A Drone Image Object Detection Algorithm Based on MSA-YOLOv10

Yingbiao Duan
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: 2523934620@qq.com

Bailin Liu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: 498194312@qq.com

***Abstract*—To address the problems of frequent missed detections and false detections caused by the small object scale, dense distribution, complex backgrounds, and susceptibility to occlusion in UAV aerial images, this.paper proposes MSA-YOLOv10, an improved object detection algorithm based on YOLOv10n. First, a Triple Feature Encoder (TFE) is introduced into the neck to enhance the representation of ulti-scale detail information. Second, a Scale Sequence Feature Fusion (ScalSeq) module is combined with a Channel and Position Attention Mechanism (CPAM) to improve multi-scale feature fusion and suppress background interference. Finally, a high-resolution P2 detection branch is introduced into the detection head and integrated with a Localization Quality Estimation (LQE) mechanism to enhance small-object detection and prediction-box ranking. Experiments on the VisDrone2019 dataset show that, compared with the original YOLOv10n, MSA-YOLOv10 improves Precision, Recall, and mAP50 by 1.0, 0.7, and 1.1 percentage points, respectively, with mAP50 increasing from 35.1% to 36.2%, while the number of parameters increases only from 2.27 M to 2.37 M. These results indicate that MSA-YOLOv10 has promising application value for object detection in UAV aerial imagery.**

Keywords-UAV Aerial Imagery; Object Detection; Yolov10n; Multi-Scale Feature Fusion; Localization Quality Estimation; Small-Object Detection

I. INTRODUCTION

In recent years, UAV technology has developed rapidly and has been widely applied in many fields. Owing to their high resolution and flexibility, low-altitude UAV remote-sensing images, when combined with three-dimensional visualization techniques, can be used in post-earthquake planning and reconstruction design [1]. However, due to the particular characteristics of UAV imaging platforms, images often contain a large number of extremely small objects that carry very limited visual information and are highly susceptible to background noise. In addition, the semantic definitions of labels from different categories are often similar, and their appearances are highly alike, making effective feature extraction difficult. Meanwhile, object distribution in images is highly uneven and typically concentrated in small local regions such as roads and squares, which leads to dense occlusion. Small-object recognition usually requires high-resolution images, but this introduces considerable computational cost. Because embedded UAV chips have limited computing power, it is difficult to achieve real-time, low-latency detection while maintaining accuracy. Therefore, designing an efficient object detection algorithm specifically for UAV imagery is of great significance for promoting the development of this field and for providing practical solutions to real-world detection problems [2].

Object detection has roughly evolved through two stages: the period of traditional object detection and the period of deep-learning-based detection. The rapid development of deep learning has driven major breakthroughs in object detection technology [3]. Existing methods can be broadly divided into two-stage detectors and one-stage detectors [4]. Representative two-stage detectors include R-CNN [5], Fast R-CNN [6], and Faster R-CNN [7]. For

example, Gu Jiaojiao et al. extracted three feature maps from successive convolutional stages of the VGG-16 backbone and concatenated them to form a multi-scale feature map with richer semantic information; they also redesigned the number and sizes of anchor boxes and proposed an improved Faster R-CNN algorithm for infrared ship detection [8]. However, because two-stage detectors generally have more complex architectures, larger numbers of parameters, and higher computational cost, their performance in real-time detection tasks is often unsatisfactory. By contrast, one-stage detectors output the bounding boxes and class information of all objects in a single forward pass and therefore exhibit clear advantages in terms of speed and real-time performance. Representative models include SSD [9] and the YOLO series [10]. Hong-Tae Choi et al. combined attention with feature-map fusion to generate an enhanced feature map block (EMB), thereby improving small-object detection [11]. Yan Zhou upgraded CSPN and PANet using a shortest–longest gradient strategy and a self-attention mechanism to enhance multi-scale object detection in complex scenes, proposed a new model named YOLO-NL, and accelerated inference through re-parameterization [12].

Nevertheless, most existing studies still focus on natural-scene images or ground surveillance videos, and research on the UAV aerial perspective remains insufficient, leaving considerable room for further exploration. On the one hand, aerial scenes are characterized by drastic scale variation, uneven object distribution, and significant class imbalance, all of which pose substantial challenges to classification. On the other hand, when processing high-resolution images, aerial platforms must also balance model computational cost against actual hardware capability to ensure that practical application requirements can be satisfied. To address these issues, Zhong Zhifeng et al. introduced a multi-scale dilated convolution residual mechanism into the backbone to improve the convolution modules and architecture, and reconstructed the neck by adding an extra P2 detection layer, thereby proposing AC-YOLO, an improved target detection algorithm based on YOLOv8s [13]. Quanyu Zhang et al. used bi-level routing attention (BRA) during feature extraction to reduce background interference and added a high-resolution small-target detection layer (STDL), proposing the BRA-YOLOv10 method for UAV detection [14]. Liu Yansong et al. introduced a wavelet-enhanced multi-branch fusion module (MANet) in the feature extraction stage, designed a FourierSPPF module in the backbone, and adopted the SDIoU regression strategy in the loss function, thereby proposing MAFS-YOLO, an enhanced detection model based on YOLO11 [15].

In view of the above research progress and the challenges posed by UAV-view imagery, this paper proposes an improved UAV object detection algorithm, termed MSA-YOLOv10. The main contributions of this study are as follows:

1) A Triple Feature Encoder (TFE) is introduced into the neck, together with 1×1 convolution for cross-layer channel alignment, to enhance detail interaction among features at different scales and improve small-object representation.

2) ScalSeq is combined with a Channel and Position Attention Mechanism (CPAM) in the neck to perform sequential fusion and adaptive weighting of multi-level features, thereby enhancing the representation of key target regions, suppressing complex background interference, and improving multi-scale feature fusion.

3) A high-resolution P2 feature layer is introduced into the detection head to expand the detection scale range, and an LQE (Localization Quality Estimation) mechanism is incorporated to jointly model classification confidence and localization quality, thereby improving small-object detection and prediction-box ranking.

II. YOLOv10 Network Architecture

The YOLO (You Only Look Once) series is one of the most widely used one-stage object detection frameworks in the current field of object detection. These models unify object localization and category recognition within a single network and directly output object positions and class information through one forward pass, thereby balancing detection accuracy and inference speed.

On this basis, Wang et al. proposed YOLOv10 in 2024. Targeting end-to-end detector deployment, this model further optimizes both network design and post-processing, significantly improving efficiency in real-time object detection tasks. Unlike earlier detectors that rely on non-maximum suppression (NMS) to remove redundant boxes, YOLOv10 realizes NMS-free end-to-end detection through a consistent dual label assignment strategy, offering a new path for efficient deployment in real-time scenarios [16].

