Heliyon 10 (2024) e32931

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Attention-based scale sequence network for small object detection

Young-Woon Lee [a], Byung-Gyu Kim [b,*]

[a] Department of Computer Engineering, Sunmoon University, Asan, Republic of Korea
[b] Division of Artificial Intelligence Engineering, Sookmyung Women's University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Small object detection
Feature pyramid network
Scale sequence
Attention mechanism
Deep learning

ABSTRACT

Recently, with the remarkable development of deep learning technology, achievements are being updated in various computer vision fields. In particular, the object recognition field is receiving the most attention. Nevertheless, recognition performance for small objects is still challenging. Its performance is of utmost importance in realistic applications such as searching for missing persons through aerial photography. The core structure of the object recognition neural network is the feature pyramid network (FPN). You Only Look Once (YOLO) is the most widely used representative model following this structure. In this study, we proposed an attention-based scale sequence network (ASSN) that improves the scale sequence feature pyramid network (ssFPN), enhancing the performance of the FPN-based detector for small objects. ASSN is a lightweight attention module optimized for FPN-based detectors and has the versatility to be applied to any model with a corresponding structure. The proposed ASSN demonstrated performance improvements compared to the baselines (YOLOv7 and YOLOv8) in average precision (AP) of up to 0.6%. Additionally, the AP for small objects (AP_S) showed also improvements of up to 1.9%. Furthermore, ASSN exhibits higher performance than ssFPN while achieving lightweightness and optimization, thereby improving computational complexity and processing speed. ASSN is open-source based on YOLO version 7 and 8. This can be found in our public repository: https://github.com/smu-ivpl/ASSN.git

1. Introduction

Thanks to recent remarkable developments in artificial neural network technology, notable achievements in many computer vision fields are being updated daily. In particular, object recognition or detection is a core technology for numerous computer vision applications, including object tracking, object segmentation, pose estimation, 3D object recognition, autonomous driving, and unmanned aerial systems.

LeCun et al. published a gradient-based learning methodology for Convolutional Neural Networks (CNN), sparking a boom in deep neural networks [1]. This also significantly changed the landscape of the object recognition field. Most early CNN-based object recognition models consisted of two steps: a region proposal step and an inference step. This configuration was highly dependent on the performance of the Region Proposal Network (RPN) and could not guarantee real-time performance due to its high computational complexity. As model structures developed, one-stage models such as You Only Look Once (YOLO) were proposed. These one-stage models were leading to improvements in inference speed and recognition rates at realistic levels [2].

* Corresponding author.
E-mail addresses: yw.lee@ivpl.sm.ac.kr (Y.-W. Lee), bg.kim@sookmyung.ac.kr (B.-G. Kim).

https://doi.org/10.1016/j.heliyon.2024.e32931
Received 13 February 2024; Received in revised form 11 June 2024; Accepted 12 June 2024

Available online 19 June 2024

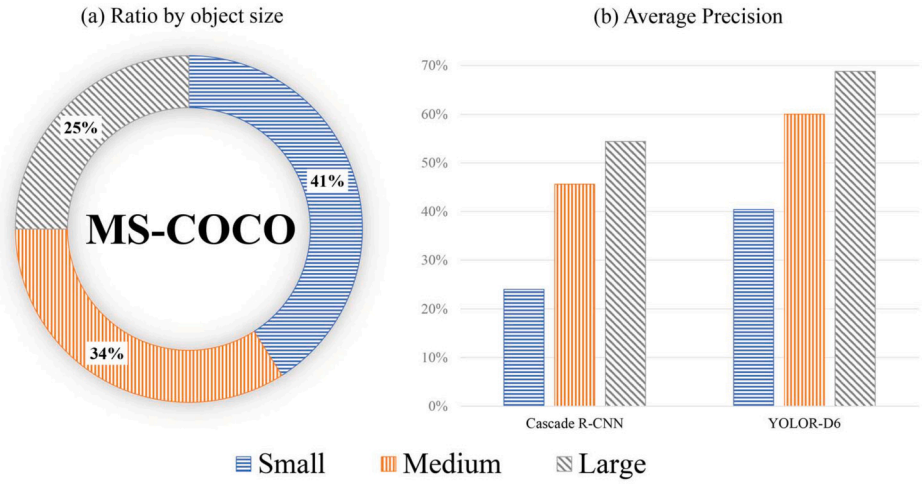

Fig. 1. Differences in distribution and recognition performance by size of the MS-COCO dataset.

The basic structure of recent object detection models is based on Feature Pyramid Networks (FPN), which construct pyramid-shaped feature maps by gradually reducing the spatial scale of the input image [3]. FPN is an approach for handling objects of various sizes in an image. However, FPN has structural problems in recognizing small objects. Specifically, compared to large objects, the probability of information disappearing for small objects increases as the spatial scale decreases. The fundamental cause of the small object problem is that as the network layer becomes deeper, features and location information about the bounding box are lost [4].

Microsoft Common Objects in Context (MS-COCO) is one of the public datasets for object detection and recognition [5]. Fig. 1 shows the ratio and the performance of data by object size in the MS-COCO. According to the definition of MS-COCO, the objects are divided into three categories as follows:

- small: segmentation mask size is 32×32 or less
- medium: segmentation mask size is more than 32×32 and 96×96 or less
- large: segmentation mask size is more than 96×96

As depicted in Fig. 1 (a), the proportion of small objects is the highest in the total data, but Fig. 1 (b) shows that the recognition rate is the lowest compared to medium/large objects. Additionally, it can be observed that although the overall recognition performance continues to increase as the model develops, the performance difference by object size exhibits a similar gap. It must be emphasized that this fact can be critical in certain application scenarios. For instance, we can consider aerial drone image processing to search for people in distress. This type of application is characterized by high resolution and small object size. Thus, making the recognition rate for small objects is the most crucial factor.

Objects of various sizes naturally exist in images, the models that is able to learn multi-scale features are required [6]. Many computer vision studies have attempted to design scale-invariant features to address this challenge. Scale-invariant features ensure consistent recognition rates even when image size changes. Thus, it is enabling effective recognition of small objects if the model can learn these features. Scale space, a multi-scale representation, is parameterized by the variance of the Gaussian kernel to extract scale-invariant features [7]. This implies that a multi-scale representation can consist of features of various resolutions.

