

A Lightweight Forest Fire Detection Algorithm Based on an Improved YOLOv11

Jingzhou Tao^{1*}, Zijing Liu², Ruoxi Lyu³, Jie Li⁴, and Xu Cao⁵

Khoury College of Computer Sciences, Northeastern University, Seattle, WA, USA^{1,2,3}

Université de Montréal, Canada⁴

University of Chinese Academy of Sciences, China⁵

Abstract—This study presents IWL-YOLOv11 (Improved Wildfire Lightweight YOLOv11), a detector that balances forest fire detection accuracy and parameter count. The model introduces three components. A Dual-Branch Gated Depthwise Convolution Block (DGCBlock) modifies the C3K2 backbone through depthwise separable convolutions and element-wise interaction between two feature branches. A Hybrid Convolution-Attention Network (HCAN) combines features at multiple scales with self-attention at the junction between the backbone and neck. The regression objective uses WDIoU, a combination of Wise Intersection-over-Union loss and Distribution Focal Loss. Experiments use a modified Flame dataset containing 8,625 images and a separately constructed ForestFire dataset containing 2,096 images. IWL-YOLOv11 achieves mAP@0.50 of 88.9 percent and 90.2 percent, respectively, exceeding the YOLOv11 baseline by 9.6 and 6.0 percentage points. Its mAP averaged over IoU thresholds from 0.50 to 0.95 is 62.4 percent on Flame and 78.2 percent on ForestFire. The model has 2,738,750 parameters, approximately 71 percent fewer than the baseline. Cumulative ablations show progressively higher detection metrics across the tested configurations. The reported results indicate a favorable accuracy-parameter trade-off on these image splits. Inference latency, edge-device performance, and uncertainty across repeated training runs are not reported, so the parameter reduction alone does not establish real-time deployment performance.

Keywords—Object detection; YOLOv11; dual-branch gated depthwise separable convolution; hybrid convolution-attention network; forest fire detection

I. INTRODUCTION

Forest fires threaten ecosystems, economies, and human life and property [1]. Timely detection is therefore an important part of disaster management and ecological protection [2]. Conventional monitoring relies on satellite imagery, ground sensors, and manual patrols [3]. These approaches face limitations in response time, coverage, and sensitivity to environmental conditions.

Deep learning provides another approach to recognizing and localizing fire and smoke in remote-sensing images, unmanned aerial vehicle (UAV) videos, and ground-camera footage [4]. Although these methods can improve detection under different environmental conditions [5], large models and complex backgrounds remain challenges. This study addresses the balance between detection accuracy and model size.

A. Related Work

Recent fire detectors combine compact convolutions, attention, feature fusion, and regression losses. FFYOLO [6] modifies YOLOv8 with channel prior dilatation attention, a mixed-classification detection head, GSConv, MPDIoU, and knowledge distillation. It reports mAP@0.50 of 88.8 percent on its custom dataset, with 25.3 percent fewer parameters and 9.3 percent higher frames per second than its baseline. These measurements illustrate why both detection accuracy and runtime matter when evaluating compact detectors.

Lei et al. [7] reduce YOLOv8 complexity using Ghost convolution and depthwise separable convolution, then incorporate dynamic upsampling, coordinate attention, and DIoU. Their study reports reductions in both parameters and FLOPs. DSS-YOLO [8] instead combines DynamicConv, SEAM attention, and SPPELAN in YOLOv8n to address small or obscured fire targets. These studies demonstrate that efficient convolutions and attention are established strategies for fire detection.

IWL-YOLOv11 evaluates a different combination: multiplicative DGCBlock branch interactions, pooled convolutional features followed by HCAN attention, and a WIoU-DFL regression objective in a YOLOv11 configuration. The contribution lies in this combined design and its reported accuracy-parameter trade-off. The cited fire-specific models were not rerun on our image splits, so their reported metrics cannot establish a direct performance ranking against IWL-YOLOv11. Dataset composition, model scale, and evaluation settings must be aligned for that comparison.

Object detectors are commonly divided into two-stage and one-stage methods. R-CNN [9], Fast R-CNN [10], Faster R-CNN [11], and Mask R-CNN [12] illustrate the two-stage family. R-CNN applies a feature extractor to candidate regions, commonly generated by Selective Search [13], before classification and bounding-box regression. Non-maximum suppression removes redundant overlapping detections [14]. Processing candidate regions separately can increase computational cost.

One-stage methods, including the YOLO series [15-25] and SSD [26], predict categories and bounding boxes directly from image features. Their detection pipeline motivates their use as a starting point for compact fire and smoke detectors. The computational cost of a specific model nevertheless depends on its architecture, input resolution, and implementation.

*Corresponding author.

B. Scope and Contributions

We investigate whether coordinated changes to convolutional feature extraction, feature fusion, and box regression can improve the accuracy-parameter trade-off of the tested YOLOv11 configuration. The task is image-based localization of fire and smoke. Detection could contribute to camera or UAV monitoring, but this study evaluates image detection rather than a complete early-warning system. The proposed model introduces three changes:

- DGCB uses depthwise separable convolution to separate spatial filtering from channel mixing and combines two feature branches through element-wise multiplication. The compact backbone is evaluated through its parameter count and detection metrics.
- HCAN combines feature information from multiple scales with self-attention at the junction between the backbone and neck. It is designed to improve feature representation for objects of different sizes.
- WDIOU combines existing WIoU and DFL terms in the regression objective. WIoU adjusts sample contributions through a dynamic, non-monotonic focusing mechanism; DFL represents box-side distances as discrete distributions.

Section II describes the architecture and loss functions. Section III presents the datasets, hyperparameter comparisons, cumulative ablation study, and quantitative and qualitative

results. Section IV discusses the conclusions, study limitations, and future validation.

II. THE PROPOSED METHOD

IWL-YOLOv11 consists of a backbone, a neck, and a detection head. Fig. 1 shows the overall architecture.

1) *Backbone*: The backbone extracts features through C3K2_DGCB, Spatial Pyramid Pooling-Fast (SPPF), and HCAN. Within C3K2_DGCB, depthwise separable convolutions reduce the cost of spatial filtering and channel mixing. DGCB combines two feature branches with Tanh activations to provide nonlinear feature interactions. HCAN processes features at multiple scales to combine local detail with broader context.

2) *Neck*: The neck fuses features from the backbone before detection. It follows a Path Aggregation Network-Feature Pyramid Network (PAN-FPN) structure, combining top-down and bottom-up propagation to integrate feature maps at different resolutions.

3) *Head*: The detection head predicts classes and bounding boxes. Its training objective contains classification and regression terms. The regression branch retains Distribution Focal Loss (DFL) [27], which represents box coordinates as probability distributions. WDIOU combines DFL with Wise Intersection-over-Union (WIoU) loss [28] to improve bounding-box regression.

