Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Lightweight YOLO Framework for Player and Ball Detection in Soccer Match Images

Jinping Wu
Guangdong Polytechnic Normal University
Xinhui Yuan
Huzhou University
Xiangyu Du
Guangdong Polytechnic Normal University
Lu Gao
02430@zjhu.edu.cn

Huzhou University

Research Article

Keywords: Soccer match images, Player-and-ball detection, YOLO11, Multi-scale feature aggregation, Lightweight object detection

Posted Date: November 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11561577/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Lightweight YOLO Framework for Player and Ball Detection in Soccer Match Images

Jinping Wu[1,†], Xinhui Yuan[2,†], Xiangyu Du[1], and Lu Gao[2,*]

[1] School of Physical Education and Health, Guangdong Polytechnic Normal University, Guangzhou 510000, China

[2] School of Information Engineering, Huzhou University, Huzhou 313000, China

[†] These authors contributed equally to this work.

*** Corresponding author:** Lu Gao, 02430@zjhu.edu.cn

Abstract

Accurate player-and-ball detection is a fundamental step in automated soccer match analysis. Small ball targets, dense player distributions, occlusion, motion blur, and complex backgrounds create a need for improved detection without increasing model complexity. This study presents PBDet-YOLO, a YOLO11-based framework that addresses this accuracy–complexity trade-off through three complementary components. Cross-Stage Partial Feature Aggregation (CSP_FA) provides multi-scale feature processing, a Lightweight Enhanced Detection Head (LE-Head) uses shared feature processing to reduce parameter redundancy, and Focaler-CIoU is adopted for bounding-box regression. Experiments were conducted on a self-constructed dataset of 11800 annotated images containing players and balls. PBDet-YOLO achieved 90.7% precision, 78.5% recall, and 85.9% mean average precision at an intersection-over-union threshold of 0.5 (mAP@0.5), with 2.51 million parameters and 6.2 GFLOPs. Relative to YOLO11, mAP@0.5 increased from 84.0% to 85.9%, while parameters decreased from 2.59 million to 2.51 million and computational complexity decreased from 6.4 to 6.2 GFLOPs. The ablation results distinguish the additional computation associated with CSP_FA from the reduction achieved by LE-Head. Together, these results indicate an improved accuracy–complexity trade-off among the lightweight YOLO configurations evaluated for soccer player-and-ball detection.

Keywords: Soccer match images; Player-and-ball detection; YOLO11; Multi-scale feature aggregation; Lightweight object detection

1 Introduction

Football combines individual actions with coordinated team tactics and generates a substantial volume of match footage within a global sports industry [1]. Extracting useful information from these images begins with locating the players and the ball. Soccer match images present distinctive visual difficulties, including large differences in target size, densely distributed players, partial occlusion, motion blur, and complex backgrounds [2]. A detector must therefore identify both small balls and larger player targets within the same image. These requirements motivate the development of a compact detection framework that balances multi-scale visual representation with the computational cost of the network.

Deep learning has expanded the design space for visual recognition. Convolutional neural networks (CNNs) and Transformer-based architectures provide different approaches to constructing visual representations; recent ConvNet designs also demonstrate the continuing relevance of convolutional backbones for detection and segmentation [3]. Object detectors

such as YOLO, DETR, Faster R-CNN, and SSD combine these representations with classification and localization components [4]. Their architectures differ in feature extraction, proposal generation, prediction heads, and computational requirements. In soccer images, small-object detection remains particularly relevant because the ball and distant players occupy much smaller regions than nearby players [5]. At the same time, increasing feature-processing capacity can increase the computational load. The research question addressed here is whether multi-scale feature aggregation, a shared lightweight detection head, and an IoU-based regression loss can improve joint player-and-ball detection while retaining the low parameter count and computational complexity of a lightweight YOLO11 baseline.

To investigate this question, this study presents PBDet-YOLO, a YOLO11-based detection framework for soccer match images. Its three components address different parts of the detection pipeline. Cross-Stage Partial Feature Aggregation (CSP_FA) replaces C3k2 feature-processing blocks and combines convolutional operations with different kernel sizes. The Lightweight Enhanced Detection Head (LE-Head) introduces shared feature processing into the prediction stage. Focaler-CIoU is used in place of CIoU for bounding-box regression in the complete configuration. The evaluation compares individual structural changes, their combination, and the addition of the regression loss. It then compares the complete model with other detectors using detection metrics, parameter counts, and GFLOPs. This organization connects the architectural choices to their measured effects on the constructed soccer dataset and positions the detector as a potential front end for subsequent player tracking, ball tracking, and tactical analysis.

The main contributions of this paper include:

1. A soccer-specific evaluation dataset containing 11800 annotated images is constructed for joint player-and-ball detection. The dataset includes two object categories and match images with variations in venue, illumination, and camera perspective, providing the experimental basis for evaluating the proposed configurations.

2. A Cross-Stage Partial Feature Aggregation (CSP_FA) module combines standard and partial convolutions with different kernel sizes within a cross-stage feature-processing structure. Its contribution is evaluated through a C3k2 replacement experiment that reports both detection performance and the associated computational cost.

3. CSP_FA is integrated with LE-Head and Focaler-CIoU to balance feature processing, prediction-head complexity, and regression optimization. The resulting configuration increases mAP@0.5 by 1.9 percentage points over YOLO11 while reducing parameters and GFLOPs by approximately 3.1%, as established by the ablation and detector-comparison experiments.

2 Related work

Object detection development

Object detection has evolved from pipelines based on handcrafted features to systems that learn visual representations and prediction functions from data [6]. Traditional sliding-window approaches examine candidate image regions at multiple positions and scales. Features such as Histogram of Oriented Gradients (HOG) or Scale-Invariant Feature Transform (SIFT) describe the appearance of these regions, while classifiers such as support vector machines (SVMs) distinguish targets from the background. This separation of feature

design and classification provides an explicit processing pipeline, but scanning many windows increases the amount of computation. Recognition also depends on how well the chosen descriptors represent the target under changes in illumination, pose, and partial occlusion. In soccer imagery, the coexistence of small balls and players of different apparent sizes makes the selection of suitable scales and visual descriptors particularly important. These considerations form part of the motivation for learned, multi-scale feature representations in subsequent detection methods.

CNN-based detectors learn feature representations jointly with the prediction task. Two-stage detectors first generate candidate regions and then classify and refine them, whereas one-stage detectors predict object categories and locations directly from image features [6]. R-CNN, Fast R-CNN, and Faster R-CNN exemplify the proposal-based family. Within this family, Light-Head R-CNN specifically investigates the computational structure of the detection head, showing that head design is itself an important aspect of detector efficiency [7]. YOLO and SSD exemplify one-stage prediction and remove the separate region-proposal stage. The complete backbone, feature-fusion, and prediction-head configuration determines the parameter count and accuracy of each detector. For compact detection networks, feature extraction and prediction-head design must be considered together. This is the architectural setting of PBDet-YOLO, which changes the feature-processing blocks and detection head of a one-stage baseline and evaluates the resulting configurations using the same set of reported metrics.

Transformer-based detection introduced a different formulation of the prediction task. Following the attention architecture of Vaswani et al. [8], DETR uses a Transformer encoder–decoder to perform object detection as set prediction and removes the conventional non-maximum suppression (NMS) stage [9]. Later real-time Transformer detectors use targeted architectural optimizations. RT-DETR, for example, introduces an efficient hybrid encoder for multi-scale features and a query-selection strategy for the decoder [10]. CNN-based and Transformer-based detectors therefore provide distinct design choices for balancing representation, prediction, and computational cost. In soccer analysis, attention has also been used in full-field player tracking and data correction [11], illustrating the broader relevance of visual representations to downstream tasks. The present comparison includes lightweight YOLO models and Rt18-DETR so that detection performance can be examined alongside the substantially different parameter counts and GFLOPs of the evaluated architectures.

