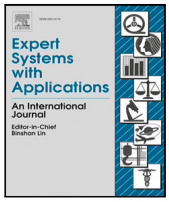
Expert Systems With Applications 256 (2024) 124848

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



EL-YOLO: An efficient and lightweight low-altitude aerial objects detector for onboard applications

Chen Xue, Yuelong Xia, Mingjie Wu, Zaiqing Chen, Feiyan Cheng, Lijun Yun *

School of Information Science and Technology, Yunnan Normal University, Kunming 650500, PR China
Engineering Research Center of Computer Vision and Intelligent Control Technology, Department of Education of Yunnan Province, Kunming 650500, PR China

ARTICLE INFO

Keywords:
Unmanned aerial vehicle
Small object detection
Edge computing
Feature pyramid network
Attention mechanism

ABSTRACT

Existing deep learning-based low-altitude small object detectors are typically complex in model architecture and demand substantial computational resources, making deployment for real-time detection tasks on edge computing devices challenging. To address this issue, we proposed EL-YOLO, an efficient and lightweight onboard applications object detector. Initially, in order to maintain a lightweight design and improve model accuracy, we propose a novel approach called Sparsely Connected Asymptotic Feature Pyramid Network(SCAFPN). This approach aims to eliminate inter-layer interference during feature fusion, thereby enhancing the model's performance. Furthermore, to establish long-range contextual relationships for small object scale information, we devise a Cross-Space Learning Multi-Head Self-Attention mechanism (CSL-MHSA). To assess EL-YOLO capability for onboard small object detection, we deploy it on the embedded NVIDIA Jetson Xavier Nx platform and employ NVIDIA TensorRT FP16 quantization acceleration. On the VisDrone2019-DET and AI-TOD datasets, EL-YOLO demonstrates a 12.4% and 1.3% improvement in mAP50 compared to YOLOv5s. In comparison with the state-of-the-art YOLOv8s proposed in 2023, it achieves a respective 2.8% and 10.7% increase in mAP50. Moreover, the inference speed for the Small model reaches 24 frames per second, while the Nano model achieves 35 frames per second. This method holds promise for direct integration onto unmanned aerial vehicles for aerial small object detection tasks in the future.

1. Introduction

In recent years, unmanned aerial vehicles (UAVs) have been equipped with increasingly powerful imaging devices. Thanks to their higher flight capabilities and flexibility, they can capture aerial image information with rich perspectives, heights, and positions, providing a growing amount of high-quality aerial imagery data for UAV aerial target detection. Currently, UAVs are widely employed in various fields such as military, rescue operations, agriculture, and surveillance, including applications like intelligent parking (Rafique et al., 2023), real-time vehicle detection (Hamzenejadi & Mohseni, 2023), and target tracking (Ma et al., 2024). Due to the unique characteristics of UAV aerial images, conventional target detectors often face challenges when detecting low-altitude aerial images. These challenges include issues such as motion blur during flight, small object sizes, uneven object distribution, and variations in environmental conditions due to multi-angle captures. Particularly, the high level of detail provided by the spatial resolution of the images presents both an advantage and a challenge. The aforementioned challenges lead to issues in well-performing detectors on low-altitude aerial datasets, such as slow inference speed, high memory usage, and large parameter sizes. Presently, deep learning methods dominate the field, with numerous emerging techniques (Lin, Goyal et al., 2017; Liu et al., 2016a; Zhang et al., 2021) indicating the future potential of this area. With the rapid development of UAVs and deep learning, their combination holds the promise of more efficient working methods. The demand for efficient algorithms and models capable of running on UAVs with onboard computing resources is becoming increasingly important. Meeting these requirements makes the development of accurate and real-time detection models that can effectively operate an urgent matter.

Target detection methods based on deep learning Convolutional Neural Networks (CNN) are commonly categorized into one-stage and two-stage approaches. One-stage methods perform direct object detection and position prediction on individual images without the need to explicitly generate candidate regions. In contrast,two-stage methods first generate candidate regions where targets may exist and then classify and perform position regression on these regions. While two-stage

* Corresponding author at: School of Information Science and Technology, Yunnan Normal University, Kunming 650500, PR China.
E-mail addresses: 2223410013@ynnu.edu.cn (C. Xue), xyl@ynnu.edu.cn (Y. Xia), 2224100006@ynnu.edu.cn (M. Wu), chenzaiqing@ynnu.edu.cn (Z. Chen), chengfy@163.com (F. Cheng), yunlijun@ynnu.edu.cn (L. Yun).

https://doi.org/10.1016/j.eswa.2024.124848
Received 8 January 2024; Received in revised form 30 June 2024; Accepted 19 July 2024
Available online 23 July 2024

methods often achieve higher detection accuracy, they are relatively slower. Addressing the challenge of recognizing objects of different sizes has been a persistent issue in target detection. The uncertainty of object sizes in images can result in the loss of detailed information, particularly for small targets. The Feature Pyramid Network (FPN) structure, notably the classic FPN (Lin, Dollár et al., 2017), has been instrumental in addressing multi-scale target detection challenges. Feature Pyramid Networks focus on resolving multi-scale problems in object detection, enhancing the performance of small object detection by modifying simple network connections. Since its introduction, FPN has paved the way for various fusion methods based on feature pyramids, such as the AFPN (Yang et al., 2023) structure. AFPN initiates by merging adjacent low-level features and progressively integrates higher-level features into the fusion process, effectively mitigating the substantial semantic gap between non-adjacent levels. However, these approaches involve complex network structures for feature extraction and fusion during inference, resulting in high model complexity and computational demands. This complexity limits their real-time applicability and deployment on resource-constrained edge devices, such as unmanned aerial vehicles (UAVs), for detection tasks.

Given the aforementioned challenges, addressing issues of computational power and memory constraints in practical applications has led to a heightened focus on model lightweighting in recent years, with rapid developments. Common lightweight models include the MobileNet (Howard et al., 2017) series, GhostNet (Han et al., 2020) series, and ShuffleNet (Zhang et al., 2018) series. In this paper, we specifically focus on aerial small target detection. While current lightweight models exhibit excellent inference speed, they may fall short of meeting the precision requirements for practical civilian applications. Therefore, striking a balance between model complexity and inference accuracy poses a challenging task. In the context of CNN-based object detection in aerial images, three major challenges are identified (Liu et al., 2021):

- The low-level feature maps exhibit a higher resolution but lack rich semantic information, while, conversely, high-level feature maps contain advanced semantic details but are insensitive to small targets, rendering them insufficient to support the performance of small target detection. Consequently, a single layer of either high or low-level feature map cannot meet the requirements of small target detection tasks. Therefore, fusing low-level and high-level features to acquire necessary spatial and semantic information represents an effective solution.
- The low resolution of images containing small targets impedes the provision of substantial information, making the identification of small objects challenging. Hence, context information plays a pivotal role in small object detection (Torralba et al., 2003). The proposed solution involves integrating contextual information into the detection network to establish long-range dependency relationships.
- The presence of class imbalance in datasets is attributed to the scarcity of ground truth bounding boxes. IoU matching strategies between ground truth and anchors, where negative examples influence positive examples, can result in models favoring the negative class. Addressing this challenge involves enhancing the balance between positive and negative examples directly during the training process.

In response to these issues, we propose an efficient and lightweight low-altitude aerial objects detector for onboard applications. The contributions of this paper are as follows:

1. In response to the first challenge, we propose a feature fusion module with a sparsely connected progressive feature fusion pyramid structure. This module aims to eliminate semantic gaps that arise during the fusion of features from non-adjacent hierarchical levels, thereby enhancing the accuracy of small target detection in the model.
2. We design a Cross-Spatial Learning Multi-Head Self-Attention mechanism (CSL-MHSA) to focus the model on detecting multi-scale targets, facilitating the extraction of richer image features through establishing long-range relationships.
3. To enhance the attention to small targets, we reorganize the backbone by employing SPD-Conv and DSConv, further optimizing the network model. This not only improves the accuracy of small target detection but also meets the requirements for model lightweighting. Additionally, we adopt a label assignment strategy based on optimization (SimOTA) to enable the network to autonomously learn the optimal label assignment.

Building upon the aforementioned innovations, we present an onboard target detector named EL-YOLO. We deploy EL-YOLO on the embedded platform NVIDIA Jetson Xavier Nx and apply NVIDIA TensorRT FP16 quantization acceleration to evaluate its onboard target detection capability.

The structure of this paper is as follows: Section 2 provides a brief introduction to target detection, particularly in the literature related to small targets. Section 3 elaborates on the innovative aspects and implementation details of our proposed model. Section 4 presents details and results of relevant experiments, along with an analysis of the outcomes. Finally, Section 5 summarizes and concludes this article.

2. Related work

2.1. Object detection

With the research and continuous development of artificial intelligence technology, techniques such as deep learning and computer vision have found widespread applications in areas such as facial recognition, modern healthcare, and autonomous driving. The goal of computer vision is to analyze and extract features from visual image information, enabling these features to be understood by computers, facilitating tasks such as classification, detection, and segmentation based on similar features. Object detection, a fundamental problem in computer vision, stands out as one of the most important and challenging tasks in the field.

Object detection models can be broadly categorized into two main approaches: two-stage detection and one-stage detection. Two-stage detection methods involve a two-step process for object detection, encompassing candidate region extraction and refined classification. In the candidate region extraction stage, a series of candidate boxes is generated, followed by classification and localization refinement in the subsequent stage. Representative models of two-stage detection include: SPPNet, introduced by He et al. (2015), innovatively incorporates a Spatial Pyramid Pooling (SPP) layer within CNNs, allowing fixed-length representations regardless of image size without rescaling. However, SPPNet's multi-stage training and focus on fine-tuning only the fully connected layers, neglecting prior layers, were addressed by the subsequent Fast R-CNN in the following year. Fast R-CNN (Region-based Convolutional Neural Networks) Girshick (2015): It operates by extracting convolutional features across the entire image and subsequently generating candidate boxes using methods such as selective search.

Faster R-CNN (Region-based Convolutional Neural Networks) (Ren et al., 2017): Faster R-CNN represents an improved version built upon the foundation of Fast R-CNN. It introduces a sub-network known as the Region Proposal Network (RPN) to generate candidate boxes, eliminating the need for external methods like selective search. In 2017, Lin, Dollár et al. (2017) introduced Feature Pyramid Networks (FPN), addressing limitations of previous deep learning detectors that focused on the top layer's feature maps. FPN employs a top-down architecture with lateral connections to integrate high-level semantics at all scales, leveraging the inherent feature pyramid structure in CNNs. Applied in a basic Faster R-CNN system, FPN achieves state-of-the-art

single-model object detection results on the COCO dataset, establishing itself as a fundamental component in contemporary detectors. Although two-stage methods, like Faster R-CNN, exhibit higher detection accuracy, their inherent characteristics often result in slower processing compared to one-stage methods. Additionally, the higher complexity in design and training demands increased computational resources.

One-stage object detection methods redefine the conventional approach by treating the entire process as a unified task, seamlessly integrating object classification and localization in a single step. This eliminates the need for a distinct candidate region generation phase. Among the noteworthy one-stage models, SSD (Single Shot MultiBox Detector) (Liu et al., 2016b) stands out for its utilization of multiple layers of feature maps. Another influential model in the realm of one-stage detection is YOLO (You Only Look Once) (Redmon et al., 2016). YOLO takes a unique approach by dividing the image into a grid and predicting both the target category and location for each grid cell. This design not only facilitates real-time performance but also maintains simplicity. YOLOv4 (Bochkovskiy et al., 2020), a variant of the YOLO series, introduces the Mosaic data augmentation method, addressing challenges such as limited samples in the dataset and enhancing the detection of smaller targets. Building upon the success of YOLOv4, the YOLOv5 (Jocher, 2020) algorithm introduces an adaptive anchor box mechanism for dynamic image scaling. Incorporating the Focus module and CSPDarknet53 structure as the backbone enhances feature extraction capabilities. Notably, YOLOv5 differentiates itself by offering five distinct model sizes, namely YOLOv5n, YOLOv5s, YOLOv5 m, YOLOv5l, and YOLOv5x. The evolution of the YOLO series continues with YOLOv6 (Li et al., 2022), where Chuyi Li and collaborators introduce a self-distillation strategy and selectively incorporate advanced techniques like label assignment, loss functions, and data augmentation to refine performance. In the subsequent months, YOLOv7 (Wang, Bochkovskiy et al., 2023), as proposed by Wang, Bogachevsky, and Liao in 2022, brings forth the ELAN (Extension Layer Aggregation Network) concept. This introduces a novel backbone named "Extension Layer Aggregation (ELAN)" and incorporates auxiliary detection heads to significantly enhance accuracy, setting it apart from other YOLO-based models. A more recent addition to the YOLO family is YOLOv8 (Jocher et al., 2023), released by the YOLOv5 team in January 2023. While research papers are currently unavailable, the model introduces a novel architecture, anchor-free object localization, C2F blocks, and innovative online image augmentation techniques, as gleaned from reference code documentation.

