# MFCA-NET: MULTI-FREQUENCY CONTENT-ADAPTIVE NETWORK FOR UAV OBJECT DETECTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Object detection performance can be improved through complementary techniques such as data augmentation, loss optimization, and image feature enhancement. Inspired by conventional image filtering, we propose a Multi-Frequency Content-Adaptive Network (MFCA-Net), which consists of Fixed-Frequency Prior Extraction (FFPE), Multi-Frequency Prior Representation (MFPR), and Content-Adaptive Frequency Fusion (CAFF). MFPR extracts three prior features using Gaussian filtering, Difference of Gaussians (DoG), and an eight-neighbor Laplacian filter. CAFF generates frequency-specific weights from the RGB and prior features, fuses them into a single-channel representation, and concatenates it with RGB as the input to the detector. We evaluate MFCA-Net with YOLOv8 and MaxViT backbones on VisDrone-DET and DroneVehicle. MFCA-Net introduces only 11,751 additional parameters while improving detection performance across different backbones and datasets. In particular, YOLOv8n+MFCA-Net improves mAP by 1.2% and mAPs by 0.9% on VisDrone-DET. These results demonstrate that the proposed frequency-based feature enhancement can improve object detection with a small increase in model complexity.

**Keywords:** Fixed Filters; Prior Features; Content-Adaptive Fusion; Multi-Frequency Representation; Object Detection; YOLOv8

## 1 INTRODUCTION

Conventional RGB-based object detectors Zhu et al. (2021a) can be broadly categorized into one-stage and two-stage detectors, and further distinguished by whether they rely on predefined anchor boxes. Recent advances in backbone architectures, attention mechanisms Carion et al. (2020), data augmentation, and loss design Lin et al. (2017b) have substantially improved the performance of one-stage detectors. Representative architectures such as YOLO Terven et al. (2023); Wang et al. (2024) and DETR Carion et al. (2020); Zhu et al. (2021b) have achieved competitive detection accuracy while retaining a relatively simple detection pipeline. More recently, stronger backbones such as MaxViT Tu et al. (2022) have also been explored to further improve the representation capability of one-stage detectors.

For a given backbone, several general techniques can be used to improve object detection performance, including data augmentation, loss function design, and multi-scale detection. Common augmentation strategies include Mosaic, random flipping, CLAHE Mishra (2021), HSV augmentation, and random affine transformations. Loss functions such as CIoU Zheng et al. (2021) and DFL Li et al. (2020) have also been widely adopted to improve bounding box regression and localization. Multi-scale detection further enhances the ability of detectors to handle objects at different scales.

Another established approach is to augment RGB inputs with handcrafted prior features. Common choices include Gaussian filtering, Laplacian filtering, Difference of Gaussians (DoG), Laplacian of Gaussian (LoG), sharpening, and Sobel filtering. We refer to these individual prior extraction operations as Fixed-Frequency Prior Extraction (FFPE). Multiple prior features can be combined to form a composite representation, which we term Multi-Frequency Prior Representation (MFPR). Since these image filters use fixed kernels, they can be formulated as 2D convolutions with non-learnable kernel weights.

We introduce MFPR features alongside the RGB input to exploit complementary prior information for object detection. To avoid a substantial increase in model parameters, the added features are



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

adaptively fused according to the RGB content. Specifically, the Content-Adaptive Frequency Fusion (CAFF) module generates a fused prior feature, which is concatenated with the RGB input to form $[\mathrm{RGB}; F_{\mathrm{fused}}]$ and fed into the detector backbone.

Our goal is to improve object detection accuracy from the UAV viewpoint while introducing as few additional parameters as possible. To transform the RGB input into $[\mathrm{RGB}; F_{\mathrm{fused}}]$, we construct MFPR using three frequency components. Gaussian filtering is used to extract low-frequency features, Difference of Gaussians (DoG) for mid-frequency features, and an eight-neighbor Laplacian filter for high-frequency features. The resulting prior features are then fused into a single-channel feature using the lightweight CAFF module and concatenated with the RGB input, yielding a four-channel input for the detector backbone.

The main contributions of this work are summarized as follows:

- We introduce Fixed-Frequency Prior Extraction (FFPE) and Multi-Frequency Prior Representation (MFPR), using Gaussian filtering, Difference of Gaussians (DoG), and an eight-neighbor Laplacian filter to extract fixed prior features at different frequencies.
- We develop a Content-Adaptive Frequency Fusion (CAFF) module that generates frequency-specific weight maps from the RGB and MFPR features. The three frequency components are fused into a single-channel feature and concatenated with RGB to form $[\mathrm{RGB}; F_{\mathrm{fused}}]$. Together, MFPR and CAFF form the proposed Multi-Frequency Content-Adaptive Network (MFCA-Net).
- We evaluate $YOLOv8n+CAFF$ against the baseline on VisDrone-DET Zhu et al. (2018; 2021a) and DroneVehicle Sun et al. (2022), and further replace the backbone with the larger MaxViT-T to examine the performance of CAFF across different backbone architectures. Despite introducing only 11,751 additional parameters, CAFF consistently improves mAP across the evaluated models and datasets.

This section introduces the background, motivation, and contributions of this work. Section II reviews related work on the datasets and methods. Section III describes the overall architecture of MFCA-Net and the training settings. Section IV presents the comparative and ablation experiments. Section V concludes the paper.

## 2 Related Work

### 2.1 Dataset

MS COCO Lin et al. (2014) with 80 common object categories in real life and scale variations of objects, clutter background and heavy occlusion, becomes the mainstream benchmark for object detection, semantic segmentation and image description. The UAVDT dataset Du et al. (2018) is utilized for object detection tasks in the field of drone inspection. MOR-UAV Mandal et al. (2020) is utilized for moving object recognition in UAV videos, and vision-based UAV applications, e.g., aerial surveillance, search and rescue.

The VisDrone-DET dataset Zhu et al. (2021a) exhibits several distinctive characteristics. The instance count per category reveals significant class imbalance, with certain categories such as "car" and "pedestrian" dominating dataset. The shape distribution of all bounding boxes shows that objects from a drone's aerial perspective are predominantly small and compact. These objects are broadly distributed across the image plane, with a slight concentration toward the center. Meanwhile, the majority of objects occupy only a small fraction of the image area, which is typical for drone-based detection tasks. Together, these statistical properties underscore the challenges of small object detection and class imbalance inherent in drone-view object detection.

### 2.2 Methods

The algorithms or models with deep convolutional neural network (CNN) or transformer have become the mainstream methods in the object detection domain Zhu et al. (2021a), e.g., Faster R-CNN Ren et al. (2017), Light-RCNN Li et al. (2017), Cascade R-CNN Cai & Vasconcelos (2018), Re-

fineDet Zhang et al. (2018), CenterNet Zhou et al. (2019); Duan et al. (2019), DETR Carion et al. (2020), YOLO Terven et al. (2023), etc.

These object detection methods can be classified according to different criteria, which comprises stage, anchor[1], network architecture, training strategy and application scenarios. Furthermore, we elaborate the explanations and representative object detection methods for different classification criteria, as shown in Table 6.

# 3 METHODS

## 3.1 MFCA-NET OVERWIEW

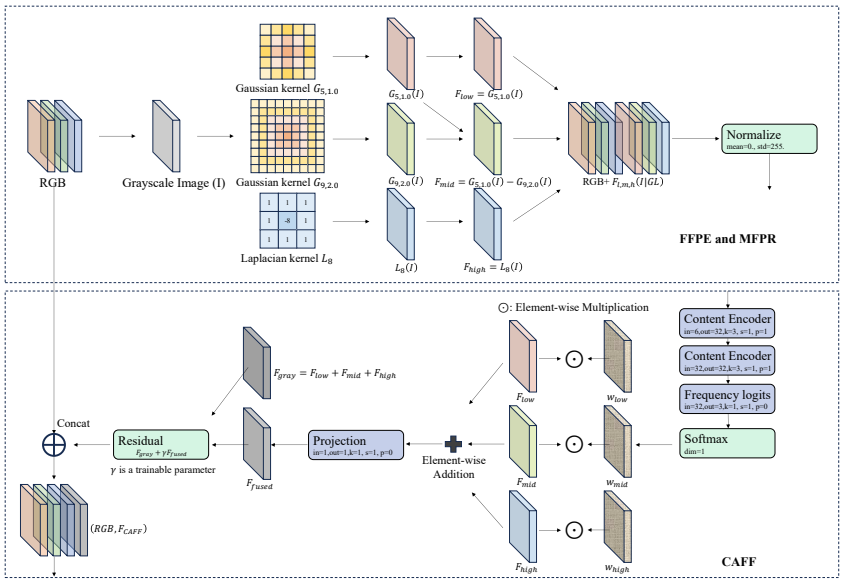



Figure 1: Overall architecture of MFCA-Net, comprising the FFPE, MFPR, and CAFF modules.

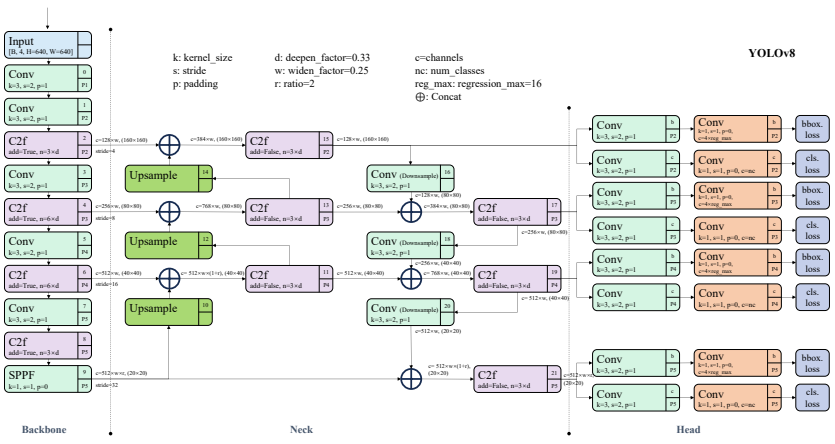



Figure 2: YOLOv8 architecture with an additional P2 detection scale at stride 4, along with the corresponding neck and detection head.

