excitation (SE) branch then generates channel weights using global average pooling, two 1×1 convolutions, ReLU, and Sigmoid, with a channel reduction ratio of four [22]. The weighted feature enters a lightweight Feed-Forward Network (FFN), in which the first 1×1 convolution doubles the channel width and the second restores it; a residual connection preserves the input information [23]. Finally, a 1×1 convolution, batch normalization, and SiLU produce the module output, followed by an outer residual connection when the input and output dimensions are equal. Figure 3 shows this sequence of input projection, LSConv, SE, FFN, output projection, and two residual paths; Figure 3a,b show LSNNetBlock and LSConv, respectively.

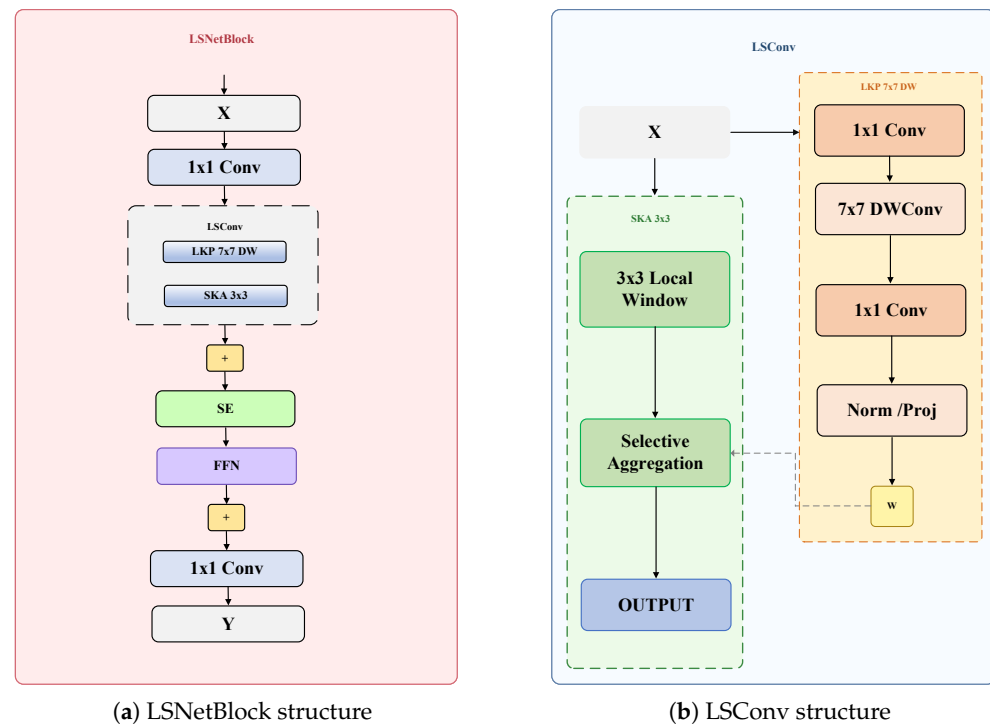


Figure 3. Structures of LSNetBlock and LSConv.

2.6. MEMFuseBlock Edge-Enhanced Fusion Module

MEMFuseBlock is placed after C3k2_LSNet at the three neck fusion scales to recover boundary details that may be smoothed during feature aggregation. Zhang et al. used a coarse-to-fine strategy to localize workpiece edges for fine-grained defect recognition [6], whereas MEMO models contours through masked edge pretraining [16]. Xception separates spatial-correlation modelling from cross-channel mapping through depthwise and pointwise convolutions [24]. These studies motivate the contour-oriented processing and separated spatial-channel operations used in MEMFuseBlock. MEMFuseBlock implements edge compensation through low-frequency separation, directional depthwise convolution, gated selection, and residual write-back. The stages in Figure 4 follow the actual forward-propagation order.

Given an input feature $X \in \mathbb{R}^{B \times C \times H \times W}$, local average pooling with kernel size three, stride one, and padding one first estimates the low-frequency component, from which the high-frequency residual is obtained:

$$L = \text{AvgPool}_{3 \times 3}(X), \quad (3)$$

$$H_f = X - L. \quad (4)$$

Because the stride and padding preserve the spatial dimensions, H_f remains in $\mathbb{R}^{B \times C \times H \times W}$. A channel-wise 3×3 depthwise convolution then refines local detail, followed by 1×3 and 3×1 depthwise convolutions for horizontal and vertical responses:

$$D = \text{DWConv}_{3 \times 3}(H_f), \quad (5)$$

$$E = \text{DWConv}_{1 \times 3}(D) + \text{DWConv}_{3 \times 1}(D). \quad (6)$$

For each depthwise convolution, the group number equals the channel number C , avoiding dense cross-channel convolution. To suppress irrelevant high-frequency responses caused by paper texture, printed characters, or reflection, a 1×1 convolution followed by Sigmoid produces a channel- and position-dependent gate:

$$G = \sigma(\text{Conv}_{1 \times 1}(E)). \quad (7)$$

The final output is

$$Y = X + \gamma(E \odot G), \quad (8)$$

where $\odot$ denotes element-wise multiplication and γ is a learnable scalar initialized to zero. Zero initialization makes the module an identity mapping at the beginning of training and allows the edge-compensation strength to be learned gradually. The five stages correspond to average pooling, detail depthwise convolution, bidirectional depthwise convolution, confidence gating, and residual write-back in Figure 4. The input and output tensor dimensions are unchanged, allowing the module to be inserted directly into the P3, P4, and P5 fusion branches.