Ball and player detection development

Early soccer-vision methods relied on color, geometric structure, and local appearance features. Kikuchi et al. [12] investigated color recognition in the RoboCup setting, while Hanek et al. [13] examined visual perception without relying on artificial color labeling. These studies addressed robotic soccer and provide historical context for visual target recognition in constrained playing environments. For match-video analysis, Baysal and Duygulu [14] developed Sentioscope, a soccer-player tracking system that includes player detection using HOG features and an SVM classifier. Manafifard et al. [15] subsequently reviewed player-tracking methods in soccer videos, bringing together the challenges of detection, occlusion, and temporal association. The distinction between robotic soccer and broadcast match imagery is relevant when interpreting these studies: camera viewpoints, target appearance, and scene composition differ. Together, these works establish the importance of reliable visual

localization, while separating image-level detection from the additional task of maintaining identities over time.

Neural-network methods extend this work through learned representations. Ribeiro et al. [16] applied a multilayer perceptron to visual recognition in RoboCup middle-size league soccer. Hurault et al. [17] proposed a self-supervised pipeline for detecting and tracking small soccer players without requiring ground-truth annotations for the adaptation procedure. Komorowski et al. [18] developed FootAndBall specifically for joint player-and-ball detection in high-resolution match footage. Its fully convolutional architecture uses a feature-pyramid design that combines higher-resolution features with features having larger receptive fields. This makes FootAndBall directly relevant to the joint detection task considered here. Sorano et al. [19] developed PassNet for automatic pass annotation, combining object detection with an LSTM-based temporal model. Lu et al. [20] investigated lightweight CNNs for player detection and classification. Zhao [21] developed an upgraded YOLOv5 approach to football and player detection incorporating SimSPPF, GhostNet, and a slim-scale detection layer. PBDet-YOLO uses a YOLO11 backbone and evaluates a different combination of multi-scale aggregation, a shared detection head, and IoU-based regression. These studies cover related but distinct objectives: visual localization, player tracking, classification, and event annotation. PBDet-YOLO concentrates on the image-level detection component, retaining a YOLO11 framework while evaluating changes to feature aggregation, the detection head, and bounding-box regression. The resulting comparison concerns an architectural accuracy–complexity trade-off within this specific task.

3 Materials and methods

Dataset collection

A self-constructed dataset was used for detecting players and soccer balls in soccer match images. Soccer-related videos were collected from online video platforms using web crawlers, and keyframes were extracted to form the image collection. The dataset contains 11800 annotated images with a resolution of 1280 × 1280 pixels. The annotations cover two target categories: players and soccer balls. The images include match scenes with differences in player distribution and apparent target size.

Data collection considered variations in match venue, illumination, and camera perspective. The images were manually annotated for players and soccer balls, and the annotations underwent repeated checks. The object annotations record the locations of players and soccer balls in the collected match images. Representative dataset samples are shown in Fig. 1.



Fig. 1 Representative samples from the soccer match image dataset

Overall network structure

PBDet-YOLO is a single-stage detector based on YOLO11 [22]. Its architecture comprises a backbone for feature extraction, a neck for multi-scale feature fusion, and a detection head for object classification and localization. As shown in Fig. 2, CSP_FA blocks are placed in both the backbone and neck, replacing the corresponding C3k2 feature-processing blocks. CSP_FA integrates standard convolution with partial convolution (PConv), an operator that applies spatial filtering to a subset of feature channels [23]. The network supplies feature maps at three prediction scales, P3, P4, and P5, to the detection head.

The multi-scale features are passed to LE-Head for box regression and classification. LE-Head uses shared intermediate feature processing followed by separate prediction branches at each scale, as illustrated in Fig. 5. The shared-head design follows the lightweight architecture described in [24], with related detection-head applications presented in [25]. The classification branches generate scores for the player and ball categories, while the regression branches produce the predicted bounding boxes. LE-Head is connected to the CSP_FA-based feature extraction and fusion stages within PBDet-YOLO.

For bounding-box regression, PBDet-YOLO adopts Focaler-CIoU based on the Focaler-IoU formulation [26]. This loss uses an interval-based mapping of the IoU term within the CIoU regression objective while retaining its center-distance and aspect-ratio terms. Section 3.5 provides the loss definition, and Fig. 2 summarizes the feature-extraction and prediction pathways of the overall network.