This section presents the overall architecture of MFCA-Net, which comprises four main components: (1) Fixed Frequency Prior Extraction (FFPE), (2) Multi-Frequency Prior Representation

[1]Anchor-boxes are a set of pre-defined bounding boxes with various scales and aspect ratios, which are used for predicting the locations and sizes of object instances Zhu et al. (2021a).

(MFPR), (3) Content-Adaptive Frequency Fusion (CAFF), and (4) an optional P2-extended detection architecture with a stride-4 P2 layer and the corresponding backbone, neck, and detection head. An overview of MFCA-Net is shown in Fig. 1.

Since YOLOv8n is used as the baseline for our comparisons and ablation studies, we show the modified YOLOv8 architecture with an additional P2 detection scale in Fig. 2.

The FFPE module provides several fixed convolutional kernels, including Gaussian, Laplacian, sharpening, and Sobel kernels, as well as composite operators such as Laplacian of Gaussian (LoG) and Difference of Gaussians (DoG). After evaluating different kernel combinations, we use $G_{5,1.0}$, $G_{9,2.0}$, and the 8-neighbor Laplacian kernel $L_8$ to construct the fourth channel in addition to RGB.

We compute the discrete Gaussian kernel $G_{k,\sigma}$ according to Eq. 1, where $k$ denotes the kernel size and $\sigma$ controls the standard deviation of the Gaussian distribution.

$$K(i,j) = \frac{e^{-\frac{x_i^2+y_j^2}{2\sigma^2}}}{\sum_m \sum_n e^{-\frac{x_m^2+y_n^2}{2\sigma^2}}} \tag{1}$$

For $G_{5,1.0}$, $x_i, y_j \in \{-2,-1,0,1,2\}$, where $i$ and $j$ are the spatial indices. Substituting these values into the Gaussian function gives the following kernel:

$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}$$

The 8-neighbor Laplacian kernel $L_8$ is given below.

$$\begin{bmatrix} 1. & 1. & 1. \\ 1. & -8. & 1. \\ 1. & 1. & 1. \end{bmatrix}$$

We construct the MFPR by selecting low-, mid-, and high-frequency features from the single-frequency candidates and augmenting the RGB input with the selected components. The resulting frequency components are defined as follows:

$$F_{l,m,h}(I|GL) = F_{low} + F_{mid} + F_{high} \tag{2}$$

where: $F_{low} = G_{5,1.0}(I),\ F_{mid} = G_{5,1.0}(I) - G_{9,2.0}(I),\ F_{high} = L_8(I).$

MFPR retains richer prior information, but concatenating RGB with $F_{l,m,h}(I|GL)$ produces a six-channel input, which introduces redundancy and additional computational and parameter overhead. We therefore introduce CAFF to compress $F_{l,m,h}(I|GL)$ into a single channel, as shown in Fig. 1.

CAFF first estimates three spatial weight maps for $F_{low}$, $F_{mid}$, and $F_{high}$ from the RGB input and $F_{l,m,h}(I|GL)$. These weight maps share the same spatial dimensions as the corresponding frequency features, as given in Eq. 3.

$$[w_{low}, w_{mid}, w_{high}] = \text{Softmax}(\text{Frequency logits}(\text{Content Encoder}(RGB; F_{l,m,h}(I|GL)))) \tag{3}$$

The weight maps are multiplied element-wise with the corresponding frequency features, and the weighted features are summed and projected through the Frequency Projection layer.

$$F_{fused} = \text{Projection}(w_{low}F_{low} + w_{mid}F_{mid} + w_{high}F_{high}) \tag{4}$$

Finally, $F_{fused}$ is added through a residual connection and concatenated with the RGB features as the backbone input.

$$+CAFF(F_{l,m,h}(I|GL)) = [RGB; F_{low} + F_{mid} + F_{high} + \gamma F_{fused}] \tag{5}$$

Here, $\gamma$ is a learnable parameter rather than manually set as a hyperparameter.

**Algorithm 1** Training Data Augmentation

**Input:** Image $I$, annotations $B$, target size $S = (640, 640)$
**Output:** Augmented image $\tilde{I}$ and annotations $\tilde{B}$

Load $I$ and corresponding bounding boxes annotations $B$.
max epoch, $close_mosaic_epochs = 10$
first training stage: current epoch $\leq$ (max epoch - $close_mosaic_epochs$)
**if** first training stage **then**
    Select three additional training images and their annotations.
    Apply Mosaic augmentation to combine the four images $\{I_k\}_{k=1}^{4}$ into one image with target size $S$ and their annotations $\{B_k\}_{k=1}^{4}$.
    Initialize an empty canvas with padding value $114$.
    **for** $k = 1$ to $4$ **do**
        Resize and crop $I_k$ according to the sampled mosaic layout.
        Transform the corresponding bounding boxes $B_k$ accordingly.
    **end for**
    Place the four transformed images on the canvas to form $I_M$.
    Merge their annotations: $B_M = \bigcup_{k=1}^{4} B_k$.
    Mosaic image $I_M$ and annotations $B_M$
    Apply random affine transformation:
        - rotation range: $[0°, 0°]$;
        - shear range: $[0°, 0°]$;
        - scaling ratio: $[1 - \alpha, 1 + \alpha]$; ($\alpha = 0.5$)
        - aspect ratio constraint: $r_{\max} = 100$.
**else**
    Resize $I$ while preserving its aspect ratio.
    Apply LetterBox padding to obtain the target size $S$.
    Apply random affine transformation using the same parameters as in the first stage.
**end if**
Apply the following image-level augmentations independently:
    - Gaussian blur with probability $0.01$;
    - median blur with probability $0.01$;
    - grayscale conversion with probability $0.01$;
    - CLAHE enhancement with probability $0.01$.
Apply random HSV augmentation.
Horizontally flip the image and annotations with probability $0.5$.
Pack the augmented image and annotations into the model input format.
**return** $\tilde{I}, \tilde{B}$

### 3.2 Training Configuration

**Detection Configuration.** We use multi-label prediction, where a detection box can be associated with multiple classes. The confidence threshold is set to 0.001, and up to 30,000 candidates are retained before NMS. Standard NMS is performed with an IoU threshold of 0.7, with at most 300 detections kept per image. Bounding boxes are represented using $xywh$, i.e., the top-left corner coordinates followed by the width and height.

**Loss Configuration.** The overall loss consists of classification, bounding box regression, and distribution focal losses. Specifically, Cross-Entropy Loss is used for classification with a weight of 0.5, while IoU Loss (CIoU) Zheng et al. (2021) is applied to bounding box regression with a weight of 7.5. Distribution Focal Loss (DFL) Li et al. (2020) is used for boundary refinement with a weight of $1.5/4$. The total loss is given by the sum of the three terms.

**Dataset Configuration.** PASCAL VOC (XML) Everingham et al. (2015); Everingham et al., YOLO (TXT) Terven et al. (2023), and COCO (JSON) Lin et al. (2014) are commonly used annotation formats. We use the standard COCO format for all datasets in our experiments.

**Data Augmentation.** We use a two-stage augmentation strategy during training, as summarized in Algorithm 1. In the first stage, four training images are randomly sampled and combined using Mosaic augmentation with a padding value of $114$, followed by a random affine transformation.

During the last 10 epochs, Mosaic augmentation is disabled and the second-stage augmentation strategy is used. Each image is resized while preserving its aspect ratio and then padded to $640 \times 640$. The random affine transformation is retained with the same settings as in the first stage.

Both stages use the same image-level augmentations, including Blur, MedianBlur, grayscale conversion, and CLAHE Mishra (2021), each applied with a probability of $0.01$. Random HSV augmentation is subsequently applied, followed by horizontal flipping with a probability of $0.5$.

For example, the HSV augmentation is defined as Eq. 6.

$$\tilde{I} = \mathrm{HSV}_{\Delta H, \Delta S, \Delta V}(I) \tag{6}$$

where $\Delta H$, $\Delta S$, and $\Delta V$ denote random perturbations applied to the hue, saturation, and value components, respectively.

For testing, each image is resized while preserving its aspect ratio and then padded to the target size $H \times W$. The bounding box annotations $B$ are transformed according to the resizing and padding operations. The processed images, annotations, and image metadata are then packed as model inputs.

**Optimization Configuration.** The loss $L$ is computed using Eq. 7, and its gradient $\nabla_\theta L$ is obtained by backpropagation. Gradient clipping is applied with $|\nabla_\theta L| \leq 10$. We use SGD with a momentum of $0.937$ and a weight decay of $0.0005$. The initial learning rate is set to $0.01$. A linear learning rate scheduler is used, with a scaling factor of $0.01$, reducing the learning rate to $0.0001$ at the end of training.

$$L = 0.5L_{cls} + 7.5L_{CIoU} + 0.375L_{DFL} \tag{7}$$

**Evaluation Configuration.** We follow the COCO evaluation protocol for object detection. The evaluation metrics include $mAP$, $mAP_{50}$, $mAP_{75}$, $mAP_s$, $mAP_m$, $mAP_l$, $AR_1$, $AR_{10}$, $AR_{100}$, $AR_s$, $AR_m$, and $AR_l$.

# 4 EXPERIMENTS

Table 1: Comparison of YOLOv8n and MaxViT-T with and without $CAFF(F_{l,m,h}(I|GL))$ on the VisDrone and DroneVehicle datasets. All metrics from mAP to $\mathrm{AR}_l$ are reported in %, with the percent signs omitted.