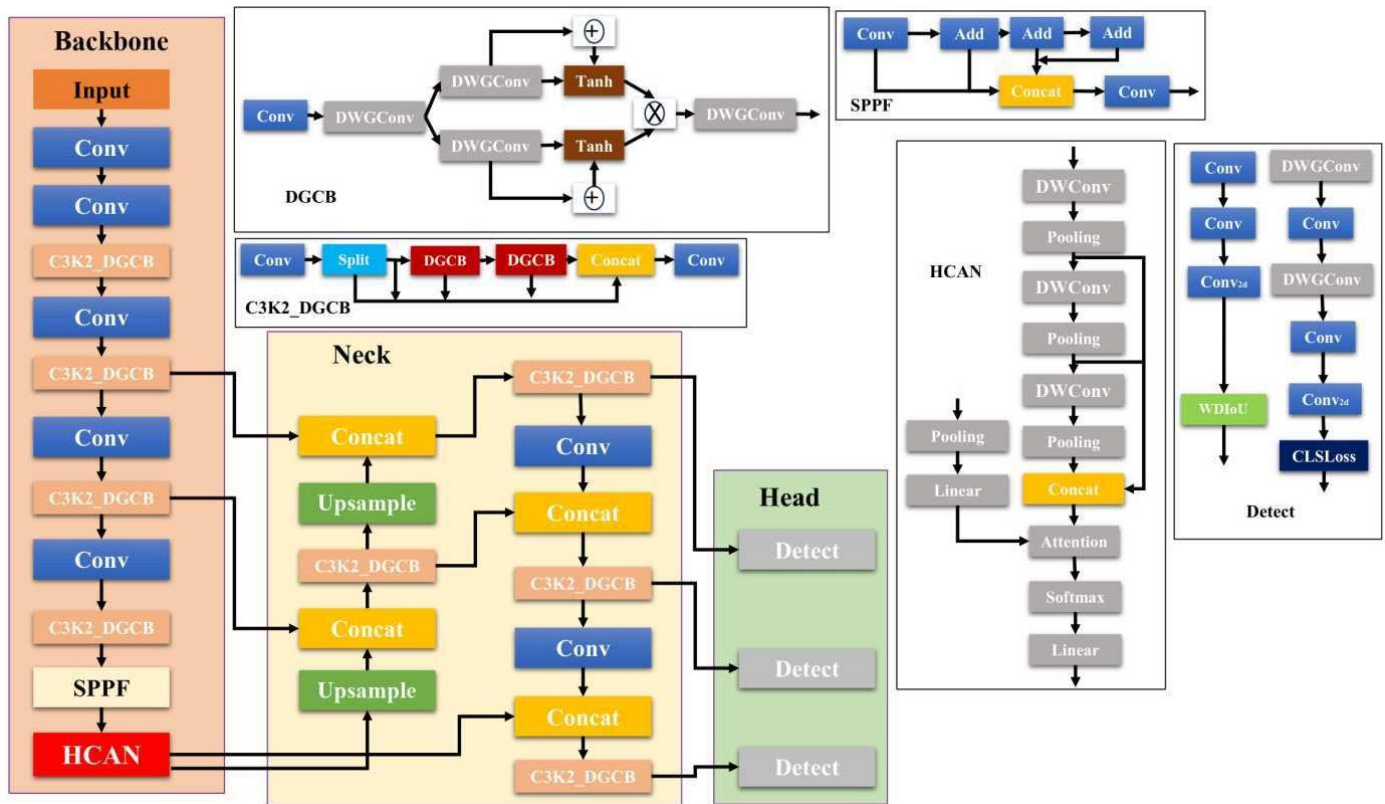


Fig. 1. Overall architecture of the proposed IWL-YOLOv11 model.

B. Dual-Branch Gated Depthwise Convolution Block (DGCB)

The C3k block used within the C3k2 structure of YOLOv11 [25] applies standard convolutions to extract

features. Fig. 2 illustrates the C3K and C3K2 structures used in this manuscript. Convolution and concatenation contribute to the parameter count and processing cost. We introduce DGCB to reduce this cost and add feature gating.

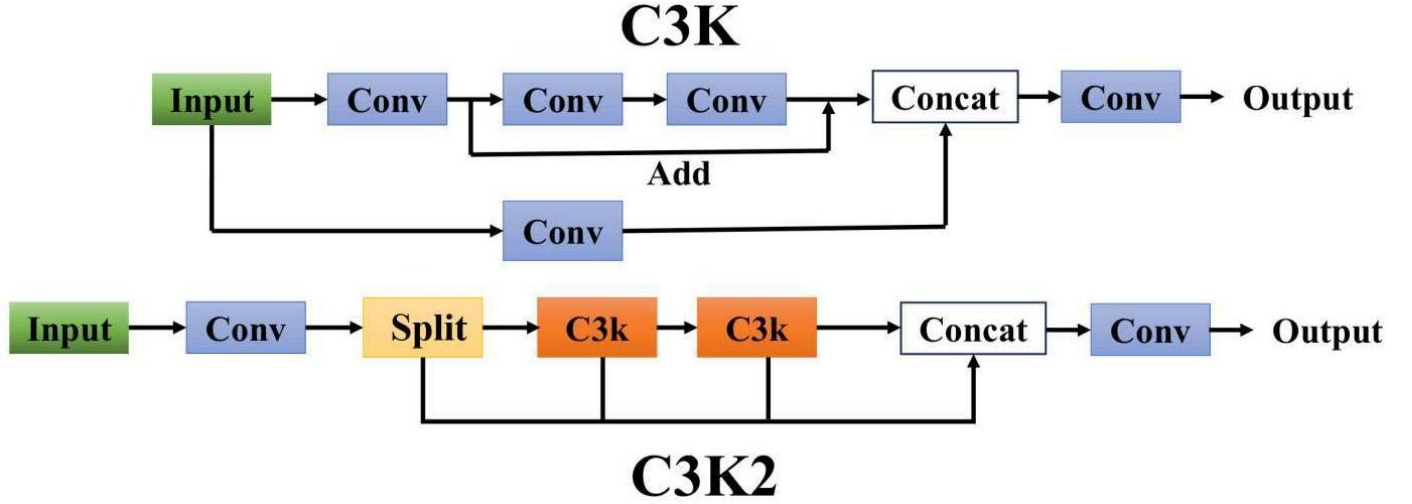


Fig. 2. Structure of the C3K and C3K2 modules.

DGCB separates spatial filtering from channel mixing through depthwise separable convolution. Its two branches interact through element-wise multiplication, providing a multiplicative gating operation. Fig. 3 shows the block. Its operations are defined below.

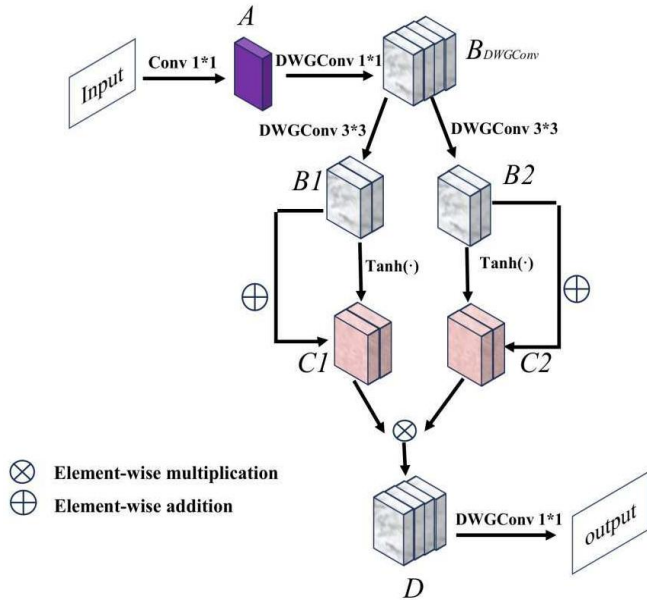


Fig. 3. Structure of the Dual-Branch Gated Depthwise Convolution Block (DGCB).

1) A 1×1 convolution transforms the input feature into A , as shown in Eq. (1):

$$A = \text{Conv}_{1 \times 1}(\text{Input}) \quad (1)$$

where, $\text{Conv}_{1 \times 1}$ denotes a 1×1 convolution and Input is the input feature.

2) Depthwise separable convolution transforms A into B_{DWGConv} . The result is split into two branches, which are processed to obtain B_1 and B_2 , as shown in Eq. (2) and (3):

$$B_{\text{DWGConv}} = \text{DWGConv}_{1 \times 1}(A) \quad (2)$$

$$B_1, B_2 = \text{DWGConv}_{3 \times 3}(\text{Split}(B_{\text{DWGConv}}, \text{dim}=1)) \quad (3)$$

where, DWGConv denotes depthwise separable convolution. The subscripts specify 1×1 in Eq. (2) and 3×3 in Eq. (3). The latter uses padding 1 and stride 1. Split partitions the tensor along the dimension specified by dim.

3) Each branch adds its Tanh-activated output to its original feature. The resulting features are C_1 and C_2 , as shown in Eq. (4):

$$\begin{aligned} C_1 &= B_1 \oplus \text{Tanh}(B_1), \\ C_2 &= B_2 \oplus \text{Tanh}(B_2) \end{aligned} \quad (4)$$

where, $\oplus$ denotes element-wise addition and Tanh is the activation function.

4) Element-wise multiplication combines C_1 and C_2 into D . A final convolution produces the block output, as shown in Eq. (5a) and (5b):

$$D = C_1 \otimes C_2 \quad (5a)$$

$$\text{output} = \text{DWGConv}_{1 \times 1}(D) \quad (5b)$$

where, DWGConv denotes depthwise separable convolution with a 1×1 kernel, and $\otimes$ denotes element-wise multiplication.

C. Hybrid Convolution-Attention Network (HCAN)

The baseline backbone ends with C2PSA, which combines convolution and attention. Fig. 4 shows the structure

represented in this manuscript. We replace this component with HCAN to incorporate pooling and convolution at multiple scales before attention. This design aims to improve feature fusion for forest fire images containing objects of different sizes.

HCAN combines convolutional feature extraction with self-attention. Pooling and depthwise convolution produce features at several scales, while attention integrates information across those features. Fig. 5 shows the query, key/value, attention, and output paths.