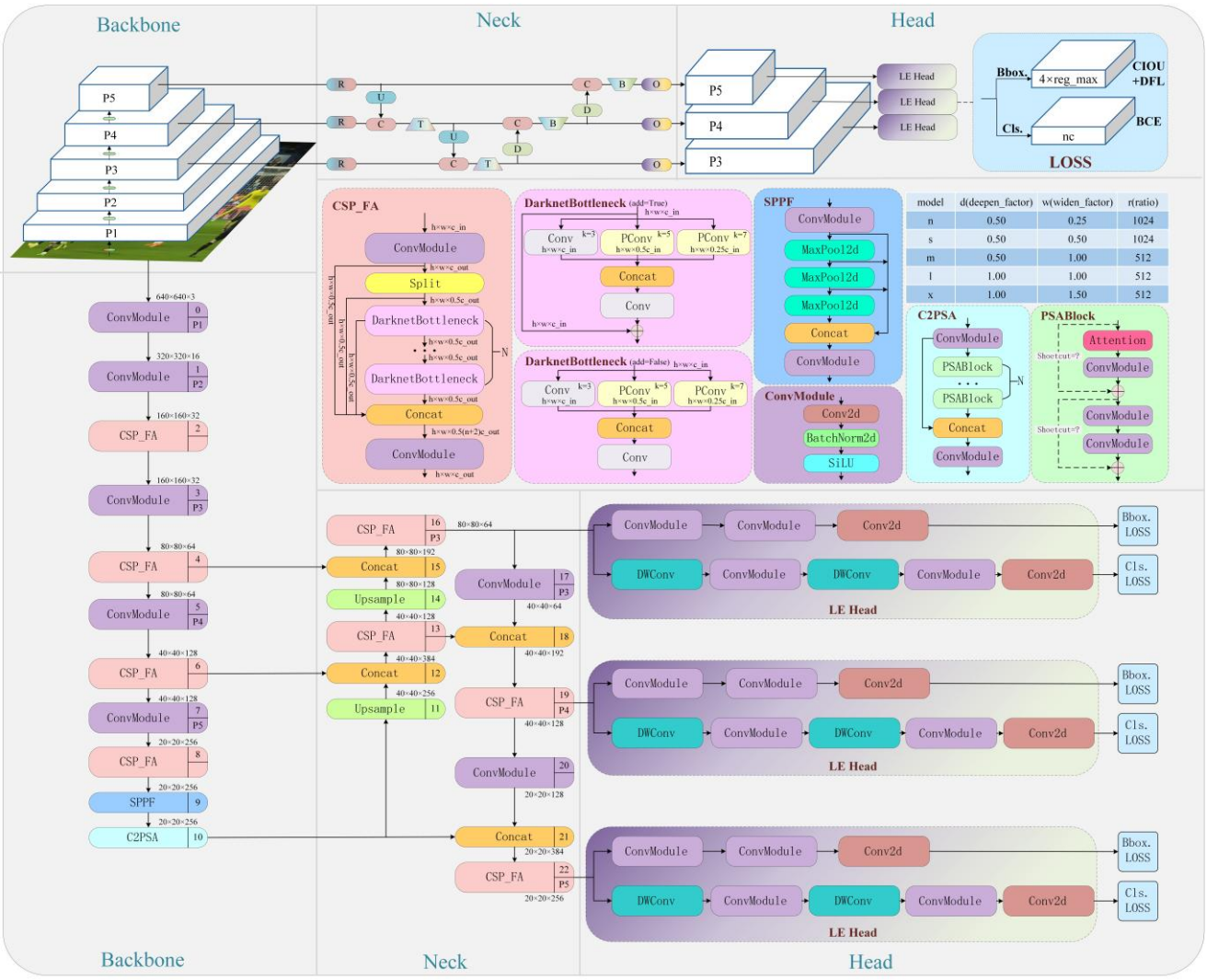


Fig. 2 Overall network architecture of PBDet-YOLO

Cross-stage partial feature aggregation block

C3k2 is a cross-stage feature-processing block in YOLO11 [22]. As shown in Fig. 3(b), an initial convolution is followed by channel splitting, a sequence of internal processing blocks, feature concatenation, and an output convolution. The figure distinguishes configurations using bottleneck blocks and configurations using C3k blocks. Both arrangements retain feature paths that feed the final concatenation. CSP_FA adopts the same general split–process–aggregate organization but changes the internal spatial-processing unit. Figure 3(a) shows the CSP_FA structure, whereas Fig. 3(b) presents the C3k2 structures used for comparison. The architectural difference lies in the organization of convolutional operations within the feature-processing paths.

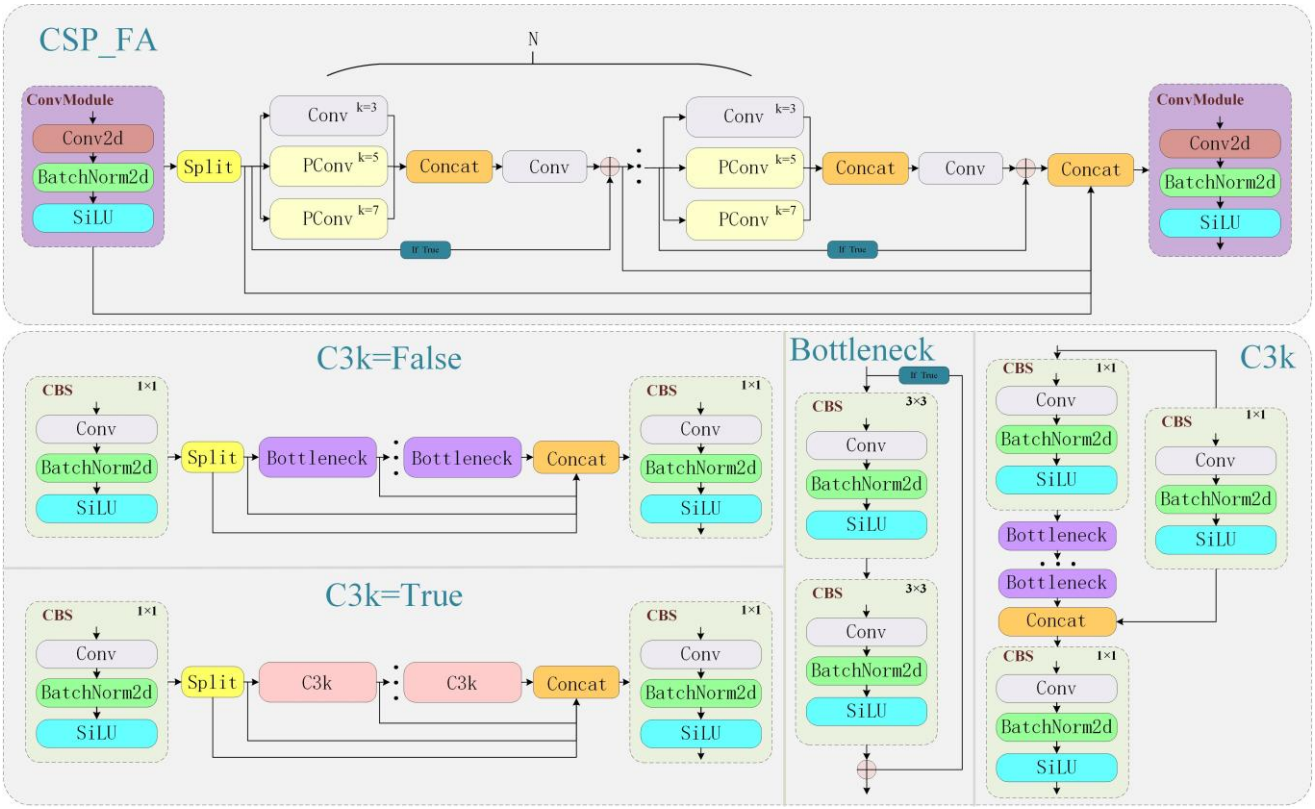


Fig. 3 Comparison of the proposed CSP_FA module and the original C3k2 module: (a) CSP_FA, shown in the upper panel; (b) C3k2 configurations, shown in the lower panel

CSP_FA draws on efficient feature-generation ideas represented by GhostNet [27] and the partial-convolution operator of FasterNet [23]. These sources contribute different ideas: GhostNet generates additional features through inexpensive operations, whereas PConv applies spatial convolution to a subset of channels. The CSP_FA structure in Fig. 3(a) begins with a ConvModule containing a convolution, batch normalization, and a SiLU activation. Its output is split into feature paths. Within the repeated processing unit, a 3 × 3 standard convolution, a 5 × 5 PConv, and a 7 × 7 PConv form parallel spatial-processing branches. Their outputs are concatenated and passed through a convolution. The diagram also includes a conditional shortcut around the processing unit. The units are arranged sequentially, and their outputs join the retained feature paths at the final concatenation. A concluding ConvModule produces the block output. The cross-stage paths preserve features for aggregation, while the parallel branches within each processing unit apply the three kernel sizes. The complete detector incorporates CSP_FA as shown in Fig. 2.

The PConv operation used in CSP_FA is illustrated in Fig. 4. Unlike a standard convolution that processes all input channels, PConv applies spatial filters to a subset of channels and leaves the other channels unchanged [23]. In CSP_FA, the 5 × 5 and 7 × 7 PConv branches are combined with the 3 × 3 standard-convolution branch. The kernel sizes therefore differ within the same processing unit, and their outputs are brought together by concatenation and convolution. The feature paths around the repeated units provide additional inputs to the final aggregation. This arrangement addresses the scale differences in soccer match images, where nearby players, distant players, and the ball appear at different sizes within the same frame. The block combines spatial processing with several kernel sizes and partial-channel computation in its 5 × 5 and 7 × 7 branches. The aggregated features are passed to the subsequent stages of the detector, as illustrated in Fig. 2.