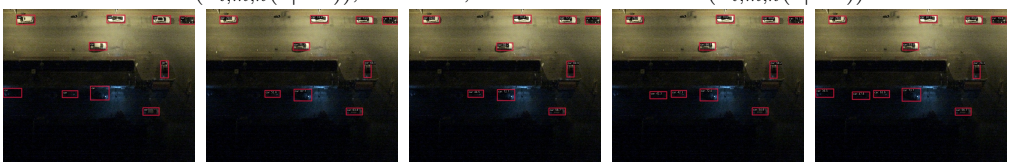
| Dataset | Backbone | Prior Feature | mAP | $\mathrm{mAP}_{50}$ | $\mathrm{mAP}_{75}$ | $\mathrm{mAP}_s$ | $\mathrm{mAP}_m$ | $\mathrm{mAP}_l$ | $\mathrm{AR}_1$ | $\mathrm{AR}_{10}$ | $\mathrm{AR}_{100}$ | $\mathrm{AR}_s$ | $\mathrm{AR}_m$ | $\mathrm{AR}_l$ | MParams | GFLOPs | FPS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| VisDrone-DET | YOLOv8n.B | RGB (**baseline**) | 16.5 | 29.6 | 16.0 | 7.7 | 26.5 | 33.8 | 7.9 | 21.6 | 28.7 | 17.8 | 44.1 | 49.3 | **3.013** | **4.057** | **85.670** |
| | | $+CAFF(F_{l,m,h}(I\|GL))$ | 17.7 | 31.3 | 17.2 | 8.6 | 27.8 | 37.5 | 8.4 | 22.7 | 30.0 | 18.9 | 45.4 | 52.9 | 3.024 | 8.665 | 72.745 |
| | MaxViT-T | RGB (**baseline**) | 24.6 | 41.9 | **24.8** | 13.8 | 37.9 | **48.0** | **10.9** | 29.1 | 37.5 | 26.3 | 53.9 | **64.2** | 36.189 | 53.078 | 18.298 |
| | | $+CAFF(F_{l,m,h}(I\|GL))$ | **24.7** | **42.1** | 24.7 | **13.9** | **38.2** | 46.4 | **10.9** | **29.4** | **37.8** | **26.4** | **54.9** | 63.9 | 36.201 | 57.713 | 18.043 |
| DroneVehicle | YOLOv8n.B | RGB (**baseline**) | 48.1 | 71.8 | 56.3 | 19.2 | 47.7 | 45.9 | 19.0 | 53.1 | 68.5 | 38.8 | 68.6 | **74.4** | **3.012** | **4.072** | **85.906** |
| | | $+CAFF(F_{l,m,h}(I\|GL))$ | 48.9 | 72.7 | 55.9 | 19.9 | 48.8 | 49.7 | 19.8 | 52.9 | 67.1 | 38.0 | 67.8 | 62.6 | 3.023 | 8.662 | 73.094 |
| | MaxViT-T | RGB (**baseline**) | 53.0 | 76.8 | 60.6 | 21.2 | 52.4 | **55.9** | 20.9 | **55.0** | 68.6 | **41.5** | 68.9 | 70.3 | 36.187 | 53.073 | 19.012 |
| | | $+CAFF(F_{l,m,h}(I\|GL))$ | **53.3** | **77.2** | **61.2** | **23.6** | **52.7** | 55.6 | **21.0** | **55.0** | **68.7** | 40.9 | **69.2** | 64.2 | 36.199 | 57.707 | 17.539 |

## 4.1 COMPARISON

We use YOLOv8n Terven et al. (2023); Varghese & M. (2024) as the baseline and compare it with our proposed $+CAFF(F_{l,m,h}(I|GL))$ method. Both models use the training configuration described in the Methods section. The detection head uses only the P3, P4, and P5 scales, without P2. We evaluate both models on the VisDrone and DroneVehicle datasets, with the results reported in Table 1.



(a) Qualitative comparison on VisDrone among ground-truth annotations, YOLOv8n, YOLOv8n+$CAFF(F_{l,m,h}(I|GL))$, MaxViT-T, and MaxViT-T+$CAFF(F_{l,m,h}(I|GL))$.



(b) Qualitative comparison on DroneVehicle among ground-truth annotations, YOLOv8n, YOLOv8n+$CAFF(F_{l,m,h}(I|GL))$, MaxViT-T, and MaxViT-T+$CAFF(F_{l,m,h}(I|GL))$.

Figure 3: Qualitative comparison examples of detection results on the two datasets.

To further evaluate CAFF with different backbones, we replace the YOLOv8n backbone with MaxViT-T Tu et al. (2022). We evaluate the resulting model on both datasets, with the results reported in Table 1.

**Parameter Analysis.** We observe that the same model architecture has different parameter counts on VisDrone and DroneVehicle. This difference comes from the number of object categories in the two datasets. For example, when using MaxViT-T as the backbone, the parameter counts of the backbone and Neck are identical for both datasets. VisDrone and DroneVehicle contain 10 and 5 object categories, respectively, resulting in a difference of $1{,}935 = 3 \times (128 + 1) \times (10 - 5)$ parameters in the classification heads.

Specifically, the final layer of each classification head at P3, P4, and P5 is a $1 \times 1$ convolution, as shown in Fig. 2. Its channel dimension changes from $128 \rightarrow 10$ for VisDrone to $128 \rightarrow 5$ for DroneVehicle. Thus, each classification head has 645 fewer parameters, resulting in a total reduction of 1,935 parameters across the three detection scales.

We further compare the parameter counts of the proposed $+CAFF(F_{l,m,h}(I|GL))$ and the baseline. The CAFF module introduces 11,751 additional parameters. For YOLOv8n, this corresponds to only a $0.39\%$ increase in the total number of parameters, while improving mAP by $1.2\%$ and $\text{mAP}_s$ by $0.9\%$ on the VisDrone-DET dataset.

**Performance Analysis.** YOLOv8n exhibits different mAP values on VisDrone and DroneVehicle, which can be attributed to the different object distributions in the two datasets. VisDrone contains many small and medium-sized objects within each image, whereas DroneVehicle generally contains fewer annotated objects per image. With only 11,751 additional parameters, our proposed $CAFF(F_{l,m,h}(I|GL))$ improves mAP across different backbones. The improvement is smaller when using MaxViT-T, which has a substantially larger backbone capacity. To further illustrate the differences between the two datasets, representative qualitative results are presented in Fig. 3.

Adding $CAFF(F_{l,m,h}(I|GL))$ to YOLOv8n improves mAP by 1.2% and $mAP_s$ by 0.9% on VisDrone-DET, and by 0.8% and 0.7% on DroneVehicle, respectively. When the backbone is replaced with MaxViT-T, the same module yields gains of 0.1% in mAP and 0.1% in $mAP_s$ on VisDrone-DET, and 0.3% in mAP and 2.4% in $mAP_s$ on DroneVehicle. These results show that $CAFF(F_{l,m,h}(I|GL))$ consistently improves detection performance across different backbone architectures, with the magnitude of the gain varying across datasets and backbones.

### 4.2 Ablation Studies

We augment the three-channel RGB input with the prior feature $CAFF(F_{l,m,h}(I|GL))$, resulting in a clear gain in mAP. Since different design choices for the prior feature can affect performance differently, we analyze them individually to quantify their respective contributions.

**Ablation Settings.** All ablation experiments use YOLOv8n, with COCO-style metrics reported on VisDrone-DET and DroneVehicle. Parameters, FLOPs, and FPS are measured using the settings

Table 2: Ablation results for different detection scales and prior feature designs. All metrics from mAP to $\mathrm{AR}_l$ are reported in %, with the percent signs omitted.