1) *Query path*: Adaptive pooling and normalization transform the input tensor X into Q . The query path then produces q_{out} , which is reshaped and permuted to obtain Q_{out} for multi-head attention, as shown in Eq. (6)-(8):

$$Q = \text{LayerNorm}(\text{AdaptiveAvgPool2d}(X)) \quad (6)$$

$$q_{out} = \text{LayerNorm}(Q) \quad (7)$$

$$Q_{out} = q_{out}.\text{reshape}(B, N_q, h, C/h).\text{permute}() \quad (8)$$

where, B is the batch size, C is the channel count, and H and W are the input feature-map dimensions. AdaptiveAvgPool2d performs adaptive average pooling, and LayerNorm performs normalization. reshape changes the tensor shape, while permute reorders its axes. N_q is the number of query elements, h is the number of attention heads, and C/h is the dimension of each head.

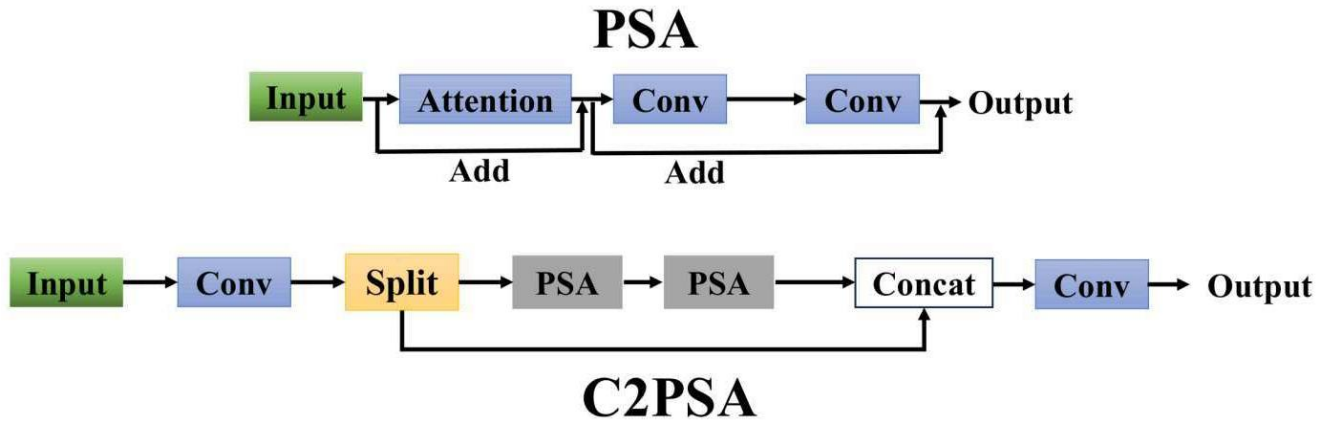


Fig. 4. Structure of the PSA and C2PSA modules.

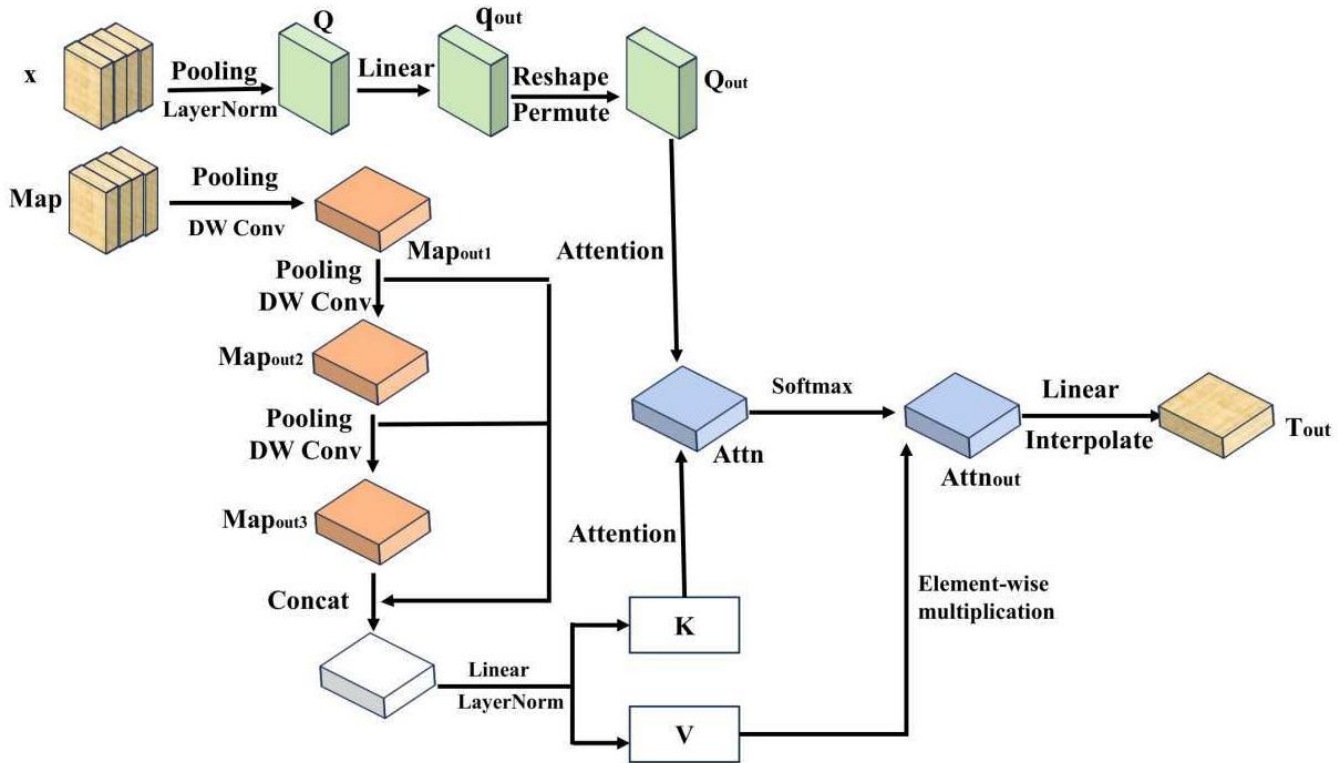


Fig. 5. Structure of the Hybrid Convolution-Attention Network (HCAN).

2) *Key/value path*: The input feature Map undergoes pyramid pooling at 11×11 , 8×8 , and 4×4 scales. A 3×3 depthwise convolution processes each pooled feature to produce Map_{out1} , Map_{out2} , and Map_{out3} . The features are concatenated, normalized, and projected to obtain the key k and value v , as shown in Eq. (9) and (10):

$$\begin{aligned} & \text{Map}_{\text{out1}}, \text{Map}_{\text{out2}}, \text{Map}_{\text{out3}} \\ & = \text{DWConv}(\text{AdaptiveAvgPool2d} \\ & \quad (\text{Map}, \text{pooled_size})) \end{aligned} \quad (9)$$

$$\begin{aligned} & k, v = \text{Linear}(\text{LayerNorm} \\ & \quad (\text{Concat}(\text{Map}_{\text{out1}}, \\ & \quad \text{Map}_{\text{out2}}, \text{Map}_{\text{out3}}))) \end{aligned} \quad (10)$$

where, *pooled_size* specifies the pooling size, *Linear* is a linear projection, *LayerNorm* is normalization, and *Concat* is concatenation.

3) *Attention*: The query Q_{out} and key k produce attention scores *attn*. *Softmax* normalizes the scores, and the resulting weights are applied to v to obtain Attn_{out} , as shown in Eq. (11) and (12):

$$\text{attn} = \frac{Q_{\text{out}} \otimes k}{\sqrt{d_k}} \quad (11)$$

$$\text{Attn}_{\text{out}} = \text{Softmax}(\text{attn}) \otimes v \quad (12)$$

where, Q_{out} , k , and v denote the query, key, and value tensors. d_k is the dimension of each attention head, and *Softmax* normalizes the attention scores.

4) *Output path*: A linear projection transforms Attn_{out} into T . Bilinear interpolation restores the original spatial resolution, as shown in Eq. (13) and (14):

$$T = \text{Linear}(\text{Attn}_{\text{out}}) \quad (13)$$

$$\begin{aligned} & T_{\text{out}} = \text{Interpolate}(T, \\ & \quad \text{size}=(H, W), \text{mode}=\text{bilinear}) \end{aligned} \quad (14)$$

where, *Linear* is a linear projection, and *Interpolate* resizes the feature map using bilinear interpolation.

D. WDIoU Loss Function

The training objective combines bounding-box regression and classification losses. Eq. (15) gives the total loss:

$$\text{Loss} = \text{Loss}_{\text{bbox}} + \text{Loss}_{\text{cls}} \quad (15)$$

where, *Loss* is the total loss, $\text{Loss}_{\text{bbox}}$ is the regression loss, and Loss_{cls} is the classification loss.