In terms of architecture, the YOLOv10 backbone inherits the staged feature extraction paradigm of CSPNet, improving gradient propagation and reducing redundant computation through an enhanced CSP structure. On this basis, YOLOv10 is jointly designed for efficiency and accuracy in several ways. First, it introduces Spatial-Channel Decoupled Downsampling, which first uses pointwise convolution to adjust channels and then applies depthwise convolution for downsampling, thereby preserving feature information as much as possible while reducing computational cost. Second, it proposes a Rank-Guided Block Design based on redundancy analysis, adopting more compact CIB modules in stages with higher redundancy so as to further reduce parameters and FLOPs. In addition, YOLOv10 employs large-kernel convolutions in small-scale models to enlarge the receptive field and introduces a Partial Self-Attention (PSA) module in low-resolution stages to enhance global modeling and feature representation.

The neck is mainly responsible for interaction and fusion across different feature levels. YOLOv10 continues to follow a multi-scale feature aggregation strategy and structurally combines the strengths of FPN [17] and PAN [18] to realize bidirectional transmission between high-level semantic information and low-level detail information. Specifically, FPN enhances the multi-scale representation of high-level semantic features through a top-down pathway and lateral connections, whereas PAN further shortens the information path between shallow localization features and deep semantic features through an additional bottom-up pathway. Benefiting from this type of multi-scale fusion mechanism, YOLOv10 can better balance semantic discrimination for large objects and detailed localization for small objects, thereby improving detection robustness in complex scenes.

The head is another key component that distinguishes YOLOv10 from traditional YOLO detectors. Rather than relying on a single detection branch, YOLOv10 introduces both a one-to-many head and a one-to-one head during training. Meanwhile, YOLOv10 adopts a lightweight classification head that replaces heavier conventional classification structures with depthwise separable convolution, thereby reducing computational cost without significantly degrading classification performance. During inference, only the one-to-one branch is retained to output detection results, so no additional NMS post-processing is required. This substantially reduces end-to-end latency and further improves real-time detection capability.

III. MSA-YOLOv10 Algorithm Architecture

Considering that YOLOv10n still has certain limitations in detecting small, densely distributed, and multi-scale objects in complex UAV aerial scenes, and thus cannot fully satisfy practical requirements for detection accuracy and robustness, this paper selects YOLOv10n as the baseline model. The rationale is that YOLOv10n provides a good balance among detection accuracy, inference speed, and computational cost. Compared with larger models, it has fewer parameters and lower computational complexity, making it more suitable as a backbone for real-time UAV object detection. To further improve feature representation and small-object detection in complex scenes, this paper enhances the original YOLOv10n from the perspectives of feature fusion structure, attention mechanism, and detection head design, and proposes the MSA-YOLOv10 detection network, whose overall architecture is shown in Fig. 1.

First, to address the insufficient cross-layer

interaction caused by the traditional upsampling-plus-Concat fusion strategy in the original YOLOv10n neck, a Triple Feature Encoder (TFE) is introduced and combined with 1×1 convolution for channel alignment across different feature levels, thereby enhancing detailed information fusion among features of different scales. Second, to further improve the extraction and representation of multi-scale target features, a Scale Sequence Feature Fusion (ScalSeq) module and a Channel and Position Attention Mechanism (CPAM) are incorporated into the neck to perform sequential fusion and adaptive weighting of features at different levels, thereby strengthening key information expression and suppressing complex background interference. Finally, to alleviate missed detections and inaccurate localization of small objects in UAV aerial images, a P2 small-object detection branch is added and the original detection head is improved with an LQE (Localization Quality Estimation) mechanism, forming a four-scale detection architecture to enhance the detection performance for small and densely distributed objects.

A. Cross-Scale Detail Enhancement Module Based on TFE

A Triple Feature Encoder (TFE) [19] is introduced into the neck. By simultaneously fusing large-, medium-, and small-scale features, it enhances the network's ability to represent fine-grained information and thereby improves the detection accuracy of densely distributed small objects.

As shown in Fig. 2, the TFE module consists of three branches, corresponding to large-, medium-, and small-scale features. First, the input large, medium, and small features are each processed by convolution blocks for channel adjustment and preliminary feature transformation, so that features of different scales share a consistent representation basis before subsequent fusion. Before feature encoding, the input feature layers are represented as in Eq. (1).

$$\begin{cases} F_l \in \mathrm{R}^{C_l \times H_l \times W_l} \\ F_m \in \mathrm{R}^{C_m \times H_m \times W_m} \\ F_s \in \mathrm{R}^{C_s \times H_s \times W_s} \end{cases} \tag{1}$$

where the three terms denote the large-, medium-, and small-scale feature maps, respectively.

To reduce channel discrepancies among features from different levels, convolutional mapping is used to align the channels of the three branches, as expressed in Eq. (2).

$$\begin{cases} \tilde{F}_l = \phi_l(F_l) \\ \tilde{F}_m = \phi_m(F_m) \\ \tilde{F}_s = \phi_s(F_s) \end{cases} \tag{2}$$

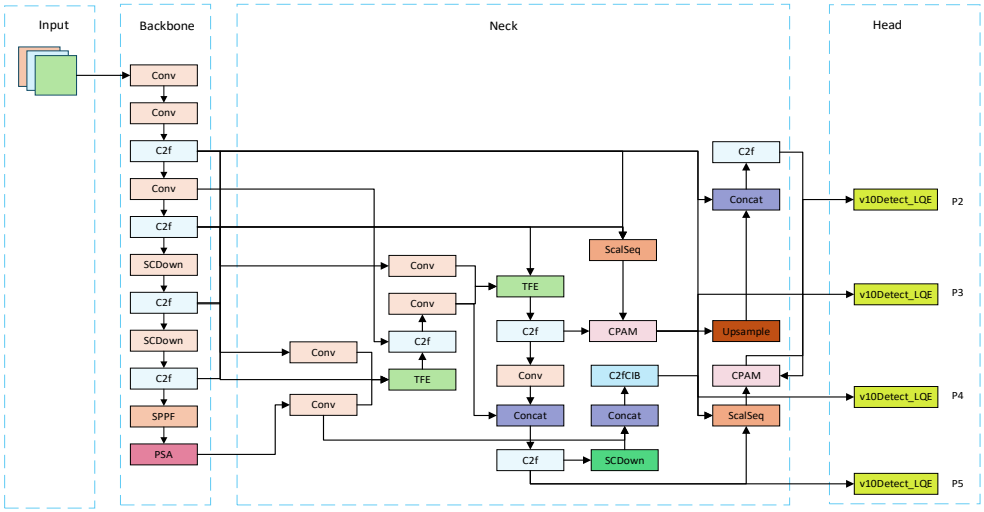


Figure 1. MSA-YOLOv10 architecture

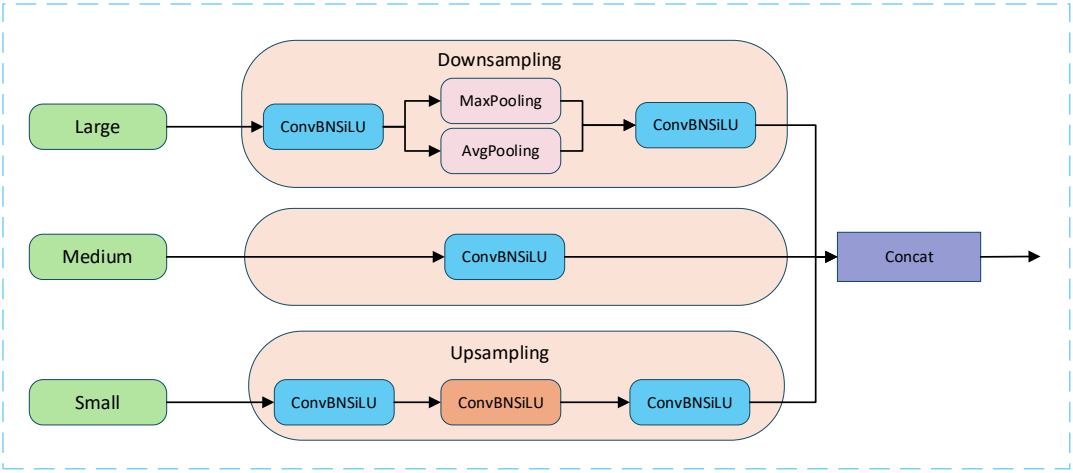


Figure 2. TFE architecture

where the three operators denote channel alignment operations. In Fig. 2, this process corresponds to the first ConvBNSiLU unit after each branch enters the module.