| Dataset | Head | Prior Feature | | mAP | $\mathrm{mAP}_{50}$ | $\mathrm{mAP}_{75}$ | $\mathrm{mAP}_s$ | $\mathrm{mAP}_m$ | $\mathrm{mAP}_l$ | $\mathrm{AR}_1$ | $\mathrm{AR}_{10}$ | $\mathrm{AR}_{100}$ | $\mathrm{AR}_s$ | $\mathrm{AR}_m$ | $\mathrm{AR}_l$ | MParams | GFLOPs↓ | FPS↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| VisDrone-DET | P3,P4,P5 | FFPE | RGB (**baseline**) | 16.5 | 29.6 | 16.0 | 7.7 | 26.5 | 33.8 | 7.9 | 21.6 | 28.7 | 17.8 | 44.1 | 49.3 | 3.013 | **4.057** | **85.670** |
| | | | $+G_{5,1.0}(I)$ | 16.6 | 29.7 | 16.0 | 7.8 | 26.2 | 36.2 | 8.2 | 21.8 | 28.8 | 17.9 | 44.0 | 50.9 | 3.013 | 4.089 | 82.526 |
| | | | $+L_4(I)$ | 16.7 | 29.9 | 15.8 | 8.0 | 26.3 | 37.2 | 8.2 | 21.7 | 29.1 | 18.4 | 43.9 | 51.2 | 3.013 | 4.089 | 82.110 |
| | | | $+L_8(I)$ | 16.8 | 30.1 | 16.0 | 7.9 | 26.6 | 36.8 | 8.2 | 21.9 | 29.0 | 18.0 | 44.2 | 50.3 | 3.013 | 4.089 | 82.500 |
| | | | $+S_9(I)$ | 16.6 | 29.8 | 16.0 | 7.8 | 26.4 | 36.5 | 8.2 | 21.8 | 28.8 | 17.8 | 44.3 | 51.5 | 3.013 | 4.089 | 81.483 |
| | | | $+S_{xy}(I)$ | 16.6 | 29.8 | 15.9 | 7.8 | 26.5 | 36.0 | 8.1 | 21.8 | 28.7 | 17.9 | 44.0 | 50.5 | 3.013 | 4.089 | 80.829 |
| | | | $+LoG_{5,1.0,4}(I)$ | 16.7 | 29.9 | 16.1 | 7.8 | 26.3 | 37.4 | 8.3 | 21.7 | 28.6 | 17.9 | 43.3 | 52.6 | 3.013 | 4.089 | 83.198 |
| | | | $+LoG_{5,1.0,8}(I)$ | 16.6 | 29.8 | 15.9 | 7.9 | 26.3 | 35.8 | 8.2 | 21.8 | 28.9 | 17.9 | 44.0 | 49.1 | 3.013 | 4.089 | 83.515 |
| | | | $+SoG_{5,1.0,9}(I)$ | 16.5 | 29.5 | 15.9 | 7.7 | 26.4 | 36.3 | 8.2 | 21.5 | 28.5 | 17.4 | 43.8 | 51.2 | 3.013 | 4.089 | 83.393 |
| | | | $+DoG_{5,1.0,9,2.0}(I)$ | 16.4 | 29.3 | 15.9 | 7.7 | 26.2 | 35.7 | 8.0 | 21.6 | 28.7 | 17.8 | 44.0 | 50.2 | 3.013 | 4.089 | 80.533 |
| | | MFPR | $+F_{l,m,h}(I\|G)$ | 16.2 | 29.4 | 15.6 | 7.6 | 26.1 | 36.5 | 8.0 | 21.3 | 28.3 | 17.4 | 43.5 | 52.5 | 3.013 | 4.119 | 78.780 |
| | | | $+F_{l,m,h}(I\|GL)$ | 16.5 | 29.8 | 15.6 | 8.0 | 26.4 | 35.5 | 8.0 | 21.7 | 28.8 | 17.9 | 44.5 | 48.4 | 3.013 | 4.119 | 77.002 |
| | | CAFF | $+CAFF(F_{l,m,h}(I\|G))$ | 17.6 | 30.9 | **17.4** | 8.5 | 28.2 | 35.3 | **8.5** | 22.7 | 29.9 | 18.7 | **45.8** | 51.5 | 3.024 | 8.665 | 72.491 |
| | | | $+CAFF(F_{l,m,h}(I\|GL))$ | **17.7** | **31.3** | 17.2 | 8.6 | **27.8** | 37.5 | 8.4 | 22.7 | 30.0 | 18.9 | 45.4 | **52.9** | 3.024 | 8.665 | 72.745 |
| | | | $+CAFF_{R=F}(F(I\|GL))$ | 17.6 | 31.1 | 16.9 | **8.8** | 27.3 | **37.6** | 8.4 | **22.8** | **30.1** | **19.1** | 45.4 | 51.0 | 3.024 | 8.665 | 73.307 |
| | +P2 | FFPE | RGB | 19.8 | 34.2 | 19.6 | 11.9 | 29.2 | **38.0** | 8.7 | 24.6 | 33.2 | 23.5 | 46.1 | 52.9 | **2.928** | 6.145 | 70.009 |
| | | | $+L_8(I)$ | 19.8 | 34.3 | 19.4 | 11.9 | 29.2 | 37.8 | 8.7 | 24.6 | 33.3 | 23.6 | 46.5 | 52.9 | **2.928** | 6.16 | 68.313 |
| | | MFPR | $+F_{l,m,h}(I\|GL)$ | 19.7 | 34.3 | 19.4 | 11.6 | 29.4 | 36.0 | 8.8 | 24.5 | 33.2 | 23.3 | 46.6 | 52.4 | **2.928** | 6.189 | 62.695 |
| | | CAFF | $+CAFF(F_{l,m,h}(I\|GL))$ | **20.3** | **34.9** | **20.1** | **12.3** | **29.9** | 36.5 | **9.0** | **25.4** | **34.2** | **24.4** | **47.4** | **56.4** | 2.939 | 10.736 | 60.572 |
| DroneVehicle | P3,P4,P5 | FFPE | RGB (**baseline**) | 48.1 | 71.8 | **56.3** | 19.2 | 47.7 | 45.9 | 19.0 | **53.1** | **68.5** | **38.8** | **68.6** | **74.4** | 3.012 | **4.072** | **85.906** |
| | | | $+L_8(I)$ | 48.2 | 72.1 | 55.0 | 19.7 | 47.8 | **50.5** | 19.6 | 52.4 | 66.6 | 38.2 | 67.2 | 68.3 | 3.012 | 4.087 | 83.980 |
| | | MFPR | $+F_{l,m,h}(I\|GL)$ | 48.3 | 72.1 | 54.6 | 19.8 | 47.9 | **50.5** | 19.7 | 52.5 | 66.7 | 38.2 | 67.4 | 62.3 | 3.012 | 4.116 | 77.822 |
| | | CAFF | $+CAFF(F_{l,m,h}(I\|GL))$ | **48.9** | **72.7** | 55.9 | **19.9** | **48.8** | 49.7 | **19.8** | 52.9 | 67.1 | 38.0 | 67.8 | 62.6 | 3.023 | 8.662 | 73.094 |
| | +P2 | FFPE | RGB | 47.1 | 70.4 | 53.9 | 20.4 | 47.1 | 46.6 | 19.4 | 52.7 | 66.9 | 38.5 | 67.6 | **61.8** | **2.927** | 6.14 | 71.467 |
| | | | $+L_8(I)$ | 47.7 | 70.9 | 54.9 | **20.6** | 47.3 | **48.6** | **19.7** | 52.7 | 67.0 | 39.7 | 67.7 | 60.9 | **2.927** | 6.154 | 69.007 |
| | | MFPR | $+F_{l,m,h}(I\|GL)$ | 48.0 | 71.4 | 54.8 | 20.4 | 47.8 | 48.0 | **19.7** | 52.7 | 67.0 | 38.0 | 67.7 | 61.7 | 2.928 | 6.184 | 65.553 |
| | | CAFF | $+CAFF(F_{l,m,h}(I\|GL))$ | **48.1** | **71.8** | **55.5** | 20.4 | **48.2** | 48.5 | 19.6 | **53.1** | **67.4** | **40.0** | **68.2** | **61.8** | 2.939 | 10.73 | 60.364 |

listed below. Unless noted otherwise, the same environment and configuration are used throughout the experiments.

| Setting | Value |
|---|---|
| GPU | NVIDIA V100 32GB |
| Input Resolution | $640 \times 640$ |
| Batch Size | 1 |
| Warm-up Iterations | 100 |
| Measured Iterations | 500 |

FPS denotes end-to-end FPS (E2E FPS), measured from the raw RGB input through data preprocessing $F_{l,m,h}(I|GL)$, CAFF, and the detection model. Thus, it accounts for the full inference pipeline rather than the detection model alone.

We add the candidate prior features, FFPE, MFPR, and CAFF in sequence. The corresponding kernel settings and configurations are listed below, where $I$ denotes the grayscale image. The baseline and ablation results are shown in Table 2. The P2 detection scale is used only in the specified configurations, and all reported metrics are included for each design choice.

**FFPE:**

**Gaussian Kernel:** $G_{k,\sigma}(I)$, where $k$ specifies the kernel size and $\sigma$ denotes the standard deviation of the Gaussian distribution.

**Laplacian Kernel:** $L_x(I)$, where $L_4(I)$ and $L_8(I)$ correspond to the four-neighbor and eight-neighbor Laplacian operators, respectively.

**Sharpening Kernel:** $S_9(I)$, which represents the response obtained using the sharpening kernel.

**Sobel Kernel:** $S_{xy}(I) = \sqrt{S_x^2(I) + S_y^2(I)}$, where $S_x(I)$ and $S_y(I)$ denote the X-horizontal and Y-vertical gradient responses, respectively.

$$S_9 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -9 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \qquad S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

**LoG Kernel:** $LoG_{k,\sigma,x}(I) = L_x(G_{k,\sigma}(I))$. For example, $LoG_{5,1.0,4}(I) = L_4(G_{5,1.0}(I))$ and $LoG_{5,1.0,8}(I) = L_8(G_{5,1.0}(I))$.

**SoG Kernel:** $SoG_{k,\sigma,9}(I) = S_9(G_{k,\sigma}(I))$, e.g., $SoG_{5,1.0,9}(I) = S_9(G_{5,1.0}(I))$.

**DoG Kernel:** $DoG_{k_1,\sigma_1,k_2,\sigma_2}(I) = G_{k_1,\sigma_1}(I) - G_{k_2,\sigma_2}(I)$. For the configuration used in our experiments, $DoG_{5,1.0,9,2.0}(I) = G_{5,1.0}(I) - G_{9,2.0}(I)$.

**MFPR:**

**Gaussian-Laplacian** $F_{l,m,h}(I|GL) = F_{low} + F_{mid} + F_{high}$, where $F_{low} = G_{5,1.0}(I)$, $F_{mid} = G_{5,1.0}(I) - G_{9,2.0}(I)$, $F_{high} = L_8(I)$.

**Only Gaussian** $F_{l,m,h}(I|G) = F_{low} + F_{mid} + F_{high}$, where $F_{low} = G_{9,2.0}(I)$, $F_{mid} = G_{5,1.0}(I) - G_{9,2.0}(I)$, and $F_{high} = I - G_{5,1.0}(I)$. Accordingly, $I = F_{low} + F_{mid} + F_{high}$.

**CAFF:**

**Only Gaussian** $CAFF(F_{l,m,h}(I|G))$.

**Gaussian-Laplacian** $CAFF(F_{l,m,h}(I|GL))$.

**No Residual** $CAFF_{R=F}(F(I|GL)) = CAFF_{R=False}(F_{l,m,h}(I|GL))$.

**Effect of +P2 on Performance.** As shown in Table 2, adding the P2 detection scale to the baseline improves the VisDrone performance by 3.3 mAP, but decreases the DroneVehicle performance by 1.0 mAP. This difference can be attributed to the object size distributions of the two datasets.

VisDrone contains a large number of small objects, for which the P2 feature map provides finer spatial details. This leads to a 3.3 mAP improvement. In contrast, DroneVehicle contains mainly medium and large objects. In this case, the high-resolution P2 feature map ($160 \times 160$) introduces more background details and noise, which reduce the detection performance by 1.0 mAP.

Importantly, in the YOLOv8 neck, the P2 features are further propagated to P3, P4, and P5, as shown in Fig. 2. Thus, the effect of P2 is not limited to the highest-resolution feature map. The small-object cues, background textures, and noise introduced at P2 can also propagate to the other detection scales, leading to different effects across datasets with different object size distributions.