The classification term uses binary cross-entropy (BCE), as shown in Eq. (16):

$$\begin{aligned} \text{Loss}_{\text{BCE}} = & -\frac{1}{N} \sum_{i=1}^N [y_i \log(p_i) \\ & + (1-y_i) \log(1-p_i)] \end{aligned} \quad (16)$$

where, N is the number of samples, y_i is the target label for sample i , and p_i is its predicted probability.

The baseline regression objective combines Complete Intersection-over-Union (CIoU) loss [29] with DFL. CIoU

accounts for overlap, center distance, and aspect-ratio discrepancy. It is expressed in Eq. (17):

$$\text{Loss}_{\text{CIoU}} = L_{\text{IoU}} + \frac{\rho^2(b, b^{\text{gt}})}{c^2} + \alpha u \quad (17)$$

where, $L_{\text{IoU}} = 1 - \text{IoU}$ is the IoU loss, u measures aspect-ratio discrepancy, and α weights that term. The symbols b and b^{gt} denote the predicted and ground-truth box centers, ρ is their Euclidean distance, and c is the diagonal length of the smallest enclosing box.

DFL represents each box-side distance as a distribution over discrete bins [27]. For a continuous target y between adjacent unit-spaced bins y_i and y_{i+1} , the loss encourages the network to allocate probability to those two bins. Eq. (18) uses the predicted probabilities S_i and S_{i+1} . Eq. (19) and (20) give the corresponding optimal target probabilities.

$$\begin{aligned} \text{DFL}(S_i, S_{i+1}) = & -((y_{i+1} - y) \log(S_i) \\ & + (y - y_i) \log(S_{i+1})) \end{aligned} \quad (18)$$

$$S_i^* = \frac{y_{i+1} - y}{y_{i+1} - y_i} \quad (19)$$

$$S_{i+1}^* = \frac{y - y_i}{y_{i+1} - y_i} \quad (20)$$

Here, $y_i \leq y \leq y_{i+1}$ and $y_{i+1} - y_i = 1$. The quantities S_i and S_{i+1} in Eq. (18) are network predictions; S_i^* and S_{i+1}^* in Eq. (19) and (20) are the target probabilities at the minimum of the loss.

WDIoU combines WIoU with DFL. The WIoU term follows the version 3 formulation with a dynamic, non-monotonic focusing coefficient [28]. This coefficient adjusts gradient contributions according to sample quality, reducing the influence of very easy and low-quality examples while emphasizing examples of intermediate quality. Eq. (21) and (22) give the WIoU term and its focusing coefficient.

$$\begin{aligned} \text{Loss}_{\text{WIoU}} = & \exp\left(\frac{(x - x_{\text{gt}})^2 + (y - y_{\text{gt}})^2}{(W_g^2 + H_g^2)^*}\right) \\ & \times \text{Loss}_{\text{IoU}} \times r \end{aligned} \quad (21)$$

$$r = \frac{\beta}{\alpha^{\beta - \varphi} \varphi} \quad (22)$$

where, (x, y) and $(x_{\text{gt}}, y_{\text{gt}})$ are the predicted and ground-truth centers. W_g and H_g are the dimensions of the smallest enclosing box. Here, $\text{Loss}_{\text{IoU}} = L_{\text{IoU}} = 1 - \text{IoU}$. The superscript $*$ denotes detachment from the computation graph. The coefficient r is controlled by the outlier degree β and hyperparameters α and φ . In the WIoU formulation, $\beta = L_{\text{IoU}}^* / \bar{L}_{\text{IoU}}$, where the denominator is the running mean IoU loss. The symbol φ corresponds to the focusing parameter called delta in [28]. The α in Eq. (22) is a focusing hyperparameter, distinct from the CIoU weight in Eq. (17).

III. EXPERIMENTS AND RESULTS

A. Experimental Platform and Datasets

The Flame dataset used in this study combines forest fire images from the FLAME dataset [30] (<https://iee-dataport.org/open-access/flame-dataset-aerial-imagery-pile-burn-detection-using-drones-uavs>) with images collected from

news websites and social media, including Twitter and Instagram. It contains 8,625 images resized to 640×640. The images were manually annotated in Labelling with two categories: fire and smoke. An 8:1:1 split produced 6,900 training images, 862 validation images, and 863 test images, as listed in Table I.

TABLE I. COMPOSITION OF THE FLAME DATASET

Split	Number of images	Proportion
Training set	6,900	80%
Validation set	862	10%
Test set	863	10%
Total	8,625	100%

ForestFire is a separate dataset constructed for evaluation against complex forest backgrounds. It includes images captured during ignition experiments, fire images collected from news websites and social media, and forest-background images used as negative examples. The training set contains 968 background images and 946 complex forest fire images; the test set contains 80 background images and 102 complex forest fire images. Table II summarizes the distribution. Training and test images do not overlap in either dataset.

TABLE II. COMPOSITION OF THE FORESTFIRE DATASET

Split	Forest-background images	Complex forest fire images	Total
Training set	968	946	1,914
Test set	80	102	182
Total	1,048	1,048	2,096

B. Results and Analysis

Experiments used 14 vCPUs on an Intel® Xeon® Gold 6330 processor at 2.00 GHz, an NVIDIA GeForce RTX 3090 GPU with 24 GB of memory, and 80 GB of system memory. The software environment used Ubuntu and PyCharm 2020.1 Professional Edition.

1) *Evaluation metrics*: The reported metrics are AP (average precision), AR (average recall), mAP@0.50, and mAP@[0.50:0.05:0.95]. AP summarizes a precision-recall curve, while AR summarizes recall under the evaluation configuration. At a fixed IoU threshold, mAP averages AP over categories. mAP@0.50 uses a threshold of 0.50. mAP@[0.50:0.05:0.95] additionally averages over the ten thresholds from 0.50 to 0.95 in steps of 0.05. Precision is defined in Eq. (23):

$$\text{Precision} = \frac{TP}{TP+FP} \quad (23)$$

where, TP counts correctly matched detections and FP counts unmatched detections, including duplicate predictions. Matching uses the chosen IoU threshold and category. Recall is defined in Eq. (24):

$$\text{Recall} = \frac{TP}{TP+FN} \quad (24)$$

where, FN counts ground-truth objects without a matched detection. A missed object need not have zero IoU with every predicted box. Eq. (25) expresses category-averaged AP at a fixed IoU threshold:

$$\text{mAP} = \frac{\sum_{j=1}^n \int_0^1 P_j(R) dR}{n} \quad (25)$$

where, n is the number of categories, $P_j(R)$ is the precision-recall curve for category j , and R denotes recall. For mAP@[0.50:0.05:0.95], the result of Eq. (25) is also averaged over the ten specified IoU thresholds.

2) *Hyperparameter settings*: Hyperparameters were compared on the Flame dataset. The settings and observations are described below.

a) *Initial learning rate*: The learning rate follows the decay schedule in Eq. (26):

$$l = l_{\text{start}} \times \gamma^{E_{\text{now}}/E_{\text{threshold}}} \quad (26)$$

where, l is the current learning rate, l_{start} is the initial value, γ is the decay factor, E_{now} is the current epoch, and $E_{\text{threshold}}$ is the decay threshold. Fig. 6(a) compares $l_{\text{start}}=0.1$ and $l_{\text{start}}=0.01$ with a decay threshold of 150 and a decay factor of 0.1. Other settings were held constant. The initial learning rate of 0.1 converged faster and produced higher mAP within the training period, so it was selected for the final configuration.

b) *Training duration*: Fig. 6(b) compares runs of 150, 200, and 250 epochs. The initial learning rate was 0.01, and the corresponding decay thresholds were 100, 150, and 200. The remaining settings were held constant. The 200-epoch configuration achieved the highest accuracy among these runs. Because the decay threshold also changed, this comparison evaluates the two settings jointly and does not isolate the effect of training duration.

c) *Dropout*: Dropout values of 0, 0.1, and 0.2 were compared with other settings unchanged. Dropout of 0 produced the highest mAP, so it was disabled in the final configuration. This observation describes the tested settings; it does not establish the mechanism behind their performance differences.

3) *Ablation study*: Table III compares four model configurations trained for 200 epochs on Flame. The baseline YOLOv11 has 9,413,574 parameters and achieves AP = 0.782, AR = 0.774, mAP@0.50 = 0.793, and mAP@[0.50:0.05:0.95] = 0.579.