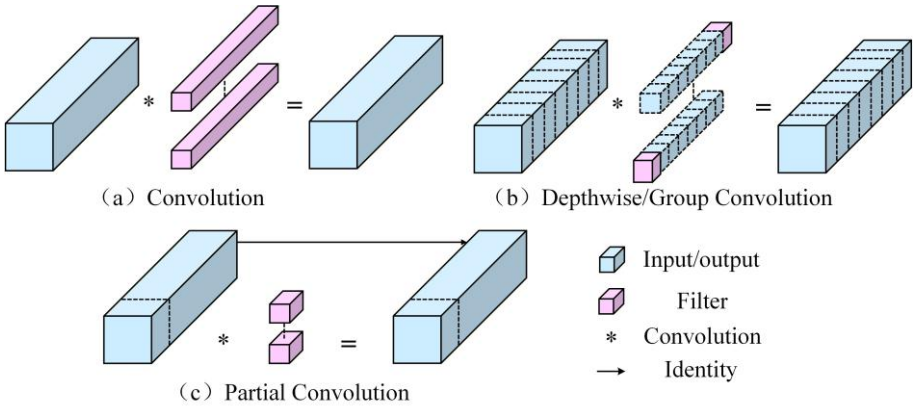


Fig. 4 Partial convolution (PConv) mechanism. Filters are applied to a subset of input channels while the remaining channels are retained unchanged

Lightweight enhanced detection head block

The prediction head converts multi-scale features into object-category scores and bounding-box outputs. YOLO11 retains the decoupled classification and regression organization used in YOLOv8 [22,28]. In this anchor-free design, classification and localization are handled by separate prediction branches rather than a separate objectness branch. Distribution Focal Loss (DFL) is associated with the distribution-based representation of box coordinates. YOLO11 also uses depthwise convolutions in its classification branch to reduce the cost of that prediction path [22]. These architectural choices form the original detection-head structure. LE-Head retains the distinction between classification and localization while changing the shared intermediate feature processing before the outputs.

LE-Head [24] uses shared feature processing to reduce redundancy in the prediction stage. Related lightweight head applications are described in [25], and RTMDet provides a reference for efficient detection-head design [29]. Figure 5 shows three feature inputs, P3, P4, and P5. Each input first passes through a convolution-and-group-normalization block. The initial 1 × 1 convolution adjusts the input channels, and the following shared Detail-Enhanced Convolution and Group Normalization blocks are denoted DEConv-GN. The figure displays two such shared processing blocks before the outputs separate into regression and classification branches at each scale. Group normalization is included in the shared processing, while the Conv-GN legend also identifies convolution and SiLU activation. This organization retains predictions at all three scales while sharing the intermediate feature-processing stage. In PBDet-YOLO, LE-Head receives the multi-scale feature maps produced by the backbone and neck and generates the classification and localization outputs.

For each prediction scale s ∈ {3, 4, 5}, let F_s denote the corresponding input feature map. The feature map is projected through a scale-specific Conv–GN operation and then processed by two shared DEConv–GN blocks. The resulting feature transformation is written as:

$$Z_s = g_\theta\big(\mathrm{ConvGN}_s(F_s)\big), \quad s \in \{3,4,5\}. \tag{1}$$

In Eq. (1), Z_s is the resulting representation, ConvGN_s denotes the scale-specific input projection, and g with parameter set θ represents the composite mapping formed by the two consecutive DEConv–GN blocks shown in Fig. 5. The parameters θ are shared across the three prediction scales. The regression and classification branches then generate predictions as follows:

$$\hat{B}_s = h_{\mathrm{reg},s}(Z_s), \quad \hat{P}_s = h_{\mathrm{cls},s}(Z_s). \tag{2}$$

In Eq. (2), the two functions denote the scale-specific box-regression and two-class classification paths. Equations (1) and (2) express the shared feature processing and the subsequent prediction branches shown in Fig. 5.

The regression and classification branches perform different prediction tasks. At each scale, the regression branch uses the distribution-based box representation associated with reg_max, while the classification branch outputs scores for the player and ball categories. Figure 5 places a Scale block after each regression convolution and a separate convolution on the classification path. The shared intermediate feature-processing layers operate on the projected P3, P4, and P5 inputs before the task-specific outputs are generated. In this way, the head retains distinct prediction branches at each scale while using shared feature processing across scales.

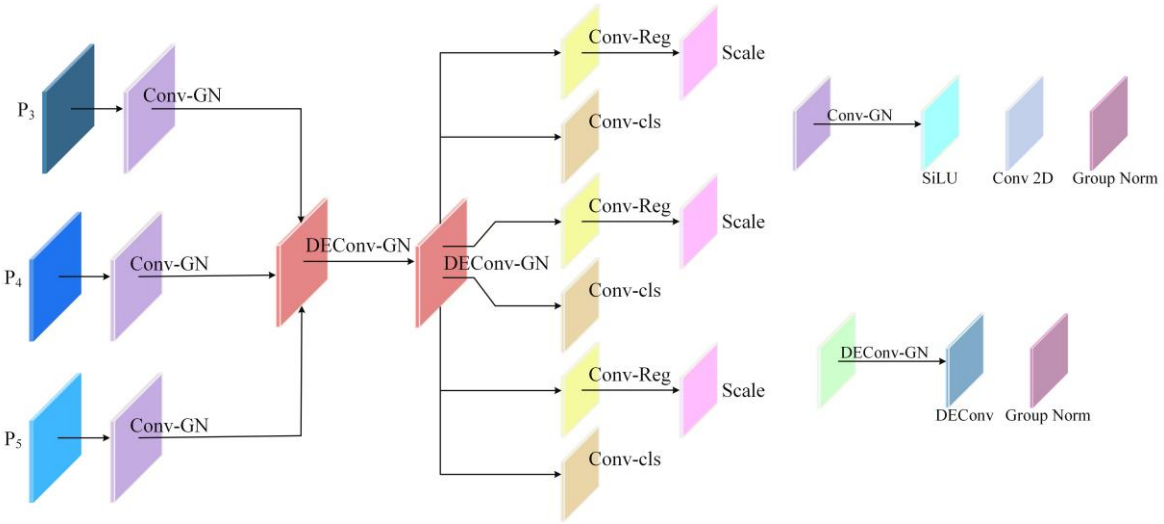


Fig. 5 Structure of LE-Head, showing the P3, P4, and P5 inputs, shared DEConv-GN processing, and separate regression and classification branches

Focaler-CIoU

Bounding-box regression optimizes the correspondence between a predicted box and its reference annotation. Intersection over union (IoU) measures the overlap of the two boxes, while CIoU also incorporates their center distance and aspect-ratio consistency [30]. These geometric terms provide the basis for localization in the baseline detector. Focaler-CIoU changes how the IoU component contributes to the regression objective through an interval-based mapping. The mapping adjusts regression emphasis across samples with different IoU values while retaining the geometric terms of CIoU.

Zhang and Zhang [26] introduced Focaler-IoU to control regression emphasis through a piecewise linear mapping of the original IoU value. Two bounds define an interval within which IoU is mapped linearly to the range from zero to one. Values below and above that interval are mapped to its lower and upper endpoints, respectively. Changing the interval changes the IoU range emphasized by this component of the objective. This formulation provides the mapped overlap term used to construct Focaler-CIoU for bounding-box regression in PBDet-YOLO.

The piecewise linear mapping is expressed as follows:

$$IoU^{focaler} = \begin{cases} 0, & IoU < d \\ \frac{IoU-d}{u-d}, & d \le IoU \le u, \\ 1, & IoU > u \end{cases} \quad (3)$$

Here, $IoU^{focaler}$ denotes the mapped overlap and IoU is the original intersection over union. The lower and upper bounds, d and u, satisfy 0 ≤ d < u ≤ 1. Within this interval, $IoU^{focaler}$ varies linearly from zero to one. The corresponding loss is defined as follows:

$$L_{Focaler\text{-}IoU} = 1 - IoU^{focaler}, \quad (4)$$

Following the Focaler-IoU formulation [26], the mapped overlap term is combined with CIoU to obtain the Focaler-CIoU loss:

$$L_{Focaler\text{-}CIoU} = L_{CIoU} + IoU - IoU^{focaler}, \quad (5)$$

In PBDet-YOLO, Focaler-CIoU is employed to optimize bounding-box regression. As expressed in Eq. (5), the IoU overlap term in CIoU is replaced by the mapped Focaler-IoU term, while the center-distance and aspect-ratio penalties are retained. The parameters d and u define the interval used in the IoU mapping in Eq. (3), allowing the regression objective to emphasize samples within different IoU ranges.