The dashed arrows in Figure 4 indicate cross-stage feature-routing paths. Specifically, X is routed to the high-frequency subtraction and final residual-addition operations, while E is routed to the gating and element-wise multiplication branches.

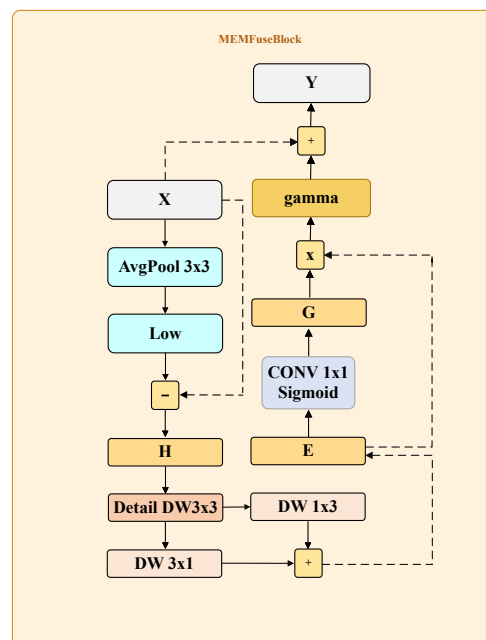


Figure 4. Structure of the proposed MEMFuseBlock.

3. Experimental Results and Analysis

3.1. Experimental Platform and Parameters

The comparison models were evaluated under a unified experimental protocol. All models were trained from scratch using the same mutually independent dataset split, input resolution, data-augmentation policy, batch size, number of epochs, optimizer, initial learning rate, hardware platform, and evaluation metrics. The test subset was not used to select training parameters. The shared configuration was selected according to validation convergence and resource feasibility and was fixed across all models. The experimental platform and main training parameters are listed in Table 1.

The hyperparameter values were fixed before final test evaluation according to defect scale, memory feasibility, and validation convergence. The input size of 1280×1280 was used to retain spatial information for small and weak-boundary defects after resizing. The batch size was fixed at 16 for all models under the common hardware configuration. All models were trained for 300 epochs. AdamW, which applies decoupled weight decay, was used with an initial learning rate of 0.001 for all models trained from scratch [25]. These

values were fixed across all compared models before the final test evaluation, and the test set was not used for hyperparameter selection.

Table 1. Experimental platform and training parameter configuration.

Parameter	Configuration
Operating system	Ubuntu
Python version	3.10
PyTorch version	2.12.0 + cu130
Graphics processing unit (GPU)	NVIDIA GeForce RTX 5070
Input image size	1280 × 1280 pixels
Batch size	16
Training epochs	300
Optimizer	AdamW [25]
Initial learning rate	0.001
Dataset split	7:2:1

The evaluation metrics include Precision, Recall, Average Precision (AP), and mean Average Precision (mAP). True positives (TP), false positives (FP), and false negatives (FN) are used to calculate Precision and Recall, while Intersection over Union (IoU) measures the overlap between the predicted and ground-truth boxes. Precision, Recall, and AP follow the Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes (PASCAL VOC) evaluation standard [26], while mAP50–95 follows the Common Objects in Context (COCO) evaluation protocol [27]. The metrics are calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad (9)$$

$$\text{Recall} = \frac{TP}{TP + FN}, \quad (10)$$

$$\text{AP} = \int_0^1 P(R) dR, \quad (11)$$

$$\text{mAP50} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i^{0.50}, \quad (12)$$

$$\text{mAP50–95} = \frac{1}{10N} \sum_{t \in \{0.50, 0.55, \dots, 0.95\}} \sum_{i=1}^N \text{AP}_i^t. \quad (13)$$

According to Equation (9), Precision reflects the correctness of model predictions, with a higher value indicating fewer false detections. According to Equation (10), Recall reflects the ability of the model to detect real defects, with a higher value indicating fewer missed detections. In Equation (11), AP denotes the area under the Precision–Recall curve for a single class at different confidence thresholds, and N denotes the number of defect classes. Equation (12) gives mAP50 as the average AP across all defect categories when the IoU threshold is fixed at 0.50. Equation (13) further averages AP over IoU thresholds from 0.50 to 0.95 at intervals of 0.05, providing a stricter measure of localization quality. Computational complexity was measured in Giga Floating-Point Operations (GFLOPs).

3.2. Comparative Experimental Results

The proposed method was evaluated against several representative object detectors under a unified experimental setting with the same dataset split, input resolution, training epochs, and evaluation metrics. The baselines included YOLO26, YOLO11n, YOLOv8, Real-Time Detection Transformer (RT-DETR) [28], Task-Aligned One-Stage Object Detection

(TOOD) [29], Generalized Focal Loss (GFL) [30], and Deformable Detection Transformer (Deformable DETR) [31].

No pretrained weights were used for the models listed in Table 2. All models were trained from scratch on the book defect dataset using the same dataset split, 1280×1280 input size, batch size of 16,300 training epochs, AdamW optimizer, initial learning rate of 0.001, and hardware platform. Specifically, YOLOv8 used the YOLOv8n architecture; RT-DETR and Deformable DETR used Residual Network-50 (ResNet-50) backbones; and TOOD and GFL used ResNet-50 Feature Pyramid Network (FPN) architectures. YOLO11n retained its original backbone, whereas LME-YOLO introduced C3k2_LSNets and MEMFuseBlock within this framework. These settings define the common training and evaluation protocol used for the comparisons in Table 2.