After channel adjustment, the resolution of the medium-scale feature is taken as the target scale. The large-scale branch is downsampled and the small-scale branch is upsampled, thereby achieving spatial alignment across multiple scales. This process is expressed in Eq. (3).

$$\begin{cases} \bar{F}_l = D(\tilde{F}_l; H_m, W_m) \\ \bar{F}_m = \tilde{F}_m \\ \bar{F}_s = U(\tilde{F}_s; H_s, W_s) \end{cases} \tag{3}$$

where the two operators denote downsampling and upsampling, respectively. As shown in Fig. 2, after channel transformation, the large-scale branch is downsampled by max pooling and average pooling and then further refined by convolution blocks to extract detailed features after downsampling. The medium-scale branch serves as the reference scale and is adjusted only through convolution blocks. The small-scale branch, after convolutional mapping, restores spatial resolution through upsampling and further enhances local texture information using subsequent convolution blocks. Through this design, the large-scale branch preserves richer high-resolution details, the medium-scale branch maintains the main structural representation, and the small-scale branch supplements deep semantic information.

Finally, the three scale-aligned features are concatenated along the channel dimension to obtain the output feature of the TFE module, as shown in Eq. (4).

$$F_{TFE} = Concat(\bar{F}_l, \bar{F}_m, \bar{F}_s) \tag{4}$$

where the former term denotes the output feature map of the TFE module and the latter denotes channel-wise concatenation. The Concat module on the right side of Fig. 3 corresponds to this process. The concatenated output maintains the same spatial resolution as the medium-scale branch while integrating complementary information from the large-, medium-, and small-scale branches, thereby jointly preserving shallow details and deep semantics and improving the description of small-object contours, textures, and locally salient regions.

B. Multi-Scale Feature Fusion Module Based on ScalSeq and CPAM

1) ScalSeq Module

To enhance the representation of multi-scale objects, this paper introduces the ScalSeq [19] module into the neck, which jointly models multi-level features in a sequential manner and thus improves the network's adaptability to objects with

varying scales.

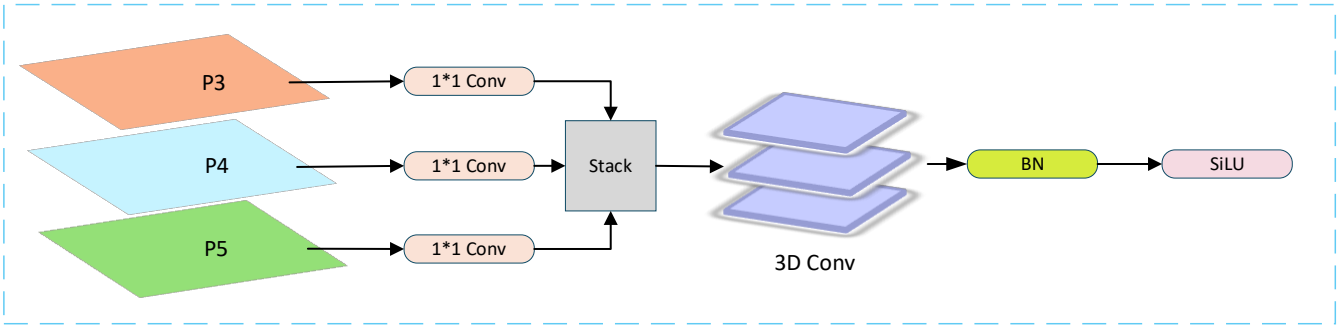


Figure 3. ScalSeq architecture

As shown in Fig. 3, the ScalSeq module takes feature maps from different levels as input. In this study, three feature scales are mainly selected for fusion. The three branches on the left side of the figure correspond to inputs from different levels. Since these features differ in both channel dimension and spatial resolution, they must be unified before sequence fusion. The input features can be represented as in Eq. (5).

$$F_1, F_2, \ldots, F_n \tag{5}$$

First, 1×1 convolution is used to map the channels of each feature level and reduce dimensional discrepancies among different levels. Then, scale alignment is performed to resize feature maps of different resolutions to a common size, allowing subsequent fusion to be carried out within a unified representation space. This process is described by Eqs. (6) and (7).

$$\tilde{F}_i = \phi_i(F_i), i = 1, 2, \ldots, n \tag{6}$$

$$\bar{F}_i = \mathbf{R}(\tilde{F}_i; H_t, W_t), i = 1, 2, \ldots n \tag{7}$$

where the first operator denotes channel mapping, the second denotes scale alignment, and the final term denotes the unified target resolution. In Fig. 3, the 1×1 Conv module following each input branch corresponds to this preprocessing step.

After channel transformation and scale alignment, the processed multi-level features are stacked sequentially along the scale dimension to form a scale sequence, as shown in Eq. (8).

$$S = Stack(\bar{F}_1, F_2, \ldots, \bar{F}_n) \tag{8}$$

where the operator denotes stacking along the scale dimension. The Stack block in the figure corresponds to this process. Essentially, it organizes features from different scales into an ordered feature sequence, so that subsequent modeling can explicitly capture relationships from the perspective of scale variation.

Subsequently, three-dimensional convolution is used to extract correlated information jointly along the scale and spatial dimensions, yielding the fused sequence feature, as expressed in Eq. (9).

$$F_{\text{seq}} = \delta\left(\text{BN}\left(\text{Conv3D}\left(S\right)\right)\right) \tag{9}$$

where the three operators denote 3D convolution, normalization, and the activation function, respectively. To be consistent with the structure shown in the figure, the activation function can be SiLU. The sequence from 3D Conv to BN and then to SiLU in Fig. 3 corresponds to the computation in Eq. (9). Through this design, the

model can learn the intrinsic relationships among features from different levels from the perspective of scale sequences, thereby enhancing multi-scale semantic representation.

2) CPAM Module

After multi-scale sequence features have been extracted, directly feeding the fused result into the detection head may still leave the model vulnerable to complex backgrounds and redundant responses. In UAV aerial imagery, small objects are often interwoven with background information such as road textures, building edges, and vegetation shadows. Therefore, relying solely on convolutional fusion results makes it difficult for the network to consistently focus on truly discriminative target regions. To further strengthen important channel responses and highlight spatially relevant target locations, a CPAM module (Channel and Position Attention Mechanism) is introduced after sequence fusion to jointly enhance the feature representations from the detail branch and the multi-scale branch.