In summary, the existing structural configurations of detection models have matured considerably, exhibiting remarkable precision. However, these models rely on intricate network architectures and substantial computational resources, posing challenges for deployment on resource-constrained edge computing platforms and hindering their suitability for practical operational needs. Though our improvements were implemented within the YOLOv5 framework, the proposed structural concepts are generalizable. Initially, we conducted comparisons based on the fundamental parameters of the models. Typically, a higher number of model parameters allows for learning richer features, but it may also lead to overfitting and increased computational demand. As observed in the experimental results presented in Chapter 4, it is evident that YOLOv8 exhibits higher accuracy than YOLOv5; however, YOLOv8's parameter and computational load are substantially higher than YOLOv5. Furthermore, as the edge computing devices utilized in this study share system memory and graphics memory, we had to consider the computational resource consumption involved in real-time video stream acquisition and video stream transcoding operations. Therefore, given the constraints of limited computational resources, we selected the relatively smaller YOLOv5 model as a baseline in order to achieve faster inference speeds.

2.2. Lightweight network

The lightweight models for object detection refer to models with smaller model sizes and lower computational resource requirements in the context of object detection tasks. This category includes model compression techniques such as pruning, quantization, low-rank decomposition, and knowledge distillation. These lightweight models aim to maintain high detection accuracy while reducing model complexity and inference time. The underlying network architecture primarily involves the use of computationally faster convolutions to decrease computational load and model complexity.

For instance, MobileNet (Howard et al., 2017), proposed by the Google team, features a core convolutional layer that utilizes depthwise separable convolution. This convolutional approach adopts a dimensionality reduction followed by dimensionality increase, significantly reducing computation compared to conventional convolutional layers while maintaining the same output feature maps. ShuffleNet (Zhang et al., 2018), introduced by Megvii Technology (formerly known as Face++) for efficient CNN models on mobile devices, addresses the substantial computational load of the 1×1 pointwise convolutions in depthwise separable convolutions. It proposes a channel-wise sparse connection method, such as group convolution, to lower computational complexity. GhostNet (Han et al., 2020), proposed by Huawei Noah's Ark Lab, tackles redundancy issues in feature maps obtained from regular convolutional operations. Limitations in addressing false detections and challenging scenarios prompted the creation of YOLONL (Zhou, 2024), which integrates dynamic label assignment, enhanced CSPNet and PANet, along with Rep-CSPNet for faster inference, improving multi-scale object detection in complex environments. Wang et al. (2024) introduce Large Kernel Attention (LKA) technology to enhance object detection for autonomous vehicles, reducing computational costs while maintaining accuracy by decoupling large kernel convolutions. However, while reducing model parameters and computation, a decline in precision inevitably occurs.

Model compression techniques arise due to limited computing power and memory in embedded devices, necessitating the compression of deep learning models for deployment on such devices. Knowledge distillation, a form of transfer learning, involves transferring "knowledge" from a complex teacher model with strong learning capabilities to a simpler student model. This approach enables small models to possess lower parameter counts and model complexity while maintaining detection accuracy comparable to larger models. Pruning involves removing relatively "unimportant" weights from the weight matrix (setting them to 0), reducing computational resource consumption, and improving real-time performance. Model quantization entails converting the neural network's floating-point algorithm into fixed-point representation, often in the form of low-precision models, such as converting from conventional FP32 (32-bit floating point, single precision) to FP16 (half-precision floating point) or INT8 (8-bit fixed-point integer). These model compression techniques optimize aspects such as reducing model size and alleviating storage pressure.

In our designed model, striking a balance between accuracy and speed, we opted for a lightweight design by refining the depth, width, and feature fusion components of the neck network structure. Additionally, we employed TensorRT quantization technology for the final model optimization.

2.3. Small object detection in aerial images

Due to the inherent flight characteristics of unmanned aerial vehicles (UAVs), captured image data is prone to complexities such as intricate backgrounds, multiple angles, varying platform heights, diverse weather conditions, lighting variations, motion blur, and the presence of small-sized targets. These factors significantly impact the visual features of targets and the overall image quality. However,

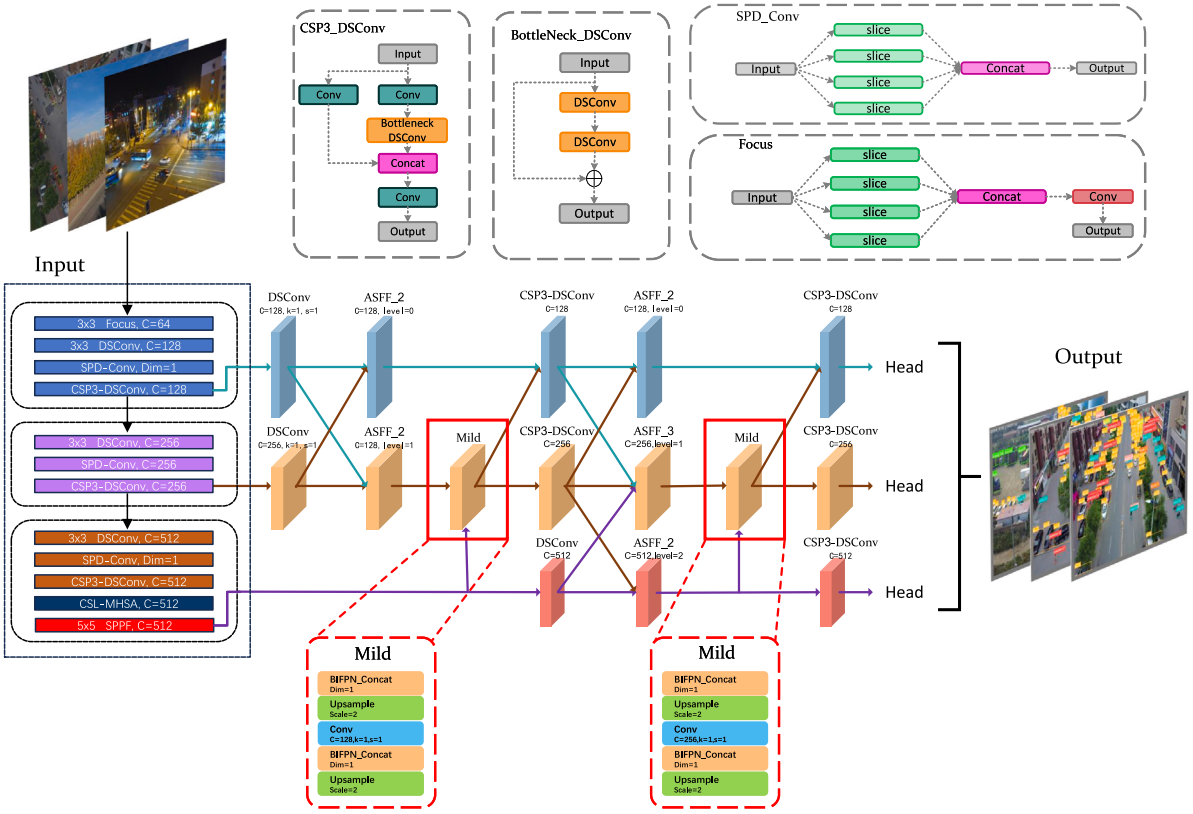


Fig. 1. The overall structure of EL-YOLO. The lines in cyan, brown, and purple represent the flow of data for each layer, corresponding to output feature map sizes of 160 × 160, 80 × 80, and 40 × 40, respectively.

relatively speaking, they also contribute to the enrichment of a more diverse set of image data.

This paper focuses specifically on the challenge of small target detection in aerial images. In recent developments addressing small target detection algorithms, TPH-YOLOv5 (Zhu, Lyu et al., 2021) integrates the Transformer concept into the YOLOv5 architecture. This integration aims to enhance the network's capability to capture more diverse and informative feature information and improve prediction accuracy. Simultaneously, an attention mechanism is employed to increase focus on small targets. QueryDet (Yang et al., 2022) adopts a query mechanism to accelerate the inference speed of the object detector. It utilizes low-resolution features to predict rough positioning, guiding high-resolution features for more precise predictive regression. SMFF-YOLO (Wang, Zou et al., 2023), employing innovative techniques such as multi-level feature fusion, ELAN-SW detection prediction heads, and bidirectional feature fusion pyramids, effectively enhances the detection accuracy of small objects. ETAM (Zhang, Xia et al., 2023), a Transformer-based model with attention mechanisms, introduces technologies like the Magnifying Glass (MG) module and Quadruple Attention (QA) to enrich feature information for small targets. FDLR-Net (Xiao et al., 2023), designed for object detection in remote sensing images, enhances object localization and classification accuracy through bidirectional feature fusion modules, feature decoupling modules, and localized optimization modules. DSNet (Zhang, Jia et al., 2023) employs an innovatively designed density-aware module to adaptively perceive vehicle density. By integrating contextual information and generating appropriate weights, it enhances vehicle features. Info-FPN (Chen et al., 2023) addresses issues such as channel information loss, feature misalignment, and computational burden. Through the introduction of modules like PSM, FAM, and SEM, it improves the effectiveness of multi-scale target feature extraction. Dong et al. (2022) proposed an enhanced lightweight version of the YOLOv5 method aimed at addressing challenges in vehicle detection technology, such as high computational load and suboptimal detection rates. The study by Mahaur et al. (2023) enhances the performance of deep learning-based object detectors in autonomous driving applications, focusing on the challenging task of detecting small objects like traffic signs and lights. The proposed method, FocusDet Shi et al. (2024), integrates a specialized backbone, feature fusion structure, and a detection head, leveraging techniques such as STCF-EANet for feature extraction and SIOU-SoftNMS for post-processing. The study by Sun et al. (2024) demonstrates that GD-PAN significantly enhances small object detection performance by integrating classic path aggregation networks with a gather-and-distribute mechanism and introducing wider coordinate attention in deformable ConvNets v2. The study by Li et al. (2024) introduces the MCSTA module, which utilizes multi-head channel and spatial trans-attention for improved feature capture in object detection for remote sensing images, aiming to enhance computational efficiency and interrelationship capture across both channel and spatial dimensions.

In summary, the current development in small object detection is advancing rapidly, with a substantial number of researchers actively involved. However, we have observed that the majority of models designed for small object detection primarily focus on improving detection accuracy, leading to significant model complexity that hinders deployment on edge computing platforms. Therefore, in addressing this issue, our research aims to balance detection accuracy and inference speed by proposing an efficient and lightweight object detection model.

3. Methodology

3.1. The overview of methods

The overall structure of the network proposed in this paper is shown in Fig. 1. The proposed structure is based on YOLOv5(Jocher,

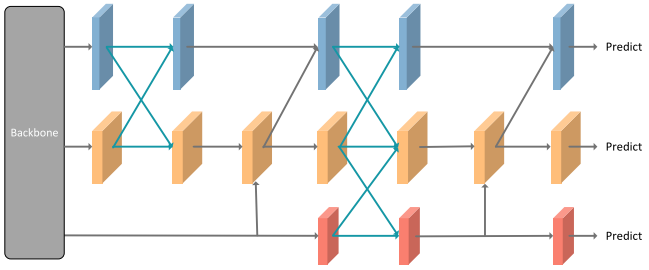


Fig. 2. The architecture of the proposed Sparsely Connected Asymptotic Feature Pyramid Network (SCAFPN). SCAFPN integrates low-level features in the initial stage, then utilizes an intermediary module to transmit high-level features to the lower levels, ultimately incorporating the highest-level features into the network. Gray arrows denote convolutional operations, while light green arrows signify adaptive spatial fusion.

2020) v7.0 which was released in 2022. Our model consists of four parts: (1) an SCAFPN for feature fusion module; (2) an CSL-MHSA module to enhance information aggregation ability of the model at different spatial dimension direction; (3) We conducted lightweight optimization of the entire network using DSConv. Additionally, we restructured the backbone by integrating SPD Conv, which is sensitive to small targets, and CSL-MHSA module to extract more comprehensive information from small targets. Finally, we optimized the network's label assignment strategy using SimOTA. (4) We employed the Tensor Runtime engine (NVIDIA TensorRT) to perform FP16 quantization on the model, thereby expediting the deep learning computations during inference. This approach has notably elevated the overall performance of our deep learning model in terms of inference speed.

This lightweight detection network comprises three components: backbone, feature fusion, and head. To address the challenge of achieving high accuracy in lightweight networks, we devised a sparsely connected progressive feature pyramid structure, commencing with the feature fusion network. Furthermore, in optimizing the neck network for enhanced lightweight characteristics in this design, we subsequently reconfigured the backbone section to align with the sparsely connected asymptotic feature pyramid structure. As a consequence of the backbone network's reorganization, leading to a challenge in capturing deeper semantic information during feature extraction, we propose a cross-spatial learning multi-head self-attention mechanism to enhance the network's feature aggregation capability. The specific details of these methods will be found in the following subsection.

3.2. The structure of sparsely connected asymptotic feature pyramid network (SCAFPN)

3.2.1. The overview of SCAFPN

The structure of SCAFPN is shown in Fig. 2. Since the advent of traditional Feature Pyramid Network (FPN) (Lin, Dollár et al., 2017), the employed method for feature fusion has consistently been top-down. However, during this process, there is no fusion of high-level features with low-level features. PANet(PAFPN) (Liu et al., 2018) was the first to propose a bottom-up secondary fusion structure. The introduction of PANet has driven the development of bidirectional fusion. Nevertheless, due to the relatively simple fusion method adopted by PANet, many subsequent studies have experimented with more complex feature fusion operations, such as AFPN (Yang et al., 2023), NAS-FPN (Ghiasi et al., 2019), and BI-FPN (Tan et al., 2020), building upon the foundation of FPN.