The latest deep learning-based object detection models, including YOLO, largely adopt a configuration of Backbone + Head. By constructing a feature pyramid while gradually reducing the spatial scale at the Backbone, the Head detects the location of the object and classifies it. Depending on the model's structure, a separate Neck module may be introduced to enhance the performance of the Head. Before detecting an object in the Head module, each object's information is processed in different pyramid layers according to its size. However, a structure that processes separately into pyramids of different scales may cause semantic gaps between each feature [8]. This semantic gap ultimately results in differences in recognition performance by object size and tends to prioritize detecting large objects. Therefore, a new structure is needed to prevent information about small objects from being lost within multi-scale features and to resolve the semantic gap.

Attention-based methodologies have become a common focus in recent deep neural network research [9,10]. However, we discovered that applying attention modules collectively to all or part of the network is not suitable for this task. Moreover, we have previously proposed a Scale Sequence Feature Pyramid Network (ssFPN) to address small object problems [11]. Building upon this prior research, we aim to introduce a novel Attention-based Scale Sequence Network (ASSN) capable of directing attention to small objects and generating meaningful scale sequences. The contributions of this paper are three-fold:

- Scale sequence feature of ASSN can intensively improve small object recognition performance while also contributing to the overall performance of FPN-based object detection model.
- ASSN is a module with versatility that can be applied to all models using the FPN structure.

- ASSN is a very lightweight additional module that guarantees the real-time performance of the existing FPN-based detection model.

2. Related works

2.1. Feature pyramid network-based object detection

The history of neural network-based object detection models can be traced back to the R-CNN [12]. The R-CNN utilized a selective search algorithm for region proposals and applied bounding box regression. Also, it retained only boxes with high scores through Non-Maximum Suppression (NMS) and employed an Support Vector Machine (SVM)-based classifier.

Since then, numerous follow-up studies have been reported one after another to improve the R-CNN. SPPNet [13] addressed R-CNN's limitations of fixed input image size and overlapping CNN structure. Fast R-CNN [14] introduced an end-to-end framework that performs CNN fine-tuning, bounding box regression, and classification within a single network. Faster R-CNN [15] proposed the RPN for replacing selective search. Additionally, the introduction of Mask R-CNN [16], which added a separate mask branch for semantic segmentation, enabled more precise inference of object locations.

Despite the rapid development of object detection models, real-time performance was still not guaranteed. Redmon et al. addressed this issue by proposing a one-stage model that simultaneously performs region proposal and classification, named You Only Look Once (YOLO) [2]. YOLO has revolutionized real-time object detection technology and is currently at version 8 (YOLOv8) due to ongoing research efforts. The characteristics of each version of YOLO can be summarized as follows:

- YOLOv1 [2]: It introduced a one-stage deep learning network for real-time object detection.
- YOLOv2 [17]: Performance improvements were achieved by replacing the Fully-Connected (FC) layer with 1×1 convolution, and introducing global average pooling to reduce the number of parameters.
- YOLOv3 [18]: This version introduced numerous changes, including the removal of the pooling layer, introduction of ResNet's skip connections [19], adoption of SSD's multi-scale feature layer [20], application of RetinaNet's FPN technique [21], and enabling multi-label classification.
- YOLOv4 [22]: To address the small object detection issue, FPN and Path Aggregation Network (PAN) were introduced. Additionally, new network structures such as Cross-Stage Partial Connections (CSP)-based backbone connection and Spatial Pyramid Pooling (SPP) were introduced to enhance performance.
- YOLOv5 [23]: Performance was enhanced by introducing the BottleneckCSP module and an improved version of CSP.
- YOLOv6 [24]: Various techniques were introduced to increase algorithm efficiency. These include dynamically adjusting the knowledge of teachers and labels so that the student model can learn more efficiently across all learning stages during the self-distillation process.
- YOLOv7 [25]: It introduced model re-parameterization, label assignment, and extend and compound scaling methodologies. Additionally, an Extended Efficient Layer Aggregation Network (E-ELAN) from a structural aspect was proposed.
- YOLOv8 [26]: NMS speed was improved by proposing an anchor-free model that directly predicts the center of the object instead of the anchor box offset. Additionally, many structural changes were implemented.

Scale-invariant features remain robust despite changes in object size [6]. The Scale Invariant Feature Transform (SIFT) is a prominent algorithm for multi-scale object processing in traditional computer vision [27]. SIFT generates multi-scale features using Gaussian filters with different scale factors. Extrema are extracted from the feature maps using the Difference of Gaussian (DoG) function as key points for matching across different scale spaces.

In this manner, the image pyramid serves as the fundamental approach for most object recognition algorithms. Structural approaches to address the small object problem can be summarized as image pyramid structures such as FPN and feature map fusion techniques such as PAN. In the following subsections, we will explore the trends in small object detection.

2.2. Small object detection

Wang et al. introduced the scale-transfer and scale-relationship modules to address changes in the scale of text images [28]. Wang et al. considered the multi-scale feature pyramid as a scale space and utilized Deformable Convolution to equalize the scale features [29]. Their approach to scale space formation was effectively adopted in our study.

Azimi et al. introduced image cascade networks for small object detection in remote sensing images [30]. They focused on extracting semantic features at various scales by leveraging the hierarchical structure of FPN. Liu et al. proposed a high-resolution detection network that adjusted the resolution of the input image [31]. They utilized backbone networks with different depths for images of varying resolutions. Their approach demonstrated notable performance on the VisDrone2019 dataset [32] which specializes in small objects.

Chen et al. also utilized a backbone network that was separated from inputs with different resolutions [33]. They introduced a mechanism to extract attention features at different scales and enhanced the detection rate for small objects by assigning high weight to low-resolution images. Tao et al. expanded upon this work and proposed a hierarchical attention structure [34]. They devised a structure to infer the attention of adjacent scale pairs in a multi-scale network.

Many studies have introduced excessive additional modules to existing models, such as distinguishing input resolutions or introducing backbones with different structures to handle different scales. However, this approach may not guarantee real-time performance due to an unnecessary increase in computation and can make precise detection of small objects challenging. Our proposed ASSN does not alter the structure of existing object detection models that use only a single image input, thereby enhancing detection performance while minimizing complexity.