Table 2 presents the comparison results on the book defect detection dataset. The proposed method achieved the highest mAP50 of 0.818, exceeding YOLO11n and YOLOv8 by 0.015 and 0.017, respectively. Its mAP50–95 reached 0.465, compared with 0.459 for YOLO11n and 0.460 for YOLOv8. The higher mAP50 and mAP50–95 values quantify the changes in defect recognition and localization performance under the unified evaluation protocol.

The proposed method contains 2.64 M parameters and requires 6.6 GFLOPs. Compared with YOLO11n, the parameter count and computational cost increased by 0.05 M and 0.2 GFLOPs, respectively. Compared with RT-DETR, TOOD, GFL, and Deformable DETR, LME-YOLO used fewer parameters and GFLOPs while achieving a higher mAP50. These measurements characterize its detection performance and computational requirements relative to the comparison models.

Table 2. Comparison results of different object detection methods on the book defect detection dataset.

Model	P	R	mAP50	mAP50–95	Params/M	GFLOPs
YOLO26	0.766	0.721	0.762	0.432	2.51	5.8
YOLO11n	0.801	0.755	0.803	0.459	2.59	6.4
YOLOv8	0.816	0.773	0.801	0.460	3.01	8.2
RT-DETR	0.752	0.713	0.739	0.433	42.80	130.5
TOOD	0.723	0.616	0.596	0.308	32.00	95.8
GFL	0.780	0.671	0.672	0.347	32.23	328.0
Deformable DETR	0.645	0.600	0.517	0.200	44.00	90.0
LME-YOLO (ours)	0.829	0.777	0.818	0.465	2.64	6.6

Figure 5 provides a radar-chart visualization of the six quantities reported in Table 2. Precision, Recall, mAP50, and mAP50–95 are plotted directly on their original 0–1 scales, for which a larger value indicates better detection performance. Because the parameter count and GFLOPs are lower-is-better quantities, they are converted to higher-is-better scores using $s_i = \min_j(x_j)/x_i$, where x_i is the parameter count or GFLOPs of model i . Consequently, a point closer to the outer boundary denotes either higher detection performance or lower model complexity.

3.3. Ablation Experimental Results

To analyze the contribution of each improved module to book defect detection, ablation experiments were conducted using YOLO11n as the baseline under the same data split, input size, training epochs, and evaluation metrics. The experiments sequentially introduced the C3k2_LSNets fine-grained feature enhancement module and the MEMFuseBlock edge enhancement module, and compared Precision, Recall, mAP50, mAP50–95, parameter count, and computational cost.

Table 3 shows the detection results under different module combinations. Compared with the YOLO11n baseline, introducing C3k2_LSNNet increased Precision from 0.801 to 0.804, Recall from 0.755 to 0.765, mAP50 from 0.803 to 0.805, and mAP50–95 from 0.459 to 0.462. The 0.010 increase in Recall indicates that more defect instances were detected. The mAP50–95 increase of 0.003 corresponds to an absolute increase of 0.3 percentage points and a relative increase of approximately 0.65%. These metric changes indicate a small improvement in detection coverage and multi-threshold localization performance after C3k2_LSNNet was introduced.