The proposal of AFPN aims to address the issue of information degradation or loss during the interaction and propagation of low-level feature information. It initiates the fusion process with low-level features and gradually incorporates high-level features. This fusion approach alleviates the large semantic gap that may exist in direct fusion between non-adjacent levels. However, semantic disparities between non-adjacent layers persist, and conflicts in multi-object information arise during feature fusion at each spatial position. To solve the above problem, this paper proposes a sparsely connected asymptotic feature pyramid network.

In the first stage, SCAFPN continues to adopt the asymptotic fusion approach used by AFPN in feature fusion. It still employs two low-level features of varying resolutions for fusion initiation. Due to the semantic gap and increased structural complexity resulting from dense cross-scale fusion, in the later stages, we restrict feature fusion to adjacent layers only. We designed an intermediate layer called 'Mild' to facilitate the gentle propagation of high-level features to lower layers. This replaces the previous dense connection method, reducing network parameters and enhancing the semantic information of lower-level features. Consequently, during the training process, the network can effectively mitigate the performance degradation caused by the large semantic gaps between non-adjacent layers and alleviate conflicts in multi-object information during the feature fusion process.

The SCAFPN is a versatile network structure applicable to other backbone networks. Due to the speed requirements addressed in this paper and YOLO's high inference speed, which better meets the demands of real-time object detection, we chose to integrate it with the single-stage YOLOv5 backbone. YOLOv5 draws inspiration from the design principles of CSPNet, employing a CSP structure in the backbone network composed of several stacked convolutional layers and C3 layers. At the highest level of the backbone network, a spatial pyramid pooling module (SPPF) is incorporated. The neck network adopts an FPN+PAN structure, where the FPN layer conveys rich semantic information from top to bottom, and the PAN layer conveys accurate localization features from bottom to top. Finally, the head network utilizes three detection heads at different scales: 20×20, 40×40, and 80×80, effectively addressing the issue of inconsistent target sizes in the images. Ultimately, employing the non-maximum suppression algorithm (NMS) results in bounding boxes for each detected target.

3.2.2. The SCAFPN structure

The target detection method based on the pyramid network generally involves fusing features extracted at different levels of the backbone structure before performing the feature fusion. Following the backbone structure of YOLOv5, the C3 module, which corresponds to the last layer of each feature extraction layer in the backbone network, is used to obtain a set of features at different scales, namely C2, C3, C4, C5 (corresponding to scales of 1/4, 1/8, 1/16, 1/32).

In the initial stage, only the low-level features C2 and C3 are utilized to enter the neck feature pyramid. During the fusion process, the Mild module is employed as an intermediate layer, consisting of upsampling, weighted feature fusion, and 1*1 convolution. The Mild module initially performs upsampling on the C4 layer that needs cross-scale fusion. Subsequently, it utilizes weighted feature fusion to concatenate the feature maps of the C3 layer. The output is then propagated upward until reaching the C2 layer. This completes the fusion of high-level features with low-level features, effectively avoiding the direct fusion of non-adjacent layer features. At this point, C4 is incorporated into the feature fusion network, and following the same sequence of operations, C5 is gradually introduced step by step. After completing the fusion operation, new multi-scale features P2, P3, P4, P5 are generated, corresponding to scales of 4, 8, 16, and 32 pixels. In particular, the YOLOv5 model inputs C3, C4, C5 into the feature fusion pyramid, resulting in P3, P4, P5 as the output features.

Before the optimization process, the SCAFPN network structure proposed is depicted in Fig. 3. As the main network progressively extracts features layer by layer, SCAFPN conducts incremental fusion on the extracted feature maps at low, medium, and high levels. The process involves initially fusing low-level features, followed by the gradual fusion of medium-level features and high-level features using the Mild intermediate layer. As noted in AFPN, the semantic gap between non-adjacent hierarchical features is larger than that between adjacent

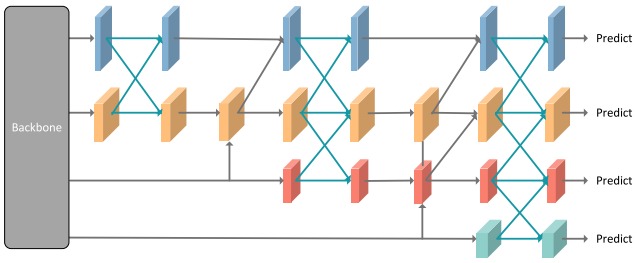


Fig. 3. The architecture of the proposed Sparsely Connected Asymptotic Feature Pyramid Network (SCAFPN), comprising four detection heads before optimization. SCAFPN integrates low-level features in the initial stage, then utilizes an intermediary module to transmit high-level features to the lower levels, ultimately incorporating the highest-level features into the network. Gray arrows denote convolutional operations, while light green arrows signify adaptive spatial fusion.

hierarchical features, especially for the bottom and top features. This leads to the poor fusion effect of non-adjacent hierarchical features directly. Therefore, it is unreasonable to directly use C2, C3, C4 and C5 for feature fusion.

The original intent behind the design of AFPN was to employ a progressive architecture to alleviate the aforementioned issues. However, we observed that dense connections still lead to the fusion of non-adjacent layers. Despite using ASFF (Liu et al., 2019) modules to enhance critical features and suppress conflicting information from different objects, we believe that this could result in an increase in model parameters and impact model performance. Therefore, we devised a Mild middleware and directly eliminated the fusion of non-adjacent layers. Each layer now exclusively performs feature information fusion with its adjacent layers, aiming to further eliminate semantic information gaps between non-adjacent layers and reduce the occurrence of multi-objective information conflicts in the feature fusion process at each spatial location.

3.2.3. The Mild middleware architecture and adaptive spatial fusion

The purpose of designing the Mild middleware is to further eliminate semantic gaps between non-adjacent layers, match the dimensions of corresponding layers, and then perform subsequent fusion operations while continuously propagating high-level features downward. This involves the stacking of nearest-neighbor interpolation upsampling, 1×1 convolutions, and weighted feature fusion modules. It is important to note that during the fusion process, the Mild middleware encounters the issue of unequal contributions from output features of different layers. To address this problem, we employ a weighted feature fusion module. We utilize the fast normalization method from BI-FPN, defined by the formula:

$$O = \frac{w_i}{\epsilon + \sum_j w_j} \cdot I_i \tag{1}$$

where $w_i \geq 0$ is ensured by applying the ReLU activation function after each w_i. A small learning rate ($e = 0.0001$) is set to avoid numerical instability.

Additionally, for the feature fusion of different layers, we utilize bilinear interpolation for upsampling, and employ different convolution kernels and strides for downsampling. For example, upsampling is achieved using the built-in bilinear interpolation in PyTorch. As for downsampling operations, 2 times downsampling is implemented via a 2×2 convolution with a stride of 2. It is important to note that, due to the structural characteristics of our approach, we do not necessitate 4 and 8 times downsampling in comparison to AFPN, and therefore it is not mentioned in this context.

In the multi-level pyramid features, the inconsistency arising from multi-scale fusion can constrain the performance of single-stage detectors. Therefore, we adopted the Adaptive Space Fusion method (ASFF) to address the impact of the aforementioned issue. The feature fusion

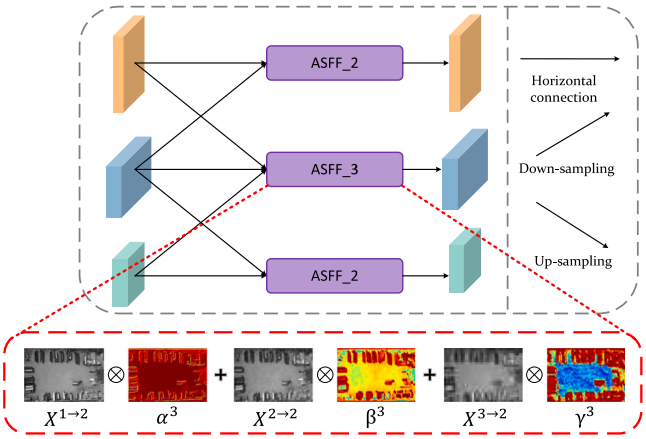


Fig. 4. The adaptive spatial fusion operation serves as an exemplification of feature fusion across three distinct levels. The method can be tailored accordingly for situations involving varying numbers of levels.

operation is conducted using the structure depicted in Fig. 4. Let $x_{ij}^{n\to l}$ denote the feature vector from level n layer to level i later at position (i, j). The final feature vector y_{ij}^{l} is obtained through adaptive space fusion of multi-level features and is defined as a linear combination of the feature vectors $x_{ij}^{1\to l}$, $x_{ij}^{2\to l}$, and $x_{ij}^{3\to l}$:

$$y_{ij}^{l} = \alpha_{ij}^{l} \cdot x_{ij}^{1\to l} + \beta_{ij}^{l} \cdot x_{ij}^{2\to l} + \gamma_{ij}^{l} \cdot x_{ij}^{3\to l} \tag{2}$$

In the context of ASFF-3 as depicted in Fig. 4, the red box illustrates the process of fusing features. Here, X^1, X^2, and X^3 represent features from level, level2, and level3, respectively. These features from different layers are multiplied by weight parameters α^3, β^3, and γ^3 and then added together to obtain the new fused feature ASFF-3. The weight parameters α_{ij}^{l}, β_{ij}^{l}, and γ_{ij}^{l} are obtained through 1×1 convolution applied to the resized features from level1 to level3. Furthermore, after concatenation, the parameters α_{ij}^{l}, β_{ij}^{l}, and γ_{ij}^{l} are processed through softmax to ensure their values lie within the range of [0,1], and satisfy the condition $\alpha_{ij}^{l} + \beta_{ij}^{l} + \gamma_{ij}^{l} = 1$.

Considering the difference in the number of fusion features at each stage of AFPN, we have incorporated an adaptive spatial fusion module tailored to a specific number of stages.

3.2.4. Optimizing the SCAFPN network structure

In our experiments, we observed that although SCAFPN reduces parameters compared to AFPN, there is also an increase in computational load. One possible explanation is that the added intermediate layer processing introduces a demand for additional computational resources. To meet the requirement for model lightweighting, we further optimized the feature fusion structure, focusing on both lightweighting and improvements for small object detection.

Given that unmanned aerial vehicle (UAV) images often involve numerous small and medium-sized objects, and as network depth increases, the corresponding receptive field also expands, making the network less sensitive to small object detection. Therefore, we deemed the contribution of higher layers to small object detection to be relatively low. Consequently, we removed the topmost layer, which is insensitive to small objects and consumes significant computational resources.

To balance the impact of removing the highest layer on model accuracy, we increased the channel capacity of the Neck network structure to accommodate more feature information, thereby compensating for the partial loss of accuracy. The final SCAFPN feature pyramid structure is illustrated in Fig. 2.

3.3. Multi-head self-attention of cross-spatial learning (CSL-MHSA)

In drone-captured images characterized by extensive scale and intricate scenes, enhancing semantic distinctiveness and reducing category confusion can be achieved by gathering and linking scene data from a broad spatial context. This approach aids in comprehending associations among different objects. However, conventional convolutional networks face limitations due to their localized convolution operations, constraining their ability to capture comprehensive global context information. Conversely, transformers excel in focusing globally on connections among image feature segments, preserving ample spatial details essential for object detection via multi-head self-attention mechanisms. Moreover, addressing viewpoint variability in drone-captured images stands as a significant challenge, necessitating detectors equipped with improved domain adaptation capabilities and adaptable receptive fields. A referenced study (Naseer et al., 2021) in the literature underscores the substantial robustness of vision transformers compared to CNNs when encountering significant occlusions, disturbances, and shifts in domains. This highlights the enhanced resilience of vision transformers in diverse and challenging image settings.

The attention mechanism primarily extracts data or important features in images rapidly based on contextual content or pixel correlations. Lin, Feng et al. (2017) proposed the self-attention mechanism and applied it to the hidden layers of bidirectional LSTM. The self-attention mechanism improves upon traditional attention mechanisms by reducing reliance on external information, excelling at capturing internal correlations within data or features. Following the introduction of single-head self-attention and MHSA by Vaswani et al. (2017), self-attention has become a research hotspot.

MHSA allocates weights to various features based on their relevance, directing network attention towards the target of interest, thereby reducing interference from unrelated background and enhancing network feature extraction performance. The multi-head self-attention mechanism is composed of multiple self-attention layers in parallel, where X and $Z_i, i = 1, 2, \ldots, n$ represent the input and output feature spaces of the self-attention layer, respectively. The input to the self-attention module is transformed through linear operations to obtain five matrices: queries (W_q), keys (W_k), values (W_v), height (R_h), and width (R_w). The module's output (Z_i) is computed as a weighted sum of similarities based on queries, keys, height, and width. The calculation formula is given by:

$$Z_i = Softmax(W_k \times W_Q + W_Q(R_h + R_w))W_v \tag{3}$$

Here, the Softmax function is a normalization exponential function that represents the similarities of queries, keys, height, and width in probability form. To ensure non-negativity of values, the similarity values are transformed using the exponential function. To ensure that the sum of all similarity values is 1, the output result represents the proportion of its exponential value in the total sum, as expressed by the equation:

$$Softmax(f_y) = p(y|x) = \frac{expf_y}{\sum_{c=1}^{C} expf_c} \tag{4}$$

Cross-spatial learning benefits from the ability to establish mutual dependencies between channels and spatial positions, coupled with recent developments in computer vision (Chen et al., 2018; Ouyang et al., 2023; Wang et al., 2018). Inspired by this, we attempt to integrate the concept of cross-spatial learning into a multi-head self-attention mechanism. We propose a cross-spatial learning multi-head self-attention mechanism with the aim of achieving more robust feature aggregation capabilities.