2.3. Feature fusion strategy

FPN extracts scale features of various resolutions and integrates them in a top-down path. Typically, in this pyramid structure, large objects can be detected at small resolutions while small objects can be detected at large resolutions, depending on the scale space characteristics. However, information inconsistencies and semantic gaps between these scales can adversely affect performance.

Liu et al. attempted to mitigate the gap by introducing bottom-up paths to FPN in their proposed PAN [8]. This method of path expansion between scale features has implications for more accurate bounding box inference. Tan et al. proposed a weighted fusion technique for feature pyramids [35]. Liu et al. proposed Composite Backbone Network Architecture (CBNet) [36]. As its name suggests, CBNet introduced multiple backbone networks for feature enhancement and fused features according to the Adjacent High-Level Composition (AHLC) method. Dai et al. introduced three attention modules: scale-aware attention, spatial-aware attention, and task-aware attention [37]. The features extracted from these are fused together to form a comprehensive view, from which the final self-attention is calculated.

This feature fusion approach has also been identified as a meaningful structure through neural Network Architecture Search (NAS) using reinforcement learning. NAS-FPN [38] showed a highly complex form of feature fusion in models derived from learning. There is a possibility that some meaningful fusion methodology exists between features extracted from these distinct pyramid paths. However, as observed in previous studies, the introduction of batch attention modules, indiscriminate skip connections, or feature fusion does not necessarily improve performance. From this perspective, we aimed to introduce an optimal scale sequence fusion methodology focused on small objects.

3. Attention-based scale sequence network

The typical structure of a detector models comprises a Backbone followed by a Head. After the Backbone generates a feature pyramid, the Head predicts the location and class of objects based on their scales. While the ASSN can be applied to any model with this structure, for the sake of clarity, this paper limits the experimental subject to YOLOv7 and YOLOv8. In previous research [11], we regarded the feature pyramid extracted from FPN as a scale space similar to the SEPC [29].

Fig. 2 illustrates the overall process of ASSN integrated into the detector model with a Backbone + Neck + Head structure. The primary objective of ASSN is to establish a new $\mathbf{P_3}$ dynamic head capable of enhancing small object detection performance. Fig. 3 shows the structural variances between the existing ssFPN and the enhanced ASSN. As depicted in the structural diagram above the dotted line, ssFPN heavily relies on the 3D convolution layer. In ssFPN, feature maps ($\mathbf{P_3} \sim \mathbf{P_5}$) with different resolutions were merged according to the scale space theory to construct a scale sequence. However, this was merely a straightforward upsampling and concatenation process. This design seemed somewhat lacking in justification for the purpose of recognizing small objects.

The bottom section of the dotted line in Fig. 3 illustrates the overall structure of the proposed ASSN. ASSN is a module designed in two stages to enhance small object detection. It generates a scale sequence from the feature pyramid involved in head calculation, refines it, and connects it to the dynamic head of $\mathbf{P_3}$. We recognized the need for a more sophisticated process in forming the scale sequence itself and thus introduced the attention method. Although the concept of applying attention to the image pyramid is not new, we reinterpreted the existing attention module to focus on small object detection. The Convolutional Block Attention Module (CBAM) [39] sequentially generates channel attention and spatial attention. Among the feature maps, $\mathbf{P_3}$ retains the most information about small objects. Therefore, to enhance features for small objects, it is effective to create an attention map specifically in $\mathbf{P_3}$. Reflecting this, the scale sequence ($\mathbf{S^2}$) features can be defined as follows:

$$\mathbf{S^2} = \{\mathbf{M_s}(\mathbf{P'_3}) \otimes \mathbf{P_i^N}\}_{\mathbf{i=3}}^{\mathbf{L}}, \quad \mathbf{N} = \mathbf{2^{i-3}}, \tag{1}$$

Here, $\mathbf{P'_3}$ corresponds to the result of applying the channel attention map to $\mathbf{P_3}$, and $\mathbf{M_s}(\mathbf{P'_3})$ represents the resulting spatial attention map. The symbol $\otimes$ denotes element-wise matrix multiplication. $\mathbf{P_i^N}$ denotes the $\mathbf{i-th}$ pyramid feature generated from the Backbone, and $\mathbf{L}$ denotes the total number of pyramids. $\mathbf{N}$ denotes the spatial resolution scale value of the feature map. For $\mathbf{i} > \mathbf{3}$, the spatial resolution of the feature map decreases by a factor of 2, so upsampling is performed to match the size of the $\mathbf{P_3}$ feature map. Specifically, $\mathbf{P_4}$ undergoes upsampling by a factor of $\times 2$, and $\mathbf{P_5}$ undergoes upsampling by a factor of $\times 4$. The process of generating a channel/spatial attention map from the $\mathbf{P_3}$ features is as follows:

$$\begin{aligned} \mathbf{P'_3} &= \mathbf{M_c}(\mathbf{P_3}) \otimes \mathbf{P_3}, \\ \mathbf{P''_3} &= \mathbf{M_s}(\mathbf{P'_3}) \otimes \mathbf{P'_3}, \end{aligned} \tag{2}$$

Here, $\mathbf{M_c}$ and $\mathbf{M_s}$ denote the channel attention map and spatial attention map, respectively. First, the process of generating a channel attention map from the $\mathbf{P_3}$ feature map is as follows:

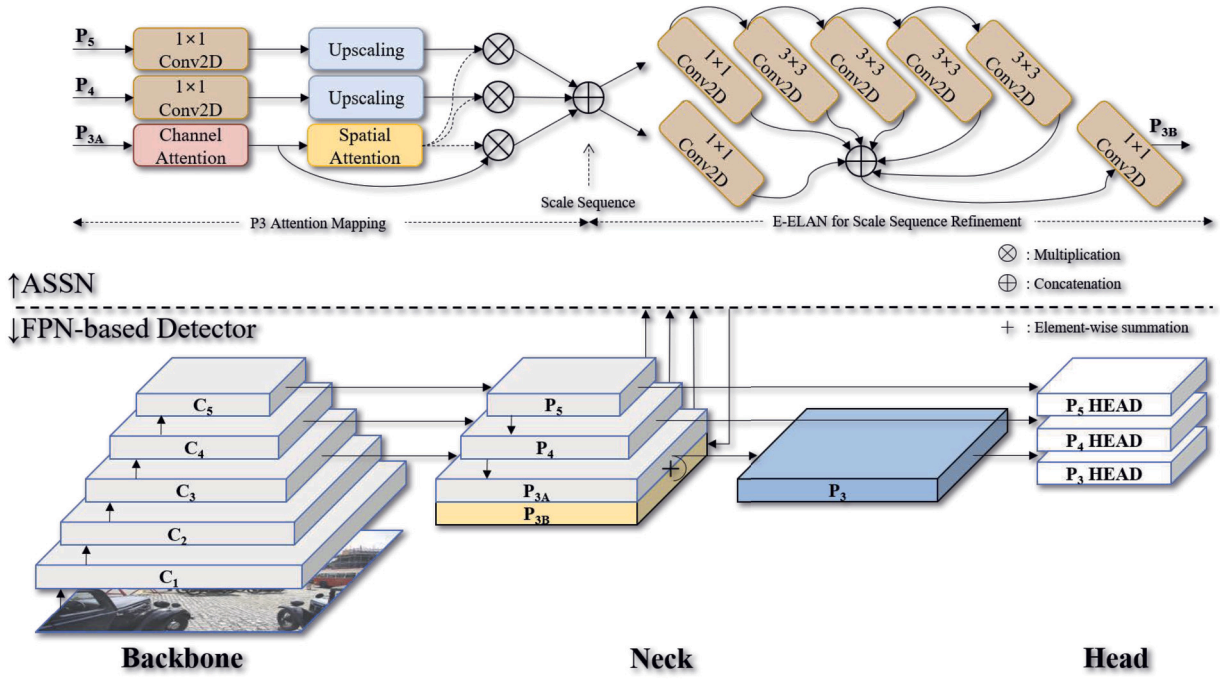

Fig. 2. Overall architecture of ASSN merged into the FPN-based object detector.

$$\begin{aligned}\mathbf{M_c}(\mathbf{P_3}) &= \sigma(MLP(AvgPool(\mathbf{P_3}))) + \sigma(MLP(MaxPool(\mathbf{P_3})))\\ &= \sigma(\mathbf{W_1}(SiLU(\mathbf{W_0}(\mathbf{P^c_{avg}})))) + \sigma(\mathbf{W_1}(SiLU(\mathbf{W_0}(\mathbf{P^c_{max}})))),\end{aligned} \tag{3}$$

Here, $AvgPool$ and $MaxPool$ denote average pooling and max pooling operations, respectively, and MLP represents a linear multi-layer perceptron. σ and $SiLU$ represent the activation functions sigmoid and swish [40], respectively. The channel attention map generated in this manner is applied to the input $\mathbf{P_3}$ to produce a spatial attention map as follows:

$$\begin{aligned}\mathbf{M_s}(\mathbf{P_3}) &= \sigma(BN(f^{7\times7}([AvgPool(\mathbf{P_3}); MaxPool(\mathbf{P_3})])))\\ &= \sigma(BN(f^{7\times7}([\mathbf{P^s_{avg}}; \mathbf{P^s_{max}}]))),\end{aligned} \tag{4}$$

Here, BN means batch normalization [41].

Fig. 4 provides an example illustrating the process of generating a $\mathbf{P_3}$ attention map as described above and creating a scale sequence containing emphasized information about small objects. It's important to understand the sequence of applying the channel attention and spatial attention maps. We introduced a method of generating an attention map solely from the $\mathbf{P_3}$ feature map and then applying it to all $\mathbf{P_i}$ feature maps involved in the Head operation. This approach significantly enhances detection performance for small objects compared to applying the attention mechanism to all network layers. Not only does it improve precision, but it also contributes to lightweighting by reducing unnecessary calculations, thereby ensuring the module's versatility. It's worth emphasizing that the attention map generation in ASSN exclusively targets the $\mathbf{P_3}$ feature map because spatial information about small objects is more likely to be present in $\mathbf{P_3}$, which has a relatively higher resolution. The spatial attention map $\mathbf{M_s}$ is applied not only to $\mathbf{P_3}$ but also to the upsampled counterparts. Consequently, the $\mathbf{S^2}$ feature map ultimately constitutes the $\mathbf{M_S}(\mathbf{P^N_i})$ set. This approach directs focused attention toward small objects, unlike the general method using CBAM. By applying the $\mathbf{P_3}$ attention map to all $\mathbf{P^N_i}$ participating in the Head, not only can the semantic gap between feature maps be bridged, but also the inference performance in the Head can be enhanced by preserving spatial information about small objects that may be lost in low resolution.

The scale sequence generated in this manner contains valuable information about small objects. But, it requires refinement before being linked to the Head, where location and class inference take place. In the case of existing ssFPN, a 3D convolution module was introduced to extract information from scale sequences. This is similar to extracting features from a video sequence by considering the correlation between adjacent frames. However, even the ssFPN exhibited significant performance in YOLOv4 and YOLOR, it is demonstrated lower performance in later YOLO versions. Consequently, we concluded that the scale sequence itself must already contain meaningful information and the subsequent refinement step should be needed.

The subsequent step for refining the scale sequence adopted the E-ELAN structure introduced in YOLOv7 [25]. Fig. 5 illustrates the module structure for refining the scale sequence from a feature map perspective. E-ELAN employs expand, shuffle, and merge cardinality to minimize computational overhead and enhance learning capability by configuring a deep network. Generally, it's known that increasing cardinality is more beneficial for performance than simply increasing the number of layers when constructing