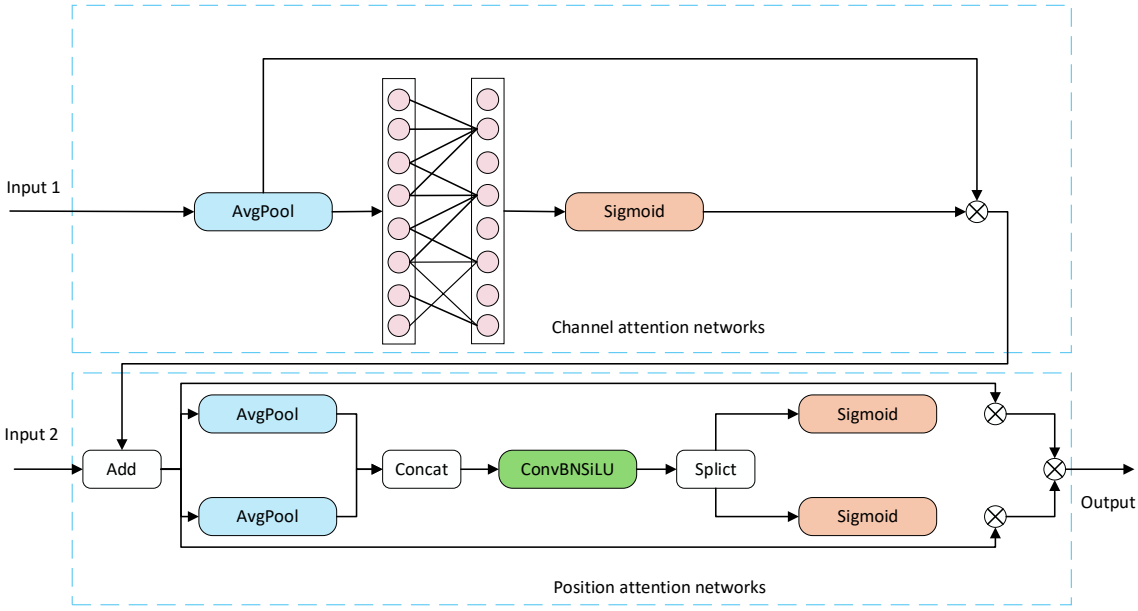


Figure 4. CPAM architecture

As shown in Fig. 5, CPAM consists of two branches: an upper channel-attention branch and a lower position-attention branch. Here, Input 1 denotes the detail-enhanced features from the TFE branch, and Input 2 denotes the multi-scale sequential features from the ScalSeq branch. Rather than weighting the two inputs independently, the module first enhances the TFE-branch feature through channel attention, then fuses it with the ScalSeq-branch feature, and finally uses position attention to further strengthen the spatial response of target regions.

First, in the channel-attention branch, global average pooling is applied to the input feature, and a one-dimensional convolution is used to model local cross-channel dependencies. The resulting channel weights are generated through a Sigmoid function, yielding the channel-enhanced feature as shown in Eq. (10).

$$F_c = M_c\left(F_{\mathrm{TFE}}\right) \tag{10}$$

where the term denotes the channel-attention map. The process in the upper part of the figure, namely "AvgPool → Conv1D → Sigmoid" followed by multiplication with the input feature, exactly corresponds to the computation in Eq. (10).

To enable the receptive range of the one-dimensional convolution to adapt to the channel

scale, the relationship between the convolution kernel size and the channel number is defined as in Eq. (11).

$$k = \left| \frac{\log_2(C) + b}{\gamma} \right|_{\text{odd}} \tag{11}$$

where the operator denotes rounding to the nearest odd number, and the remaining terms are proportional control parameters. This formula corresponds to the Conv1D setting in the channel-attention branch, allowing the range of local cross-channel interaction to be adjusted automatically according to the channel scale.

After the channel-enhanced feature is obtained, it is added element-wise to the sequential fusion feature from the ScalSeq branch to produce the fused feature, as expressed in Eq. (12).

$$F_f = F_c + F_{\text{seq}} \tag{12}$$

The Add block in Fig. 4 represents this process. Through this operation, detail features and multi-scale semantic features are jointly represented before entering the position-attention branch. Subsequently, in the position-attention branch, average pooling is performed along the width and height directions of the fused feature, respectively, so as to preserve the spatial structural information of the target region along the two directions. Let the spatial size of the fused feature be H × W. Then, the two directional descriptors can be expressed by Eqs. (13) and (14), respectively.

$$p_w(i) = \frac{1}{H} \sum_{j=1}^{H} F_f(i, j) \tag{13}$$

$$p_h(j) = \frac{1}{W} \sum_{i=1}^{W} F_f(i, j) \tag{14}$$

where the two terms denote the positional descriptors aggregated along the horizontal and vertical directions, respectively. The two AvgPool branches in the lower part of the figure and the adjacent labels correspond to this process.

Next, the two directional descriptor vectors are concatenated and jointly encoded through convolution, normalization, and activation operations to obtain the positional attention coordinates, as shown in Eq. (15).

$$P(a_w, a_h) = \text{Conv}\left[\text{Concat}(p_w, p_h)\right] \tag{15}$$

After joint encoding, a split operation is performed to obtain the positional weights for the two directions, as expressed in Eq. (16).

$$\begin{cases} s_w = \text{Split}(a_w) \\ s_h = \text{Split}(a_h) \end{cases} \tag{16}$$

Finally, the two directional positional weights are applied to the fused feature, producing the output of CPAM, as shown in Eq. (17).

$$F_{\text{CPAM}} = F_f \otimes s_w \otimes s_h \tag{17}$$

where the operator denotes element-wise multiplication. In the lower part of the figure, the outputs of the two Sigmoid branches are respectively multiplied with the feature and are then aggregated into the final output. Through this process, CPAM first strengthens more meaningful semantic responses in the channel dimension and then highlights the spatial regions that are truly relevant to the target, thereby effectively suppressing irrelevant interference introduced by complex backgrounds.

C. Improved P2-LQE Detection Head

UAV aerial images are characterized by a high viewpoint, small target scale, dense object distribution, and complex backgrounds, and targets such as vehicles and pedestrians often occupy only a few pixels. As the network is repeatedly downsampled, edge, texture, and positional

information in shallow features can be weakened during deep semantic extraction, which leads to missed detections, localization bias, and unreasonable confidence ranking in small-object detection. On the basis of the original detection layers, this paper further introduces a high-resolution feature layer for prediction. Let the high-level fused feature enhanced by the aforementioned neck be denoted by one term and the shallow high-resolution feature output by the backbone be denoted by another. Then, the construction of the newly added P2 branch is described by Eqs. (18) and (19).

$$\hat{F}_{P2} = \mathrm{Concat}\left(\mathrm{U}\left(F_{P3}\right), F_{P2}\right) \tag{18}$$

$$F_{P2}^{*} = \psi\left(\hat{F}_{P2}\right) \tag{19}$$

where the three operators denote upsampling, channel-wise concatenation, and the subsequent convolutional feature fusion module, respectively. Equation (18) indicates that the enhanced high-level feature is upsampled and fused with the shallow backbone feature, while Eq. (19) describes the final P2 detection feature obtained through feature integration. Because the P2 layer has higher spatial resolution and can preserve more local texture and edge information, it is better suited to describing small and weak targets in UAV aerial images. Ultimately, the detection head is extended from the original three-scale outputs to four-scale outputs, thereby enhancing the network's unified detection capability for objects of different scales.