4 Results

Experimental platform and parameter settings

The model was trained using PyTorch 1.8 and Python 3.8. The server environment used Ubuntu 22.04, an Intel Xeon E5-2650 v3 CPU running at 2.30 GHz, 64 GB of memory, and an Nvidia V100 GPU with 32 GB of VRAM. The software environment also included CUDA 11.8 and cuDNN 8.2.

The local environment used an Intel i9-14900HX processor running at 5.2 GHz, 32 GB of memory, and a mobile Nvidia RTX 4060 GPU with 8 GB of VRAM. CUDA 11.8 and Python 3.8 were also installed in this environment.

The training input resolution was 640 × 640, with a batch size of 128 and a training duration of 200 epochs. The SGD optimizer was used. The 11800-image dataset was divided into 9,440 training images, 1,180 validation images, and 1,180 test images, corresponding to an 8:1:1 ratio. The learning rate followed the linear-decay schedule listed in Table 1. Momentum and weight decay were set to 0.937 and 5 × 10^{-4}, respectively. The initial and final learning rates were 10^{-2} and 10^{-4}. The table also specifies three warm-up epochs, warm-up momentum of 0.8, and a warm-up bias learning rate of 0.0.

Table 1 Hyperparameters of the PBDet-YOLO model.

Hyperparameter	Value
Epochs	200
Optimizer	SGD
Momentum	0.937
Batch size	128
Weight decay	5×10^{-4}
Warm-up epochs	3
Warm-up momentum	0.8
Warm-up bias learning rate	0.0
Initial learning rate	10^{-2}
Final learning rate	10^{-4}
Learning rate schedule	Linear decay

Ablation studies

The ablation study evaluates the three components through five model configurations on the soccer dataset. Model A is the original YOLO11 baseline. Model B replaces its C3k2 blocks with CSP_FA, whereas Model C replaces only the original detection head with LE-Head. Model D combines CSP_FA and LE-Head. Model E retains this combined structure and replaces CIoU with Focaler-CIoU for bounding-box regression. Table 2 reports the parameter count, precision, recall, mAP@0.5, and GFLOPs for each configuration. The A-to-B and A-to-C comparisons describe the two individual structural replacements; the B-to-D and C-to-D comparisons describe their combination; and the D-to-E comparison concerns the regression loss. Detection metrics are considered together with model size and computational complexity so that both the gains and costs of the structural changes remain visible.

Table 2 Model ablation experiment.

Model	CSP_FA	LE-Head	Focaler-CIoU	Parameters (M)	Precision (%)	Recall (%)	mAP@0.5 (%)	GFLOPs
A				2.59	91.0	77.4	84.0	6.4
B	✓			2.64	91.0	78.9	84.5	7.8
C		✓		2.26	92.9	77.3	84.4	6.0
D	✓	✓		2.51	89.5	79.4	84.9	6.2
E	✓	✓	✓	2.51	90.7	78.5	85.9	6.2

Table 2 and Fig. 6 show distinct effects for the two structural replacements. Adding CSP_FA alone increases mAP@0.5 from 84.0% to 84.5% and recall from 77.4% to 78.9%, with precision unchanged at 91.0%. The parameter count increases from 2.59 million to 2.64 million, and GFLOPs increase from 6.4 to 7.8. Model B therefore combines an improvement in

the reported detection metrics with additional computation. Replacing only the head produces a different profile: Model C uses 2.26 million parameters and 6.0 GFLOPs while achieving 84.4% mAP@0.5. Relative to Model A, these are parameter and GFLOPs reductions of approximately 12.7% and 6.3%, respectively. Its precision is 92.9% and recall is 77.3%. Combining both components in Model D yields 84.9% mAP@0.5 with 2.51 million parameters and 6.2 GFLOPs. Relative to Model B, this combination reduces GFLOPs by 1.6 while increasing mAP@0.5 by 0.4 percentage points. Relative to Model C, it adds 0.25 million parameters and 0.2 GFLOPs for a 0.5-percentage-point mAP@0.5 increase. These comparisons describe the different accuracy–complexity profiles of the individual components and their combination.

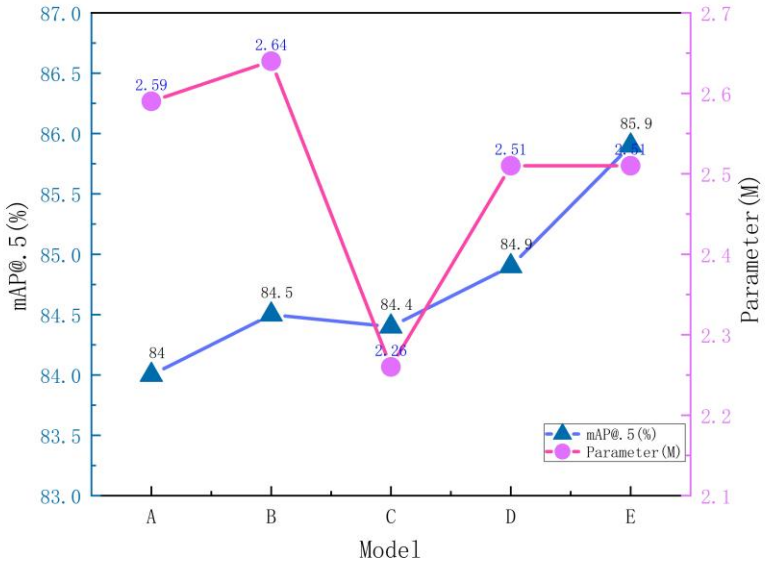


Fig. 6 Ablation comparison of the different model configurations

Introducing Focaler-CIoU in Model E raises mAP@0.5 from 84.9% to 85.9% while retaining the 2.51 million parameters and 6.2 GFLOPs of Model D. Precision increases from 89.5% to 90.7%, whereas recall changes from 79.4% to 78.5%. Thus, the loss change improves mAP@0.5 and precision in this comparison, with a 0.9-percentage-point decrease in recall. Across the five configurations, Model E has the highest mAP@0.5, Model C the highest precision, and Model D the highest recall. Relative to the YOLO11 baseline, the complete PBDet-YOLO configuration improves mAP@0.5 by 1.9 percentage points and recall by 1.1 percentage points, while precision changes from 91.0% to 90.7%. Parameters decrease by 0.08 million and GFLOPs by 0.2, corresponding to reductions of approximately 3.1% in each case. The complete configuration consequently improves the reported mAP@0.5 while preserving a model size and computational requirement close to those of the lightweight baseline.

Comparative experiments

Comparison of training effects with the YOLO11 model

Figure 7 compares the training and validation loss curves of PBDet-YOLO and YOLO11 over the 200-epoch training schedule. It separately presents box-regression loss, classification loss, and Distribution Focal Loss. These curves describe the progression of the optimization

objectives, complementing the detection metrics and complexity measurements in Tables 2 and 3.