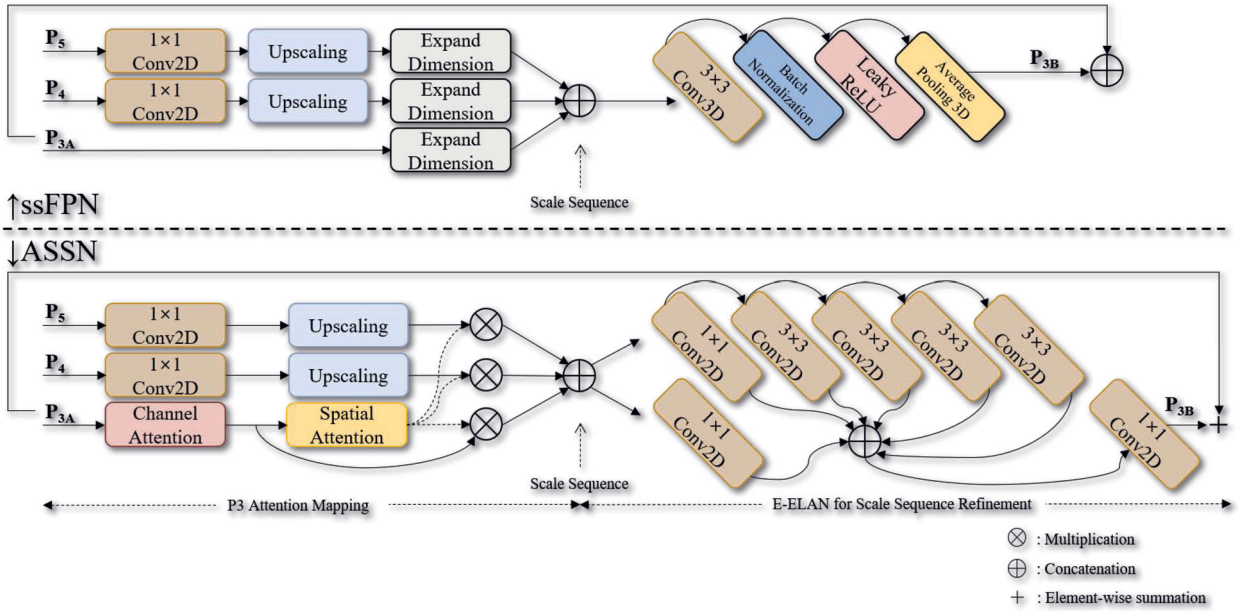

Fig. 3. Architectural comparison between ssFPN and ASSN.

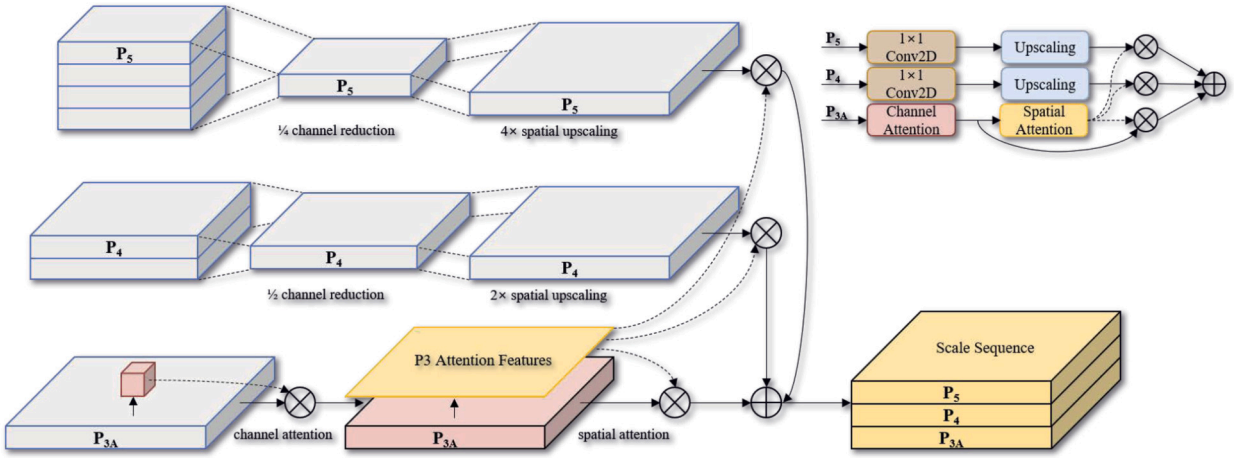

Fig. 4. $\mathbf{P_3}$ attention map creation process from a feature map perspective.

a deep network. The refinement stage structurally forms a deeper layer compared to the 3D convolution module of ssFPN, but it incurs lower computational cost.

In Fig. 3, it's notable that the operations for connecting the $\mathbf{P_3}$ feature maps generated by ssFPN and ASSN as the dynamic head differ. While ssFPN combined the feature maps of $\mathbf{P_{3B}}$ and $\mathbf{P_{3A}}$ generated in the module through concatenation, ASSN employed an add-based fusion method. This implies that ASSN itself functions as a residual block to compensate for information loss in the existing $\mathbf{P_3}$ pyramid feature. Through experimentation, we confirmed that the add-based fusion method outperforms the concatenation method. Add-based fusion also provided significant advantages in terms of computational complexity and contributed to the lightweight nature of ASSN.

3.1. Dataset and evaluation metrics

All experiments were conducted using the MS-COCO 2017 dataset [5] and the Pascal VOC 2007/2012 dataset [42,43]. The MS-COCO dataset comprises 80 object categories, with 118k images in the training set, 5k images in the validation set and 20k images in the test set. The Pascal VOC dataset consists of 20 object categories, with 17k images in the training set and 5k images in the validation set. Evaluation of all results was performed using the average precision (AP) metrics on both the MS-COCO and Pascal VOC datasets.