As illustrated in Fig. 5, we employ 1 × 1, 1 × 1, and 3 × 3 convolutions to compute q, k, and v, respectively. Specifically, we associate two 1 × 1 convolutions with local receptive fields and the 3 × 3 convolution with a global multi-scale receptive field. These operations are hierarchically layered according to a predefined number of heads, resulting in W_Q, W_K, and W_V. Subsequent operations are performed on each layer individually. We compute the dot product of W_Q and W_K, denoted as $W_{Q \cdot K}$, followed by softmax activation and matrix multiplication with W_V. Simultaneously, W_V, acting as a global receptive field, undergoes softmax activation and matrix multiplication with $W_{Q \cdot K}$. The final output for a specific layer is obtained by summing the results of these two operations. The overall output is derived by summing the outputs of all layers. The mathematical expression for this process is given by:

$$\begin{aligned} Z_i = {} & Softmax(W_{Q \cdot K} + W_Q(R_h + R_w))W_v \\ & + Softmax(W_Q \cdot K)(W_{Q \cdot K} + W_Q \times (R_h + R_w)) \end{aligned} \tag{5}$$

3.4. Other strategies

In order to further reduce the computational demands of the model, we have employed DSConv (Nascimento et al., 2019) (Distribution Shifting Convolution). DSConv achieves lower memory consumption and higher computational speed. We have enhanced the C3 module by applying DSConv, which maintains the same output as the original convolution by employing kernel-based and channel-based distribution offsets. This modification is illustrated in Fig. 1. DSConv is applied across the entire network, leading to a significant reduction in the computational load of the model.

To effectively extract channel and spatial information for small target features and adapt to the input of the SCAFPN structure, we reorganize the backbone structure of YOLOv5 using SPDConv (Sunkara & Luo, 2023), CSL-MHSA, and DSConv, as depicted in Fig. 1. In images captured by unmanned aerial vehicles (UAVs), a broader scene coverage often introduces more complex backgrounds and object occlusions. The aforementioned approach mitigates these challenges.

Upon analyzing the VisDrone2019 dataset, we observed the presence of dense target scenes, indicating a substantial occurrence of ambiguous labels, where a single anchor may correspond to multiple targets. Conventional manual rule-based allocation methods fail to consider variations in size, shape, or boundary occlusion, and lack a global perspective. Assigning any label under such methods may adversely impact network learning. To address these issues, we employ the SimOTA (Ge et al., 2021) label assignment strategy proposed by the YOLOX team.

Most deep learning frameworks default to using 32-bit floating-point algorithms for training. In 2017, NVIDIA researched a method for Automatic Mixed Precision (AMP) training, which combines single precision (FP32) with half precision (FP16) during network training, achieving nearly the same accuracy as FP32 while using the same hyperparameters. Purely using FP16 precision for training poses specific challenges: 1.Overflow errors, arising from the significantly narrower dynamic range of FP16 compared to FP32, leading to potential overflow and underflow problems, resulting in "NaN" during computation. 2.Round-off errors, occurring when gradients become too small to be represented within the current interval, potentially leading to failed gradient updates. To further conserve memory usage and accelerate training computations, an automatic mixed precision training method has been employed.

3.5. Applying TensorRT for float16 quantization of the model

In response to the operational demands in airborne scenarios, our model needs to be deployed on edge computing devices, characterized by compact size and relatively lower computational power compared to traditional servers. The small-scale model we propose initially achieves an inference speed of only 14 frames per second (FPS) without acceleration. Hence, we apply TensorRT quantization acceleration to enhance the model's performance.

TensorRT, developed by NVIDIA, is a neural network inference library that offers various optimizations for fully convolutional neural

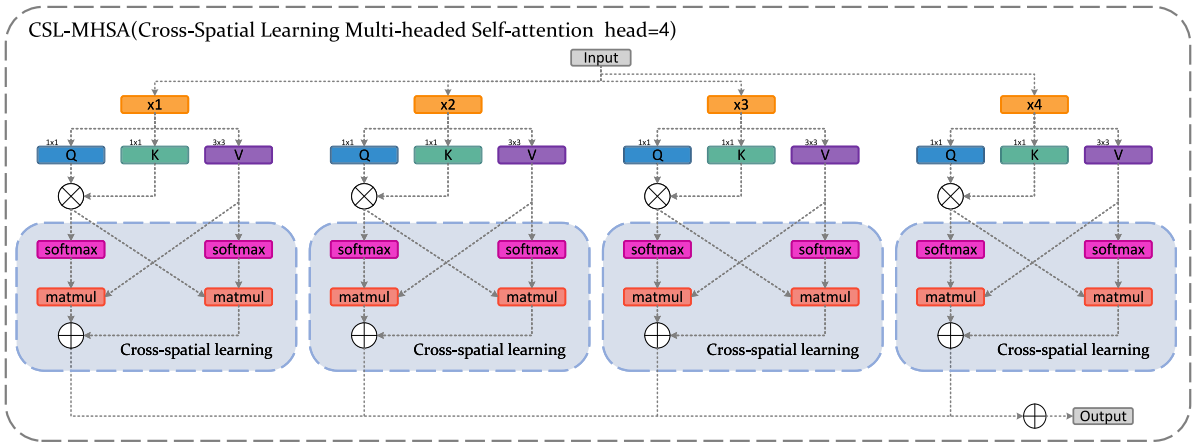


Fig. 5. CSL-MHSA with Head = 4: Structure Example of Multi-Head Self-Attention in Cross-Spatial Learning. The operations for Q, K, and V are accomplished through 1 × 1, 1 × 1, and 3 × 3 convolution, respectively.

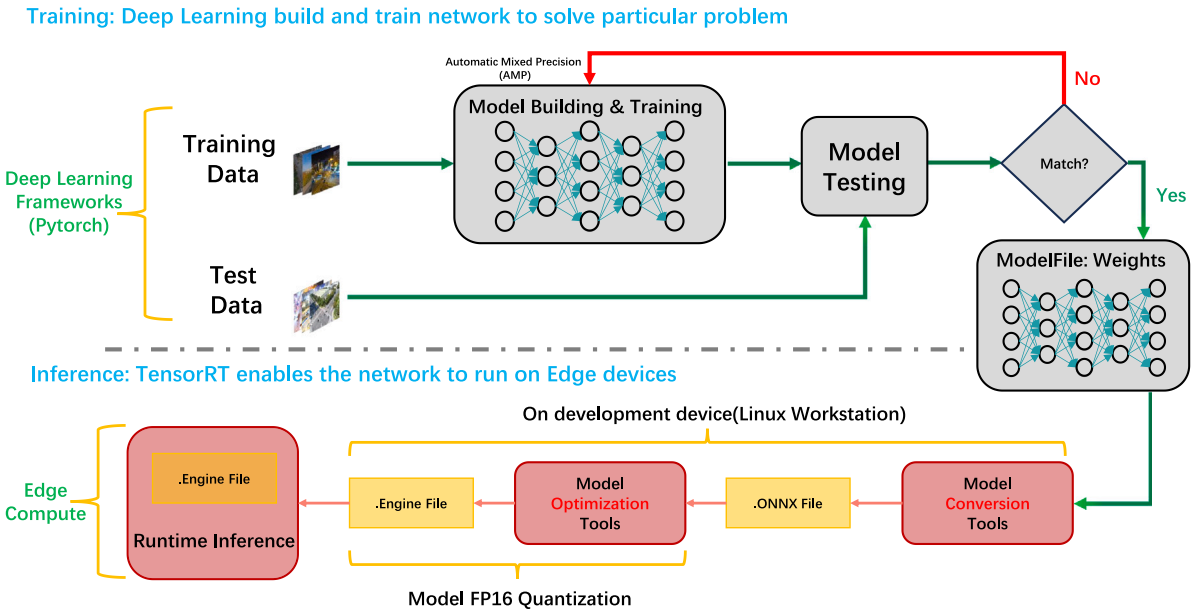


Fig. 6. TensorRT Acceleration Workflow Steps Presentation.

network structures. These optimizations significantly boost the performance of detection algorithms and improve the effectiveness of detection models on devices with limited processing capabilities. The overall deployment process for TensorRT-accelerated inference is illustrated in Fig. 6. After training the neural network model on a server and generating a model with FP32 precision, we import it directly into TensorRT from frameworks such as Caffe, UFF, and ONNX through parsers for Caffe, TensorFlow, and other deep learning frameworks. This process results in the creation of a serialized file and the formation of an optimized engine. During execution, the engine can be invoked to perform inference.

Quantization primarily involves converting numbers of type float32 to float16 or uint8 to reduce computational load. While the conversion between float32 and float16, both being floating-point numbers, is relatively straightforward with minimal precision loss, the conversion to uint8 leads to increased precision loss due to the significantly reduced numerical range. The basic principles of quantization are as follows: Let r represent a floating-point real number, q represent the quantized fixed-point integer, S denote the scale factor between real and integer values, and Z represent the integer corresponding to 0 in the quantized real numbers. The calculations for S and Z are shown in Eqs. (6) and (7), while the conversion formulas for r and q are presented in Eqs. (8) and (9), allowing for quantization at different bit levels.

$$S = \frac{r_{max} - r_{min}}{q_{max} - q_{min}} \tag{6}$$

$$Z = round(q_{max} - \frac{r_{max}}{S}) \tag{7}$$

$$r = S(q - Z) \tag{8}$$

$$q = round(\frac{r}{S} + Z) \tag{9}$$

Quantization acceleration includes FP32, FP16, and INT8, where the reduction in precision corresponds to an increase in the model's inference speed. Our experiments validate that INT8 quantization has the greatest impact on precision, resulting in severe accuracy degradation and calibration challenges. Therefore, in this paper, we opted for FP16 quantization acceleration for the model.

4. Experiment and result

4.1. Experimental details

4.1.1. Datasets

The experimental data in this paper were validated using two datasets, VisDrone2019 (Zhu, Wen et al., 2021) and AI-TOD (Wang, Xu et al., 2021; Wang, Yang, et al., 2021). The VisDrone dataset, characterized as a standard and challenging dataset, comprises images captured from the real world using unmanned aerial vehicles. These images encompass various angles and heights, taken during both daytime and nighttime, and at different resolutions. The dataset is divided into three parts, with 6471 images selected for the training set, 548 images for the validation set, and 1580 images for the test set. AI-TOD comprises 28,036 aerial images with a total of 700,621 object instances distributed across eight categories, including airplane, bridge, storage-tank, ship, swimming-pool, vehicle, person, and windmill. Notably, this dataset is tailored for assessing detector performance on tiny object detection, with the mean size of objects measuring about 12.8 pixels, markedly smaller than other existing aerial image object detection datasets. Moreover, in AI-TOD, the largest object size is constrained to be smaller than 64 pixels, and over half of the objects are less than 16 pixels in size.

4.1.2. Environment and training parameter settings

The proposed model in this paper is implemented using the PyTorch framework. All models are trained on a computing device equipped with an NVIDIA GeForce RTX RTX4090 GPU, an Intel(R) Xeon(R) Platinum 8352 CPU, and 90 GB of memory during the training phase. The testing phase is conducted on the NVIDIA Jetson Xavier Nx edge computing platform. The NVIDIA Jetson Xavier Nx features 384 CUDA cores, 48 Tensor Cores, and 2 NVDLA engines, delivering 14 TOPS at 10 watts and 21 TOPS at 15 watts, making it suitable for resource-constrained systems. The implementation on the NVIDIA Jetson Xavier Nx uses PyTorch 1.12.0 based on Python 3.8, Jetpack version 5.1, and CUDA version 11.4.

EL-YOLO is trained for 300 epochs on the VisDrone2019 dataset, with three epochs dedicated to warm-up training. The network is trained using the SGD gradient descent optimizer with an input size of 640 × 640 pixels, a batch size of 16, and unspecified hyperparameters consistent with YOLOv5.

Training on the AI-TOD dataset consists of 100 epochs, with three epochs for warm-up training. The network is trained using the SGD optimizer with an input size of 640 × 640 pixels, a batch size of 16, an initial learning rate of 0.01, a decay factor of 0.12 for the final epochs, momentum set to 0.9, and the following loss function coefficients: box (0.1), cls (0.3), obj (1.3). Unspecified hyperparameters remain consistent with YOLOv5.

4.2. Evaluation metrics

In this section, we employ several key metrics, including precision, recall rate, F1 score, mAP (0.5), mAP (0.5:0.95), frames per second (FPS), floating-point operations per second (FLOPs), and model parameter quantity, to evaluate the performance of the model on different datasets.

The accuracy of a model in object detection hinges on the correctness of predictions, assessed through IoU and a threshold $\in$ [0, 1], utilizing TP (True Positive), FP (False Positive), and FN (False Negative) metrics. TP signifies a correct prediction when IoU surpasses the specified threshold, while FP denotes an erroneous prediction of a non-existent object or the detection of an object with an IoU lower than the threshold. FN identifies an object in the actual image that the model fails to predict. Notably, True Negative (TN) is not utilized in object detection due to the infinite number of prediction boxes that should not be anticipated in each image. Evaluation of object detection models predominantly relies on Precision (P) and Recall (R) criteria. Furthermore, the primary metric for assessing the accuracy of object detection models is Mean Average Precision (mAP), calculated based on the mean of Average Precision (AP).