Adding DGCB reduces the parameter count to 2,778,487. AP increases to 0.790, AR to 0.783, mAP@0.50 to 0.848, and mAP@[0.50:0.05:0.95] to 0.588. Introducing WDIoU raises these metrics to 0.795, 0.790, 0.854, and 0.594, respectively, with 2,757,215 parameters.

The complete model adds HCAN and achieves AP = 0.827, AR = 0.826, mAP@0.50 = 0.889, and mAP@[0.50:0.05:0.95] = 0.624, with 2,738,750 parameters. Each successive configuration improves the reported metrics. The complete model provides the best results in this ablation comparison while retaining a substantially smaller parameter count than the baseline.

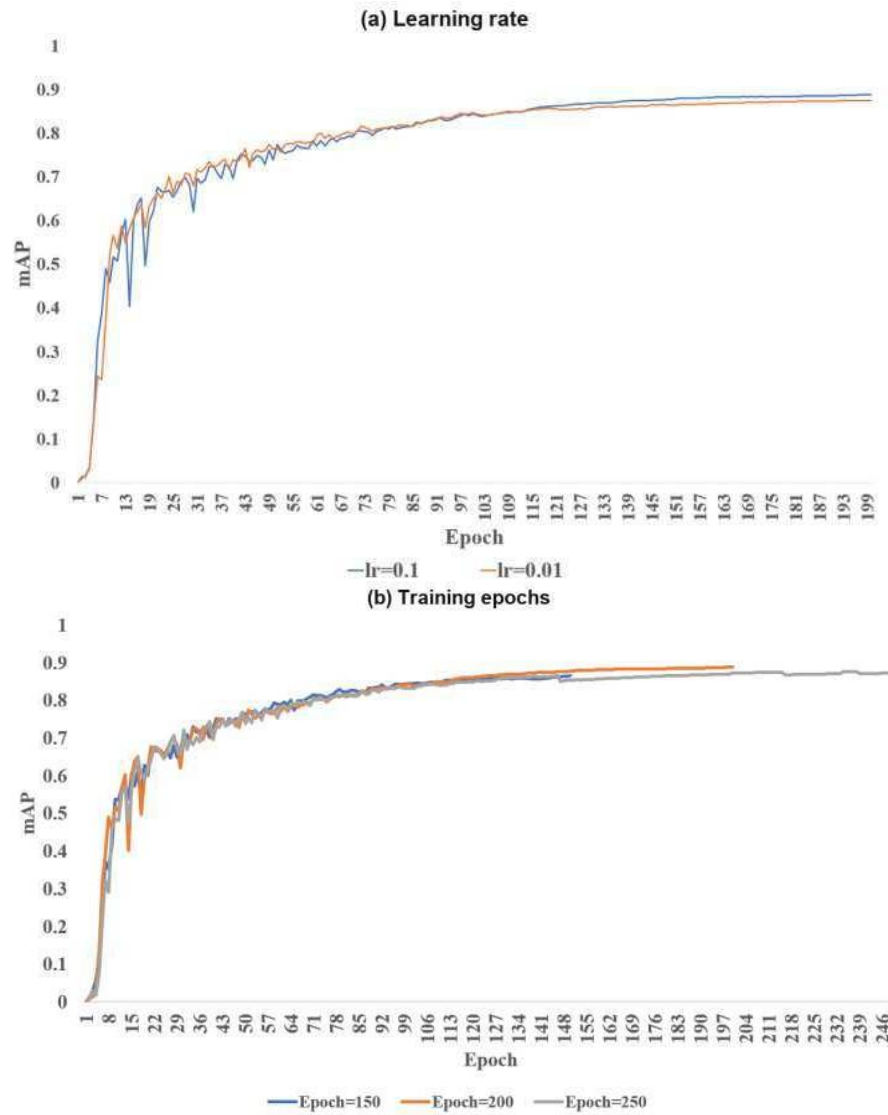


Fig. 6. Accuracy comparison for hyperparameter selection: (a) initial learning rate; (b) maximum number of training epochs.

TABLE III. ABLATION STUDY ON THE FLAME DATASET (200 EPOCHS)

Model	AP	AR	mAP@0.50	mAP@ [0.50:0.05:0.95]	Parameters
YOLOv11 (baseline)	0.782	0.774	0.793	0.579	9,413,574
YOLOv11 + DGCB	0.790	0.783	0.848	0.588	2,778,487
YOLOv11 + DGCB + WDIoU	0.795	0.790	0.854	0.594	2,757,215
YOLOv11 + DGCB + WDIoU + HCAN (Ours)	0.827	0.826	0.889	0.624	2,738,750

4) *Comparison with state-of-the-art methods:* Fig. 7- 9 compare detectors trained for 200 epochs on Flame and ForestFire. Fig. 8 reports their parameter counts. Faster R-CNN uses region proposals followed by classification and regression. Its tested configuration has the largest parameter count, at 43.7 M. On Flame, its AP, AR, mAP@0.50, and mAP@[0.50:0.05:0.95] are 0.723, 0.792, 0.758, and 0.498. On ForestFire, the corresponding values are 0.672, 0.721, 0.611, and 0.592.

YOLOv5 has 7.5 M parameters. It achieves AP = 0.776, AR = 0.792, mAP@0.50 = 0.723, and mAP@[0.50:0.05:0.95] = 0.587 on Flame. On ForestFire, the values are 0.798, 0.752, 0.699, and 0.692, respectively.

YOLOv8 [22] uses an anchor-free head and C2f blocks. The tested model has 11.2 M parameters. Its AP, AR, mAP@0.50, and mAP@[0.50:0.05:0.95] are 0.791, 0.803, 0.813, and 0.592 on Flame, and 0.863, 0.878, 0.857, and 0.729 on ForestFire.

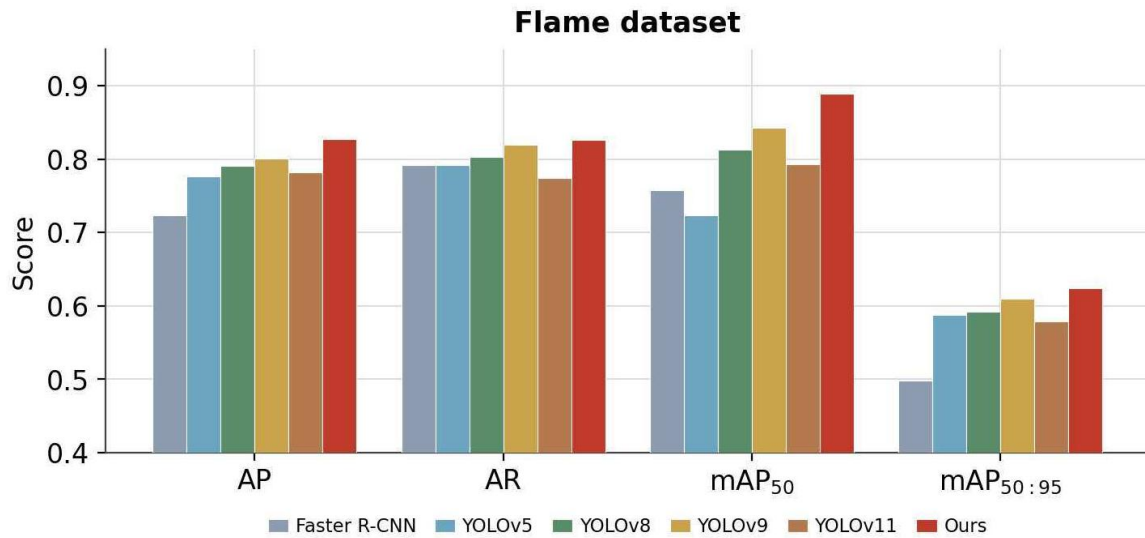


Fig. 7. Overall metric comparison of the different detectors on the Flame dataset.

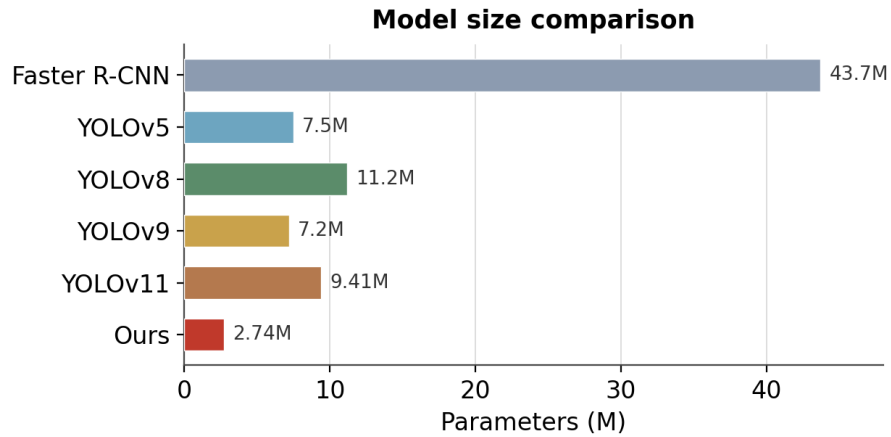


Fig. 8. Comparison of the number of parameters of the different detectors.