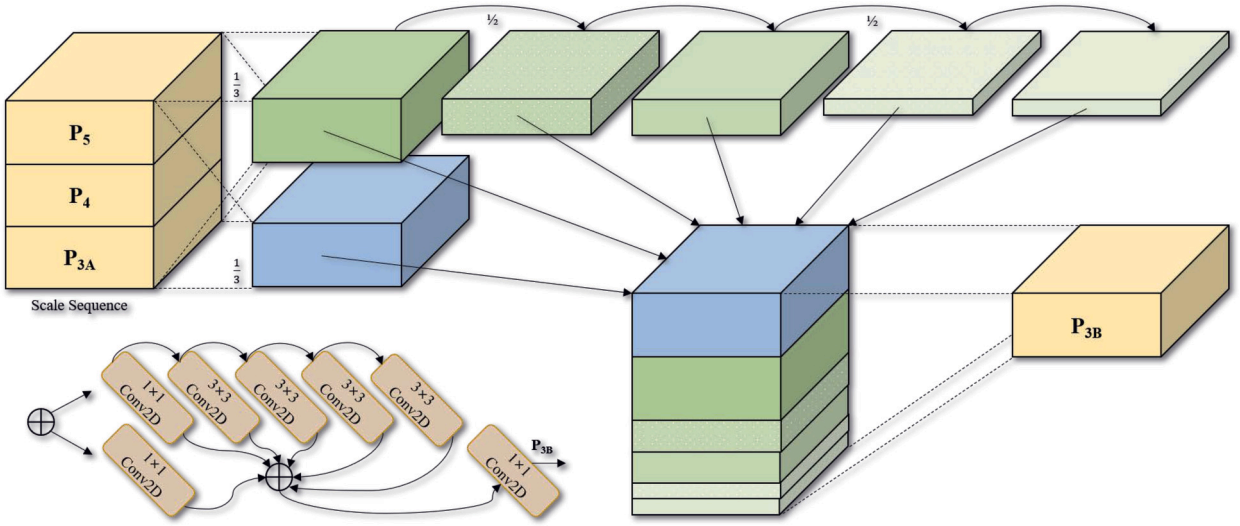

Fig. 5. Scale sequence refinement process from a feature map perspective.

Table 1
Environment for model training.

CPU	Intel® Core™ i9-10900X CPU @ 3.70GHz × 20
RAM	256 GB
GPU	NVIDIA GeForce RTX 3090 24GB × 4
OS	Ubuntu 20.04.6 LTS
Python	3.9.18
PyTorch	2.1.0
CUDA	12.1
CUDNN	8.9.2

We measured the average of the Intersection over Union (IoU) metrics for three types as follows. In addition to the average AP across the IoU range of 0.50 to 0.95, AP_{50} and AP_{75} were specifically calculated at IoU thresholds of 0.50 and 0.75, respectively. Furthermore, AP_S, AP_M, and AP_L were reported for small, medium, and large objects, respectively, based on their size categories.

3.2. Implementation details

We selected YOLOv7 and YOLOv8 as the baseline models to evaluate the performance of the proposed ASSN. Table 1 provides a summary of the experimental environment. PyTorch [44] was utilized as the deep learning framework for implementation. Transfer learning was applied using the pre-trained weights of YOLOv7 and YOLOv8. The hyper-parameter settings for training were configured to match the default settings of YOLOv7 and YOLOv8.

4. Experimental results

4.1. Performance analysis

Tables 2, 3, 4 and 5 show the performance comparison among the baseline models, ssFPN and the proposed ASSN. The experiment encompassed four P5 models (YOLOv7, YOLOv7-X, YOLOv8n, YOLOv8s) and one P6 model (YOLOv7-W6). Additionally, to verify the performance improvement of ASSN compared to ssFPN, we included the experimental results of ssFPN for certain baselines. Tables 3 and 4 provide a summary of the AP performance for MS-COCO's validation set (5k images) and test set (20k images), respectively.

When ssFPN was applied, the performance was decreased by -0.2% and -0.1% in YOLOv7, respectively. But, the performance was improved by +0.1% and +0.2% in the case of YOLOv7-X, respectively. Although these results are specific to the P5 model, it is verified that the ssFPN based on 3D convolution caused a performance degradation in YOLOv7. Conversely, higher performance improvements were verified of +0.3% in YOLOv7, +0.6% in YOLOv7-X and +0.4% in YOLOv7-W6 in validation set of MS-COCO, respectively. Also, better performances than baseline were shown of +0.5% in YOLOv7, +0.4% in YOLOv7-X and +0.3% in YOLOv7-W6 in test set of MS-COCO, respectively. Based on the test set of MS-COCO, the detection performance for small objects (AP_S) was enhanced by +0.6% in YOLOv7, +0.5% in YOLOv7-X, and +0.6% in YOLOv7-W6. Regarding YOLOv8, Table 3 demonstrates the performance integration with ASSN. AP performance was improved by 0.1% in YOLOv8s, while AP_S performance showed an enhancement of 1.9% in YOLOv8n and 1.3% in YOLOv8s, respectively. The experimental results clearly indicate that ASSN

Table 2
Complexity and processing speed comparison between the baselines (YOLOv7 and YOLOv8), the ssFPN, and proposed ASSN.

Model	Size	#Params (M)	FLOPs (G)	Batch32 (ms)	Batch1 (fps)
YOLOv7	640	36.9	104.7	7.1	77
YOLOv7-X	640	71.3	189.9	8.9	63
YOLOv7-W6	1280	70.4	89.9	13.7	47
YOLOv8n	640	3.2	8.7	1.6	83
YOLOv8s	640	11.2	28.6	2.8	87
ssFPN + YOLOv7	640	39.9 (+ 3.0)	175.9 (+ 71.2)	8.7 (+ 1.6)	65 (-12)
ssFPN + YOLOv7-X	640	74.8 (+ 3.5)	297.9 (+108.0)	11.3 (+ 2.4)	55 (- 8)
ssFPN + YOLOv7-W6	1280	86.1 (+15.7)	249.2 (+159.3)	29.1 (+16.6)	31 (-16)
ASSN + YOLOv7	640	39.5 (+ 2.6)	125.9 (+ 21.2)	7.7 (+ 0.6)	69 (- 8)
ASSN + YOLOv7-X	640	74.2 (+ 2.9)	219.5 (+ 29.6)	9.9 (+ 1.0)	58 (- 5)
ASSN + YOLOv7-W6	1280	84.4 (+14.0)	123.4 (+ 33.5)	19.6 (+ 5.9)	35 (-12)
ASSN + YOLOv8n	640	3.3 (+ 0.1)	9.9 (+ 1.4)	1.9 (+ 0.3)	68 (-15)
ASSN + YOLOv8s	640	11.6 (+ 0.4)	33.3 (+ 4.7)	3.3 (+ 0.5)	66 (-21)

Table 3
Performance comparison on MS-COCO validation set (5K images) between the baseline (YOLOv7, v8), the ssFPN, and proposed ASSN.

Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv7	51.2	69.7	55.5	35.2	56.0	66.7
YOLOv7-X	52.9	71.1	57.5	36.9	57.7	68.6
YOLOv7-W6	54.6	72.3	59.5	40.1	59.0	68.6
YOLOv8n	37.4	52.9	40.3	18.6	41.0	53.5
YOLOv8s	44.9	62.1	48.3	25.9	49.9	61.0
ssFPN + YOLOv7	51.0 (−0.2)	69.5 (−0.2)	55.3 (−0.2)	35.1 (−0.1)	55.8 (−0.2)	65.8 (−0.9)
ssFPN + YOLOv7-X	53.0 (+0.1)	71.1 (-)	57.7 (+0.2)	36.2 (-0.7)	57.7 (-)	68.6 (-)
ASSN + YOLOv7	51.5 (+0.3)	70.0 (+0.3)	56.1 (+0.6)	35.7 (+0.5)	56.3 (+0.3)	65.8 (-0.9)
ASSN + YOLOv7-X	53.5 (+0.6)	71.6 (+0.5)	58.2 (+0.7)	36.7 (-0.2)	58.3 (+0.6)	69.7 (+1.1)
ASSN + YOLOv7-W6	55.0 (+0.4)	72.7 (+0.4)	60.1 (+0.6)	40.0 (-0.1)	59.5 (+0.5)	68.2 (-0.4)
ASSN + YOLOv8n	37.4 (-)	53.2 (+0.3)	40.5 (+0.2)	20.5 (+1.9)	41.6 (+0.6)	51.6 (-1.9)
ASSN + YOLOv8s	45.0 (+0.1)	62.3 (+0.2)	49.1 (+0.8)	27.2 (+1.3)	50.3 (+0.4)	59.7 (-1.3)

Table 4
Performance comparison on MS-COCO test set (20K images) between the baseline (YOLOv7), the ssFPN, and proposed ASSN.

Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv7	51.4	69.7	55.9	31.8	55.5	65.0
YOLOv7-X	53.1	71.2	57.8	33.8	57.1	67.4
YOLOv7-W6	54.9	72.6	60.1	37.3	58.7	67.1
ssFPN + YOLOv7	51.3 (-0.1)	69.7 (-)	55.8 (-0.1)	32.0 (+0.2)	55.2 (-0.3)	64.5 (-0.5)
ssFPN + YOLOv7-X	53.3 (+0.2)	71.4 (+0.2)	58.0 (+0.2)	33.9 (+0.1)	57.2 (+0.1)	67.5 (+0.1)
ASSN + YOLOv7	51.9 (+0.5)	70.3 (+0.6)	56.5 (+0.4)	32.6 (+0.6)	55.7 (+0.3)	65.4 (+0.7)
ASSN + YOLOv7-X	53.5 (+0.4)	71.5 (+0.3)	58.2 (+0.4)	34.3 (+0.5)	57.5 (+0.4)	67.4 (-)
ASSN + YOLOv7-W6	55.2 (+0.3)	72.9 (+0.3)	60.5 (+0.4)	37.9 (+0.6)	58.8 (+0.1)	67.5 (+0.4)

significantly enhances the detection performance, demonstrating its effectiveness not only for small objects but also for objects of all sizes.

In the case of YOLOv7-W6, which is a P6 model, the feature pyramid generated from the Neck module consists of 2 stages. Please note that ASSN was applied to only one stage instead of both stages to maintain the lightweight nature of ASSN. Hence, it can be inferred that the performance of the P6 model has potential for further enhancement, leaving it as a topic for future research.

Table 5 shows the performance of YOLOv8 on the MS-COCO and Pascal VOC datasets. Since Pascal VOC does not define metrics for object sizes, only the performance for AP and AP_{50} was compared. From the result, it is verified that ASSN exhibits better performance regardless of the dataset type.

In terms of lightweighting, ASSN demonstrated a much lighter model structure and faster processing speed compared to ssFPN. The #Params column in Table 2 compares the number of learning parameters of ssFPN and ASSN relative to the baseline, expressed in millions (M). While ssFPN exhibited an increase in the number of parameters by +3.0M, +3.5M and +15.7M for each model, ASSN only increased the parameter count by +2.6M, +2.9M and +14.0M, respectively. Moreover, the computational complexity (FLOPs) shows a much larger decrease due to differences in operations. ssFPN displayed a significant increase in computational complexity

Table 5
Performance comparison by dataset.

Model	MS-COCO (validation set, 5K images)		Pascal-VOC (validation set, 5K images)	
	AP	AP_{50}	AP	AP_{50}
YOLOv8n	37.4	52.9	54.6	76.1
YOLOv8s	44.9	62.1	60.2	81.3
ASSN + YOLOv8n	37.4 (-)	53.2 (+0.3)	54.9 (+0.3)	76.3 (+0.2)
ASSN + YOLOv8s	45.1 (+0.1)	62.3 (+0.2)	60.5 (+0.3)	81.5 (+0.2)

Fig. 6. Visual performance comparison. *Left*: YOLOv7-W6. *Right*: ASSN + YOLOv7-W6. Only detection results for small objects smaller than 32×32 were visualized. It can be seen that ASSN is detecting more small objects in the area of green rectangles in each figure.

of +71.2G, +108.0G and +159.3G, respectively. But, ASSN showed only a relatively small increase in complexity of +21.2G, +29.6G and +33.5G, respectively. Notable disparities were also showed in the number of Frames Per Second (FPS) and processing speed for 32 batches. Using the YOLOv7-W6 model as a reference, which is the heaviest model, ssFPN exhibited a processing speed of 31 FPS and 29.1 ms. But, ASSN demonstrated a fast processing speed of 42 FPS and 13.5 ms.

Fig. 6 illustrates the small object recognition performance of the proposed ASSN. For visualization, images were sampled from MS-COCO's test-dev2017 dataset. The detection performance of the P6 model, YOLOv7-W6, and the ASSN-integrated model were compared. The detection confidence value was set to 0.25 and the internal image size was 1280×1280.

The bounding box was displayed only for inferred small objects (size 32×32 or less). The green rectangles highlighted in the figures at the same locations indicate that more small objects are being detected by the ASSN. Through this visual comparison, it is obvious that the proposed ASSN effectively enhanced the detection performance for small objects compared to the baseline detector.

5. Conclusions

We have proposed the Attention-based Scale Sequence Network (ASSN) to improve small object detection performance and evaluated its performance. Small object detection is a crucial task in real-world applications but it is still challenging. The process of training scale-invariant features constitutes the basic foundation of an FPN-based detection model. We reinterpreted attention module to focus on small objects. Consequently, a lightweight structure was applied to ASSN. But, the performance was more effective than baselines. ASSN has the versatility to be applied to all FPN-based models.