Precision, outlined in Eq. (10), gauges the accurate proportion of all predicted bounding boxes, while Recall, defined in Eq. (11), signifies the ratio of correctly located and identified objects within the ground truth. F1 score, articulated in Eq. (12), serves as the harmonic average of Precision and Recall, providing a comprehensive evaluation of algorithmic performance.

Additionally, mAP(0.5) measures the average precision of predicting boxes when the Intersection over Union (IoU) exceeds 0.5, as defined by Eq. (13), representing the integral area under the Precision–Recall curve (P–R curve). The mAP(0.5:0.95) takes into consideration IoU values between 0.5 and 0.95.

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F1 - score = \frac{2PR}{P + R} \tag{12}$$

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i \tag{13}$$

Finally, the metrics for evaluating model complexity and inference speed are considered:

1. Frames Per Second (FPS) refers to the number of frames a model can infer per second. Generally, depending on the specific application, real-time object detection models should be capable of processing video frames at a speed of at least 20–30 FPS to align with practical task requirements.
2. FLOPs, or Floating Point Operations Per Second, represent the required number of floating-point calculations per second. It is a crucial metric indicating the speed and computational capability of neural network models. Lower FLOPs imply lower computational complexity.
3. The use of megabyte units to denote the size and complexity of the model is one of the key reference metrics for lightweight models. In general, models with fewer parameters can execute tasks more quickly and are suitable for deployment on resource-constrained edge devices.

4.3. Ablation experiment

In this subsection, we conducted ablation experiments on the VisDrone2019 dataset to validate the effectiveness of our proposed method, EL-YOLO. Table 1 demonstrates that employing SCAFPN improves mAP50 to 33.4%, an 5.2% increase compared to the baseline model. Despite a 37% increase in GFLOPs, the parameter count decreases by 53%. Table 2 compares SCAFPN with other structures, revealing a 3% improvement in mAP50 over AFPN. While GFLOPs are slightly higher, SCAFPN outperforms AFPN by 5 frames per second (FPS). Eliminating the semantic gap between non-adjacent layers and reducing the potential for multi-target information conflicts has a positive impact on the network, as indicated by Table 2, considering the influence of GFLOPs and parameter count on inference speed.

Next, restructuring the backbone network with the inclusion of SPD-Conv raises mAP50 to 38.5%. Further integration of our proposed attention mechanism, CSL-MHSA, boosts mAP50 to 40.1%, indicating a 1.6% improvement. This suggests that incorporating attention mechanisms during feature extraction positively influences bounding box regression with minimal impact on FPS. Table 3 illustrates the performance of EL-YOLO with CSL-MHSA and MHSA attention mechanisms. CSL-MHSA outperforms MHSA with a 0.5% increase in mAP50, showcasing better performance on the VisDrone2019 dataset.

Table 1
Experimental investigation on VisDrone2019 Test Set: Ablation study of each component (Deployed and tested on NVIDIA Jetson Xavier Nx platform). EL-YOLO represents the model after the previous improvement (YOLOv5s+SCAFPN+SPD-Conv+CSL-MHSA).

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs	FPS
YOLOv5s	34.9	28.5	29	28.2	14.7	7.04	**15.8**	**32**
YOLOv5s+SCAFPN	38.8	33.2	35	33.4	18.6	**3.3**	26.8	22
YOLOv5s+SCAFPN+SPD-Conv	45.7	37.7	40	38.5	22	3.56	44.2	16
YOLOv5s+SCAFPN+SPD-Conv+CSL-MHSA	48.6	39	42	40.1	22.8	4.29	46.5	14
EL-YOLO+SimOTA	47.3	**39.3**	42	40.5	23.2	4.29	46.5	14
EL-YOLO+SimOTA+DSConv	**48.6**	39	42	**40.8**	**23.3**	4.29	24.7	16
EL-YOLO+SimOTA+DSConv+FP16 Quantization	48.1	38.8	**42**	40.6	23.2	4.29	24.7	24

Table 2
Experimental evaluation on VisDrone2019 test set: A comparative analysis with AFPN network. Image input size: 640 × 640 pixels.

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs	FPS
FPN+PAN	34.9	28.5	29	28.2	14.7	7.04	**15.8**	**31**
AFPN	35.7	30.6	31	30.4	16.9	7.16	19.5	16
SCAFPN	**38.8**	**33.2**	**35**	**33.4**	**18.6**	**3.3**	26.8	22

Although the model complexity has increased, it has not significantly impacted the model's inference speed. To further illustrate the influence of CSL-MHSA on the model, Table 4 presents a comparison of single-class mAP50 accuracy for five challenging small object categories in the VisDrone dataset. Examining the table, it is observed that "bicycle" exhibits the lowest accuracy among all categories. In comparison to MHSA, CSL-MHSA demonstrates a 1.7% improvement in mAP50 for this particular category. Across these five classifications, CSL-MHSA exhibits superior performance in handling challenging small objects.

Backbone reorganization enhances accuracy by 6.7%, highlighting the importance of restructuring for addressing complex backgrounds and object occlusion issues, with attention mechanisms strengthening model generalization.

Introducing the SimOTA label assignment strategy pushes mAP50 to 40.5%, with mAP50:95 increasing to 22.8%, while maintaining inference speed and model complexity. Thus, it demonstrates the effectiveness of this change in improving bounding box prediction accuracy. Lightweight modifications to the network backbone and neck using DSConv result in a 47% decrease in GFLOPs, simultaneously accelerating model inference speed and achieving a 0.3% increase in mAP50, thereby significantly reducing model computational requirements.

Finally, the entire network undergoes FP16 quantization for better deployment on edge computing devices. Despite a slight decrease in mAP50, the model achieves a substantial increase in FPS to 24 frames/s. Although some improvements may lead to a certain degree of accuracy reduction, overall, the speed enhancement outweighs the precision decrease, yielding a more optimized model.

In summary, EL-YOLO demonstrates a 12.4% overall improvement in mAP50 compared to the baseline model, with a 8.5% increase in mAP50:95. Despite the increased computational demand in terms of GFLOPs compared to the baseline model, EL-YOLO exhibits clear advantages in terms of parameter efficiency and high accuracy, making it suitable for various applications, especially those requiring high performance and precision. In the subsequent section, we will showcase the Nano model of EL-YOLO, which achieves an inference speed of 35 frames per second (FPS) following quantization, while maintaining commendable model performance.

4.4. Comparative experiments on the VisDrone dataset

4.4.1. Comparison results on edge computing platform

The model proposed in this paper was ported to the NVIDIA Jetson Xavier Nx edge computing platform for practical deployment and testing. However, due to significant hardware compilation and software support requirements, as well as resource constraints when compared to the reference model, edge devices were unable to fulfill the operational needs. Consequently, we conducted comparative tests by porting the proposed model, YOLOv5 series, and state-of-the-art YOLOv8 (SOTA) series onto edge devices. To comprehensively validate our model's performance, we further conducted comparative experiments in the subsequent section using the NVIDIA GeForce RTX 4090 24G graphics card, contrasting with some classical algorithms.

Initially, the results from various models on the VisDrone2019 validation set were obtained, as shown in Table 5. Both the YOLOv5 and YOLOv8 series proposed five models of different sizes. To better showcase our model's performance, we presented five models of varying sizes based on the depth and width parameters of the YOLO series (arranged from small to large: n-Nano, s-Small, m-Medium, l-Large, x-Extra Large). Table 5 indicates that our model exhibits higher accuracy within the same level. Specifically, the proposed X-scale model achieved the highest precision at 54.3% on the mAP50 metric, surpassing the YOLOv8x model by 4.5%, with a reduction of approximately 45% in parameter count. The proposed S-scale model, compared to the YOLOv5s baseline, showed an improvement of 13.2% in mAP50, surpassing YOLOv8s by 5.1%. Table 5 illustrates that the proposed S-scale model outperforms YOLOv5x by 6.3% on mAP50, with a 95.14% reduction in parameters, an 88% decrease in GFLOPs, a 380% increase in inference speed, and a performance advantage over the latest YOLOv8 series, showcasing significant advantages and high performance on the VisDrone2019 dataset.

A more in-depth analysis of the results in Table 5 reveals that, despite EL-YOLOn (Nano model) having the highest FPS and being the lightest model among all, the proposed model demonstrates higher accuracy compared to models of the same size, outperforming YOLOv5n and YOLOv8n by 12.7% and 4.5%, respectively. This is attributed to the SCAFPN, which adds branches with higher resolution feature maps, removes output layers designed for detecting large objects, and focuses the model more on small object detection, resulting in high-performance outcomes. Additionally, the reduction in model size weakens the model's semantic representation capabilities, explaining the significant accuracy gap between the proposed Nano model and the Small model.

Given the crucial role of target localization in successful object detection and classification, Table 6 presents the efficiency of different models in this regard, calculating mAP50 for different categories in the VisDrone2019 dataset. As anticipated, our proposed method exhibits higher mAP50 for most categories compared to baseline models of the same level.

Moreover, results Table 7 obtained on the VisDrone2019 test set demonstrate that our proposed X-scale model achieves an impressive precision of 44.1% on the mAP50 metric, surpassing the YOLOv8x model by a notable margin of 2%. In comparison to the YOLOv5s baseline, our introduced S-scale model exhibits a remarkable enhancement of 9.7% in mAP50, outperforming YOLOv8s by 3%. The enduring excellence of EL-YOLO performance on its respective test set could be attributed to the reconfiguration of the backbone network in the feature extraction stage, showcasing the commendable generalization capability of the proposed model.

In summary, our model achieved superior performance across various parameters when compared to other models. For real-time detection models, we offer Small and Nano models for selection. Notably, the

Table 3
Experimental evaluation on VisDrone2019 test set: A comparative analysis with MHSA module. Image input size: 640 × 640 pixels.

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs	FPS
EL-YOLO+MHSA	46.3	38.4	41	39.6	22.8	**3.78**	**44.8**	**14.895**
EL-YOLO+CSL-MHSA	**48.6**	**39**	**42**	**40.1**	**22.8**	4.29	46.5	14.36

Table 4
Experimental assessment on VisDrone2019 test set: Comparative analysis with MHSA module for single-class detection of smaller targets. Image input size: 640 × 640 pixels.

Model	Bicycle	Tricycle	Awning-tricycle	Motor	Van
EL-YOLO+MHSA	15.2	22.3	24.8	**39.6**	42.4
EL-YOLO+CSL-MHSA	**16.9**	**22.5**	**26.6**	38.9	**43.8**

Nano model achieves an inference speed of 35 frames per second and can be deployed on edge computing devices, meeting the requirements of the majority of real-time tasks.

4.4.2. Comparison results on the server

To further validate the performance of EL-YOLO, we introduced three evaluation metrics, namely Aps, APm, and Apl, corresponding to small, medium, and large-sized targets. Under the same training environment, we conducted comparisons with some of the current classical two-stage algorithms, such as Faster-RCNN (Ren et al., 2017), RetinaNet Lin, Goyal et al. (2017), Libra R-CNN (Pang et al., 2019), and anchor-free detector Fcos (Tian et al., 2019). Additionally, we compared with recent single-stage algorithms for small object detection, including FE-YOLOv5 (Wang, Yang et al., 2023) and TPH-YOLOv5 (Zhu, Lyu et al., 2021), on the VisDrone2019 dataset. In all aspects, EL-YOLO outperformed classical object detectors.

Due to its focus on achieving a balance between accuracy and speed, EL-YOLO aims to be a deployable lightweight detector. However, when compared to two-stage algorithms and larger models, there is still a discernible gap. This discrepancy arises from the fact that single-stage object detectors merge classification and localization tasks, sacrificing accuracy to gain speed. It is important to consider that single-stage networks, such as YOLOv5 and YOLOv8, inherently face challenges in achieving the same level of precision as two-stage networks in object detection tasks. Two-stage networks tend to have separate proposal and refinement stages, allowing for more meticulous object localization and classification. Consequently, while single-stage networks like YOLOv5 and YOLOv8 excel in real-time applications due to their speed, they might not achieve the same high precision levels as two-stage networks in certain scenarios, particularly with smaller object detection in remote sensing datasets. On the other hand, large models are unsuitable for deployment on edge computing devices due to their substantial resource requirements. Furthermore, on the APs metric (Table 8), the X-scale model consistently outperforms the other listed algorithms, underscoring the attention our proposed model pays to the detection of small objects.

4.5. Comparative experiments on the AI-TOD dataset

To enhance the comprehensive evaluation of the proposed model, conduct comparisons using a highly challenging AI-TOD remote sensing dataset. It is essential to note that metrics such as model size and GFLOPs are intrinsic to the model itself and exhibit no reliance on the characteristics of the dataset. Therefore, in the subsequent analysis, our focus will solely be on assessing accuracy-related metrics, specifically precision, recall, mAP50, and mAP95, utilizing the AI-TOD datasets.