Although adding a shallow detection branch can alleviate the loss of small-object detail information to some extent, the ranking of candidate boxes in dense scenes still depends heavily on classification scores. In UAV aerial imagery, adjacent objects are often affected by occlusion, overlap, and complex background interference. If candidate boxes are ranked solely according to classification confidence, cases may arise where a box has a high classification score but poor localization quality, which may cause truly high-quality predictions to be wrongly suppressed during non-maximum suppression (NMS). To address this issue, the LQE (Localization Quality Estimation) mechanism [20] is further introduced into the detection head to jointly model the output of the classification branch and the localization quality of the predicted boxes, thereby improving the rationality and stability of prediction ranking.

As shown in Fig. 5, the LQE mechanism mainly consists of a classification branch, a distribution regression branch, and a quality prediction branch based on distribution statistics. The upper part of the figure is the classification branch, whose output denotes the classification confidence for each category. The lower part is the distribution regression branch, whose output denotes the discrete distribution representation of the four sides of the bounding box. The red dashed box in the middle corresponds to the quality prediction branch, which extracts statistical features from the regression distribution and generates a localization quality estimate. This estimate is then multiplied with the output of the classification branch to obtain the final joint representation.

The output of the classification branch is given in Eq. (20).

$$C = \left[C_1, C_2, \ldots, C_m\right], C_i \in \left[0,1\right] \tag{20}$$

where one term denotes the number of categories and another denotes the classification confidence of the i-th category. Let the localization quality estimate output by the quality prediction branch be denoted accordingly. Then, the detection head uses the joint representation of classification and localization quality for prediction scoring, as expressed in Eq. (21).

$$J = C \times I \tag{21}$$

where the term denotes the joint representation used for both training and inference. In the top "Classification" branch of the figure, the classification output is fused at the multiplication node with the localization quality estimate

generated by the quality prediction branch inside the red dashed box, producing the final joint score. This explicitly decoupled-then-fused strategy suppresses candidate boxes with high classification scores but poor localization quality, so that truly well-localized predictions obtain more reasonable comprehensive scores.

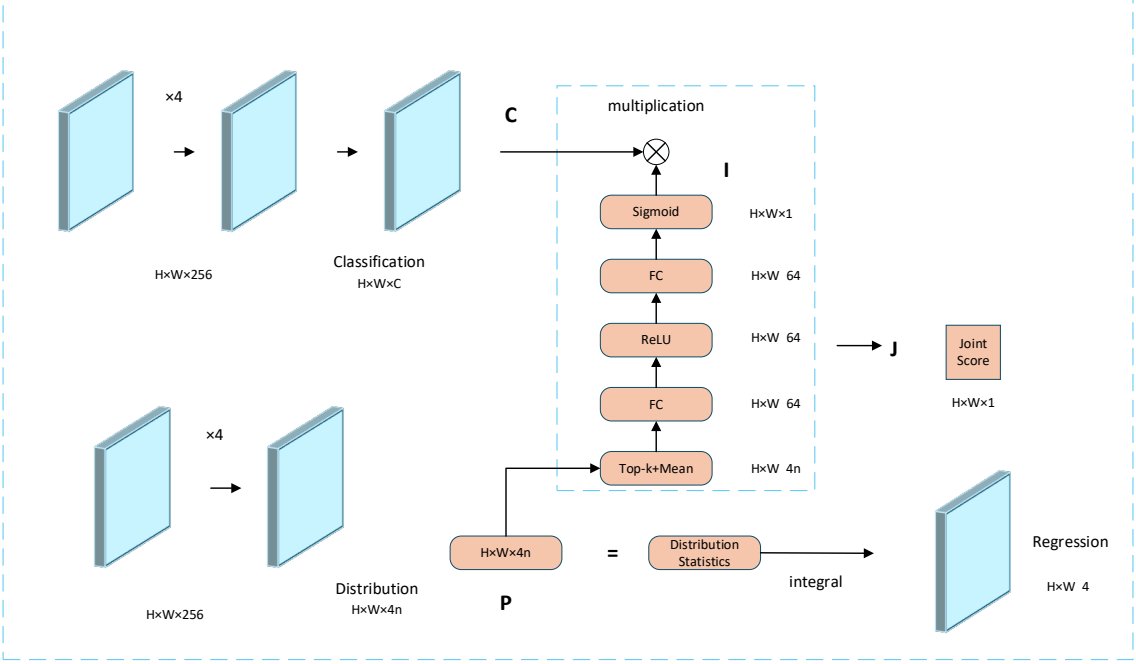


Figure 5. LQE architecture

In the regression branch, the four sides of the target box are represented separately, and their corresponding discrete probability distribution vectors are used to derive the regression results of each boundary through distribution integral, as described by Eqs. (22) and (23).

$$P^w = \left[P^w(y_0), P^w(y_1), \ldots, P^w(y_n) \right] \quad (22)$$

$$\hat{y} = \sum_{i=0}^{n} P(y_i) y_i \quad (23)$$

where the term denotes the boundary prediction value obtained by integrating the discrete distribution. In the lower "Distribution" branch of the figure, the regression result is output after the integral operation, corresponding to the bounding-box regression process in the detection head. Therefore, its output should be interpreted as a regression result rather than a joint score.

Because the concentration of the boundary distribution is closely related to the true localization quality of the predicted box, a sharper distribution usually indicates more accurate localization, whereas a flatter distribution suggests greater uncertainty. Based on this property, the Top-k values from the distributions of the four directions and their means can be extracted to form the statistical features required for localization quality estimation, as expressed in Eq. (24).

$$F = \text{Concat}\left\{ \text{Topk}_m\left(P^w\right) \mid w \in \{l, r, t, b\} \right\} \quad (24)$$

where the first operator denotes extracting the Top-k values from the distribution vector and computing their mean, and the second denotes channel-wise concatenation. The "Top-k + Mean" module at the bottom of the red dashed box in Fig. 5 corresponds to this process and is used to extract

statistical descriptors from the distribution regression results.

Furthermore, the statistical feature is fed into a lightweight fully connected subnetwork to generate the localization quality estimate, as shown in Eq. (25).

$$I = \sigma\left(W_2\, \delta\left(W_1 F\right)\right) \tag{25}$$

where the two functions denote the ReLU and Sigmoid activation functions, respectively, and the remaining symbols denote the parameters of the two fully connected mappings. Corresponding to the process inside the red dashed box, the statistical feature is passed through FC → ReLU → FC → Sigmoid in sequence to generate the localization quality estimate. This branch is lightweight and can adaptively produce a more reliable quality score according to the shape of the bounding-box distribution without significantly increasing inference cost.