Through experimental results, we have verified that the ASSN was able to improve not only the detection performance of small objects but also the overall performance. In addition, through a lightweight model design, performance was improved without compromising the real-time nature of the model. We plan to continue experiments to further develop the technology and derive more generalized performance across various FPN-based models.

CRediT authorship contribution statement

Young-Woon Lee: Writing – original draft, Visualization, Validation, Software, Methodology, Conceptualization. **Byung-Gyu Kim:** Writing – review & editing, Supervision, Resources, Project administration.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Byung-Gyu Kim reports writing assistance was provided by Sookmyung Women's University. Byung-Gyu Kim reports a relationship with Heliyon Journal that includes: board membership. Byung-Gyu Kim has patent N/A pending to N/A. The corresponding author, Prof. Byung-Gyu Kim, is an associate editor of the Heliyon Journal. 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.

References

[1] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.
[2] J. Redmon, S.K. Divvala, R.B. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 779–788.
[3] T.-Y. Lin, P. Dollár, R.B. Girshick, K. He, B. Hariharan, S.J. Belongie, Feature pyramid networks for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 936–944.
[4] K. Tong, Y. Wu, F. Zhou, Recent advances in small object detection based on deep learning: a review, Image Vis. Comput. 97 (2020) 103910.
[5] T.-Y. Lin, M. Maire, S.J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: common objects in context, in: European Conference on Computer Vision, 2014.
[6] D.G. Lowe, Object recognition from local scale-invariant features, in: Proceedings of the Seventh IEEE International Conference on Computer Vision 2, vol. 2, 1999, pp. 1150–1157.
[7] T. Lindeberg, Scale-Space Theory in Computer Vision, Lecture Notes in Computer Science, 1993.
[8] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, Path aggregation network for instance segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 8759–8768.
[9] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-excitation networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017, pp. 7132–7141.
[10] A. Vaswani, N.M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Neural Information Processing Systems, 2017.
[11] H.-J. Park, J.-W. Kang, B.-G. Kim, ssfpn: scale sequence (s2) feature-based feature pyramid network for object detection, Sensors (Basel, Switzerland) 23.
[12] R.B. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 580–587.
[13] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE Trans. Pattern Anal. Mach. Intell. 37 (2014) 1904–1916.
[14] R.B. Girshick, Fast r-cnn, 2015.
[15] S. Ren, K. He, R.B. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2015) 1137–1149.
[16] K. He, G. Gkioxari, P. Dollár, R.B. Girshick, Mask r-cnn, 2017.
[17] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 6517–6525.

[18] J. Redmon, A. Farhadi, Yolov3: an incremental improvement, arXiv:1804.02767 [abs].
[19] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 770–778.
[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S.E. Reed, C.-Y. Fu, A.C. Berg, Ssd: single shot multibox detector, in: European Conference on Computer Vision, 2015.
[21] T.-Y. Lin, P. Goyal, R.B. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2999–3007.
[22] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: optimal speed and accuracy of object detection, arXiv:2004.10934 [abs].
[23] G. Jocher, Yolov5 by ultralytics, 2020, https://github.com/ultralytics/yolov5.
[24] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, X. Wei, Yolov6: a single-stage object detection framework for industrial applications, arXiv:2209.02976 [abs].
[25] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, 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 (CVPR), 2022, pp. 7464–7475.
[26] G. Jocher, A. Chaurasia, J. Qiu, Ultralytics yolov8, 2023, https://github.com/ultralytics/ultralytics.
[27] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2004) 91–110.
[28] Y. Wang, H. Xie, Z. Fu, Y. Zhang, Dsrn: a deep scale relationship network for scene text detection, in: International Joint Conference on Artificial Intelligence, 2019.
[29] X. Wang, S. Zhang, Z. Yu, L. Feng, W. Zhang, Scale-equalizing pyramid convolution for object detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 13356–13365.
[30] S.M. Azimi, E. Vig, R. Bahmanyar, M. Körner, P. Reinartz, Towards multi-class object detection in unconstrained remote sensing imagery, in: Asian Conference on Computer Vision, 2018.
[31] Z. Liu, G. Gao, L. Sun, Z. Fang, Hrdnet: high-resolution detection network for small objects, in: 2021 IEEE International Conference on Multimedia and Expo (ICME), 2020, pp. 1–6.
[32] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, H. Ling, Detection and tracking meet drones challenge, IEEE Trans. Pattern Anal. Mach. Intell. 44 (11) (2021) 7380–7399.
[33] L.-C. Chen, Y. Yang, J. Wang, W. Xu, A.L. Yuille, Attention to scale: scale-aware semantic image segmentation, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3640–3649.
[34] A. Tao, K. Sapra, B. Catanzaro, Hierarchical multi-scale attention for semantic segmentation, arXiv:2005.10821 [abs].
[35] M. Tan, R. Pang, Q.V. Le, Efficientdet: scalable and efficient object detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 10778–10787.
[36] Y. Liu, Y. Wang, S. Wang, T. Liang, Q. Zhao, Z. Tang, H. Ling, Cbnet: a novel composite backbone network architecture for object detection, arXiv:1909.03625 [abs].
[37] X. Dai, Y. Chen, B. Xiao, D. Chen, M. Liu, L. Yuan, L. Zhang, Dynamic head: unifying object detection heads with attentions, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 7369–7378.
[38] G. Ghiasi, T.-Y. Lin, R. Pang, Q.V. Le, Nas-fpn: Learning scalable feature pyramid architecture for object detection, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7029–7038.
[39] S. Woo, J. Park, J.-Y. Lee, I.-S. Kweon, Cbam: convolutional block attention module, arXiv:1807.06521 [abs].
[40] P. Ramachandran, B. Zoph, Q.V. Le, Swish: a self-gated activation function, arXiv: Neural and Evolutionary Computing.
[41] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv:1502.03167 [abs].
[42] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, A. Zisserman, The Pascal visual object classes challenge 2007 (VOC2007) results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.
[43] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, A. Zisserman, The Pascal visual object classes challenge 2012 (VOC2012) results, http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.
[44] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: an imperative style, high-performance deep learning library, in: Neural Information Processing Systems, 2019.