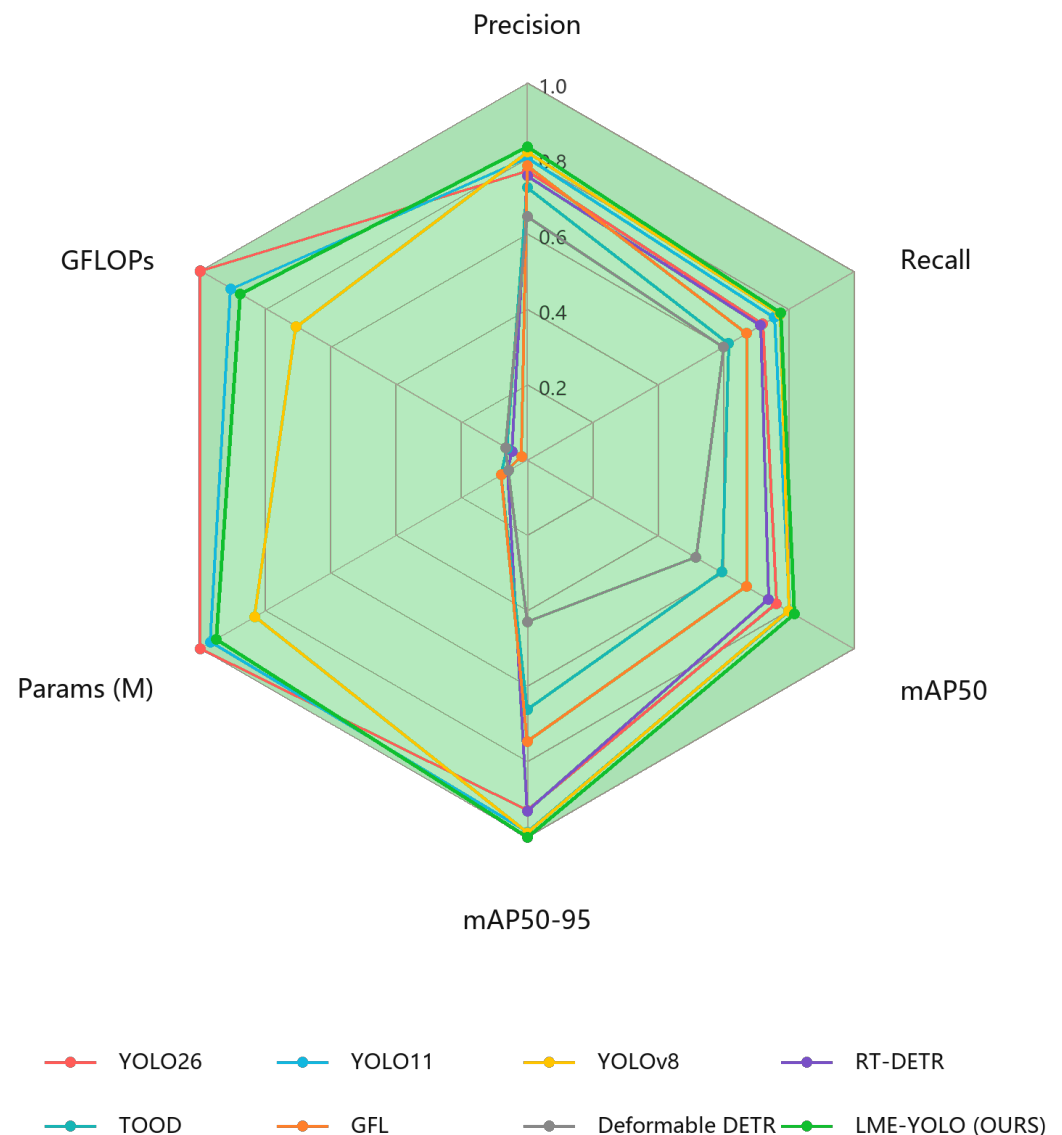


Figure 5. Radar-chart comparison derived from Table 2.

After MEMFuseBlock was added to YOLO11n+C3k2_LSNNet, Precision increased from 0.804 to 0.829, Recall from 0.765 to 0.777, mAP50 from 0.805 to 0.818, and mAP50–95 from 0.462 to 0.465, corresponding to absolute increases of 0.025, 0.012, 0.013, and 0.003, respectively. MEMFuseBlock was the only architectural change between these two settings; therefore, the metric changes quantify its contribution under the controlled ablation protocol. The increase in mAP50–95 suggests improved bounding-box localization performance across stricter IoU thresholds.

Compared with YOLO11n, the complete model increased Precision by 0.028 (3.50%), Recall by 0.022 (2.91%), mAP50 by 0.015 (1.87%), and mAP50–95 by 0.006 (1.31%), showing

consistent improvements across all four detection metrics. Adding MEMFuseBlock increased the parameter count from 2.54 M to 2.64 M and the computational cost from 6.4 to 6.6 GFLOPs, corresponding to increases of 3.94% and 3.13%, respectively. Relative to the YOLO11n baseline, the complete model added 0.05 M parameters (1.93%) and 0.2 GFLOPs (3.13%). Moreover, it used 12.29% fewer parameters and 19.51% fewer GFLOPs than YOLOv8. These results show that the complete model improves detection performance across multiple metrics while maintaining relatively low parameter and computational requirements.

Table 3. Ablation experiments.

Model	P	R	mAP50	mAP50–95	Params/M	GFLOPs
YOLO11n	0.801	0.755	0.803	0.459	2.59	6.4
YOLO11n+C3k2_LNet	0.804	0.765	0.805	0.462	2.54	6.4
YOLO11n+C3k2_LNet+MEMFuseBlock	0.829	0.777	0.818	0.465	2.64	6.6

Figure 6 shows the training loss and validation metrics of the YOLO11n baseline and the complete model. The curves were smoothed using the Exponential Moving Average (EMA). The two models exhibit similar convergence trajectories because they share the overall YOLO11n framework and training protocol. Both models were trained for 300 epochs under the same training schedule.

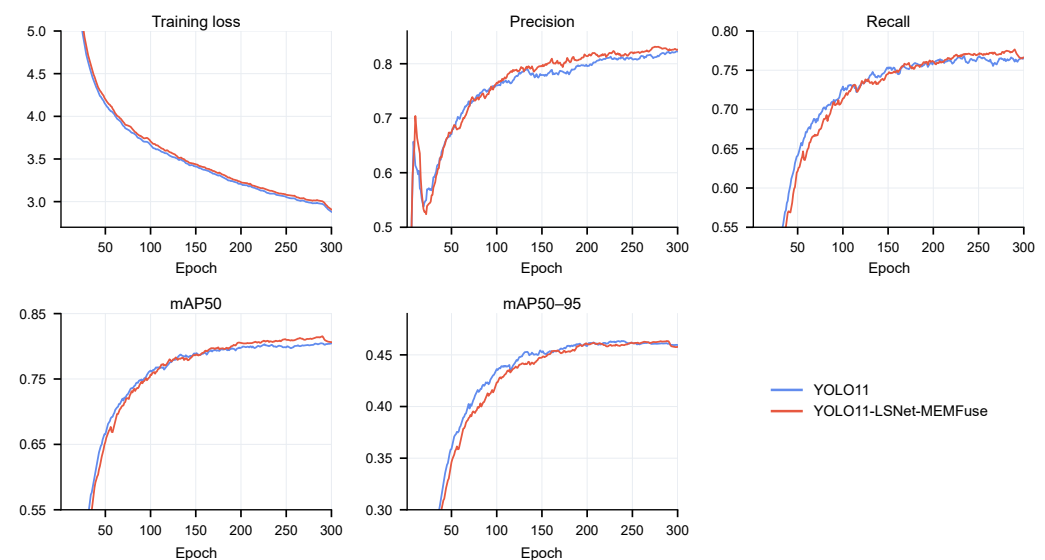


Figure 6. Training loss and validation metric curves of YOLO11n and the complete improved model (EMA).