Table 3: Parameter counts of individual modules with and without the P2 detection scale. Both models use YOLOv8n with RGB as the only feature and exclude additional prior features and the CAFF module. Values are reported in millions (M).

| | P3,P4,P5 | +P2 |
|---|---|---|
| Backbone | 1.2727 | 1.2727 |
| Neck | 0.9869 | 1.0367 |
| Head | 0.7532 | **0.6185** |
| Total | 3.0128 | **2.9279** |

**Effect of +P2 on Parameters.** As shown in Table 2, adding the P2 detection scale reduces the number of parameters, while increasing FLOPs due to the larger $160 \times 160$ P2 feature map. To investigate the source of the parameter reduction, we compare the parameter counts of individual modules in YOLOv8n before and after adding P2, as reported in Table 3. The number of parameters in the backbone remains unchanged, while the Neck introduces additional parameters and the Head uses fewer parameters. Overall, the reduction in the Head parameters is larger than the increase in the Neck parameters, resulting in a lower total parameter count.

Specifically, the channel width of the classification branch in the Head is determined by $\max(\text{in_channels}[0], \text{num_classes})$. After adding the P2 detection scale, the input channels of the first detection level are reduced from 64 (P3) to 32 (P2). As a result, the classification channels at all detection levels are reduced from 64 to 32. This change saves substantially more parameters in the classification branches than are introduced by the additional P2 branch, leading to an overall reduction in the model parameters.

## 5 Conclusion

MFCA-Net is built around two core components, MFPR and CAFF. MFPR derives multi-frequency representations from the RGB input, while CAFF adaptively fuses them into a single feature, which is then concatenated with the original RGB image. The entire network introduces only 11,751 additional parameters. With YOLOv8 as the detector, MFCA-Net improves mAP on both VisDrone-DET and DroneVehicle. Replacing the backbone with MaxViT-T also yields mAP gains on both datasets, indicating that the proposed feature enhancement is not limited to a specific backbone architecture. For example, YOLOv8n+MFCA-Net improves mAP by 1.2% and $\text{mAP}_s$ by 0.9% on VisDrone-DET.

## REFERENCES


Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks, 2016. URL https://arxiv.org/abs/1511.02853.

Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6154–6162, 2018. doi: 10.1109/CVPR.2018.00644.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. URL https://arxiv.org/abs/2005.12872.

Chen Chen, Jiahao Qi, Xingyue Liu, Kangcheng Bin, Ruigang Fu, Xikun Hu, and Ping Zhong. Weakly misalignment-free adaptive feature alignment for uavs-based multimodal object detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26826–26835, 2024. doi: 10.1109/CVPR52733.2024.02534.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020a. URL https://arxiv.org/abs/2002.05709.

Yihong Chen, Zheng Zhang, Yue Cao, Liwei Wang, Stephen Lin, and Han Hu. Reppoints v2: Verification meets regression for object detection. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5621–5631. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3ce3bd7d63a2c9c81983cc8e9bd02ae5-Paper.pdf.

Jiankang Deng, Jia Guo, Evangelos Ververas, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-shot multi-level face localisation in the wild. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5202–5211, 2020. doi: 10.1109/CVPR42600.2020.00525.

Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. Uavdt dataset, 2018. URL https://sites.google.com/view/grli-uavdt/%E9%A6%96%E9%A1%B5.

Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6568–6577, 2019. doi: 10.1109/ICCV.2019.00667.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

Sakshini Hangloo and Bhavna Arora. Multimodal fusion techniques: Review, data representation, information fusion, and application areas. *Neurocomputing*, 649(000):22, 2010.

Irtiza Hasan, Shengcai Liao, Jinpeng Li, Saad Ullah Akram, and Ling Shao. Generalizable pedestrian detection: The elephant in the room, 2020. URL https://arxiv.org/abs/2003.08799.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning, 2020. URL https://arxiv.org/abs/1911.05722.

Wei Hua, Dingkang Liang, Jingyu Li, Xiaolong Liu, Zhikang Zou, Xiaoqing Ye, and Xiang Bai. Sood: Towards semi-supervised oriented object detection, 2023. URL https://arxiv.org/abs/2304.04515.

Soonmin Hwang, Jaesik Park, Namil Kim, Yukyung Choi, and In So Kweon. Multispectral pedestrian detection: Benchmark dataset and baselines. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints, 2019. URL https://arxiv.org/abs/1808.01244.

Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, 2020. URL https://arxiv.org/abs/2006.04388.

Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector, 2017. URL https://arxiv.org/abs/1711.07264.

Jianghang Lin, Yunhang Shen, Bingquan Wang, Shaohui Lin, Ke Li, and Liujuan Cao. Weakly supervised open-vocabulary object detection, 2023. URL https://arxiv.org/abs/2312.12437.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, editor="Fleet David Zitnick, C. Lawrence", Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pp. 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944, 2017a. doi: 10.1109/CVPR.2017.106.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, 2017b. doi: 10.1109/ICCV.2017.324.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 21–37, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46448-0.

Murari Mandal, Lav Kush Kumar, and Santosh Kumar Vipparthi. Mor-uav: A benchmark dataset and baselines for moving object recognition in uav videos. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, pp. 2626–2635, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379885. doi: 10.1145/3394171.3413934. URL https://doi.org/10.1145/3394171.3413934.

Akshansh Mishra. Contrast limited adaptive histogram equalization (clahe) approach for enhancement of the microstructures of friction stir welded joints, 2021. URL https://arxiv.org/abs/2109.00886.

Eliraz Orfaig, Inna Stainvas, and Igal Bilik. Rgbx-diffusiondet: a framework for multi-modal rgb-x object detection using diffusiondet. *Pattern Recognition*, 172:112460, April 2026. ISSN 0031-3203. doi: 10.1016/j.patcog.2025.112460. URL http://dx.doi.org/10.1016/j.patcog.2025.112460.

Tao Peng, Qing Li, and Pengfei Zhu. Rgb-t crowd counting from drone: A benchmark and mmccn network. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017. doi: 10.1109/TPAMI.2016.2577031.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Tahira Shehzadi, Ifza, Didier Stricker, and Muhammad Zeshan Afzal. Semi-supervised object detection: A survey on progress from cnn to transformer, 2024. URL https://arxiv.org/abs/2407.08460.

Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection, 2020. URL https://arxiv.org/abs/2005.04757.

Yiming Sun, Bing Cao, Pengfei Zhu, and Qinghua Hu. Drone-based rgb-infrared cross-modality vehicle detection via uncertainty-aware learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(10):6700–6713, 2022. doi: 10.1109/TCSVT.2022.3168279.

Juan Terven, Diana-Margarita Córdova-Esparza, and Julio-Alejandro Romero-González. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716, November 2023. ISSN 2504-4990. doi: 10.3390/make5040083. URL http://dx.doi.org/10.3390/make5040083.

Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9626–9635, 2019. doi: 10.1109/ICCV.2019.00972.

Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer, 2022. URL https://arxiv.org/abs/2204.01697.

Hoang Van Thanh, Tu Minh Phuong, Ha-Phuong Nguyen, Vu Huu Dao, and Kang-Hyun Jo. An efficient object detection model based on nanodet. In *2025 International Workshop on Intelligent Systems (IWIS)*, pp. 1–5, 2025. doi: 10.1109/IWIS66215.2025.11142424.

Rejin Varghese and Sambath M. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pp. 1–6, 2024. doi: 10.1109/ADICS58448.2024.10533619.

Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information, 2024. URL https://arxiv.org/abs/2402.13616.

Wei Xu, Xiaodong Du, Ruochen Li, and Lei Xing. Dtc-yolo: Multimodal object detection via depth-texture coupling and dynamic gating optimization. *Sensors*, 25(18), 2025. ISSN 1424-8220. doi: 10.3390/s25185731. URL https://www.mdpi.com/1424-8220/25/18/5731.

Qihang Yang, Yang Zhao, and Hong Cheng. Mmlf: Multi-modal multi-class late fusion for object detection with uncertainty estimation, 2024. URL https://arxiv.org/abs/2410.08739.

Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9656–9665, 2019. doi: 10.1109/ICCV.2019.00975.

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection, 2022. URL https://arxiv.org/abs/2203.03605.

Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li. Single-shot refinement neural network for object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4203–4212, 2018. doi: 10.1109/CVPR.2018.00442.

Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9756–9765, 2020. URL https://api.semanticscholar.org/CorpusID:208637257.

Zixiao Zhang, Xiaoqiang Lu, Guojin Cao, Yuting Yang, Licheng Jiao, and Fang Liu. Vit-yolo:transformer-based yolo for object detection. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 2799–2808, 2021. doi: 10.1109/ICCVW54120.2021.00314.

Zhaohui Zheng, Ping Wang, Dongwei Ren, Wei Liu, Rongguang Ye, Qinghua Hu, and Wangmeng Zuo. Enhancing geometric factors in model learning and inference for object detection and instance segmentation, 2021. URL https://arxiv.org/abs/2005.03572.

Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points, 2019. URL https://arxiv.org/abs/1904.07850.

Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Ling, and Qinghua Hu. Vision meets drones: A challenge, 2018. URL https://arxiv.org/abs/1804.07437.

Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7380–7399, 2021a.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection, 2021b. URL https://arxiv.org/abs/2010.04159.

Zhanhong Zhuo, Ruitao Lu, Yongxiang Yao, Siyu Wang, Zhi Zheng, Jing Zhang, and Xiaogang Yang. Taf-yolo: A small-object detection network for uav aerial imagery via visible and infrared adaptive fusion. *Remote Sensing*, 17(24), 2025. ISSN 2072-4292. doi: 10.3390/rs17243936. URL https://www.mdpi.com/2072-4292/17/24/3936.

# SUPPLEMENTARY MATERIALS

## 5.1 DATASET

Abundant large-scale datasets are available to image object detection track based on monocular and binocular UAV, which comprise ImageNet (ILSVRC) Russakovsky et al. (2015) exceeding 100GB, MS COCO Lin et al. (2014), VisDrone Zhu et al. (2018; 2021a), etc.