IV. Experiments and Results

A. Dataset

The VisDrone2019 dataset [21] was jointly constructed by the Machine Learning and Data Mining Laboratory of Tianjin University and the AISKYEYE team. The data were collected in multiple cities across China using UAVs under different viewpoints and diverse scenes. The dataset contains 10,209 images with a resolution of 2000×1500 pixels and covers various weather conditions and illumination environments, making it one of the most widely used benchmark datasets in UAV aerial image analysis. The dataset defines 10 object categories, including pedestrians, motor vehicles, bicycles, and public facilities, and some targets are divided into finer-grained subclasses. Notably, the dataset contains a large number of small objects, which are typically densely distributed and spatially uneven. In this study, the dataset is divided into a training set, a validation set, and a test set, containing 6,471, 548, and 1,610 images, respectively.

B. Experimental Environment

1) Experimental Configuration

The experimental environment used in this study is listed in Table 1.

TABLE I. Experimental Environment Configuration

Class	Version
Operating System	Windows 10
CPU	Intel Core i7-14650HX
GPU	GeForce RTX 5060
PyTorch	2.9.1
CUDA	12.8
Programming language	Python 3.11

2) Parameter Settings

The experimental parameter settings are listed in Table 2.

TABLE II. Experimental Parameter Settings

Name	Parameter
Epochs	200
Initial learning rate	0.01
Optimizer	SGD
Batch Size	4
Momentum	0.937

3) Evaluation Metrics

To objectively evaluate the detection performance and inference efficiency of the proposed model, this study adopts Precision, Recall, Average Precision (AP), mean Average Precision (mAP), and Frames Per Second (FPS) as evaluation metrics. Their definitions are given in Eqs. (26)–(29).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{26}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{27}$$

where the three quantities denote the numbers of true positives, false positives, and false negatives, respectively. Precision mainly reflects the accuracy

of model predictions, whereas Recall characterizes the model's ability to detect target samples.

For object detection, AP describes the detection accuracy of a single category and is essentially the area under the Precision–Recall curve, whereas mAP is the mean of AP values over all categories and comprehensively reflects overall detection performance. In general, mAP50 denotes the mean Average Precision computed at an Intersection over Union (IoU) threshold of 0.5. The corresponding expressions are given by Eqs. (28) and (29), respectively.

$$AP = \int_0^1 \text{Precision}\left(\text{Recall}\right) d\left(\text{Recall}\right) \quad (28)$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP \quad (29)$$

where one term denotes the total number of target categories in the dataset and the other denotes the average precision of the i-th category.

In addition to detection accuracy, runtime efficiency is also crucial in practical applications. To measure the real-time inference capability of the proposed model, FPS is adopted as the speed evaluation metric. FPS represents the number of image frames processed by the model per unit time; a larger value indicates faster inference and better real-time performance.

C. Ablation Experiments

To systematically evaluate the actual effects of each proposed module and their different combinations on baseline performance, ablation experiments are conducted using YOLOv10n as the base network. By comparing the results of different experimental settings, the contribution of each module to performance improvement, as well as the synergistic effects among modules, can be more clearly analyzed. The corresponding results are presented in Table 3. In the table, "√" indicates that the corresponding module is introduced into the model, whereas "–" indicates that the module is not used. Here, M1, M2, and M3 denote the TFE module, the ScalSeq-CPAM module, and the P2-LQE module, respectively.

TABLE III. RESULTS OF ABLATION EXPERIMENTS

Baseline	M1	M2	M3	P/%	R/%	mAP50%	Params/M	FPS	GFLOPs/G
√	-	-	-	45.6	35.2	35.1	2.27	329.04	6.5
√	√	-	-	46.1	34.8	35.4	2.27	300.52	6.7
√	√	√	-	46.3	34.7	35.4	2.31	333.93	7.0
√	√	√	√	46.6	35.9	36.2	2.37	190.60	10.8

As presented in Table 3, the overall detection performance of the model improves progressively with the sequential introduction of the proposed modules, although this improvement is accompanied by increased model complexity and reduced inference speed. The baseline YOLOv10n achieves a Precision of 45.6%, a Recall of 35.2%, and an mAP50 of 35.1%, with 2.27 M parameters, 6.5 GFLOPs, and an inference speed of 329.04 FPS.

After introducing the TFE module, the Precision and mAP50 increase to 46.1% and 35.4%, respectively, indicating that TFE can enhance multi-scale feature fusion and improve the representation of fine-grained details in small objects, while introducing only marginal computational overhead. When the ScalSeq-CPAM module is further incorporated, the Precision rises to 46.3%, whereas the Recall decreases slightly and the mAP50 remains unchanged. This suggests that ScalSeq-CPAM contributes to improving the discriminative ability of the model to a certain extent, but its overall performance gain is relatively limited.

After further integrating the P2-LQE module,

the model achieves the most significant performance improvement, with Precision, Recall, and mAP50 reaching 46.6%, 35.9%, and 36.2%, respectively. These results demonstrate that the P2-LQE module is more effective in enhancing small-object detection and localization quality. However, this improvement comes at the cost of higher computational burden: the number of parameters and GFLOPs increase to 2.37 M and 10.7, respectively, while the inference speed drops to 190.6 FPS. Overall, P2-LQE serves as the primary contributor to the final performance gain, but it is also the main source of increased model complexity and reduced inference efficiency.

D. Comparative Experiments

To further evaluate the practical effectiveness of the proposed method for UAV aerial object detection, the public VisDrone2019 dataset is used as the experimental platform, and the improved MSA-YOLOv10 is compared with several representative detection algorithms in the field. The corresponding results are presented in Table 4.

As shown by the comparative results in Table 4, the proposed model achieves the best detection accuracy on the VisDrone2019 dataset. Compared with several mainstream detection algorithms, the proposed method obtains an mAP50 of 36.2%, which is higher than those of Faster R-CNN (35.8%), YOLOv8n (35.9%), YOLOv10n (35.1%), and YOLOv5n (32.4%). This demonstrates that the proposed improvements can effectively enhance the ability of the model to detect small and densely distributed objects in UAV aerial scenes.