In addition, to analyze the influence of bounding-box regression loss on detection results, Complete IoU (CIoU), Generalized IoU (GIoU), Scylla IoU (SIoU), Efficient IoU (EIoU), Wise IoU (WIoU), Alpha-IoU, Focal-EIoU, and Minimum Point Distance IoU (MPDIoU) were compared using the same network structure, data split, and training parameters [32–38]. The final loss function was selected by jointly considering Precision, Recall, mAP50, and mAP50–95. As shown in Table 4, CIoU ranked first in Precision and mAP50, with values of 0.829 and 0.818, respectively. Compared with MPDIoU, CIoU increased Precision by 0.022 and mAP50 by 0.011, while its Recall and mAP50–95 were lower by 0.005 and 0.002, respectively. Compared with GIoU, CIoU increased Precision by 0.027 and mAP50 by 0.016, while Recall decreased by 0.011 and mAP50–95 remained unchanged. CIoU therefore provided the highest Precision and mAP50 while maintaining Recall and mAP50–95 close to the highest observed values, and it was used in the final configuration.

Table 4. Comparison results of different loss functions on the book defect detection dataset.

Method	P	R	mAP50	mAP50–95
CIoU (ours)	0.829	0.777	0.818	0.465
GIoU	0.802	0.788	0.802	0.465
SIoU	0.814	0.768	0.802	0.461
EIoU	0.799	0.781	0.808	0.458
WIoU	0.821	0.778	0.810	0.465
AlphaIoU	0.819	0.759	0.807	0.466
FocalEIoU	0.810	0.773	0.798	0.462
MPDIoU	0.807	0.782	0.807	0.467

To further analyze the performance difference between CIoU and MPDIoU, the CIoU loss is defined as [33]

$$L_{\text{CIoU}} = 1 - \text{IoU} + \frac{\rho^2(b, b^{\text{gt}})}{c^2} + \alpha v, \quad (14)$$

In Equation (14), b and b^{gt} denote the centre points of the predicted and ground-truth boxes, respectively. The variable c is the diagonal length of their smallest enclosing box. The term v measures aspect-ratio consistency, and α is an adaptive weight, defined as

$$v = \frac{4}{\pi^2} \left(\arctan \frac{w^{\text{gt}}}{h^{\text{gt}}} - \arctan \frac{w}{h} \right)^2, \quad \alpha = \frac{v}{1 - \text{IoU} + v}. \quad (15)$$

According to Equation (15), CIoU jointly constrains box overlap, centre-point distance, and aspect-ratio consistency.

The MPDIoU loss is defined as [38]

$$L_{\text{MPDIoU}} = 1 - \text{IoU} + \frac{d_1^2}{W^2 + H^2} + \frac{d_2^2}{W^2 + H^2}, \quad (16)$$

In Equation (16), d_1 and d_2 denote the Euclidean distances between the corresponding upper-left and lower-right corners of the two boxes. The variables W and H are the input-image width and height. MPDIoU directly penalizes deviations between corresponding box corners, whereas CIoU jointly constrains overlap, centre-point distance, and aspect-ratio consistency.

Book defects such as Tears and Ink Stains often have irregular, blurred, or gradually changing boundaries, so their annotated box corners can vary with defect morphology and annotation extent. Consequently, MPDIoU may be more sensitive to corner placement, whereas CIoU combines overlap, centre-point distance, and aspect-ratio consistency. This geometric distinction may partly explain the different trade-offs between strict-IoU localization and overall detection metrics observed in Table 4.

3.4. Class-Wise Detection Results

To further analyze recognition differences across defect categories, Figure 7 presents the normalized confusion matrix on the test set. The columns correspond to the ground-truth classes, whereas the rows correspond to the predicted classes. The diagonal values for Ink Stain, Folded Corner, and Hole are 0.86, 0.88, and 0.86, respectively, indicating that the model recognizes these categories with relatively stable performance. The diagonal value for Tear is 0.74, lower than those of the other categories, showing that slender cracks and weak-boundary regions remain more difficult to identify.

The error distribution shows that 25% of the true Tear samples were predicted as background. During dataset construction and partitioning, the sample distribution of the four defect categories was checked and controlled to keep the categories balanced across the dataset subsets. Tear contains the largest number of samples among the four defect

categories. The misclassified Tear regions typically showed low contrast and textures, colors, and edge patterns similar to the normal paper background. Their irregular shapes, slender structures, and variable appearance further increased recognition difficulty. These observations associate the Tear-to-background errors with visual similarity and morphological variation. Ink Stain also showed some confusion with background, which may be related to the local similarity between paper stains, shadows, and printed textures. Overall, the confusion matrix is consistent with the quantitative metrics and visualization results, further indicating that LME-YOLO has stable recognition performance for the main defect categories, while slender cracks and weak-boundary targets still leave room for improvement.