Table 9 presents the results of running the model on the AI-TOD dataset with an input size of 640 × 640 pixels and training conducted for 100 epochs. The table demonstrates that our proposed model outperforms all evaluated models on this dataset. In comparison to the baseline model, the proposed model exhibits significant improvements across all accuracy metrics. Compared to the EL-YOLOs and YOLOv5s models, our model shows an increase of 1.3% and 3% in mAP50 and mAP95, respectively. Furthermore, the EL-YOLOn outperforms the YOLOv5n with accuracy improvements of 5.9% and 5% for the two metrics, respectively. Table 9 also highlights the good generalization performance of YOLOv5, as it demonstrates stable performance on two different datasets. This aspect played a crucial role in selecting YOLOv5 as the baseline model.

Table 9 presents noteworthy findings, revealing suboptimal performance of the latest YOLOv8 on this dataset. In light of these results, we have initiated a detailed discussion, addressing specific aspects as follows: 1.YOLOv5 may outperform YOLOv8, suggesting better generalization capabilities. YOLOv8, being a recent release, may require further optimization for improved performance over time. 2.Concerns arise due to the larger size and complexity of YOLOv8 compared to YOLOv5, posing a risk of overfitting. The heightened model complexity in YOLOv8 may affect precision, especially with small objects in the AI-TOD remote sensing dataset, potentially lowering overall accuracy. 3. The observed decline in YOLOv8 accuracy on the AI-TOD remote sensing dataset could be attributed to the heightened computational demands of the larger model. This becomes particularly critical in situations where edge computing devices face resource constraints, potentially resulting in a degradation of model accuracy.

4.6. Float16 quantization of the model

This subsection presents a comparative analysis of FP32 and FP16 quantization acceleration results, without considering INT8 quantization. This omission is primarily due to significant accuracy degradation observed with INT8 precision after model quantization. Although INT8 quantization can potentially offer improved inference speed, it fails to meet the required accuracy specifications, thus ruling out its adoption. Table 11 showcases the performance comparison between pre- and post-quantization acceleration for the models. It is evident that FP16 quantization yields the best results with minimal performance loss. The Nano model achieves an inference speed of 35 frames per second (fps), representing an improvement of 11 fps. Similarly, the Small model achieves an inference speed of 24 fps, an increase of 10 fps. Table 11 also exhibits unsatisfactory acceleration results for FP32 quantization, which could be attributed to the following factors:

1. The additional operations involved in quantization and dequantization introduce computational overhead.
2. The presence of unquantizable operators in the model, where quantization inference focusing only on Convolution and Fully Connected layers may not yield desired outcomes.
3. The hardware platform may possess better optimization for FP16 precision, leading to subpar results for FP32 acceleration. This performance decline could also stem from excessive memory consumption, given that the edge computing device utilized in this study shares memory with graphics memory.

The above illustrates the quantization acceleration during the model inference stage. In addition, we conducted experimental testing for model training using the Automatic Mixed Precision (AMP) approach. Specifically, we compared the model's training using single-precision FP32 and mixed-precision FP32+FP16 training. As shown in Table 10, the AMP approach effectively reduces the GPU memory footprint and speeds up training computations. Additionally, the metrics MAP50 and MAP95 show reductions of 0.3% and 0.4%, respectively, with comparatively small loss in accuracy, thereby accelerating model training. In

Table 5
Performance comparison of EL-YOLO with YOLOv5 series and the latest YOLOv8 series detectors on VisDrone2019 validation set. The highest numerical values are highlighted in bold, and 'para' indicates the model's parameter count (parameters). (Deployed and tested on NVIDIA Jetson Xavier Nx platform).

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs	FPS
YOLOv3-tiny	24.8	20.6	21	21.7	10.2	8.69	12.9	–
YOLOv5-n	36.7	28.3	27	30.2	15.9	1.77	**4.2**	26
YOLOv5-s	44	33.8	33	36.1	20.2	7.04	15.8	23
YOLOv5-m	47.7	36.8	37	39.4	23	20.88	48	14
YOLOv5-l	50.7	38.6	39	41.4	24.6	46.15	107.8	8
YOLOv5-x	52.1	40.4	41	43	26	86.23	203.9	5
YOLOv6-s	–	–	–	37.7	22.1	17.2	44.2	–
YOLOv7-tiny	47.6	37.3	41	35.8	18.8	6.04	13.3	–
YOLOv7	57	49.5	53	45.8	26.5	37.2	105.3	–
YOLOv7x	58.9	**50.4**	54	47.1	27.6	70.84	188.2	–
YOLOv8-n	44.9	34.1	38	38.4	23.8	3	8.1	25
YOLOv8-s	52.1	38.4	43	44.2	28.2	11.12	28.5	18
YOLOv8-m	53.5	41.1	46	46.5	30.1	25.84	78.7	8
YOLOv8-l	56.4	43.8	48	49.3	32.3	43.61	164.9	4
YOLOv8-x	56.5	44.5	49	49.8	32.6	68.13	257.4	3
Proposed Model-n	48.8	40.3	43	42.9	24.8	**1.08**	6.7	**35**
Proposed Model-s	54.7	46.6	50	49.3	29.4	4.29	24.7	24
Proposed Model-m	58.1	49.3	53	52.7	32.2	10.93	54	8
Proposed Model-l	**59.8**	50.3	54	54	33.4	21.75	94.6	5
Proposed Model-x	59.5	50.1	**54**	**54.3**	**34**	37.6	146.5	4

Table 6
Single-class comparison of EL-YOLO with YOLOv5 series and the latest YOLOv8 series detectors on VisDrone2019 test set. The highest numerical values are highlighted in bold. (Deployed and tested on NVIDIA Jetson Xavier Nx platform).

Model	Pedestrian	People	Bicycle	Car	Van	Truck	Tricycle	Awning-tricycle	Bus	Motor
YOLOv3-tiny	16.7	15	8	53.8	14.9	12.1	7.05	10.5	37.4	15.1
YOLOv5-n	25.7	19.2	6.78	65.3	23	22.3	12.9	13.6	50.9	22.2
YOLOv5-s	31.2	22.4	10	70.6	31.5	29.7	15	17.1	56.7	26.5
YOLOv5-m	33.4	24.9	11.8	72.8	34.4	34.8	18.3	19.7	58.8	29.3
YOLOv5-l	34.8	25.1	11.7	74.3	36.8	36.9	19.9	22.4	59.5	32
YOLOv5-x	36.2	26.1	14.2	75.4	38	39.3	18.5	24.1	61.4	33.4
YOLOv7-tiny	24.5	17.4	6.46	70	27.7	29	12.2	12.8	48.7	26.1
YOLOv7	36.9	26.2	15.1	78	38.5	46.1	22.2	20.4	57.5	39.1
YOLOv7x	37.9	26.9	15.5	78.8	39.2	49	25	21.7	59.9	41.4
YOLOv8-n	33.6	21.4	11.1	71.7	33.7	35.2	16.8	20.9	59.3	29.4
YOLOv8-s	38.4	25.6	15.2	75.1	40.2	43.2	23.1	24.7	62.6	35.3
YOLOv8-m	40.3	26.6	16.8	76.4	42.1	48.6	22.1	25.5	65.5	37.1
YOLOv8-l	42.1	28.8	19.2	77.5	43.7	52.2	22.6	**26.6**	67.2	40.9
YOLOv8-x	42	28.7	19.5	78.2	45.3	**52.2**	26.4	24.2	67.6	39.9
Proposed Model-n	51.8	45.2	14.8	82.3	45	34.6	29.2	15.3	57.7	52.6
Proposed Model-s	59.1	49.7	22.9	86.3	51.7	40.2	34.9	20	68.2	60.2
Proposed Model-m	63.7	53.4	27.5	87.6	53.7	46	40.2	21.8	70	62.9
Proposed Model-l	64.4	**54.8**	28.7	87.8	**55.4**	48.2	41	24	72.4	63.7
Proposed Model-x	**65.1**	54.5	**30.4**	**88.1**	55.1	48	**42**	22.5	**73.3**	**64.3**

conclusion, based on the comparative analysis presented in this section, it is evident that FP16 quantization demonstrates favorable results with minimal performance loss. Therefore, it is a viable quantization scheme worth considering.

4.7. Discussion

Firstly, despite EL-YOLO exhibiting superior performance in accuracy and speed compared to YOLOv5 and YOLOv8, its training speed is considerably slower, approximately three times the gap, essentially trading time for model performance. Secondly, the model exhibits slow convergence speed. Although the improved model shows a reduction in parameters compared to the baseline model, it concurrently introduces additional hierarchical structures, necessitating more iterations and adjustments during the learning process to converge to the optimal solution. The increased complexity of the model may complicate and prolong the process of the gradient descent algorithm searching for the minimum value in parameter space, thereby decelerating the convergence speed.

Furthermore, the introduction of new trainable parameters in the network may require additional time to identify the optimal hyperparameter configuration, further influencing the model's convergence speed. In our experiments, we observed that the proposed model exhibits twice the GFLOP count compared to the baseline model, a consequence of the proposed architecture. To enhance accuracy in detecting smaller targets, a greater number of modules were placed within the lower layers of the model. This placement aimed to enable the model to capture more intricate features of these smaller targets. Lower-level feature maps typically possess higher resolutions, as they extract fine details from the input images during the early stages of network processing. Consequently, the increased pixel count in these feature maps unavoidably leads to a rise in computational load during convolutional operations and other computations. This is an area of focus for our future research, aimed at optimizing these issues. Finally, despite the aforementioned challenges, the model brings forth high-performance accuracy and speed.

Fig. 7 illustrates the confusion matrix of the proposed model trained on the VisDrone dataset, alongside YOLOv5s. According to the outcomes, the model demonstrates effective performance in accurately distinguishing between various classes.

Fig. 8 demonstrates superior performance of our proposed model (EL-YOLO) compared to other baseline models, particularly in environments with intense lighting conditions. The performance improvement is more pronounced when objects are extremely small. It is evident from Fig. 8 that, in comparison to the two alternative models, EL-YOLO effectively reduces false negatives (FN), thereby significantly enhancing overall model performance.

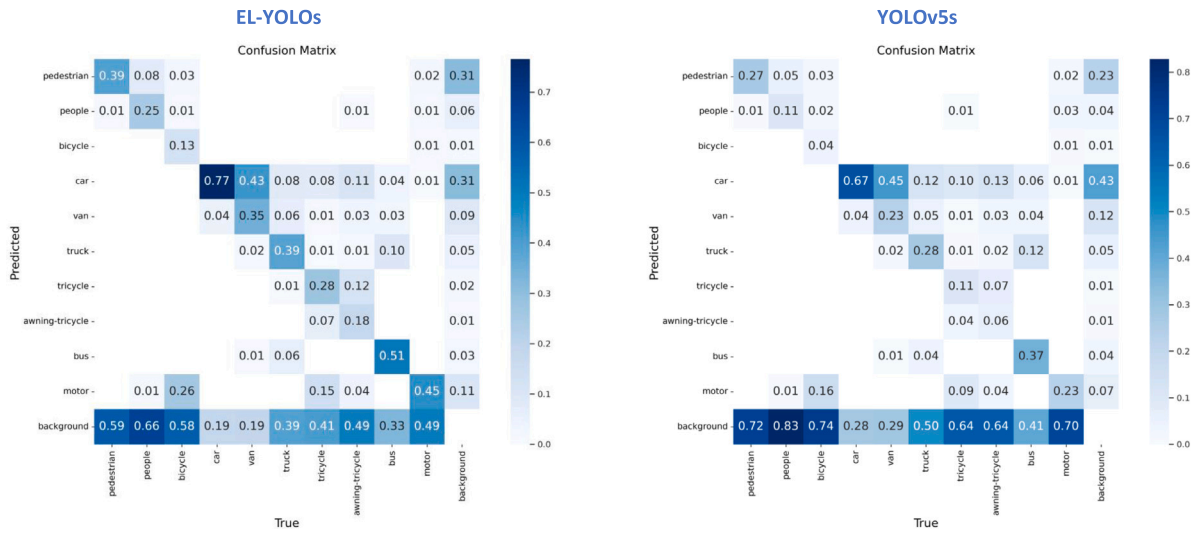


Fig. 7. Confusion matrix for the proposed model and YOLOv5s trained on the VisDrone2019 dataset.