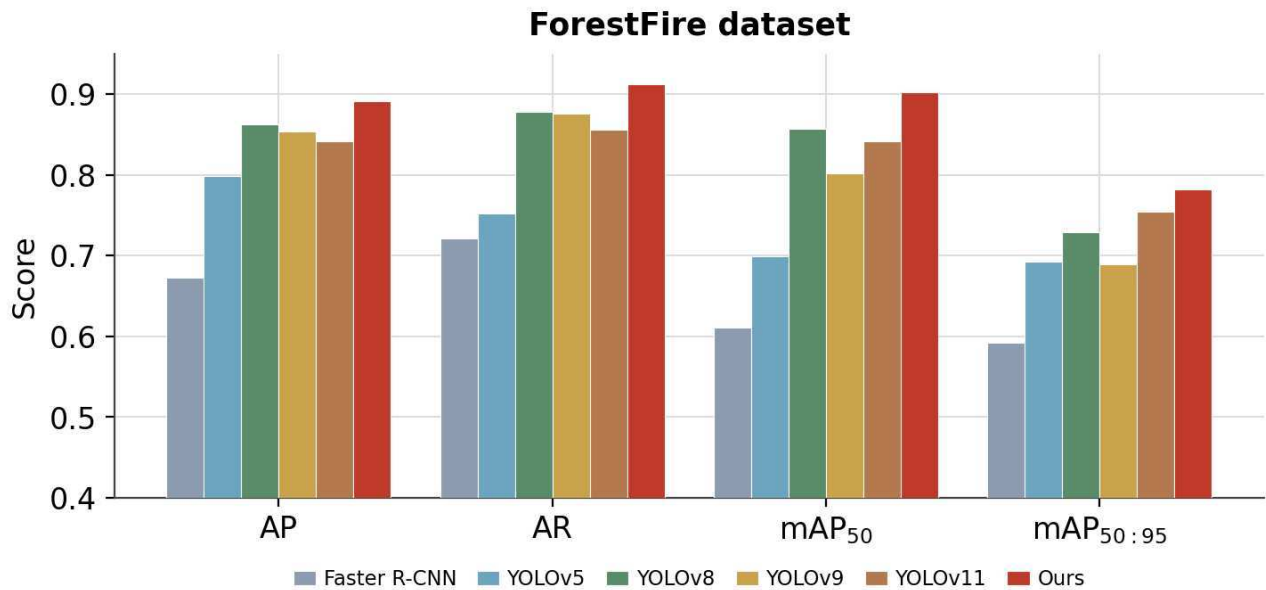


Fig. 9. Overall metric comparison of the different detectors on the ForestFire dataset.

YOLOv9 [23] introduces Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN). With 7.2 M parameters, it achieves $AP = 0.801$, $AR = 0.819$, $mAP@0.50 = 0.843$, and $mAP@[0.50:0.05:0.95] = 0.609$ on Flame. The corresponding ForestFire values are 0.854, 0.876, 0.802, and 0.689.

The baseline YOLOv11 has approximately 9.41 M parameters. It achieves $AP = 0.782$, $AR = 0.774$, $mAP@0.50 = 0.793$, and $mAP@[0.50:0.05:0.95] = 0.579$ on Flame. On ForestFire, these values are 0.842, 0.856, 0.842, and 0.754.

IWL-YOLOv11 combines DGCB, HCAN, and WDIoU. On Flame, it achieves $AP = 0.827$, $AR = 0.826$, $mAP@0.50 = 0.889$, and $mAP@[0.50:0.05:0.95] = 0.624$, with approximately 2.74 M parameters. Relative to YOLOv11, these values improve by 4.5, 5.2, 9.6, and 4.5 percentage points, respectively. The parameter count decreases by approximately 6.67 M.

On ForestFire, IWL-YOLOv11 achieves $AP = 0.891$, $AR = 0.912$, $mAP@0.50 = 0.902$, and $mAP@[0.50:0.05:0.95] = 0.782$. The corresponding gains over YOLOv11 are 4.9, 5.6, 6.0, and 2.8 percentage points. The proposed model has the highest reported values for all four metrics on both datasets and the smallest parameter count among the compared configurations.

5) *Grad-CAM heatmap comparison*: Fig. 10 compares Grad-CAM [31] heatmaps for the same fire image. Faster R-CNN concentrates mainly on the lower-left region and provides scattered coverage. YOLOv5 gives a more

concentrated response but misses parts of the fire region. YOLOv8 improves coverage, although some regions remain weakly represented. YOLOv9 highlights the fire more strongly but still misses fine detail. The YOLOv11 response is relatively uniform and leaves gaps within the fire region.

The proposed model concentrates its response more closely on the fire and covers most of the visible region. This observation is consistent with the intended role of HCAN in combining local features and broader context. The heatmaps provide a qualitative illustration of feature attention; the quantitative detection results are reported separately.

6) *Detection result comparison*: Fig. 11 compares detections for a forest fire image. Faster R-CNN produces relatively low confidence scores for smoke and fire. YOLOv5 improves smoke confidence in this example, while YOLOv8 produces higher smoke confidence and more consistent fire detections. YOLOv9 further improves some predictions, although smoke overlapping the tree background remains difficult. YOLOv11 produces relatively high smoke confidence but does not localize every boundary accurately.

The proposed model produces tighter boxes and higher confidence scores for the illustrated objects. WDIoU is intended to improve regression by adjusting sample contributions through its non-monotonic focusing coefficient. The example complements the dataset-level comparisons in Fig. 7-9.

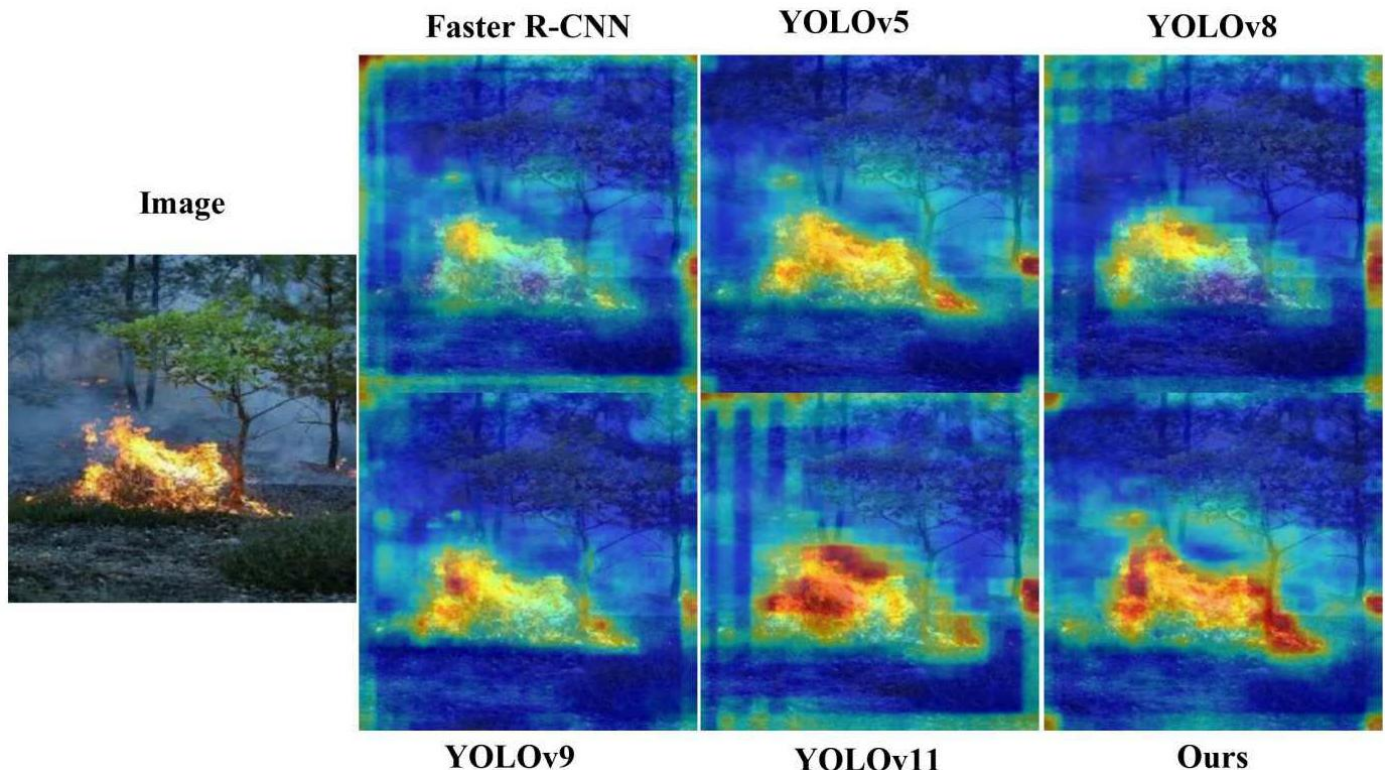


Fig. 10. Grad-CAM heatmap comparison of the different models.