Figure 7. Normalized confusion matrix of LME-YOLO on the book defect test set.

3.5. Visualization Analysis

Gradient-weighted Class Activation Mapping++ (Grad-CAM++) was used to visualize the image regions contributing to the predictions [39]. Figure 8 presents paired results for YOLO11n and LME-YOLO using identical test images and the same color mapping. Red and yellow indicate relatively high response intensity, whereas blue indicates relatively low response intensity. In the Ink Stain and Hole examples, the high-response regions of LME-YOLO overlap more closely with the annotated defect regions, while the baseline retains additional responses on surrounding page content. In the Folded Corner and Tear examples, the responses of LME-YOLO extend more continuously along the geometric deformation or damaged edges. These visual patterns are consistent with the increases in Recall and mAP50–95 reported in the ablation study. Grad-CAM++ is a qualitative

visualization method; accordingly, the spatial response patterns in Figure 8 are considered together with the detection metrics when interpreting model behaviour.

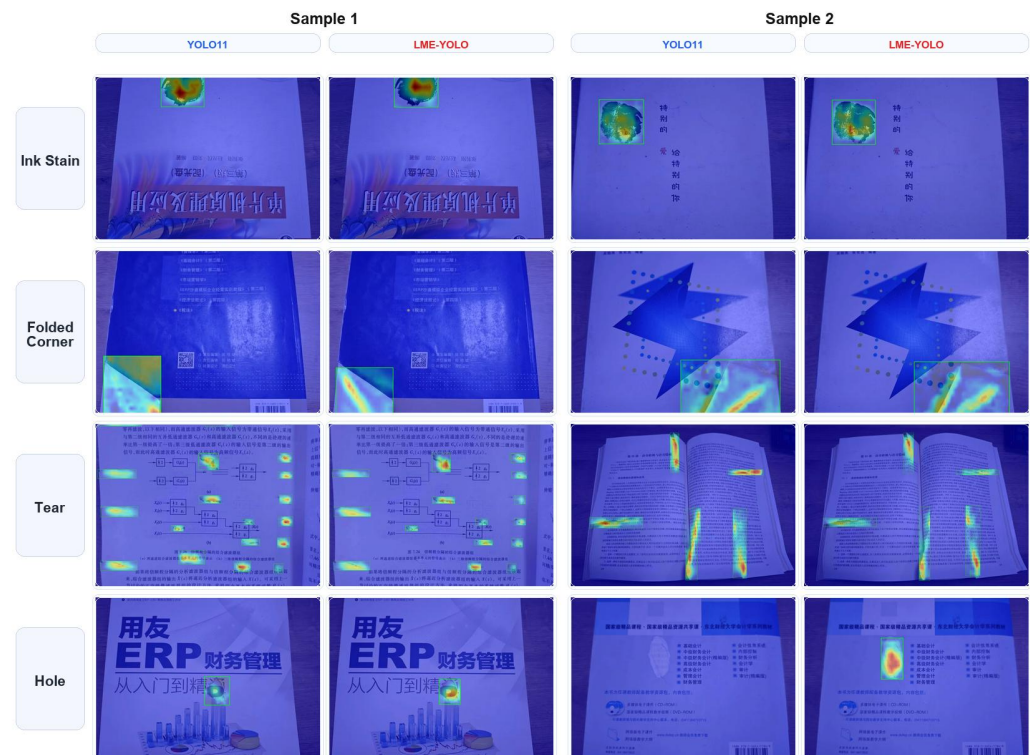


Figure 8. Grad-CAM++ visualization comparison between YOLO11n and LME-YOLO on different book defect samples.

4. Discussion

Controlled ablation and Grad-CAM++ visualization were used to examine the contributions of the proposed modules in relation to established detection methods. The results were obtained from a single data split without repeated-run uncertainty estimates; therefore, the reported metric differences describe performance under the present experimental setting.

This finding is consistent with previous work on multi-scale feature representation. FPN uses a top-down pathway and lateral connections to combine semantic information across feature levels when detecting small objects and targets at different scales [17]. EfficientDet further improved this idea by using a weighted bidirectional feature pyramid, which strengthens multi-scale fusion while maintaining computational efficiency [18]. LME-YOLO retains the feature-pyramid structure of YOLO11n and adds local aggregation and edge supplementation within the existing framework. This configuration targets feature enhancement at the neck fusion scales while preserving the compact backbone.

The NVIDIA GeForce RTX 5070 was used as the unified platform for training and experimental evaluation. Based on the model size and computational cost reported in Table 3, the estimated inference time on a GTX 1060-class GPU is approximately 20–40 ms per image, corresponding to about 25–50 frames per second (FPS). This estimate may vary depending on the hardware architecture, inference framework, input resolution, batch size, and deployment optimization. For deployment on lower-capacity devices, ONNX/TensorRT conversion, pruning, and quantization can be further adopted to improve inference efficiency.

The feature-selection mechanisms used in the proposed modules are consistent with established channel- and spatial-recalibration methods. CBAM, for example, uses channel and spatial attention to reweight intermediate features and emphasize discriminative responses [40]. In this study, controlled ablation quantifies changes in detection performance, whereas Grad-CAM++ characterizes the corresponding spatial response patterns. Together, these analyses relate the proposed feature-processing mechanisms to the measured detection results.