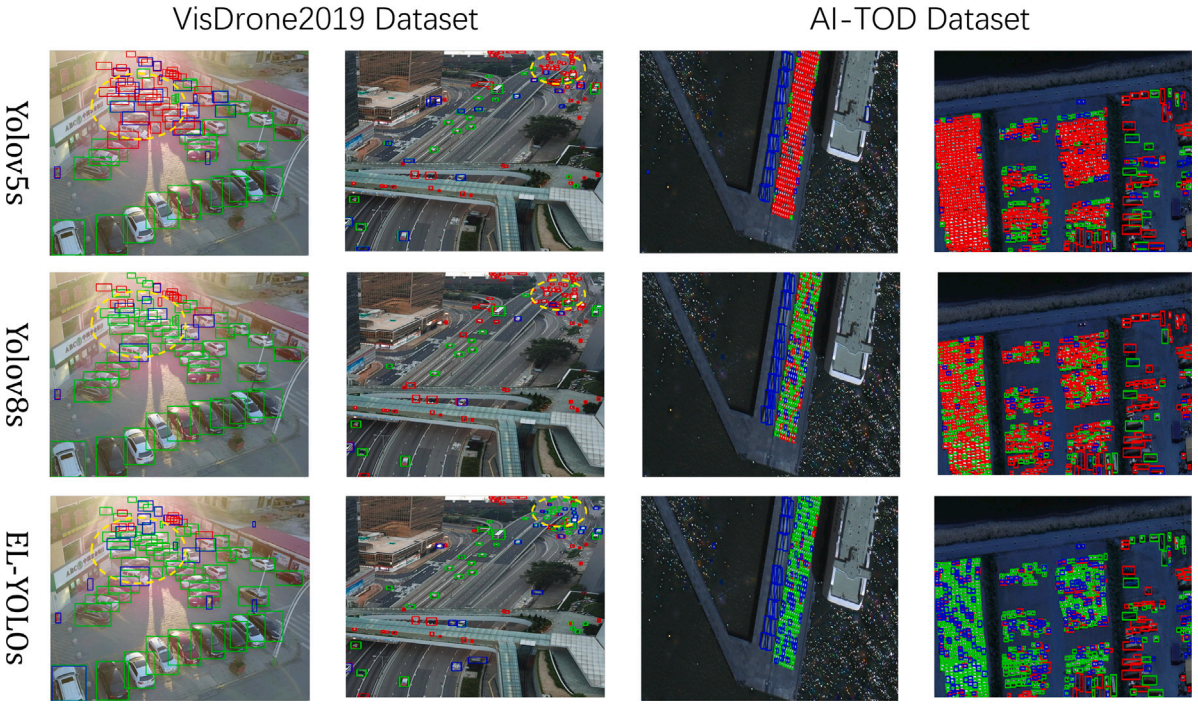


Fig. 8. Visualization of Detection Results Using VisDrone2019 and AI-TOD Datasets. The green, blue and red boxes denote true positive (TP), false positive (FP) and false negative (FN) predictions, respectively.

In summary, we conducted a comprehensive evaluation of the EL-YOLO model using two datasets and multiple performance metrics. The results indicate that the proposed model aims to strike a balance between accuracy, speed, and model size, finding a small object detection model that performs better and meets practical requirements within the YOLO series. EL-YOLO exhibits outstanding performance with fewer parameters and higher accuracy. For real-time detection tasks, we present Nano and Small models to address scenarios with limited computational resources that require real-time performance, high efficiency, and accuracy.

To validate the portability of the proposed algorithm, we deployed the model on the NVIDIA Jetson Xavier Nx edge computing platform and tested its performance on these resource-constrained devices, demonstrating excellent model performance. Fig. 9 illustrates the operational power consumption of the PyTorch, TensorRT 32-bit, and TensorRT 16-bit models for the deployment of EL-YOLO on NVIDIA Jetson Xavier NX, evaluated on the VisDrone test set. All models operate with an input resolution of 640 × 640. Fig. 10 presents the memory usage during the aforementioned deployment process.

It is evident that the TensorRT 16-bit quantized model demonstrates superior performance.

Additionally, to assess the model's generalization performance, we conducted comparative experiments using the AI-TOD dataset. Ultimately, the experimental results confirm the effectiveness and practicality of the proposed model.

5. Conclusion

This paper proposes an EL-YOLO model for target detection in low-altitude aerial images captured by airborne platforms, and it undergoes evaluation on two distinct datasets: VisDrone2019 and AI-TOD.

Table 7
Performance comparison of EL-YOLO with YOLOv5 series and the latest YOLOv8 series detectors on VisDrone2019 test set. The highest numerical values are highlighted in bold, and 'para' indicates the model complexity in terms of parameters. (Deployed and tested on NVIDIA Jetson Xavier Nx platform).

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs	FPS
YOLOv3-tiny	20.8	20.4	19	19.1	8.63	8.69	12.9	–
YOLOv5-n	32.7	25.7	28	26.2	13.5	1.77	**4.2**	34
YOLOv5-s	38.4	30.5	34	31.1	16.9	7.04	15.8	31
YOLOv5-m	42.5	32.6	37	33.8	19.1	20.88	48	16
YOLOv5-l	44.5	34.3	38.7	35.3	20.3	46.15	107.8	8
YOLOv5-x	46.1	36	40.4	36.7	21.3	86.23	203.9	5
YOLOv7-tiny	43.5	33.5	37	27.5	14.4	6.04	13.3	–
YOLOv7	52.2	43.6	47	38	21.2	37.2	105.3	–
YOLOv7x	53.4	**44.7**	**48**	39.5	22.3	70.84	188.2	–
YOLOv8-n	38.9	29.9	33.8	32.6	19.4	3	8.1	33
YOLOv8-s	46.6	33.4	38.9	37.8	23.1	11.12	28.5	18
YOLOv8-m	47.5	36.2	41.1	39.7	24.5	25.84	78.7	9
YOLOv8-l	50.3	37.7	43.1	41.7	26.1	43.61	164.9	5
YOLOv8-x	50.3	38.1	43.4	42.1	**26.3**	68.13	257.4	3
Proposed Model-n	41.8	34.1	36	34.9	19.4	**1.08**	6.7	**35**
Proposed Model-s	48.6	39	42	40.6	23.3	4.29	24.7	24
Proposed Model-m	52.1	41.1	45	43.7	25.3	10.93	54	8
Proposed Model-l	52.3	41	45	43.2	25.4	21.75	94.6	5
Proposed Model-x	**52.5**	41.4	46	**44.1**	26.1	37.6	146.5	3

Table 8
Comparison of traditional object detectors and small object detectors on the VisDrone2019 dataset (Tested on NVIDIA GeForce RTX 4090 platform).

Model	mAP95	mAP50	mAP75	APs	APm	APl
RetinaNet	15	26.4	15.3	6.3	25.6	34.4
Libra R-CNN	14.9	25.2	15.2	5.9	25.6	31.4
Fcos	17.9	30.4	18.3	9.2	27.6	35.4
ATSS	20.4	33.8	20.9	11.6	31.7	36.7
TridentNet	19.8	35	19.5	11.4	29.6	36.6
FE-YOLOv5	21	37	20.7	13.2	29.5	39.1
Faster-RCNN	26.7	43.7	28.4	14.8	37.5	41.6
TPH-YOLOv5	27.8	46.6	28.1	18.9	**38.5**	**41.6**
ARFP	20.4	33.9	21.8	–	–	–
Proposed Model-n	19.2	35.7	17.7	13.2	26.1	31.7
Proposed Model-s	24.3	43.3	23.5	17.8	32	39.1
Proposed Model-m	26.2	45.4	25.9	19.3	34.4	43
Proposed Model-l	27.6	47.8	27.3	20.6	36.3	40
Proposed Model-x	**28.7**	**48.7**	**28.9**	**21.8**	37.7	40.4

Table 9
Performance comparison of EL-YOLO with YOLOv5 series and the latest YOLOv8 series detectors on the AI-TOD test set. The highest numerical values are highlighted in bold, and 'para' indicates the model complexity in terms of parameters. (Deployed and tested on NVIDIA Jetson Xavier Nx platform).

Model	P	R	F1	mAP50	mAP95	Para(M)	GFLOPs
YOLOv3-tiny	42.8	19.8	23	20.9	7.35	8.68	12.9
YOLOv5-n	53.7	40.7	44	42.6	17.2	1.77	**4.2**
YOLOv5-s	**63.7**	47.9	54	52.9	23.3	7.03	15.8
YOLOv6-n	–	–	–	22.8	9.71	4.3	11.1
YOLOv6-s	–	–	–	25.7	11.5	17.2	44.2
YOLOv8-n	39.6	29.8	33	34.4	15.8	3	8.1
YOLOv8-s	50.6	38.1	42	43.5	20.6	11.1	28.5
Proposed Model-n	58	45.2	47	48.5	22.2	**1.08**	6.7
Proposed Model-s	61.9	**49.7**	**53**	**54.2**	**26.3**	4.29	24.7

In comparison to the YOLOv5 baseline model, the proposed model incorporates the following enhancements: (a) Proposes a sparsely connected progressive feature fusion pyramid structure in the feature fusion module to eliminate semantic gaps during non-adjacent hierarchical feature fusion, directing the model's attention to small object detection and enhancing accuracy. (b) Improvement of the multi-head self-attention mechanism (MHSA) through the design of a Cross-Space Learning (CSL-MHSA) structure, facilitating the establishment of long-range contextual relationships for the extraction of richer image features. (c) Restructuring of the backbone feature extraction using

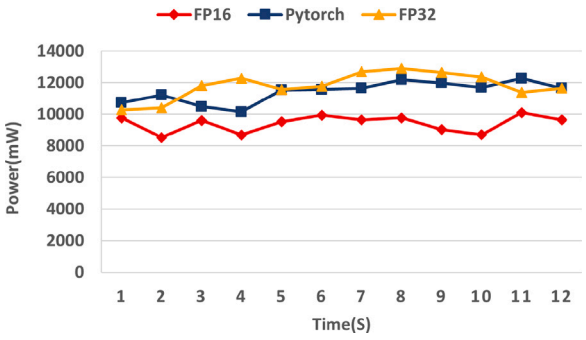


Fig. 9. Power Consumption Performance of EL-YOLO Model Across Different Deployment Frameworks. Each 'Time' represents the sampled results at one-second intervals.

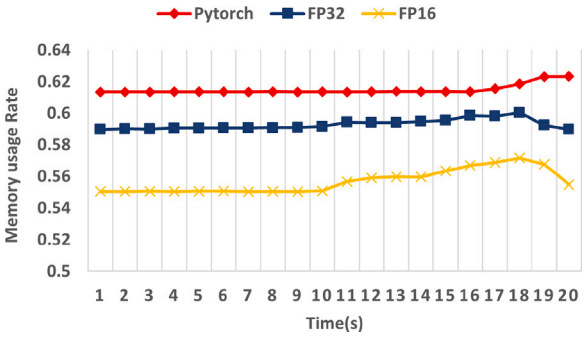


Fig. 10. Comparison of Memory Usage Rates for EL-YOLO Model Across Different Deployment Frameworks. Each 'Time' represents the sampled results at one-second intervals.

Table 10
Experimental assessment on VisDrone2019 Test Set: Model training phase comparative experiment using different precisions. Epoch=100, Batch size=4.Image input size: 640 × 640 pixels.

	Training time	GPU Memory	MAP50	MAP95
FP32	**34 h**	**16.8G**	**41**	**23.6**
AMP	32 h	12.1G	40.7	23.2

SPD-Conv, DSConv, and CSL-MHSA to enhance the model's feature extraction capabilities and enrich semantic information. (d) Adoption of an optimal transport-based label assignment strategy, SimOTA, to address the lack of global consideration in label assignments. This mitigates potential negative impacts on network learning when dealing with ambiguous labels during assignment. (e) Finally, to further improve model inference speed, we utilize NVIDIA TensorRT for FP16 quantization. These improvements aim to increase the model's focus on small targets while minimizing model complexity, facilitating the development of more efficient models for resource-scarce airborne tasks. In comparison to the YOLOv5s baseline model, our approach excels in precision, recall, mAP50, and mAP50:95. Despite an increase in computational load, the model's complexity significantly decreases. Our proposed Small model achieves an inference speed of 24 frames per second (FPS), while the Nano model reaches 35 FPS, meeting real-time task demands while maintaining high performance and accuracy. Finally, the deployment of the model on an airborne edge computing platform (NVIDIA Jetson Xavier Nx) validates the practical applicability of our proposed method.

Table 11
Experiment on the VisDrone2019 test set: Performance comparison of models after FP32 and FP16 quantization. 'Pre' denotes preprocessing time, and 'Size' indicates the model file size.

Init	FP32	FP16	Scale	Pre	Inference	NMS	mAP50	FPS	Size
✔			Nano	1.1	36.8	**3**	34.9	24	**3.12M**
	✔		Nano	1.1	41.4	4.1	34.8	21	9.12M
		✔	Nano	**1**	**24.7**	3.1	34.8	**35**	5.18M
✔			Small	1.1	64.8	3	**40.8**	14	9.26M
	✔		Small	1	83.6	4.1	40.6	11	27M
		✔	Small	1.1	40.4	3.9	40.6	24	11M

CRediT authorship contribution statement

Chen Xue: Conceptualization, Methodology, Software, Investigation, Data curation, Writing – original draft. **Yuelong Xia:** Validation, Formal analysis, Writing – review & editing. **Mingjie Wu:** Validation, Writing – review & editing. **Zaiqing Chen:** Validation, Supervision. **Feiyan Cheng:** Validation, Supervision. **Lijun Yun:** Formal analysis, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Feiyan Cheng reports financial support was provided by National Natural Science Foundation of China. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (No. 62265017). This study was supported by Key Project of Yunnan Basic Research Program, grant number 202401AS070034, and Yumnan Normal University graduate research innovation fund project, grant number YJSJJ23-B130.

References

Bochkovskiy, A., Wang, C., & Liao, H. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. CoRR abs/2004.10934, URL: https://arxiv.org/abs/2004.10934.

Chen, Y., Kalantidis, Y., Li, J., Yan, S., & Feng, J. (2018). A^2-nets: Double attention networks. CoRR abs/1810.11579, URL: http://arxiv.org/abs/1810.11579.