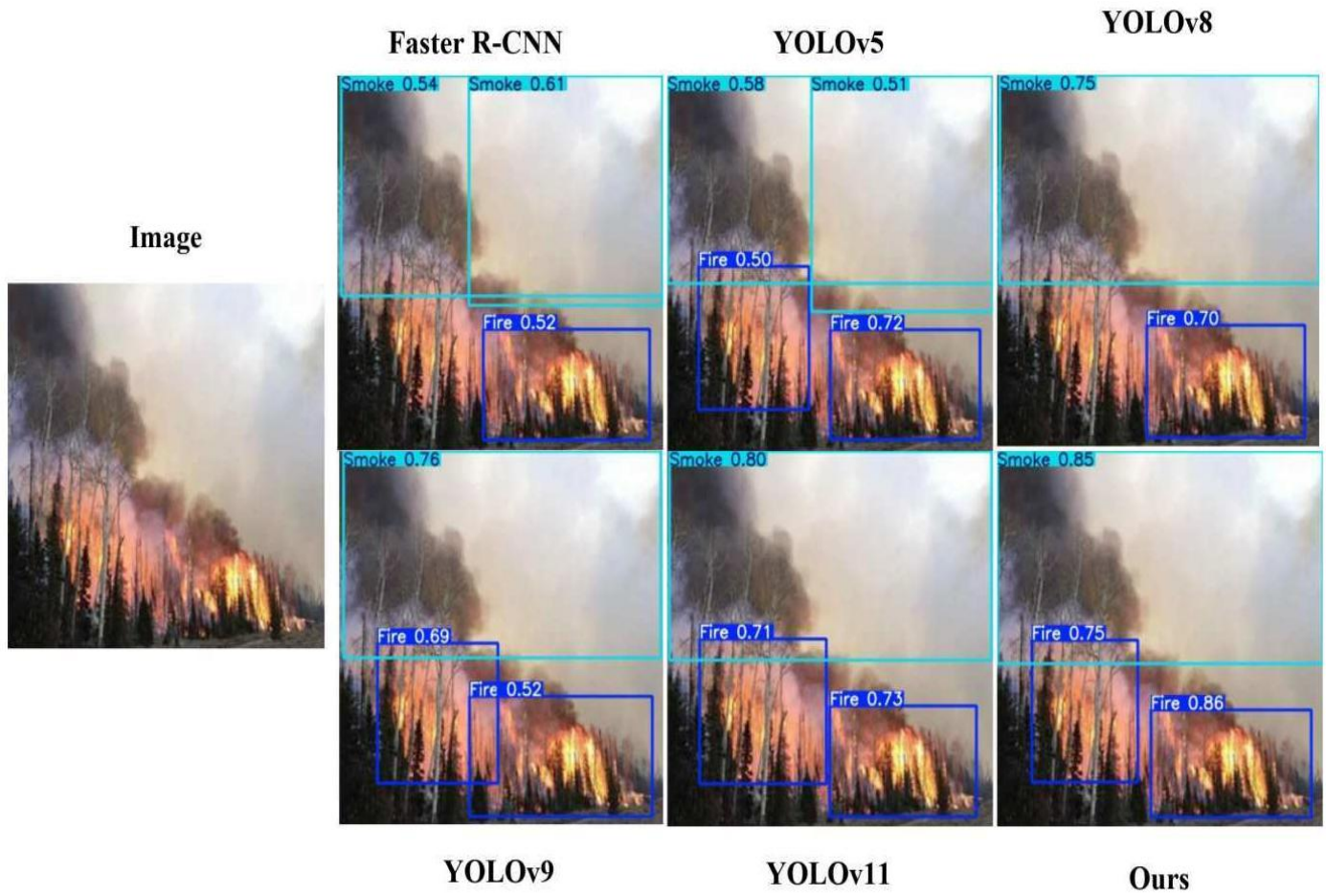
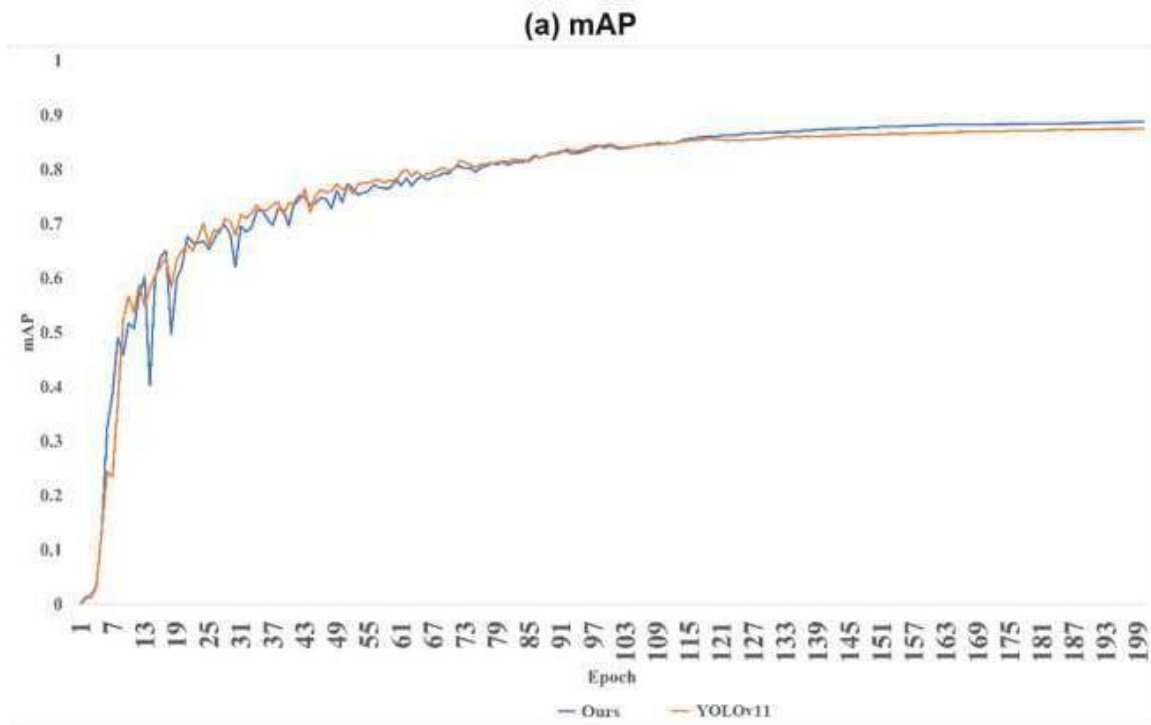


Fig. 11. Detection result comparison of the different models.



