



RESEARCH ARTICLE | MAY 04 2023

Compact Sparse R-CNN: Speeding up sparse R-CNN by reducing iterative detection heads and simplifying feature pyramid network

Zihang He  ; Xiang Ye  ; Yong Li 

AIP Advances 13, 055205 (2023)

<https://doi.org/10.1063/5.0146453>View
OnlineExport
Citation

CrossMark

Articles You May Be Interested In

Autonomous corrosion detection of inside and outside steel pipeline by using YOLO as fast algorithm on image processing

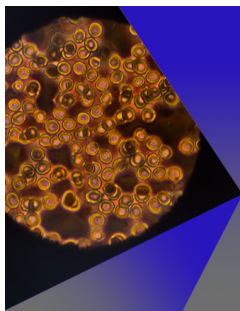
AIP Conference Proceedings (May 2023)

A detection method for impact point water columns based on improved YOLO X

AIP Advances (June 2022)

A data-driven deep learning approach for predicting separation-induced transition of submarines

Physics of Fluids (February 2022)



AIP Advances

Special Topic: Medical Applications
of Nanoscience and Nanotechnology

Submit Today!

 AIP
Publishing

Compact Sparse R-CNN: Speeding up sparse R-CNN by reducing iterative detection heads and simplifying feature pyramid network

Cite as: AIP Advances 13, 055205 (2023); doi: 10.1063/5.0146453

Submitted: 14 February 2023 • Accepted: 11 April 2023 •

Published Online: 4 May 2023



View Online



Export Citation



CrossMark

Zihang He,  Xiang Ye,  and Yong Li^{a)}

AFFILIATIONS

School of Electronic Engineering, Beijing University of Posts and Telecommunications, 10 Xitucheng Road, Beijing, China

^{a)} Author to whom correspondence should be addressed: yli@bupt.edu.cn

ABSTRACT

Processing a large number of proposals usually takes a significant proportion of inference time in two-stage object detection methods. Sparse regions with CNN features (Sparse R-CNN) was proposed using a small number of learnable proposals to replace the proposals derived from anchors. To decrease the missing rate, Sparse R-CNN uses six iterative detection heads to gradually regress the detection boxes to the corresponding objects, which hence increases the inference time. To reduce the number of iterative heads, we propose the iterative Hungarian assigner that encourages Sparse R-CNN to generate multiple proposals for each object at the inference stage. This decreases the missing rate when the number of iterative heads is small. As a result, Sparse R-CNN using the proposed assigner needs fewer iterative heads but gives higher detection accuracy. Also, we observe that the multi-layer outputs of the feature pyramid network contribute little to Sparse R-CNN and propose using a single-layer output neck to replace it. The single-layer output neck further improves the inference speed of Sparse R-CNN without the cost of detection accuracy. Experimental results show that the proposed iterative Hungarian assigner together with the single-layer output neck improves Sparse R-CNN by 2.5 AP₅₀ on the Microsoft common objects in context (MS-COCO) dataset and improves Sparse R-CNN by 3.0 AP₅₀ on the PASCAL visual object classes (VOC) dataset while decreasing 30% floating point operations (FLOPs).

© 2023 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>). <https://doi.org/10.1063/5.0146453>

I. INTRODUCTION

Object detection based on deep neural networks has been drawing much research attention in recent years.^{1,2} The inference speed of the detection methods plays a critical role in many applications. Object detection methods can be classified into single-stage methods³⁻¹¹ and two-stage methods.¹²⁻¹⁹ Single-stage methods directly produce detection boxes by classifying and regressing anchors that are manually set at each pixel on the feature map. Different from single-stage methods, two-stage methods produce detection boxes by classifying and regressing proposals. Specifically, two-stage methods have an additional region proposal network (RPN) and use it to classify the anchors into foreground anchors and background anchors, and only the foreground anchors as proposals are retained and fed into the detection network, as shown in Fig. 1. The proposals generated by RPN are derived from anchors and hence are written as “anchor-derived proposals” by us. Although

the number of proposals is much smaller than the anchors, processing them in the detection network still takes a significant proportion of inference time, even though the detection heads are relatively lightweight compared with the backbone and neck.