Chen, S., Zhao, J., Zhou, Y., Wang, H., Yao, R., Zhang, L., & Xue, Y. (2023). Info-FPN: An informative feature pyramid network for object detection in remote sensing images. *Expert Systems with Applications*, *214*, Article 119132. http://dx.doi.org/10.1016/j.eswa.2022.119132, URL: https://www.sciencedirect.com/science/article/pii/S0957417422021509.

Dong, X., Yan, S., & Duan, C. (2022). A lightweight vehicles detection network model based on YOLOv5. *Engineering Applications of Artificial Intelligence*, *113*, Article 104914. http://dx.doi.org/10.1016/j.engappai.2022.104914, URL: https://www.sciencedirect.com/science/article/pii/S0952197622001415.

Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). YOLOX: Exceeding YOLO series in 2021. URL: http://arxiv.org/abs/2107.08430, arXiv:2107.08430 [cs].

Ghiasi, G., Lin, T.-Y., & Le, Q. V. (2019). NAS-FPN: Learning scalable feature pyramid architecture for object detection. In *2019 IEEE/CVF conference on computer vision and pattern recognition* (pp. 7029–7038). http://dx.doi.org/10.1109/CVPR.2019.00720.

Girshick, R. (2015). Fast R-CNN. In *2015 IEEE international conference on computer vision* (pp. 1440–1448). http://dx.doi.org/10.1109/ICCV.2015.169.

Hamzenejadi, M. H., & Mohseni, H. (2023). Fine-tuned YOLOv5 for real-time vehicle detection in UAV imagery: Architectural improvements and performance boost. *Expert Systems with Applications*, *231*, Article 120845. http://dx.doi.org/10.1016/j.eswa.2023.120845, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417423013477.

Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., & Xu, C. (2020). GhostNet: More features from cheap operations. In *2020 IEEE/CVF conference on computer vision and pattern recognition* (pp. 1577–1586). http://dx.doi.org/10.1109/CVPR42600.2020.00165.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(9), 1904–1916. http://dx.doi.org/10.1109/TPAMI.2015.2389824.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. URL: http://arxiv.org/abs/1704.04861, arXiv:1704.04861 [cs].

Jocher, G. (2020). Ultralytics YOLOv5. http://dx.doi.org/10.5281/zenodo.3908559, URL: https://github.com/ultralytics/yolov5.

Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLOv8. URL: https://github.com/ultralytics/ultralytics.

Li, M., Chen, Y., Zhang, T., & Huang, W. (2024). TA-YOLO: a lightweight small object detection model based on multi-dimensional trans-attention module for remote sensing images. *Complex & Intelligent Systems*, http://dx.doi.org/10.1007/s40747-024-01448-6.

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., & Wei, X. (2022). YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. http://dx.doi.org/10.48550/arXiv.2209.02976, arXiv e-prints, arXiv:2209.02976.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *2017 IEEE conference on computer vision and pattern recognition* (pp. 936–944). http://dx.doi.org/10.1109/CVPR.2017.106.

Lin, Z., Feng, M., dos Santos, C. N., Yu, M., Xiang, B., Zhou, B., & Bengio, Y. (2017). A structured self-attentive sentence embedding. CoRR abs/1703.03130, URL: http://arxiv.org/abs/1703.03130.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *2017 IEEE international conference on computer vision* (pp. 2999–3007). http://dx.doi.org/10.1109/ICCV.2017.324.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016a). SSD: Single shot MultiBox detector. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Computer vision – ECCV 2016* (pp. 21–37). Cham: Springer International Publishing.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016b). *Vol. 9905*, *SSD: Single shot MultiBox detector* (pp. 21–37). http://dx.doi.org/10.1007/978-3-319-46448-0_2, URL: http://arxiv.org/abs/1512.02325, arXiv:1512.02325 [cs].

Liu, S., Huang, D., & Wang, Y. (2019). Learning spatial fusion for single-shot object detection. CoRR abs/1911.09516, URL: http://arxiv.org/abs/1911.09516.

Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path aggregation network for instance segmentation. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 8759–8768). http://dx.doi.org/10.1109/CVPR.2018.00913.

Liu, Y., Sun, P., Wergeles, N., & Shang, Y. (2021). A survey and performance evaluation of deep learning methods for small object detection. *Expert Systems with Applications*, *172*, Article 114602. http://dx.doi.org/10.1016/j.eswa.2021.114602, URL: https://www.sciencedirect.com/science/article/pii/S0957417421000439.

Ma, S., Zhao, B., Hou, Z., Yu, W., Pu, L., & Yang, X. (2024). SOCF: A correlation filter for real-time UAV tracking based on spatial disturbance suppression and object saliency-aware. *Expert Systems with Applications*, *238*, Article 122131. http://dx.doi.org/10.1016/j.eswa.2023.122131, URL: https://www.sciencedirect.com/science/article/pii/S0957417423026337.

Mahaur, B., Mishra, K., & Kumar, A. (2023). An improved lightweight small object detection framework applied to real-time autonomous driving. *Expert Systems with Applications*, *234*, Article 121036. http://dx.doi.org/10.1016/j.eswa.2023.121036, URL: https://www.sciencedirect.com/science/article/pii/S0957417423015385.

Nascimento, M. G. D., Prisacariu, V., & Fawcett, R. (2019). Dsconv: Efficient convolution operator. In *2019 IEEE/CVF international conference on computer vision* (pp. 5147–5156). http://dx.doi.org/10.1109/ICCV.2019.00525.

Naseer, M., Ranasinghe, K., Khan, S. H., Hayat, M., Khan, F. S., & Yang, M. (2021). Intriguing properties of vision transformers. CoRR abs/2105.10497, URL: https://arxiv.org/abs/2105.10497.

Ouyang, D., He, S., Zhang, G., Luo, M., Guo, H., Zhan, J., & Huang, Z. (2023). Efficient multi-scale attention module with cross-spatial learning. In *ICASSP 2023 - 2023 IEEE international conference on acoustics, speech and signal processing* (pp. 1–5). http://dx.doi.org/10.1109/ICASSP49357.2023.10096516.

Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., & Lin, D. (2019). Libra R-CNN: towards balanced learning for object detection. CoRR abs/1904.02701, URL: http://arxiv.org/abs/1904.02701.

Rafique, S., Gul, S., Jan, K., & Khan, G. M. (2023). Optimized real-time parking management framework using deep learning. *Expert Systems with Applications*, *220*, Article 119686. http://dx.doi.org/10.1016/j.eswa.2023.119686, URL: https://www.sciencedirect.com/science/article/pii/S0957417423001872.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 779–788). http://dx.doi.org/10.1109/CVPR.2016.91.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. http://dx.doi.org/10.1109/TPAMI.2016.2577031.

Shi, Y., Jia, Y., & Zhang, X. (2024). FocusDet: an efficient object detector for small object. *Scientific Reports*, *14*(1), 10697. http://dx.doi.org/10.1038/s41598-024-61136-w.

Sun, F., He, N., Li, R., Wang, X., & Xu, S. (2024). GD-PAN: a multiscale fusion architecture applied to object detection in UAV aerial images. *Multimedia Systems*, *30*(3), 143. http://dx.doi.org/10.1007/s00530-024-01342-8.

Sunkara, R., & Luo, T. (2023). No more strided convolutions or pooling: A new CNN building block for low-resolution images and small objects. In M. Amini, S. Canu, A. Fischer, T. Guns, P. Novak, & G. Tsoumakas (Eds.), *Lecture notes in artificial intelligence*: *Vol. 13715*, *Machine learning and knowledge discovery in databases, ecml pkdd 2022, pt iii* (pp. 443–459). Salesforce; ASML; Confianceai; Expedia Grp; Google; CEA; Naver Labs; Criteo AI Lab; KNIME; Biomerieux; PythIA; Univ Grenoble Alps; Inria; Persyval 2; Soc Savante Francophone Apprentissage Machine; Normastic; Grenoble Ensimag INP; Litis; Springer; Inst Natl Sci Appliquees Rouen Normandie, http://dx.doi.org/10.1007/978-3-031-26409-2_27, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD), Grenoble, FRANCE, SEP 19-23, 2022.

Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *2020 IEEE/CVF conference on computer vision and pattern recognition* (pp. 10778–10787). http://dx.doi.org/10.1109/CVPR42600.2020.01079.

Tian, Z., Shen, C., Chen, H., & He, T. (2019). FCOS: Fully convolutional one-stage object detection. In *2019 IEEE/CVF international conference on computer vision* (pp. 9626–9635). http://dx.doi.org/10.1109/ICCV.2019.00972.

Torralba, Murphy, Freeman, & Rubin (2003). Context-based vision system for place and object recognition. *Vol. 1*, In *Proceedings ninth IEEE international conference on computer vision* (pp. 273–280). http://dx.doi.org/10.1109/ICCV.2003.1238354.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *NIPS'17, proceedings of the 31st international conference on neural information processing systems* (pp. 6000–6010). Red Hook, NY, USA: Curran Associates Inc..

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *2023 IEEE/CVF conference on computer vision and pattern recognition* (pp. 7464–7475). http://dx.doi.org/10.1109/CVPR52729.2023.00721.

Wang, X., Girshick, R., Gupta, A., & He, K. (2018). Non-local neural networks. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 7794–7803). http://dx.doi.org/10.1109/CVPR.2018.00813.

Wang, Z., Li, Y., Liu, Y., & Meng, F. (2024). Improved object detection via large kernel attention. *Expert Systems with Applications*, *240*, Article 122507. http://dx.doi.org/10.1016/j.eswa.2023.122507, URL: https://www.sciencedirect.com/science/article/pii/S0957417423030099.

Wang, J., Xu, C., Yang, W., & Yu, L. (2021). A normalized Gaussian Wasserstein distance for tiny object detection. arXiv preprint arXiv:2110.13389.

Wang, J., Yang, W., Guo, H., Zhang, R., & Xia, G.-S. (2021). Tiny object detection in aerial images. (pp. 3791–3798).

Wang, M., Yang, W., Wang, L., Chen, D., Wei, F., KeZiErBieKe, H., & Liao, Y. (2023). FE-YOLOv5: Feature enhancement network based on YOLOv5 for small object detection. *Journal of Visual Communication and Image Representation*, *90*, Article 103752. http://dx.doi.org/10.1016/j.jvcir.2023.103752, URL: https://www.sciencedirect.com/science/article/pii/S1047320323000020.

Wang, Y., Zou, H., Yin, M., & Zhang, X. (2023). SMFF-YOLO: A scale-adaptive YOLO algorithm with multi-level feature fusion for object detection in UAV Scenes. *Remote Sensing*, *15*(18), 4580. http://dx.doi.org/10.3390/rs15184580, URL: https://www.mdpi.com/2072-4292/15/18/4580.

Xiao, J., Yao, Y., Zhou, J., Guo, H., Yu, Q., & Wang, Y.-F. (2023). FDLR-Net: A feature decoupling and localization refinement network for object detection in remote sensing images. *Expert Systems with Applications*, *225*, Article 120068. http://dx.doi.org/10.1016/j.eswa.2023.120068, URL: https://www.sciencedirect.com/science/article/pii/S0957417423005705.

Yang, C., Huang, Z., & Wang, N. (2022). QueryDet: Cascaded sparse query for accelerating high-resolution small object detection. In *2022 IEEE/CVF conference on computer vision and pattern recognition* (pp. 13658–13667). http://dx.doi.org/10.1109/CVPR52688.2022.01330.

Yang, G., Lei, J., Zhu, Z., Cheng, S., Feng, Z., & Liang, R. (2023). AFPN: Asymptotic feature pyramid network for object detection. http://dx.doi.org/10.48550/arXiv.2306.15988, arXiv e-prints, arXiv:2306.15988.

Zhang, Y., Jia, R.-S., Yang, R., & Sun, H.-M. (2023). DSNet: A vehicle density estimation network based on multi-scale sensing of vehicle density in video images. *Expert Systems with Applications*, *234*, Article 121020. http://dx.doi.org/10.1016/j.eswa.2023.121020, URL: https://www.sciencedirect.com/science/article/pii/S0957417423015221.

Zhang, H., Wang, Y., Dayoub, F., & Sünderhauf, N. (2021). VarifocalNet: An iou-aware dense object detector. In *2021 IEEE/CVF conference on computer vision and pattern recognition* (pp. 8510–8519). http://dx.doi.org/10.1109/CVPR46437.2021.00841.

Zhang, J., Xia, K., Huang, Z., Wang, S., & Akindele, R. G. (2023). ETAM: Ensemble transformer with attention modules for detection of small objects. *Expert Systems with Applications*, *224*, Article 119997. http://dx.doi.org/10.1016/j.eswa.2023.119997, URL: https://www.sciencedirect.com/science/article/pii/S0957417423004992.

Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 6848–6856). http://dx.doi.org/10.1109/CVPR.2018.00716.

Zhou, Y. (2024). A YOLO-NL object detector for real-time detection. *Expert Systems with Applications*, *238*, Article 122256. http://dx.doi.org/10.1016/j.eswa.2023.122256, URL: https://www.sciencedirect.com/science/article/pii/S0957417423027586.

Zhu, X., Lyu, S., Wang, X., & Zhao, Q. (2021). TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In *2021 IEEE/CVF international conference on computer vision workshops* (pp. 2778–2788). http://dx.doi.org/10.1109/ICCVW54120.2021.00312.

Zhu, P., Wen, L., Du, D., Bian, X., Fan, H., Hu, Q., & Ling, H. (2021). Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(11), 7380–7399.