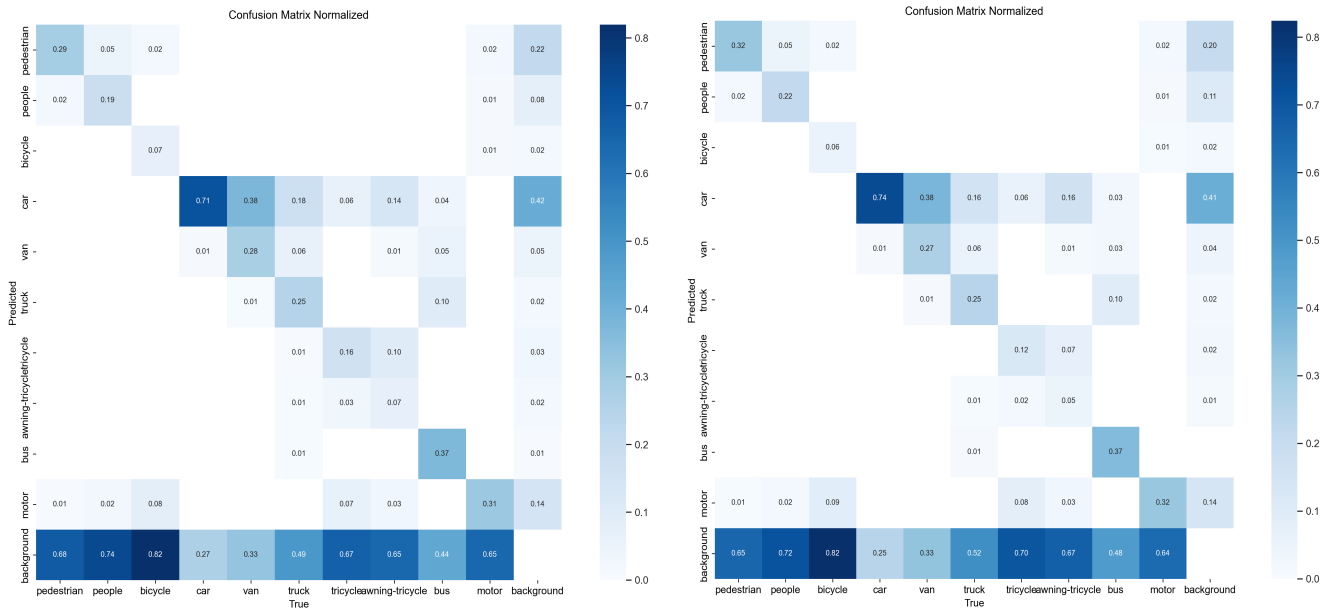
TABLE IV. RESULTS OF COMPARATIVE EXPERIMENTS

Algorithms	mAP50/%	Params/M	GFLOPs/G	FPS
Faster-RCNN	35.8	41.88	204.85	20
YOLOv5n	32.4	1.92	4.9	186
YOLOv8n	35.9	3.01	8.1	230.94
YOLOv10n	35.1	2.27	6.5	329.04
Ours	36.2	2.37	10.8	190.60

Specifically, Faster R-CNN achieves an mAP50 of 35.8%, which is close to that of the proposed model, but it requires as many as 41.88 M parameters and 204.85 GFLOPs, with an inference speed of only 20 FPS. By contrast, the proposed model attains higher detection accuracy while using only 2.37 M parameters and 10.8 GFLOPs, indicating that the proposed method can obtain favorable detection performance while greatly reducing model complexity. Among the one-stage YOLO-series models, YOLOv5n has relatively low parameter count and computational cost, but its mAP50 is only 32.4%. YOLOv8n improves mAP50 to 35.9%, but with increased parameters and computational cost. The original YOLOv10n has clear advantages in lightweight design and real-time performance, with 2.27 M parameters, 6.5 GFLOPs, and 329.04 FPS, but its mAP50 is 35.1%. The proposed model increases mAP50 to 36.2% on the basis of YOLOv10n, improving the baseline by 1.1 percentage points while increasing the parameter count by only 0.10 M, which confirms that the proposed strategy effectively enhances detection accuracy with a very small parameter overhead.

E. Visualization of Experimental Results

To visually demonstrate the detection performance of the improved network, confusion matrices and visualized detection results are compared. Figure 6 shows the confusion matrices of YOLOv10n and MSA-YOLOv10, where the diagonal entries indicate the proportions of correctly detected samples.

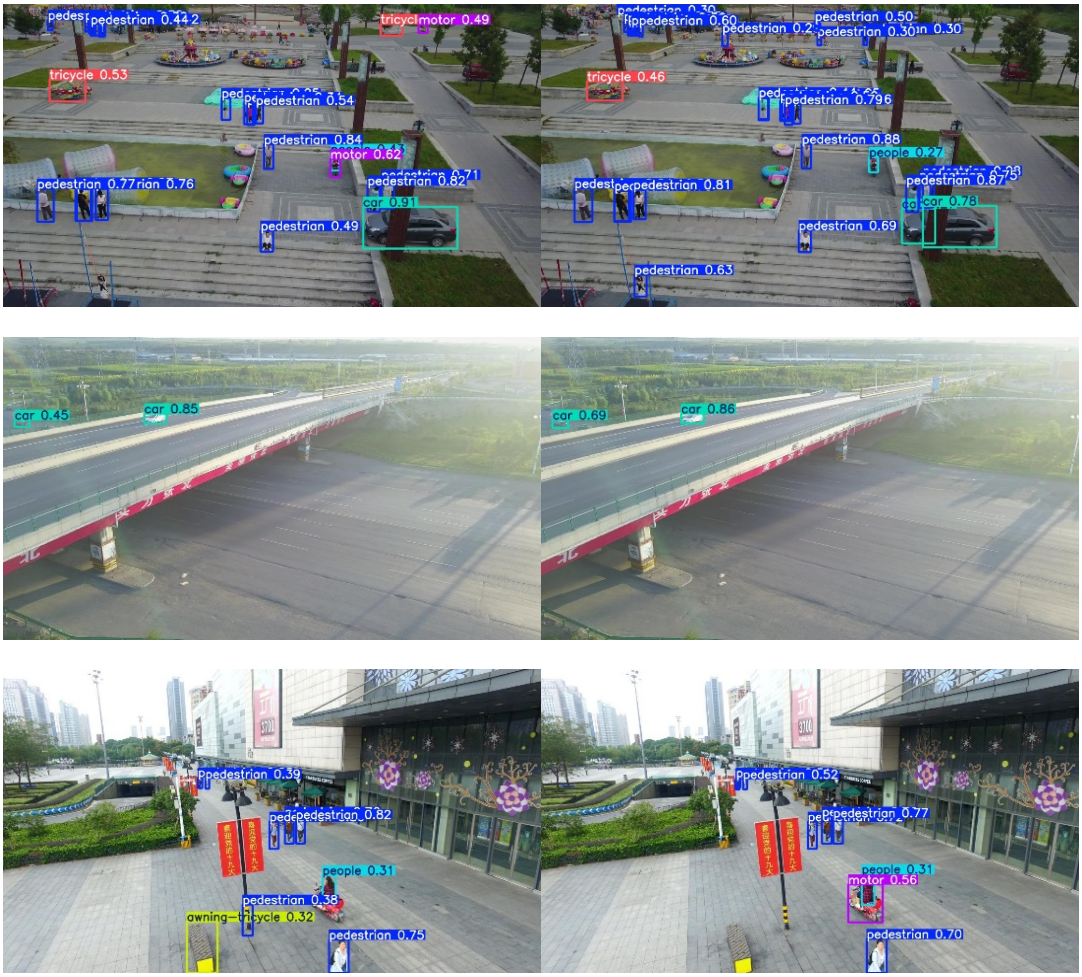


(a) YOLOv10n (b) MSA-YOLOv10

Figure 6. Confusion matrices of YOLOv10n and the improved MSA-YOLOv10 model

The improved model achieves an overall increase in detection accuracy across categories, indicating that appropriately increasing the loss proportion of hard samples during training can, to some extent, suppress false detections. Meanwhile, the number of cases in which MSA-YOLOv10 incorrectly classifies some target categories as background is reduced, showing that the model's ability to discriminate target features has been enhanced. Nevertheless, owing to complex background interference and the small scale of targets, some categories still exhibit relatively high missed-detection rates. To more intuitively demonstrate the effectiveness of the proposed method in complex scenes, three representative groups of densely distributed target images are selected from the VisDrone2019 test set for visualization analysis, and the results are shown in Fig. 7.