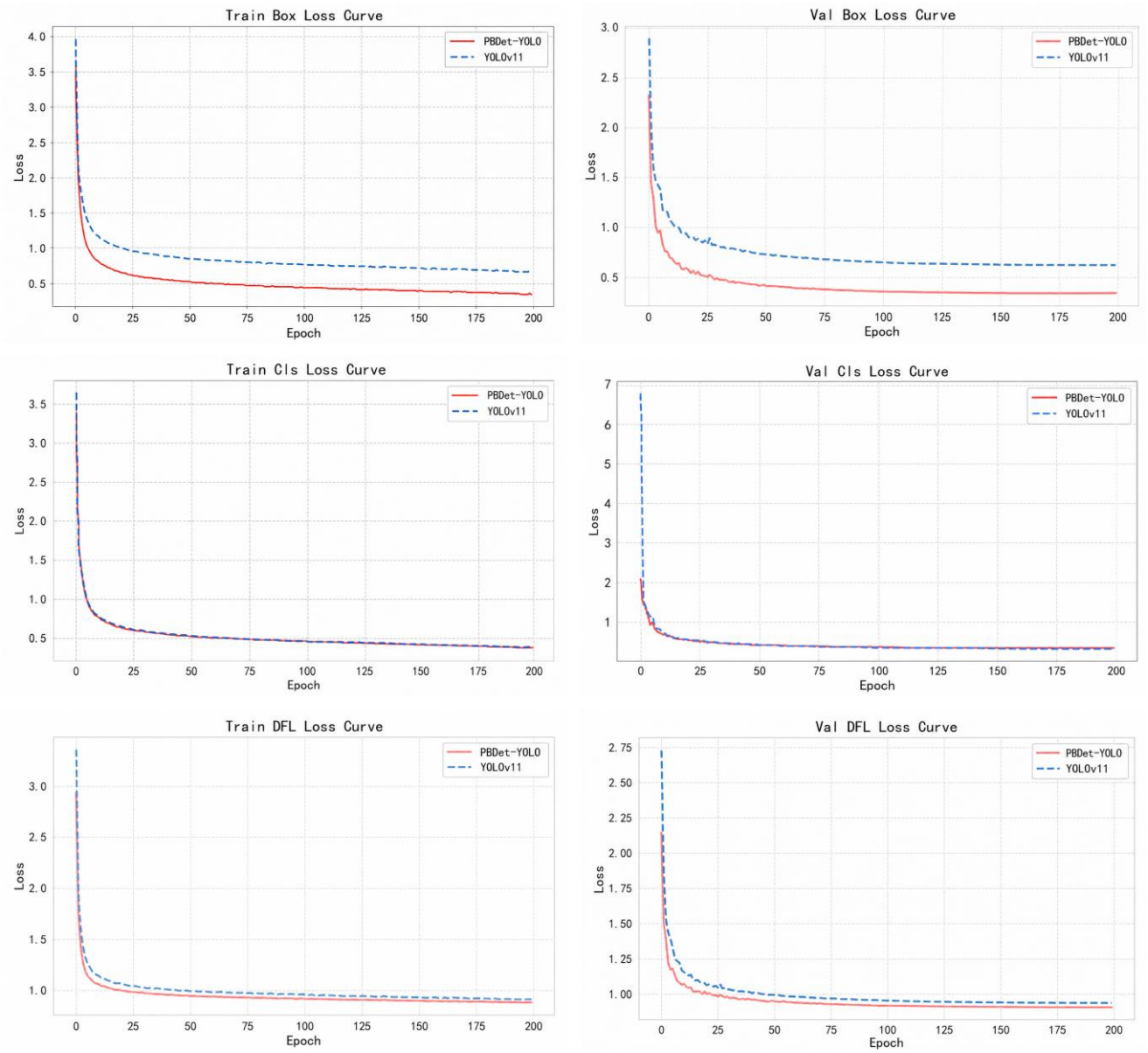


Fig. 7 Training curves of PBDet-YOLO and YOLO11

Both models show a rapid reduction in loss during the initial training stage, followed by more gradual changes toward the end of training. The box-loss curves are visibly separated, with PBDet-YOLO exhibiting lower plotted values. The classification-loss curves are much closer over most of the schedule, while the DFL curves show a smaller separation than the box-loss curves. The training and validation panels present this progression for each objective. The box-loss values refer to each model's respective regression objective; detection performance is compared using the common mAP@0.5 metric in Tables 2 and 3.

Performance comparison among different detectors

PBDet-YOLO is compared with YOLOv5, YOLOv8, YOLO11, and Rt18-DETR. Table 3 reports the parameter count, precision, recall, mAP@0.5, and GFLOPs for the five detectors, and Fig. 8 visualizes mAP@0.5 and recall. The comparison includes the directly modified YOLO11 baseline, other lightweight YOLO configurations, and the higher-complexity Rt18-

DETR configuration. This permits examination of both performance within the lightweight group and the accuracy obtained by the larger model.

Table 3 Comparative experiments with different models.

Model	Parameters (M)	Precision (%)	Recall (%)	mAP@0.5 (%)	GFLOPs
Rt18-DETR	20.18	94.4	89.2	93.2	58.6
YOLOv5	2.50	91.0	76.4	82.7	7.1
YOLOv8	3.00	92.6	76.5	83.1	8.1
YOLO11	2.59	91.0	77.4	84.0	6.4
PBDet-YOLO	2.51	90.7	78.5	85.9	6.2

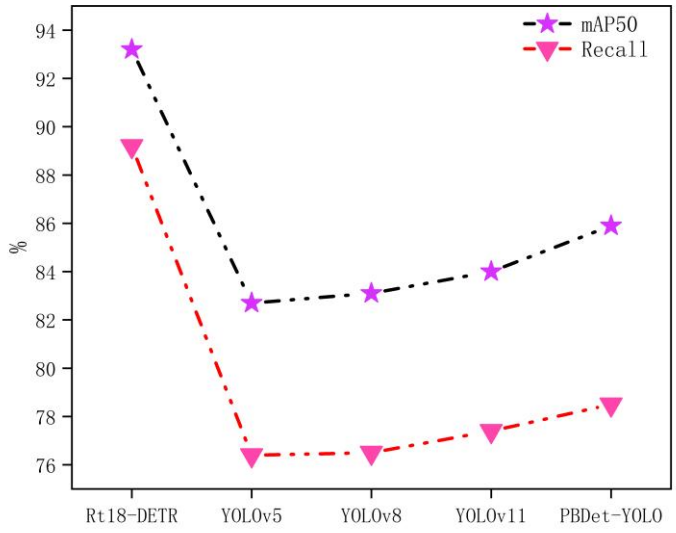


Fig. 8 Performance comparison among different detectors

Among the lightweight YOLO models in Table 3, PBDet-YOLO obtains the highest mAP@0.5 at 85.9%. The increases over YOLOv5, YOLOv8, and YOLO11 are 3.2, 2.8, and 1.9 percentage points, respectively. PBDet-YOLO also has the highest recall within this group at 78.5%. Its 2.51 million parameters are close to the 2.50 million of YOLOv5 and below the 3.00 million of YOLOv8 and 2.59 million of YOLO11. Its computational requirement of 6.2 GFLOPs is the lowest among these YOLO configurations. Precision is 90.7%, compared with 91.0%, 92.6%, and 91.0% for YOLOv5, YOLOv8, and YOLO11, respectively. The resulting performance profile combines the highest mAP@0.5 and recall within the evaluated lightweight YOLO group with the lowest GFLOPs.

Rt18-DETR achieves the highest absolute detection performance in Table 3, with 94.4% precision, 89.2% recall, and 93.2% mAP@0.5. It also has the largest parameter count and computational requirement: 20.18 million parameters and 58.6 GFLOPs. Its mAP@0.5 exceeds PBDet-YOLO by 7.3 percentage points, while its parameter count and GFLOPs are approximately 8.0 and 9.5 times those of PBDet-YOLO. These results identify different

operating points in terms of accuracy and model complexity. PBDet-YOLO improves mAP@0.5 relative to the evaluated lightweight YOLO models, whereas Rt18-DETR offers higher accuracy with a substantially larger computational footprint. This comparison places the proposed detector's contribution within its intended low-complexity design objective.

Analysis of the detection results

Figure 9 presents three match scenes and the corresponding outputs of PBDet-YOLO, YOLO11, YOLOv8, and YOLOv5.