Industrial anomaly detection studies also suggest that anomalies in real quality-inspection scenarios rarely have a unified morphology. The MVTec AD dataset covers multiple object and texture categories and includes scratches, contamination, structural changes, and other anomaly types, reflecting the complex defect morphologies and large sample differences in practical visual inspection tasks [41]. Book defect detection faces a similar problem. Ink Stain is closer to local texture contamination, Tear and Hole are closer to structural damage, and Folded Corner involves both geometric deformation and illumination variation. These morphological differences motivate the combined modelling of local texture, contextual information, and damaged edges. The present aggregate ablation results characterize the overall module contributions across the four defect categories.

The complete model achieved a 0.006 increase in mAP_{50–95} over YOLO11n, indicating that localization at stricter IoU thresholds remains a challenge. Tear, Hole, and Folded Corner often have irregular boundaries that are not fully represented by rectangular bounding boxes. Ink Stain may also resemble paper texture, printed patterns, or light reflections. Consequently, high-precision localization remains constrained by annotation granularity and defect morphology.

This study still has several limitations. First, the current dataset remains limited in scale, and both the sample size and the range of scenes can be further expanded. Different paper materials, printing styles, imaging devices, and lighting conditions may affect model generalization. In addition, this study mainly focuses on visible page appearance defects. More complex publication quality problems, such as binding quality, cover lamination, and spine damage, were not systematically studied.

5. Conclusions

The continuing development of China's publishing "going global" strategy has placed higher requirements on the physical quality of books, while traditional manual inspection is increasingly unable to meet the demands of large-scale production and international circulation. To support automated book appearance quality inspection, this study constructed a dedicated book defect detection dataset and developed LME-YOLO based on the YOLO11n framework for four defect categories: Ink Stain, Folded Corner, Tear, and Hole. The model incorporates C3k2_LSNNet and MEMFuseBlock to enhance the representation of local details, boundary information, and contextual features. On the dataset containing 632 images, LME-YOLO achieved an mAP₅₀ of 0.818 and an mAP_{50–95} of 0.465, with 2.64 M parameters and 6.6 GFLOPs.

Future work will expand the dataset by including additional book types, paper materials, imaging devices, lighting conditions, and defect categories, and will use external datasets to evaluate generalization to unseen books and acquisition environments. The model will also be evaluated on lower-capacity GPUs and embedded devices by reporting FPS, inference latency, memory consumption, and power consumption. In addition, contour-based or segmentation annotations and boundary-aware optimization will be investigated to improve the localization of Tear, Hole, and other defects with irregular or weak boundaries.

Author Contributions: Conceptualization, W.L.; methodology, W.L. and Q.X.; software, W.H.; formal analysis, W.L.; investigation, L.L.; resources, Q.X.; writing—original draft, W.L.; writing—review and editing, W.L., W.H., Q.X., T.W., X.W., L.L. and L.J.; visualization, L.L. and W.H.; supervision, T.W., Q.X., X.W. and L.J.; project administration, Q.X.; funding acquisition, Q.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Education Humanities and Social Sciences Project (Grant No. 21YJCZH191).

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Acknowledgments: The authors gratefully acknowledge the individuals who contributed to book image collection, defect annotation, and data verification for this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Lu, X. Research on the “Going Global” Strategy of China’s Publishing Industry. Ph.D. Thesis, Wuhan University, Wuhan, China, 2010.
2. Hao, T.; Qian, P. Regional publishing status and development thoughts of Chinese theme books “going global”: A case study of local publishing houses in Shaanxi Province. *Publ. Print.* **2023**, *4*, 17–27. [\[CrossRef\]](#)
3. Czimmermann, T.; Ciuti, G.; Milazzo, M.; Chiurazzi, M.; Roccella, S.; Oddo, C.M.; Dario, P. Visual-based defect detection and classification approaches for industrial applications—A survey. *Sensors* **2020**, *20*, 1459. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Hütten, N.; Alves Gomes, M.; Hölken, F.; Andricevic, K.; Meyes, R.; Meisen, T. Deep learning for automated visual inspection in manufacturing and maintenance: A survey of open-access papers. *Appl. Syst. Innov.* **2024**, *7*, 11. [\[CrossRef\]](#)
5. Ren, Z.; Fang, F.; Yan, N.; Wu, Y. State of the art in defect detection based on machine vision. *Int. J. Precis. Eng. Manuf.-Green Technol.* **2022**, *9*, 661–691. [\[CrossRef\]](#)
6. Zhang, P.; Wang, H.; Cheng, W.; Li, J. Accurate edge localization of complex workpiece based on coarse-to-fine principle. In Proceedings of the 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Beijing, China, 13–15 October 2018; pp. 1–8. [\[CrossRef\]](#)
7. Wang, H.; Sun, S.; Kan, T.; Sun, G. Damage detection system of ancient paper books based on machine vision. In Proceedings of the 2022 2nd International Conference on Algorithms, High Performance Computing and Artificial Intelligence (AHPCAI), Guangzhou, China, 21–23 October 2022; pp. 412–415. [\[CrossRef\]](#)
8. Hong, S.; Jeong, J.-W. Automatic examination of condition of used books with YOLO-based object detection framework. *Comput. Syst. Sci. Eng.* **2023**, *47*, 1611–1632. [\[CrossRef\]](#)
9. Tomas, J.P.; Albano, K.A.; Alday, J.S.; Corbe, M.R.; Pangan, G.N. Using YOLOv8 book defect detection model with MIRNet LLI. In Proceedings of the 14th International Workshop on Computer Science and Engineering (WCSE 2024), Phuket, Thailand, 19–21 June 2024; pp. 27–35. [\[CrossRef\]](#)
10. Bhatt, P.M.; Malhan, R.K.; Rajendran, P.; Shah, B.C.; Thakar, S.; Yoon, Y.J.; Gupta, S.K. Image-based surface defect detection using deep learning: A review. *J. Comput. Inf. Sci. Eng.* **2021**, *21*, 040801. [\[CrossRef\]](#)
11. Liu, Y.; Zhang, C.; Dong, X. A survey of real-time surface defect inspection methods based on deep learning. *Artif. Intell. Rev.* **2023**, *56*, 12131–12170. [\[CrossRef\]](#)
12. Peng, G.; Song, T.; Cao, S.; Zhou, B.; Jiang, Q. A two-stage defect detection method for unevenly illuminated self-adhesive printed materials. *Sci. Rep.* **2024**, *14*, 20547. [\[CrossRef\]](#) [\[PubMed\]](#)
13. Liu, J.; Cai, Z.; He, K.; Huang, C.; Lin, X.; Liu, Z.; Li, Z.; Chen, M. An efficient printing defect detection based on YOLOv5-DCN-LSK. *Sensors* **2024**, *24*, 7429. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Khanam, R.; Hussain, M. YOLOv11: An overview of the key architectural enhancements. *arXiv* **2024**, arXiv:2410.17725. [\[CrossRef\]](#)
15. Wang, A.; Chen, H.; Lin, Z.; Han, J.; Ding, G. LSNet: See large, focus small. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 11–15 June 2025; pp. 9718–9729. [\[CrossRef\]](#)
16. Cheng, J.; Wu, Y.; Zhou, Y. MEMO: Human-like crisp edge detection using masked edge prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Denver, CO, USA, 3–7 June 2026; pp. 27740–27749. [\[CrossRef\]](#)
17. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125. [\[CrossRef\]](#)

18. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual Conference, 14–19 June 2020; pp. 10781–10790. [\[CrossRef\]](#)
19. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 8759–8768. [\[CrossRef\]](#)
20. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861. [\[CrossRef\]](#)
21. Ding, X.; Zhang, X.; Han, J.; Ding, G. Scaling up your kernels to 31×31 : Revisiting large kernel design in CNNs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–24 June 2022; pp. 11963–11975. [\[CrossRef\]](#)
22. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141. [\[CrossRef\]](#)
23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 5998–6008.
24. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807. [\[CrossRef\]](#)
25. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019. [\[CrossRef\]](#)
26. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [\[CrossRef\]](#)
27. Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014; pp. 740–755. [\[CrossRef\]](#)
28. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs beat YOLOs on real-time object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024; pp. 16965–16974. [\[CrossRef\]](#)
29. Feng, C.; Zhong, Y.; Gao, Y.; Scott, M.R.; Huang, W. TOOD: Task-aligned one-stage object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual Conference, 10–17 October 2021; pp. 3490–3499. [\[CrossRef\]](#)
30. Li, X.; Wang, W.; Wu, L.; Chen, S.; Hu, X.; Li, J.; Tang, J.; Yang, J. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. In Proceedings of the Advances in Neural Information Processing Systems, Virtual Conference, 6–12 December 2020; Volume 33, pp. 21002–21012.
31. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable transformers for end-to-end object detection. In Proceedings of the International Conference on Learning Representations (ICLR), Virtual Conference, 3–7 May 2021. [\[CrossRef\]](#)
32. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 658–666. [\[CrossRef\]](#)
33. Zheng, Z.; Wang, P.; Ren, D.; Liu, W.; Ye, R.; Hu, Q.; Zuo, W. Enhancing geometric factors in model learning and inference for object detection and instance segmentation. *IEEE Trans. Cybern.* **2022**, *52*, 8574–8586. [\[CrossRef\]](#) [\[PubMed\]](#)
34. Gevorgyan, Z. SIoU loss: More powerful learning for bounding box regression. *arXiv* **2022**, arXiv:2205.12740. [\[CrossRef\]](#)
35. Zhang, Y.-F.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* **2022**, *506*, 146–157. [\[CrossRef\]](#)
36. Tong, Z.; Chen, Y.; Xu, Z.; Yu, R. Wise-IoU: Bounding box regression loss with dynamic focusing mechanism. *arXiv* **2023**, arXiv:2301.10051. [\[CrossRef\]](#)
37. He, J.; Erfani, S.M.; Ma, X.; Bailey, J.; Chi, Y.; Hua, X.-S. Alpha-IoU: A family of power intersection over union losses for bounding box regression. In Proceedings of the Advances in Neural Information Processing Systems, Virtual Conference, 6–14 December 2021; Volume 34, pp. 20230–20242.
38. Ma, S.; Xu, Y. MPDIoU: A loss for efficient and accurate bounding box regression. *arXiv* **2023**, arXiv:2307.07662. [\[CrossRef\]](#)
39. Chattopadhyay, A.; Sarkar, A.; Howlader, P.; Balasubramanian, V.N. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 839–847. [\[CrossRef\]](#)

40. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBAM: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19. [[CrossRef](#)]
41. Bergmann, P.; Fauser, M.; Sattlegger, D.; Steger, C. MVTec AD: A comprehensive real-world dataset for unsupervised anomaly detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 9592–9600. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.