In the three groups of detection results shown in Fig. 7, the left column presents the results of the baseline YOLOv10n model, and the right column presents those of the improved MSA-YOLOv10 model. As indicated by the red-marked regions, the baseline model still has certain limitations in complex UAV aerial scenes. It tends to miss distant small objects, densely distributed objects, and partially occluded objects. Moreover, when background textures are complex or targets are close to one another, false detections may also occur. By contrast, the proposed model produces more complete detection results in the above scenarios, effectively reducing both missed detections and false detections, and providing more accurate discrimination of small-object contours and target categories. Especially in regions with small object scales, pronounced occlusion, and strong background interference, the improved model demonstrates stronger target perception capability and more stable detection performance. Taken together, the three groups of comparisons show that the proposed method can more effectively improve detection accuracy for UAV aerial object detection and exhibits better robustness in complex scenes.



(a) YOLOv10n (b) MSA-YOLOv10

Figure 7. Comparison of YOLOv10n and MSA-YOLOv10 for object detection in different scenarios

V. CONCLUSIONS

To address the problems of small target scale, dense object distribution, complex backgrounds, and susceptibility to occlusion in UAV aerial images, this paper takes YOLOv10n as the baseline and proposes an improved object detection method, MSA-YOLOv10. The proposed method improves the model from two main aspects—multi-scale feature representation and detection head optimization—with the aim of enhancing the detection capability for small and densely distributed objects. First, a Triple Feature Encoder (TFE) module is introduced into the neck to jointly fuse large-, medium-, and small-scale features, thereby enhancing the network's representation of fine-grained information and alleviating the deficiency of conventional feature pyramids in representing small-object details. Second, a ScalSeq sequential multi-scale feature fusion module is introduced and combined with CPAM (Channel and Position Attention Mechanism) to enhance fused features in both channel and spatial-position dimensions, thereby strengthening multi-scale semantic modeling and effectively suppressing irrelevant responses in complex backgrounds. Finally, a high-resolution P2 detection branch and an LQE (Localization Quality Estimation) mechanism are introduced into the detection head. By expanding the detection scale range and jointly modeling classification confidence and localization quality, the proposed model further improves small-object detection and the rationality of prediction-box ranking.

Experimental results show that the proposed MSA-YOLOv10 achieves better detection performance than the baseline model on the VisDrone2019 dataset. Compared with the original YOLOv10n, the improved model increases mAP50 from 35.1% to 36.2%, Precision from 45.6% to 46.6%, and Recall from 35.2% to 35.9%, demonstrating that the proposed improvement strategy can effectively enhance small-object detection performance in UAV aerial scenes. In terms of model complexity, the parameter count of MSA-YOLOv10 increases only slightly from 2.27 M to 2.37 M, indicating that the improved model remains lightweight. Meanwhile, the computational cost rises from 6.5 GFLOPs to 10.8 GFLOPs, and the inference speed decreases from 329.04 FPS to 190.60 FPS, showing that the gain in accuracy is achieved at the cost of increased computation. Overall, the proposed method effectively improves detection accuracy while maintaining a relatively low parameter scale, which verifies the effectiveness of the designed modules for UAV aerial image object detection.

Comprehensive quantitative and visualization results indicate that MSA-YOLOv10 exhibits better detection performance for distant small objects, densely distributed objects, and partially occluded objects, and that both missed detections and false detections are reduced to some extent. These findings demonstrate that the proposed method has good robustness and applicability in complex scenes. Future work can further focus on lightweight design and inference efficiency optimization, for example by exploring more efficient feature fusion strategies, improving the detection head, or incorporating techniques such as model pruning and knowledge distillation, so as to further reduce computational cost while maintaining detection accuracy and thereby facilitate practical deployment in real-time UAV object detection.

REFERENCES

[1] Lu H, Li Y, Li H, et al. Digital processing of UAV imagery and its application in reconstruction planning after earthquake disasters [J]. Journal of Southwest Jiaotong University, 2010, 45(04): 533-538+573.

[2] Leng J, Mo M, Zhou Y, et al. A survey of object detection from the UAV perspective [J]. Journal of Image and Graphics, 2023, 28(09): 2563-2586.

[3] Zou Z, Chen K, Shi Z, et al. Object detection in 20 years: A survey [J]. Proceedings of the IEEE, 2023, 111(3): 257-276.

[4] Zhao Z Q, Zheng P, Xu S, et al. Object detection with deep learning: A review [J]. IEEE Transactions on Neural Networks and Learning Systems, 2019, 30(11): 3212-3232.

[5] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014: 580-587.

[6] Girshick R. Fast R-CNN [C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1440-1448.

[7] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, 39(6): 1137-1149.

[8] Gu J, Li B, Liu K, et al. Infrared ship target detection algorithm based on improved Faster R-CNN [J]. Infrared Technology, 2021, 43(2): 170-178.

[9] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector [C]//European Conference on Computer Vision. Cham: Springer International Publishing, 2016: 21-37.

[10] Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, real-time object detection [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 779-788.

[11] Choi H T, Lee H J, Kang H, et al. SSD-EMB: An improved SSD using enhanced feature map block for object detection [J]. Sensors, 2021, 21(8): 2842.

[12] Zhou Y. A YOLO-NL object detector for real-time detection [J]. Expert Systems with Applications, 2024, 238: 122256.

[13] Zhong Z, Peng Z, Huang P, et al. UAV small-object detection algorithm with multi-scale feature perception [J]. Computer Engineering, 1-15 [2026-04-01].

[14] Zhang Q, Wang X, Shi H, et al. BRA-YOLOv10: UAV small target detection based on YOLOv10 [J]. Drones, 2025, 9(3): 159.

[15] Liu Y, Gao S, Wei D. MAFS-YOLO: A UAV-view small-object detection algorithm based on improved YOLO11 [J]. Computer Science and Exploration, 1-13 [2026-04-01].

[16] Wang A, Chen H, Liu L, et al. YOLOv10: Real-time end-to-end object detection [J]. Advances in Neural Information Processing Systems, 2024, 37: 107984-108011.

[17] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 2117-2125.

[18] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 8759-8768.

[19] Kang M, Ting C M, Ting F F, et al. ASF-YOLO: A novel YOLO model with attentional scale sequence fusion for cell instance segmentation [J]. Image and Vision Computing, 2024, 147: 105057.

[20] Li X, Wang W, Hu X, et al. Generalized focal loss v2: Learning reliable localization quality estimation for

dense object detection [C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 11632-11641.

[21] Du D, Zhu P, Wen L, et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results [C]//Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. 2019.