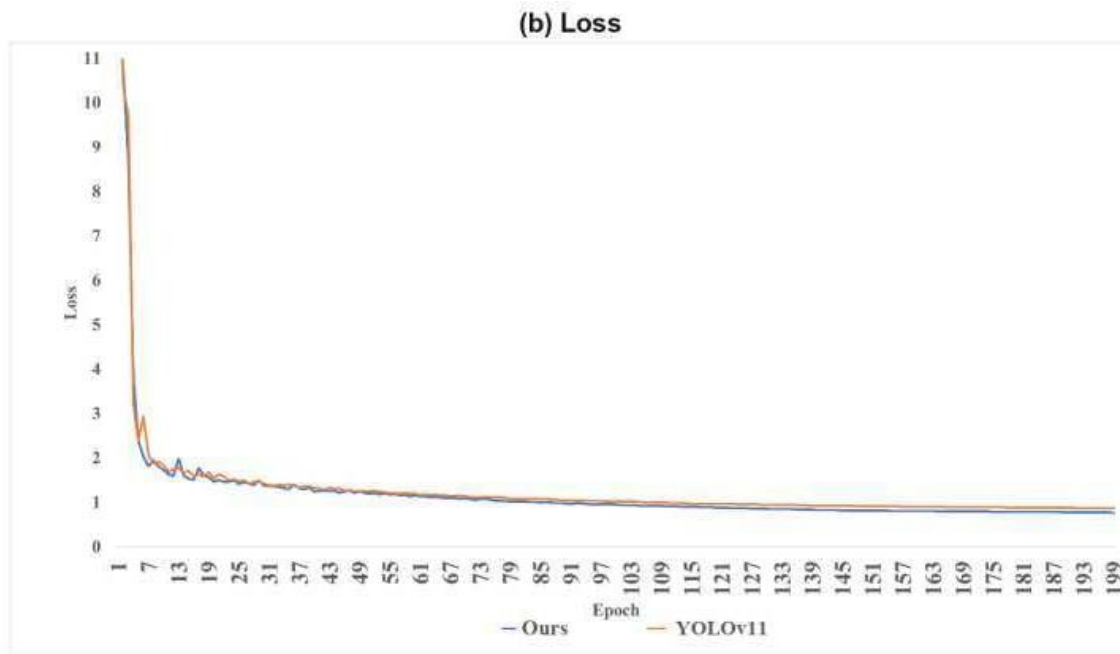


Fig. 12. Comparison of the training process between the proposed model and YOLOv11: (a) mAP; (b) loss.

7) *Training process analysis*: Fig. 12(a) compares mAP during the 200-epoch training runs of IWL-YOLOv11 and YOLOv11 on Flame. The proposed model approaches its highest mAP near epoch 150 and remains relatively stable afterward. Its curve is generally above the baseline curve. These observations support improved convergence and detection performance for the complete model.

Fig. 12(b) compares the corresponding loss curves. Both decrease during training, and the proposed model reaches a lower loss toward the end of the run. Because the model uses a modified regression objective, the loss values should be interpreted together with mAP and the ablation results rather than as an independent measure of detection accuracy.

C. Study Limitations

The reported evaluation has several limitations. The ForestFire test set contains 182 images. The modified Flame and ForestFire collections include Internet images, and image-level separation does not rule out correlated video frames or near-duplicates. A source-aware duplicate and frame-correlation audit is needed to assess that risk. The reported point estimates have no accompanying variance across training seeds, and the evaluation does not provide per-class or object-size results. Generalization to independent geographic regions and continuous video therefore remains unestablished.

The ablation is cumulative and does not independently isolate every component or its interactions. Single-image heatmaps and detections are qualitative examples. Parameter count measures network size but cannot determine inference latency, FLOPs, memory demand, or energy use; HCAN attention may affect these measures differently from convolution. These deployment measures, controlled comparisons with recent fire-specific detectors, and systematic

false-positive and false-negative analyses remain necessary for evaluating practical monitoring performance.

IV. CONCLUSION AND FUTURE WORK

This study introduced IWL-YOLOv11 for forest fire detection by combining DGCB, HCAN, and a WIoU-DFL regression objective. On the tested image splits, the model achieves mAP@0.50 of 88.9 percent on Flame and 90.2 percent on ForestFire, exceeding YOLOv11 by 9.6 and 6.0 percentage points, respectively. Its mAP@[0.50:0.05:0.95] is 62.4 percent and 78.2 percent, respectively. The 2,738,750 parameters represent an approximately 71 percent reduction from the baseline. These results support an accuracy-parameter trade-off for the evaluated configurations.

Detection remains difficult when images contain many small, densely distributed objects. Future work will examine small-object feature extraction and data augmentation, then evaluate their effects on detection accuracy and parameter count. Further validation should use source-aware image splits, multiple training seeds, independent geographic data, and continuous video. Deployment studies should measure latency, memory, and computational cost on the intended hardware, together with false alarms and missed detections under changing illumination and weather.

DECLARATIONS

Generative AI was used to assist with language editing, manuscript formatting, and equation transcription. The authors remain responsible for the accuracy, originality, and integrity of the manuscript.

REFERENCES

- [1] K. Gajendiran, S. Kandasamy, and M. Narayanan, "Influences of wildfire on the forest ecosystem and climate change: a comprehensive study," *Environmental Research*, vol. 240, Art. no. 117537, 2024.

- [2] G. Zhai, L. Dou, Y. Gu, et al., "From spark to suppression: An overview of wildfire monitoring, progression prediction, and extinguishing techniques," *International Journal of Applied Earth Observation and Geoinformation*, vol. 140, Art. no. 104600, 2025.
- [3] B. Aldughayfiq, I. Alrashdi, M. A. Tawfeek, et al., "Real-time forest fire detection, monitoring, and mitigation to safeguard humanity using YOLO and federated learning," *International Journal of Distributed Sensor Networks*, vol. 2025, no. 1, Art. no. 4707734, 2025.
- [4] L. He, Y. Zhou, L. Liu, Y. Zhang, and J. Ma, "Research and application of deep learning object detection methods for forest fire smoke recognition," *Scientific Reports*, vol. 15, no. 1, Art. no. 16328, 2025.
- [5] R. N. Vasconcelos, W. J. S. Franca Rocha, D. P. Costa, et al., "Fire detection with deep learning: a comprehensive review," *Land*, vol. 13, no. 10, Art. no. 1696, 2024.
- [6] B. Yun, Y. Zheng, Z. Lin, and T. Li, "FFYOLO: A lightweight forest fire detection model based on YOLOv8," *Fire*, vol. 7, no. 3, Art. no. 93, 2024, doi: 10.3390/fire7030093.
- [7] L. Lei, R. Duan, F. Yang, and L. Xu, "Low complexity forest fire detection based on improved YOLOv8 network," *Forests*, vol. 15, no. 9, Art. no. 1652, 2024, doi: 10.3390/f15091652.
- [8] H. Wang, X. Fu, Z. Yu, and Z. Zeng, "DSS-YOLO: An improved lightweight real-time fire detection model based on YOLOv8," *Scientific Reports*, vol. 15, Art. no. 8963, 2025, doi: 10.1038/s41598-025-93278-w.
- [9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142-158, 2016, doi: 10.1109/TPAMI.2015.2437384.
- [10] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440-1448.
- [11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.
- [12] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2961-2969.
- [13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, pp. 154-171, 2013.
- [14] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, IEEE, vol. 3, pp. 850-855, 2006.
- [15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779-788.
- [16] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7263-7271.
- [17] J. Redmon and A. Farhadi, "YOLOv3: an incremental improvement," *arXiv:1804.02767*, 2018.
- [18] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," *arXiv:2004.10934*, 2020.
- [19] G. Jocher, A. Stoken, J. Borovec, et al., "ultralytics/yolov5: v3.0," *Zenodo*, 2020, doi: 10.5281/zenodo.3983579.
- [20] C. Li, L. Li, H. Jiang, et al., "YOLOv6: A single-stage object detection framework for industrial applications," *arXiv:2209.02976*, 2022.
- [21] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 7464-7475.
- [22] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," 2023. <https://github.com/ultralytics/ultralytics>.
- [23] C. Y. Wang, I. H. Yeh, and H. Y. M. Liao, "YOLOv9: learning what you want to learn using programmable gradient information," in *Computer Vision - ECCV 2024, Lecture Notes in Computer Science*, vol. 15089, pp. 1-21, 2025, doi: 10.1007/978-3-031-72751-1_1.
- [24] A. Wang, H. Chen, L. Liu, et al., "YOLOv10: real-time end-to-end object detection," *Advances in Neural Information Processing Systems*, vol. 37, pp. 107984-108011, 2024.
- [25] G. Jocher and J. Qiu, "Ultralytics YOLO11," 2024. <https://github.com/ultralytics/ultralytics>.
- [26] W. Liu, D. Anguelov, D. Erhan, et al., "SSD: single shot multibox detector," in *Computer Vision - ECCV 2016*, Springer, 2016, pp. 21-37.
- [27] X. Li, W. Wang, L. Wu, et al., "Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21002-21012, 2020.
- [28] Z. Tong, Y. Chen, Z. Xu, and R. Yu, "Wise-IoU: bounding box regression loss with dynamic focusing mechanism," *arXiv:2301.10051*, 2023.
- [29] Z. Zheng, P. Wang, W. Liu, et al., "Distance-IoU loss: faster and better learning for bounding box regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 12993-13000, 2020.
- [30] A. Shamsoshoara, F. Afghah, A. Razi, et al., "Aerial imagery pile burn detection using deep learning: The FLAME dataset," *Computer Networks*, vol. 193, Art. no. 108001, 2021.
- [31] R. R. Selvaraju, M. Cogswell, A. Das, et al., "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618-626.