PASCAL Visual Object Classes (VOC) Everingham et al. (2015); Everingham et al. is one of the most popular benchmarks in generic object detection, image classification and semantic segmentation tasks, including 20 categories, e.g., person, aeroplane, car and tvmonitor.

ImageNet datasets Russakovsky et al. (2015) was first released by Li Fei-Fei et al. in 2009 and has been continuously improved since 2010 at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). ImageNet Russakovsky et al. (2015) is currently the largest image recognition database in the world, primarily used for image classification and object detection in the field of computer vision. ImageNet datasets Russakovsky et al. (2015) comprises a total of 22,000 image categories, where each category consists of hundreds or even thousands of images, with an average of over 500 per category, approximately 15 million images in total.

VisDrone Zhu et al. (2018; 2021a) is a mainstream benchmark dataset, consists of 4 categories, as follows: (1) Image object detection track (DET), (2) Video object detection track (VID), (3) Single object tracking track (SOT) and (4) Multi-object tracking track (MOT), where DET is utilized in our method. VisDrone enables extensive evaluation and investigation of visual analysis algorithms for the drone platform.

RGB and modalities including panchromatic, infrared, radar, and multispectral are frequently employed. The detailed characteristics of these image types, specifically their visible-light sensitivity, number of channels, and underlying imaging principles, are summarized in Table 4.

Table 4: The Detailed Information on Multimodal Image Types

| Image type | Visible light | Channels | Imaging principle |
|---|---|---|---|
| Visible light image | ☑ | 3 (RGB) | Simulate human perception and capture the red, green and blue light information reflected by objects |
| Panchromatic image | ☑ | 1 (grayscale) | Capture black and white images of the entire visible light spectrum (wider range) |
| Infrared (IR) image | ☒ | 1 (heatmap/grayscale) | The heat radiated by object itself (thermal infrared) or reflected infrared light (near-infrared) |
| Radar image (SAR[2]/millimeter wave) | ☒ | 1 (grayscale/point cloud) | Actively emitting electromagnetic waves and receiving echoes, imaging by calculating the flight time of the beam. |
| Multispectral/ Hyperspectral images | Partially | 3 to hundreds | Divide the spectrum into dozens or even hundreds of very narrow bands for imaging |

Table 5: The Single-modal and Bimodal Image Object Detection Dataset with Details. (1k=1,000, 1p = a pair of images for both modalities)

| Modality | Data | scenario | number of images | | | categories | resolution | space size | year |
|---|---|---|---|---|---|---|---|---|---|
| | | | train | val | test | | | | |
| RGB | PASCAL VOC2012 Everingham et al. | life | 17125 | | - | 20 | $469 \times 387$ | 2GB | 2012 |
| | ImageNet (ILSVRC2012) Russakovsky et al. (2015) | life | 1300k | 50k | 100k | 200 | $482 \times 415$ | 155GB | 2012 |
| | ImageNet (ILSVRC2013) Russakovsky et al. (2015) | life | 395909 | 20121 | 40152 | 200 | $482 \times 415$ | 60GB | 2013 |
| | MS COCO (2017) Lin et al. (2014) | life | 118k | 5k | 41k | 91 | $640 \times 640$ | 28GB | 2017 |
| | UAVDT-DET Du et al. (2018) | drone | 24143 | 0 | 53676 | 4 | $\frac{1080 \times}{540}$ | 13GB | 2018 |
| | VisDrone, DET Zhu et al. (2018; 2021a) | drone | 6471 | 548 | 1610+1580 | 10 | $\frac{2000 \times}{1500}$ | 2GB | 2019 |
| RGB+IR | KAIST Hwang et al. (2015) | life | 50187p | 0 | 45141p | 3 | $640 \times 480$ | 36GB | 2015 |
| | DroneRGBT Peng et al. (2020) | drone | 1807p | 0 | 1800p | 1 | $640 \times 512$ | 1GB | 2020 |
| | DroneVehicle Sun et al. (2022) | drone | 17990p | 1469p | 8980p | 5 | $840 \times 712$ | 13GB | 2022 |

We summarize the corresponding single-modal object detection datasets in terms of various attributes as shown in the Table 5 and evaluate our method on some datasets.

[2]Synthetic Aperture Radar, SAR. The high-resolution images generated by SAR, which can be imaged under various weather conditions and have all-weather observation capabilities.

To evaluate the generalization capability of our model, we further collected and experimented with multiple bimodal datasets. A summary of the detailed information for each dataset is shown in Table 5.

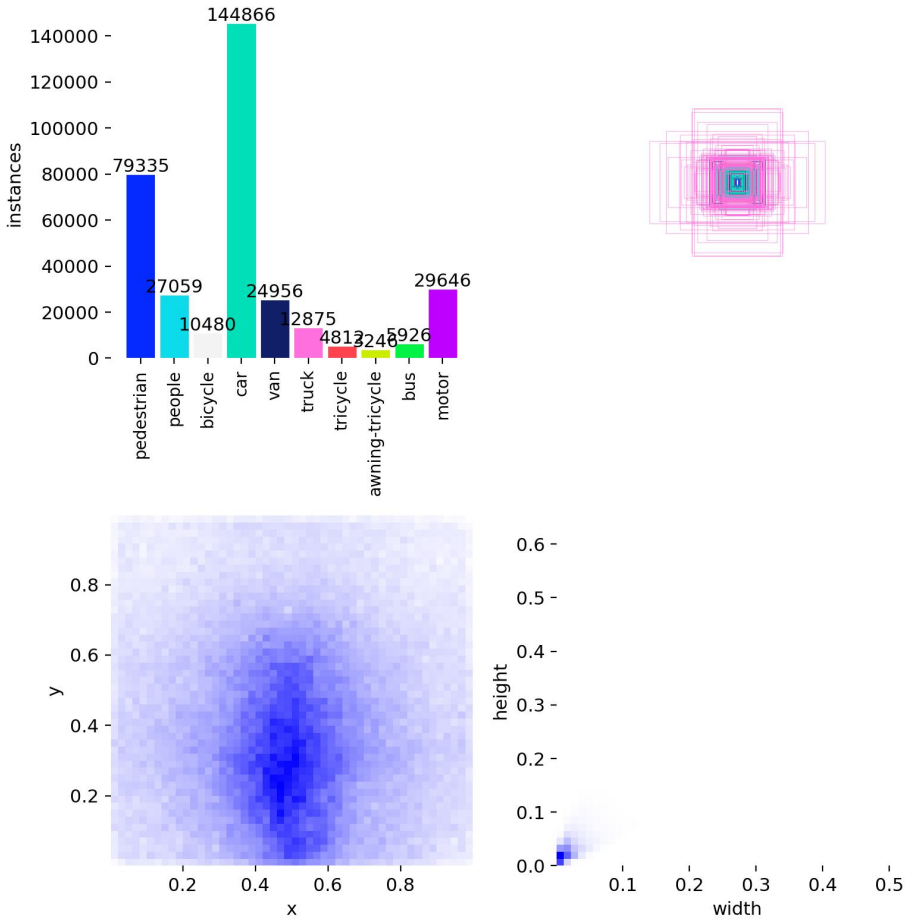



Figure 4: The detailed information of object boxes labels in VisDrone-DET dataset: (1) the instances number for each type of object boxes, (2) the shape of all boxes, (3) the distribution of center coordinates $(cx, cy)$ for all object boxes, and (4) the distribution of width and height for all boxes, these are all normalized format.

**Algorithm 2** Test Data Preprocessing

**Input:** Image $I$, annotations $B$, target size $S = (640, 640)$
**Output:** Transformed image $\tilde{I}$ and annotations $\tilde{B}$

1: Load the input image $I$ and its corresponding annotations $B$.
2: Resize $I$ while preserving its aspect ratio, followed by LetterBox padding to the target size $S$ without scaling up.
3: Transform the bounding boxes $B$ according to the applied resizing and padding.
4: Pack the processed image, annotations, and image metadata as model inputs.
5: **return** $\tilde{I}, \tilde{B}$

For testing, each image is resized while preserving its aspect ratio and then padded to the target size $H \times W$. The bounding box annotations $B$ are transformed according to the resizing and padding operations. The processed images, annotations, and image metadata are then packed as model inputs. The complete procedure is described in Algorithm 2.

### 5.2 METHODS

Furthermore, existing multimodal detectors including TAF-YOLO Zhuo et al. (2025), OAFA Chen et al. (2024), DTC-YOLO Xu et al. (2025), and RGBX-DiffusionDet Orfaig et al. (2026), leverage complementary information from RGB, infrared, or radar modalities, thereby substantially improving detection robustness in challenging environments.

Multimodal image feature fusion methods can be broadly categorized into three paradigms Hangloo & Arora (2010): image-level, feature-level, and decision-level fusion, as illustrated in Fig.5. Notably, the predicted bounding boxes from all feature layers are subsequently processed by non-maximum suppression (NMS) to obtain the final detection results.

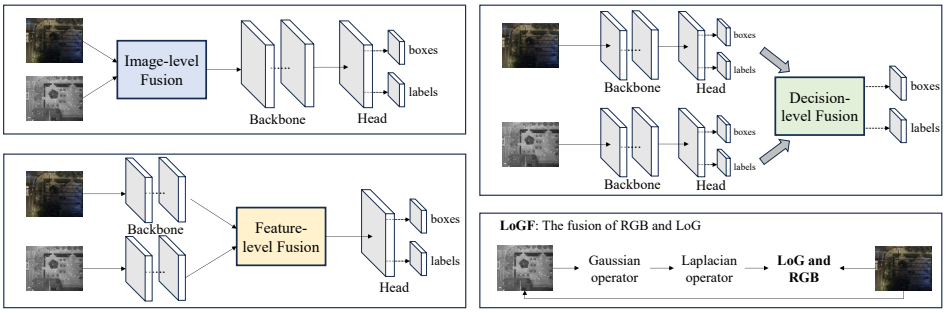



Figure 5: The three types of bimodal feature fusion methods and our proposed LoGF only using unimodal.