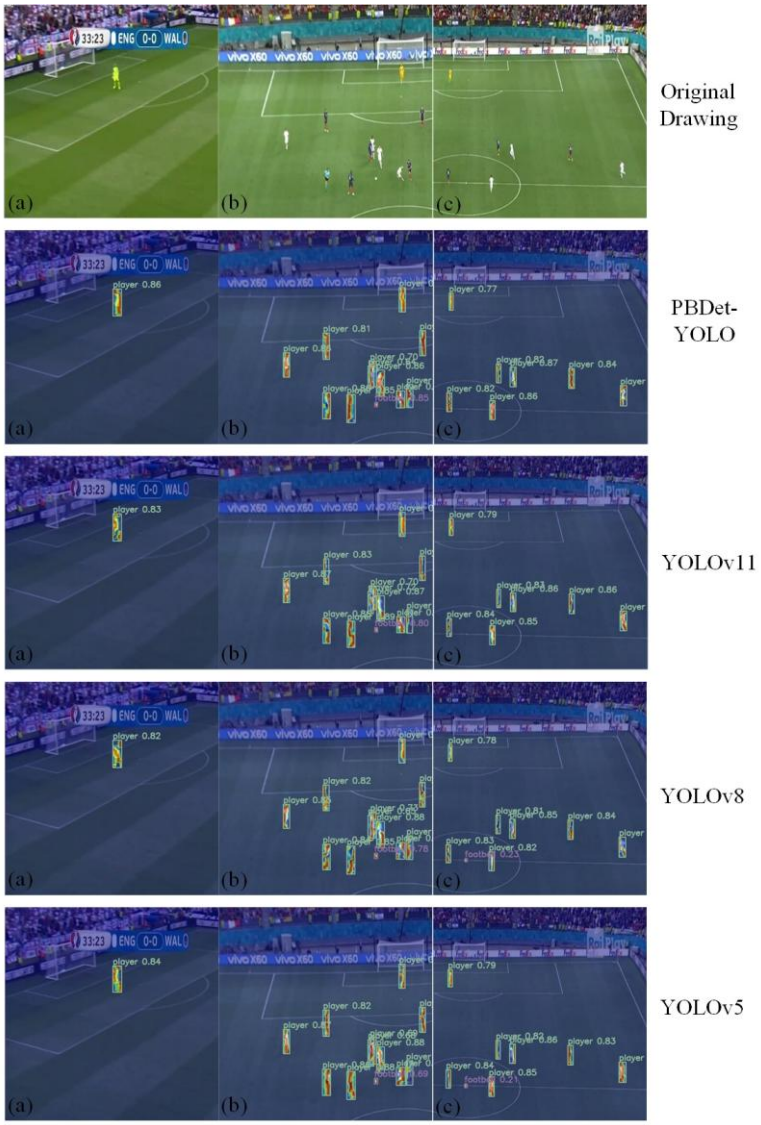


Fig. 9 Qualitative detection results from different detection models

Panel (a) contains an isolated goalkeeper near the goal. The displayed player-confidence scores are 0.86 for PBDet-YOLO, 0.83 for YOLO11, 0.82 for YOLOv8, and 0.84 for YOLOv5. Panel (b) contains multiple players in and around the penalty area, while panel (c) shows players distributed more widely across the field. The bounding boxes and category labels allow the outputs for each scene to be inspected alongside the input image. These examples illustrate the appearance of the detections under different player distributions and viewing scales. Figure 9 illustrates representative predictions, while Table 3 summarizes quantitative performance across the evaluated detectors.

5 Discussion

The central result of this study is an improvement in joint player-and-ball detection within the computational range of the lightweight YOLO11 baseline. PBDet-YOLO increases mAP@0.5 from 84.0% to 85.9%, while reducing the parameter count from 2.59 million to 2.51 million and GFLOPs from 6.4 to 6.2. These results answer the research question through the complete configuration rather than through any single component. The detector combines multi-scale feature processing, a shared prediction head, and an interval-based regression objective. The value of this combination lies in the measured balance between detection performance and model complexity on the soccer dataset.

The ablation study clarifies the different contributions of the three components to the accuracy–complexity balance. CSP_FA improves the reported detection metrics with additional computational cost, while LE-Head reduces the model's parameter count and GFLOPs in the corresponding ablation configuration. Combining the two retains the multi-scale feature-processing design with fewer GFLOPs than CSP_FA alone. The Focaler-CIoU configuration further increases mAP@0.5 without changing the reported parameter count or GFLOPs. These results distinguish the roles of feature aggregation, shared head processing, and regression-loss optimization in the complete detector.

The detection metrics also reveal distinct preferences among the tested configurations. Model C produces the highest precision, Model D the highest recall, and Model E the highest mAP@0.5. In particular, replacing CIoU with Focaler-CIoU raises precision and mAP@0.5 but lowers recall relative to Model D. The selection of the complete model follows its highest mAP@0.5 within the ablation study and its lower parameter count and GFLOPs relative to the baseline. Reporting precision and recall alongside mAP@0.5 makes this trade-off explicit. The comparison with Rt18-DETR further separates a low-complexity objective from maximum absolute accuracy: the larger detector obtains stronger detection metrics, whereas PBDet-YOLO remains within the parameter range of the lightweight YOLO alternatives.

For soccer analysis, image-level detections provide object locations for subsequent video-processing tasks. PBDet-YOLO produces player-and-ball predictions in a common framework, and the three scenes in Fig. 9 illustrate its outputs under different player distributions. Existing work has examined player tracking, full-field data correction, and possession-sequence analysis [11,15,31]. These applications establish a context for obtaining object locations from match imagery.

Future work can extend PBDet-YOLO from image-level detection to player and ball tracking, multi-view information fusion, and posture analysis. Tracking would associate detections across successive frames to obtain trajectories, while posture analysis would characterize player configurations and movements. Additional match footage and external datasets would extend the evaluation of detection accuracy, and measurements on intended deployment hardware would characterize runtime performance. These directions connect the detector with broader soccer video analysis tasks.

Together, the ablation results and detector comparisons provide a quantitative reference for the accuracy–complexity trade-offs of the proposed soccer-specific configuration. They establish a basis for evaluating future extensions of player-and-ball detection systems.

6 Conclusions

This study presented PBDet-YOLO, a YOLO11-based framework for player-and-ball detection in soccer match images. The framework combines CSP_FA for multi-scale feature processing, LE-Head for shared prediction-head processing, and Focaler-CIoU for bounding-box regression. On the self-constructed dataset of 11800 annotated images, PBDet-YOLO achieved 90.7% precision, 78.5% recall, and 85.9% mAP@0.5 with 2.51 million parameters and 6.2 GFLOPs. Relative to YOLO11, mAP@0.5 increased by 1.9 percentage points while parameters and GFLOPs each decreased by approximately 3.1%. The ablation study showed that CSP_FA improved the reported detection metrics with additional computational cost, LE-Head reduced model complexity, and Focaler-CIoU increased mAP@0.5 in the combined configuration. Among the lightweight YOLO models evaluated, the complete detector achieved the highest mAP@0.5 and recall and the lowest GFLOPs, establishing the accuracy–complexity trade-off targeted by the study.

Future work will extend the evaluation to external datasets and continuous video, examine small-ball detection under severe occlusion, and measure performance on deployment hardware. Integration with player tracking and tactical-analysis systems will build on the image-level detection framework established here.

Statements and Declarations

Funding

This work was supported by the Science Foundation of Ministry of Education of China (Grant No. 2510294350) and the Natural Science Foundation of Huzhou (Grant No. 2022GZ36).

Competing Interests

The authors declare no competing interests.

Data Availability

The PBDet-Soccer dataset used in this study is publicly available on Zenodo at https://doi.org/10.5281/zenodo.21821917.