Each fusion paradigm offers distinct advantages. Image-level fusion enables the exploitation of powerful off-the-shelf detection architectures, e.g. YOLO Terven et al. (2023) and DETR Carion et al. (2020), by directly combining multimodal inputs at the raw data level. Feature-level fusion Chen et al. (2024); Orfaig et al. (2026) provides the flexibility to adaptively weigh the contributions of two modalities, down to a per-pixel level, thereby enabling dynamic feature re-calibration. In contrast, Decision-level fusion Yang et al. (2024) preserves the full representational capacity of each modality by integrating predictions only at the decision stage.

Zhuo et al.Zhuo et al. (2025) proposed an image-level fusion method called Two-branch Adaptive Fusion Network (TAFNet) along with the TAF-YOLO object detector. This approach adaptively integrates visible and infrared information at both pixel and channel levels prior to feature extraction. By maximizing complementary information while minimizing redundancy across modalities, the fused representation is subsequently fed into the backbone network. DTC-YOLO Xu et al. (2025), adopts an image-level fusion strategy to achieve cross-channel feature alignment between RGB and LiDAR data. It constructs a joint depth-texture representation space before input into the backbone network.

Chen et al.Chen et al. (2024) proposed an offset-guided adaptive feature alignment (OAFA) method, which adaptively adjusts the relative positions between multimodal features. OAFA operates after the backbone network but before the detection head, making it a representative feature-level fusion approach. In RGBX-DiffusionDet Orfaig et al. (2026), each modality is processed by its own backbone consisting of ResNet and FPN. The corresponding feature maps from each modality are fused at each level before being passed to the detection head, which is a feature-level fusion methods.

In addition, there are other methods used to assist in object detection, e.g. the method of labeling bounding boxes for each instance in images and selecting high-quality samples. Adaptive Training Sample Selection (ATSS) Zhang et al. (2020), a classification method of training data, which significantly improves the performance of anchor-based and anchor-free detectors to bridges the gap between them, through automatically select positive and negative samples according to statistical characteristics of object.

Table 6: The Taxonomy of Object Detection Methods.

| Classification criteria | Subcategory | Explanation | Representative methods |
|---|---|---|---|
| Stage | One-stage methods. | The approaches directly predict objects by sampling over dense locations, instead of using the time-consuming region proposal generation module. | YOLOv9 Wang et al. (2024); Terven et al. (2023), SSD Liu et al. (2016), RetinaNet Lin et al. (2017b), FCOS Tian et al. (2019), CenterNet Zhou et al. (2019); Duan et al. (2019), DETR Carion et al. (2020). |
| | Two-stage methods. | The approaches consist of two modules, the first module generate proposal regions for object detection, and the second module predict the accurate bounding boxes and the corresponding category label. | Faster R-CNN Ren et al. (2017), Cascade R-CNN Cai & Vasconcelos (2018), Light-RCNN Li et al. (2017), FPN Lin et al. (2017a), RefineDet Zhang et al. (2018). |
| Anchor[1] | Anchor-based | The methods predefine a set of anchor boxes, and then predict the bounding box location and category label for each instance in an image by anchor boxes. | Faster R-CNN Ren et al. (2017), SSD Liu et al. (2016), RetinaNet Lin et al. (2017b), FPN Lin et al. (2017a). |
| | Anchor-free | The methods directly use various points to predict the bounding box and location for each object instance instead of anchor boxes, e.g. corner-points Law & Deng (2019), center-points Duan et al. (2019) and key-points. | CornerNet Law & Deng (2019), CenterNet Zhou et al. (2019); Duan et al. (2019), FCOS Tian et al. (2019), RepPoints Yang et al. (2019) and its v2 Chen et al. (2020b), YOLOv9 Wang et al. (2024). |
| Network architecture | CNN-based | The algorithms or models with deep convolutional neural network (CNN), this methods employ CNN as the main framework. | Faster R-CNN Ren et al. (2017), SSD Liu et al. (2016), RetinaNet Lin et al. (2017b), YOLOv9 Wang et al. (2024), Cascade R-CNN Cai & Vasconcelos (2018), Light-RCNN Li et al. (2017). |
| | Transformer-based | The object detection methods with Transformer encoder or decoder, and the algorithms or models employ Transformer as the main framework. | DETR Carion et al. (2020), DINO Zhang et al. (2022), Deformable DETR Zhu et al. (2021b), ViT-YOLO Zhang et al. (2021). |
| Training strategy | Supervised learning | The methods require a large amount of labeled image with precise bounding boxes annotation. | YOLOv9 Wang et al. (2024), RetinaNet Lin et al. (2017b), RefineDet Zhang et al. (2018), Faster R-CNN Ren et al. (2017), SSD Liu et al. (2016), Cascade R-CNN Cai & Vasconcelos (2018), Light-RCNN Li et al. (2017), etc. |
| | Weakly supervised learning | The annotation information of the training data for the model is incomplete, imprecise or inaccurate. The model improves its performance by mining the implicit information of existing annotations or utilizing additional knowledge sources. | WSOD Lin et al. (2023), WSDDN Bilen & Vedaldi (2016). |
| | Semi-supervised learning Shehzadi et al. (2024) | The training data of the model comprises a small amount of precisely labeled data and a large amount of unlabeled data. The generalization ability of model is improved by combining labeled and unlabeled data. | SOOD Hua et al. (2023), STAC Sohn et al. (2020). |
| | Self-supervised learning | The model utilizes unlabeled data to generate labels for pre-training, and then fine-tunes on a small amount of labeled data, thereby reducing labeling costs while improving detection performance. | MoCo He et al. (2020), SimCLR Chen et al. (2020a). |
| Application scenarios | General object detection | The methods for detecting common objects. | MS COCO detector Lin et al. (2014) for 80-class object detection, ImageNet Russakovsky et al. (2015) for 200 categories. |
| | Specific domain | The methods for detection in specific domains, e.g. faces, pedestrians and vehicles detection. | RetinaFace Deng et al. (2020), Pedestron Hasan et al. (2020), UAVDT Du et al. (2018), VisDrone Zhu et al. (2021a). |
| | Real-time detection | The methods for real-time object detection. | YOLO-NAS Terven et al. (2023), NanoDet Van Thanh et al. (2025). |
| Modality | RGB | The only RGB-based methods. | YOLO series Terven et al. (2023), CenterNet Zhou et al. (2019); Duan et al. (2019), DETR Carion et al. (2020), DINO Zhang et al. (2022), Deformable DETR Zhu et al. (2021b), etc. |
| | RGB+IR | The methods based on RGB and IR. | TAF-YOLO Zhuo et al. (2025), DTC-YOLO Xu et al. (2025), OAFA Chen et al. (2024), RGBX-DiffusionDet Orfaig et al. (2026). |

### 5.3 EVALUATION METRICS

The core metrics for evaluating performance of object detection methods and semantic segmentation algorithms, comprises **A**verage **P**recision (AP) across different object categories and Intersection over Union (IoU), **A**verage **R**ecall (AR) across different number of predicted boxes. These metrics primarily originate from the MS COCO Lin et al. (2014), where the mean Average Precision (mAP) is the main metric commonly used.

The AP, AR and their derived metrics are calculated based on the Precision and Recall rate with the following two core concepts. Define the IoU threshold $\tau$ and confidence threshold $\theta$.

- IoU: The intersection over Union between the predicted and ground truth boxes, which is used to measure the accuracy of localization.
- Confidence Level: The degree of certainty the model predicts object and category within the bounding boxes.

The trained model outputs a large number of prediction boxes with categories, confidence scores and coordinates for each image. To evaluate the trained model, all prediction boxes are categorized into True-Positive (TP), False-Positive (FP), False-Negative (FN) and True-Negative (TN) samples according to IoU and confidence threshold. The detailed meanings of the four fields are shown in the Table 7.

In the research field of object detection, the positive means that real object box or that having been successfully detected. The detailed meanings of TP, FP, FN, TN are as follows.

- TP: When the prediction box label is correctly predicted and the IoU is greater than corresponding threshold $\tau$, the object prediction box with the highest confidence score is defined as TP samples.

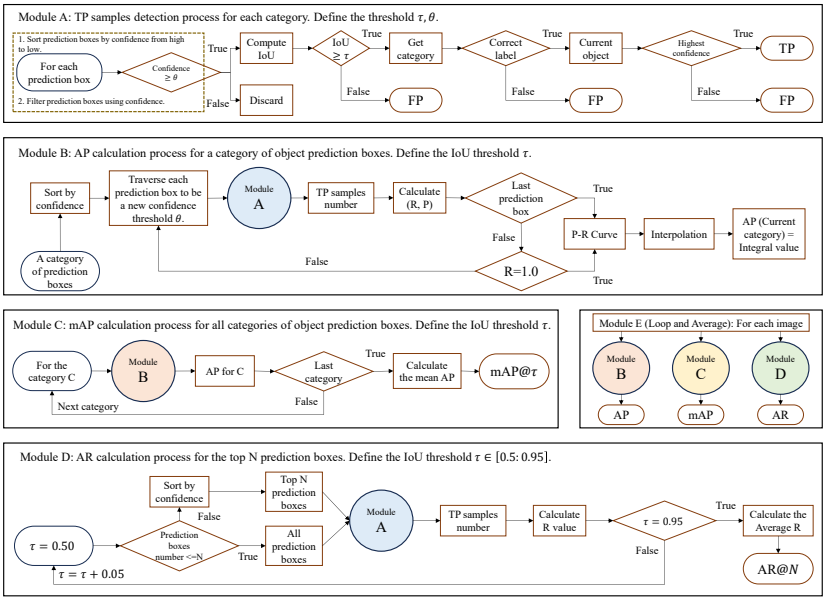



Figure 6: The calculation processes of various evaluation metrics, which includes TP, AP@$\tau$, mAP@$\tau$, AR@N, correspond to modules A, B, C and D. The module E represents the loop execution of modules B, C, D for all image, and calculates the average value to get the corresponding metrics.

Table 7: The Confusion Matrix of TP, FN, FP, TN.

| | Predicted as positive | Predicted as negative |
|---|---|---|
| Actually positive | TP | FN |
| Actually negative | FP | TN |

- FP: The prediction category error under IoU is above the threshold $\tau$, or the prediction box label is correct but the IoU is below the threshold $\tau$, or the prediction box except the highest confidence score if IoU is above the threshold $\tau$, which is defined as FP samples.

- FN: The real object bounding box but not detected by the model, which is defined as FN samples.

- TN: The object detection does not count the number of TN, due to the large amount of negative samples in the background area with no real object boxes and not detected by model.

Furthermore, the Accuracy (Acc), Precision (P) and Recall (R) are calculated based on the above four metrics, as shown in Eq.8. Given the IoU threshold $\tau$, R and P are calculated according to the change of confidence threshold $\theta$ from high to low, this process can generate a series of (R, P) points, i.e. the P-R curve, and then calculate the AP metric for each category. TP is the critical metrics to calculate P-R values from the Eq.8, AP is the integral value of P-R curve. When TP is the maximum, $\text{TP} = \min\{\text{all real boxes}, \text{all prediction boxes}\}$, and then $\text{P} = 1$ or $\text{R} = 1$.

$$\begin{cases} Acc = \dfrac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \\ P = \dfrac{\text{TP}}{\text{TP+FP}} = \dfrac{\text{TP}}{\text{The number of all prediction boxes}} \\ R = \dfrac{\text{TP}}{\text{TP+FN}} = \dfrac{\text{TP}}{\text{The number of all real boxes}} \end{cases} \tag{8}$$

AP is the evaluation metric for one category, mAP is the mean AP across all categories, and AR is the average value of the Recall across all IoU thresholds $\tau = [0.5 : 0.95, 0.05]$. The calculation processes of various evaluation metrics are shown in Fig.6, comprise TP, AP@$\tau$ and mAP@$\tau$, AR@N.

The common AP metrics of object detection include $\text{AP}_{50}$ (AP@0.5) and $\text{AP}_{75}$ (AP@0.75), where $\text{AP}_{50}$ is calculated under IoU threshold $\tau = 0.5$, similarly the $\text{AP}_{75}$ means that IoU threshold $\tau = 0.75$. Correspondingly, mAP@0.5 is mean value of AP@0.5 across all categories, mAP@0.75 is the same principle for more higher IoU and more precise location.

Furthermore, originated from the COCO standard Lin et al. (2014), AP@ $[0.5 : 0.95, 0.05]$ and mAP@ $[0.5 : 0.95, 0.05]$ mean that calculate the metrics at 10 different IoU thresholds from $\tau = 0.5$ to $\tau = 0.95$ with a step size of 0.05, and then calculate average to get results. The mAP@ $[0.5 : 0.95, 0.05]$ can comprehensively evaluate the performance of model under various rigorous requirements, is the most stringent comprehensive indicator Lin et al. (2014).

The common AR metrics include $\text{AR}_1$, $\text{AR}_{10}$, $\text{AR}_{100}$ and $\text{AR}_{500}$, the calculation process is presented in Fig.6. AR@$N$ means that calculate the AR of model across different IoU thresholds under the maximum number $N$ of detection boxes allowed to be output per image.

mAP measures the proportion of positive samples among all predicted bounding boxes, averaged over all categories and confidence thresholds. In contrast, AR quantifies the fraction of ground-truth objects that are successfully detected by the model among all truth bounding boxes. Furthermore, $\text{mAP}_s$, $\text{mAP}_m$, and $\text{mAP}_l$ denote the average precision evaluated separately on small, medium, and large objects, where object size is defined by its pixel area. Specifically, small objects are those with area $< 32^2$ pixels, medium objects satisfy $32^2 \leq$ area $< 96^2$ (1024 to 9216 pixels), and large objects have area $\geq 96^2$ pixels.

### 5.4 Model Configuration

**Loss function.** The bounding box regression loss, termed the box loss, is responsible for localization and is implemented using the Complete IoU (CIoU) Loss Zheng et al. (2021). This metric not only accounts for the intersection over union (IoU) between the predicted and ground-truth boxes but also incorporates the distance between their center points and the consistency of their aspect ratios.

The classification (cls) loss handles the identification of predicted box labels and employs the Binary Cross-Entropy with Logits Loss. The distribution focal loss (DFL) is designed for boundary refinement, with particular efficacy for small objects or those with ambiguous edges, thereby enhancing sub-pixel localization accuracy. Unlike conventional regression approaches that directly predict a single coordinate value, DFL learns a probability distribution over coordinate values.

### 5.5 Training Configuration

The same mean and standard deviation are used for image normalization throughout all experiments.

mean=[0., 0., 0., 0.] or mean=[0., 0., 0., 0., 0., 0.]

std=[255., 255., 255., 255.] or std=[255., 255., 255., 255., 255., 255.]

### 5.6 Experiment Supplement

kernel size=5, $\sigma = 1.0$, Gaussian kernel initialization:

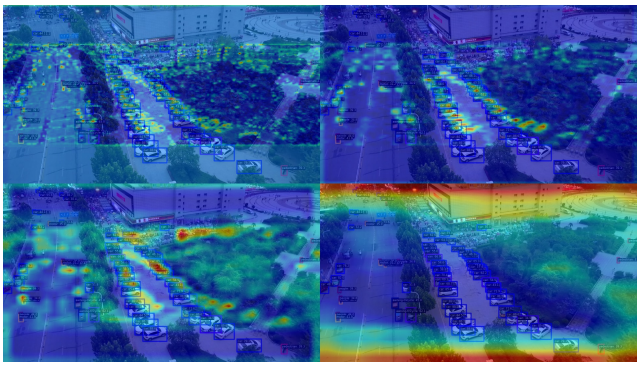
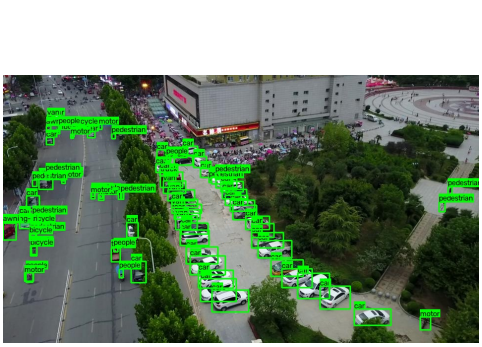


Figure 7: The backbone feature maps at different downsampling stride, which includes P2 (stride=4, size=160 $\times$ 160), P3, P4, P5, respectively.



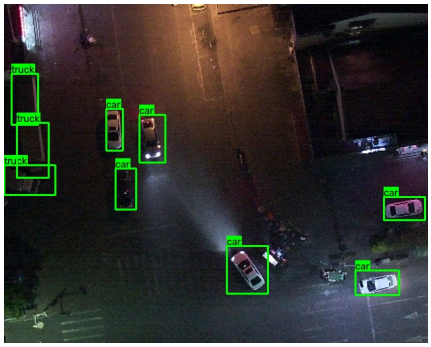


Figure 8: Ground-truth examples from two datasets: VisDrone on the left and DroneVehicle on the right.

$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}$$

$G_{k,\sigma}$ means Gaussian Kernel Size $k$ and $\sigma$.

$L_4$ means Four neighborhood kernel and $L_8$ eight neighborhood kernel.

$L_4$:

$$\begin{bmatrix} 0. & 1. & 0. \\ 1. & -4. & 1. \\ 0. & 1. & 0. \end{bmatrix}$$

$L_8$:

$$\begin{bmatrix} 1. & 1. & 1. \\ 1. & -8. & 1. \\ 1. & 1. & 1. \end{bmatrix}$$

Moreover, the P2 downsampling layer provides finer-grained features, as shown in Fig. 7, obtained by averaging across all channels. These features are particularly useful for small-object detection, but come at the cost of a substantial increase in parameters and computation. Thus, whether to use P2 depends on the specific application.

**Effect of +P2 on Performance.** As shown in Table 2, adding the P2 detection scale to the baseline improves the VisDrone performance by 3.3 mAP, but decreases the DroneVehicle performance by 1.0 mAP. This difference can be attributed to the object size distributions of the two datasets, as illustrated in Fig. 8.

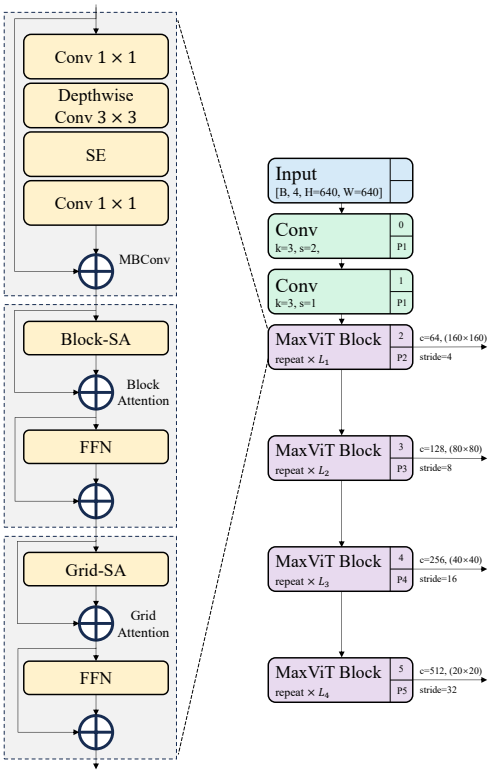



Figure 9: MaxViT architecture. For MaxViT-T, the numbers of blocks in the four stages are $(L_1, L_2, L_3, L_4) = (2, 2, 5, 2)$, with output channels of 64, 128, 256, and 512 for the P2–P5 feature levels, respectively.

The MaxViT-T architecture is shown in Fig. 9, while the Neck and Head remain unchanged. The P2 detection head is not used.

To limit the additional parameter overhead, we use three channels to represent the introduced frequency features. The selection and combination of frequency components could be further explored to improve the representation. Nevertheless, the above experiments consistently show that introducing prior frequency features improves mAP across different backbone architectures and datasets.