Consent to Publish declaration

not applicable.

Ethics and Consent to Participate declarations

not applicable.

Author Contributions

J.W. and L.G. contributed to the conceptualization and methodology of the study. L.G. provided resources and acquired funding. X.D. was responsible for data curation. X.Y. and J.W. developed the software. X.D. and L.G. conducted the formal analysis and investigation. J.W. and L.G. supervised the research. L.G. and X.Y. performed validation. X.Y. prepared the visualizations. X.Y. and J.W. prepared the original manuscript, managed the project, and contributed to manuscript review and editing. All authors read and approved the final manuscript.

References

1. Pifer, N.D., Wang, Y., Scremin, G., Pitts, B.G., Zhang, J.J.: Contemporary global football industry: An introduction. In: The Global Football Industry. Routledge (2018). ISBN 978-1-351-11798-2
2. Naik, B.T., Hashmi, M.F., Bokde, N.D.: A comprehensive review of computer vision in sports: Open issues, future trends and research directions. Applied Sciences 12, 4429 (2022). https://doi.org/10.3390/app12094429
3. Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11966–11976 (2022). https://doi.org/10.1109/CVPR52688.2022.01167
4. Hosain, M.T., Zaman, A., Abir, M.R., Akter, S., Mursalin, S., Khan, S.S.: Synchronizing object detection: Applications, advancements and existing challenges. IEEE Access 12, 54129–54167 (2024). https://doi.org/10.1109/ACCESS.2024.3388889
5. Tong, K., Wu, Y.: I-YOLO: A novel single-stage framework for small object detection. The Visual Computer 40, 8927–8944 (2024). https://doi.org/10.1007/s00371-024-03284-8
6. Zou, Z., Chen, K., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. Proceedings of the IEEE 111(3), 257–276 (2023). https://doi.org/10.1109/JPROC.2023.3238524
7. Li, Z., Peng, C., Yu, G., Zhang, X., Deng, Y., Sun, J.: Light-head R-CNN: In defense of two-stage object detector. arXiv preprint arXiv:1711.07264 (2017). https://doi.org/10.48550/arXiv.1711.07264
8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
9. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020, pp. 213–229. Springer, Cham (2020) https://doi.org/10.1007/978-3-030-58452-8_13
10. 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, pp. 16965–16974 (2024). https://doi.org/10.1109/CVPR52733.2024.01605
11. Yang, C., Yang, M., Li, H., Jiang, L., Suo, X., Li, Z., et al.: Soccer player tracking and data correction based on attention with full-field videos. The Visual Computer 40, 9141–9153 (2024). https://doi.org/10.1007/s00371-024-03300-x
12. Kikuchi, T., Umeda, K., Ueda, R., Jitsukawa, Y., Osumi, H., Arai, T.: Improvement of color recognition using colored objects. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005: Robot Soccer World Cup IX, pp. 537–544. Springer, Berlin (2006) https://doi.org/10.1007/11780519_51
13. Hanek, R., Schmitt, T., Buck, S., Beetz, M.: Toward RoboCup without color labeling. AI Magazine 24, 47–50 (2003). https://doi.org/10.1609/aimag.v24i2.1703
14. Baysal, S., Duygulu, P.: Sentioscope: A soccer player tracking system using model field particles. IEEE Transactions on Circuits and Systems for Video Technology 26, 1350–1362 (2016). https://doi.org/10.1109/TCSVT.2015.2455713

15. Manafifard, M., Ebadi, H., Abrishami Moghaddam, H.: A survey on player tracking in soccer videos. Computer Vision and Image Understanding 159, 19–46 (2017). https://doi.org/10.1016/j.cviu.2017.02.002
16. Ribeiro, P.R.A., Lopes, G., Ribeiro, F.: Neural network in computer vision for RoboCup middle size league. Journal of Software Engineering and Applications 9, 319–325 (2016). https://doi.org/10.4236/jsea.2016.97022
17. Hurault, S., Ballester, C., Haro, G.: Self-supervised small soccer player detection and tracking. In: Proceedings of the 3rd International Workshop on Multimedia Content Analysis in Sports (2020). https://doi.org/10.1145/3422844.3423054
18. Komorowski, J., Kurzejamski, G., Sarwas, G.: FootAndBall: Integrated player and ball detector. In: Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, pp. 47–56 (2020) https://doi.org/10.5220/0008916000470056
19. Sorano, D., Carrara, F., Cintia, P., Falchi, F., Pappalardo, L.: Automatic pass annotation from soccer video streams based on object detection and LSTM. In: Dong, Y., Ifrim, G., Mladenić, D., Saunders, C., Van Hoecke, S. (eds.) Machine Learning and Knowledge Discovery in Databases: Applied Data Science and Demo Track, pp. 475–490. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-67670-4_29
20. Lu, K., Chen, J., Little, J.J., He, H.: Lightweight convolutional neural networks for player detection and classification. Computer Vision and Image Understanding 172, 77–87 (2018). https://doi.org/10.1016/j.cviu.2018.02.008
21. Zhao, K.: Enhancing the performance and accuracy in real-time football and player detection using upgraded YOLOv5 architecture. International Journal of Computational Intelligence Systems 17, 163 (2024). https://doi.org/10.1007/s44196-024-00565-x
22. Khanam, R., Hussain, M.: YOLOv11: An overview of the key architectural enhancements. arXiv preprint arXiv:2410.17725 (2024). https://doi.org/10.48550/arXiv.2410.17725
23. Chen, J., Kao, S., He, H., Zhuo, W., Wen, S., Lee, C.H., et al.: Run, don't walk: Chasing higher FLOPS for faster neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12021–12031 (2023) https://doi.org/10.1109/CVPR52729.2023.01157
24. Yuan, X., Xu, Q., Wang, T., Gao, L., Zhao, G., Jia, L., et al.: RiceDetect-Net: A lightweight real-time detection framework for rice diseases. BMC Plant Biology 26, 555 (2026). https://doi.org/10.1186/s12870-026-08341-z
25. Huan, M., Wang, T., Xu, Q., Gao, L., Lu, Z., Shi, X., et al.: An efficient lightweight object detection algorithm for defect identification in Paeoniae Radix Alba with rapid area and diameter estimation using OpenCV. Food Science & Nutrition 13, e70947 (2025). https://doi.org/10.1002/fsn3.70947
26. Zhang, H., Zhang, S.: Focaler-IoU: More focused intersection over union loss. arXiv preprint arXiv:2401.10525 (2024). https://doi.org/10.48550/arXiv.2401.10525
27. Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: GhostNet: More features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1580–1589 (2020) https://doi.org/10.1109/CVPR42600.2020.00165

28. Sohan, M., Sai Ram, T., Rami Reddy, C.V.: A review on YOLOv8 and its advancements. In: Jacob, I.J., Piramuthu, S., Falkowski-Gilski, P. (eds.) Data Intelligence and Cognitive Informatics, pp. 529–545. Springer Nature, Singapore (2024) https://doi.org/10.1007/978-981-99-7962-2_39
29. Lyu, C., Zhang, W., Huang, H., Zhou, Y., Wang, Y., Liu, Y., et al.: RTMDet: An empirical study of designing real-time object detectors. arXiv preprint arXiv:2212.07784 (2022). https://doi.org/10.48550/arXiv.2212.07784
30. 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 Transactions on Cybernetics 52(8), 8574–8586 (2022). https://doi.org/10.1109/TCYB.2021.3095305
31. Athanesious, J.J., Kiruthika, S.: Perspective transform based YOLO with weighted intersect fusion for forecasting the possession sequence of the live football game. IEEE Access 12, 75542–75558 (2024). https://doi.org/10.1109/ACCESS.2024.3402370