To reduce invalid and duplicate proposals while keeping valid proposals, sparse regions with CNN features¹⁷ (Sparse R-CNN) replaced the anchor-derived proposals with the learnable proposals and successfully reduced the number of proposals to 300. The learnable proposals are embedded as parameters in the networks and can be trained, so when proposals are fewer, they are easier to be regressed than non-learnable anchor-derived proposals. Nevertheless, the learnable proposals are too sparse to overlap with all objects, and the non-overlap objects will be ignored in the training under the prevalent max intersection over union (Max-IoU) assigner. Thus, Sparse R-CNN applied the Hungarian assigner proposed in detection Transformer²⁰ (DETR) to assign each object with a single proposal. Also, six iterative detection heads are applied to

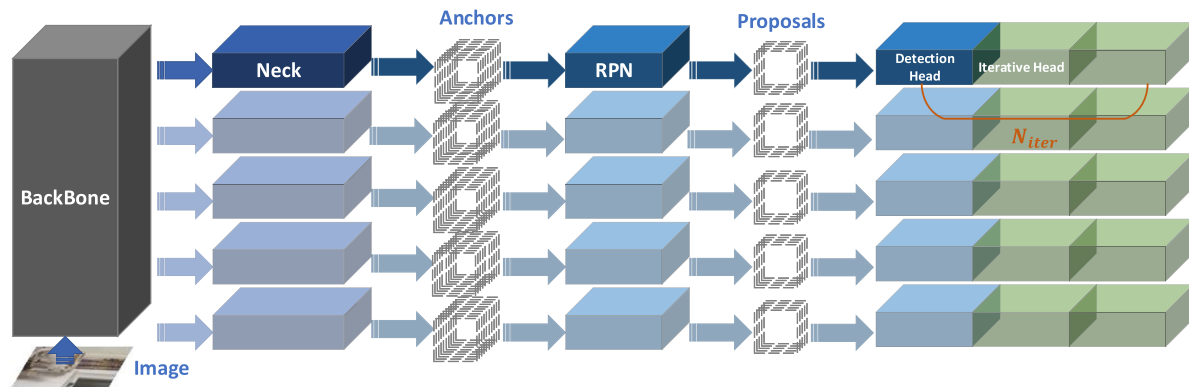


FIG. 1. The pipeline of two-stage methods. The backbone extracts feature maps from images and feeds them in the neck network. Then, the neck network outputs feature maps on multiple scales and sets anchors on them. RPN classifies and regresses the anchors to produce proposals, and they are fed into detection heads to generate the detection boxes.

ensure the single proposal gradually regresses to the object. Iterative detection heads were proposed in Cascade R-CNN¹⁵ (mentioned as the detection stage) in which the current detection head is fed in the boxes predicted by the former detection head, except that the first detection head is fed in proposals.

Although the iterative detection head guarantees the detection accuracy of object detection models, it increases the computation of processing proposals in detection heads by N_{iter} times where N_{iter} denotes the number of iterative heads for each proposal. Existing detection head networks typically require three iterative heads to finely localize objects for anchor-derived proposals, but Sparse R-CNN using learnable proposals requires six iterative heads due to fewer proposals.

To decrease the number of iterative heads when using learnable proposals, we proposed the iterative Hungarian assigner (IHA) that assigns multiple positive proposals (exclusive for different objects) instead of one single proposal to each object during the training process. IHA encourages the trained backbone to generate multiple proposals for each object at the inference stage, and one of them is easier to be regressed than the proposal generated without IHA. Therefore, fewer iterative heads are needed and N_{iter} can be reduced to 3 or 4.

Since the learnable proposal is capable of learning, modules that make up for shortcomings of the anchor-derived proposal may be ineffective for the learnable proposal. Specifically, the anchor-derived proposals yielded from anchors that have fixed sizes, and so cannot be regressed to a size very different from them, e.g., a proposal with the size of $[32, 32]$ is challenging to be regressed to $[128, 128]$ to detect the target with a similar size. For detecting objects of various sizes, the feature pyramid network (FPN) scales feature maps to different sizes and outputs them to region of interest (RoI) heads at multi-layers. Thus, the targets on feature maps are together scaled, and anchor-derived proposals with limited sizes can be easily regressed to one of the scaled targets. However, the sizes of learnable proposals are gradually adjusted through the training and, thus, are diverse, covering most anchor-derived proposals. As a result, learnable proposals are more easily to be regressed to targets than the anchor-derived proposals on a single-scale feature

map. Observing that multi-layer outputs of the FPN²¹ contribute little to the structure employing the learnable proposals, we propose using a single-layer output neck (SLON). SLON gives the same detection accuracy while eliminating the computation brought by the multi-layer outputs.

In conclusion, we propose speeding up Sparse R-CNN by simplifying redundant structures, including the iterative detection heads and FPN, without damaging the detection performance of it.

The main contribution of this paper is summarized as follows:

- (1) Sparse R-CNN finely regresses the single proposal by iterative heads. To speed the inference, we reduce the number of iterative heads and propose the iterative Hungarian assigner (IHA) that assigns multiple positive proposals instead of one single proposal to each object. At the inference stage, one of the produced multiple proposals may have a comparable regression accuracy to the finely regressed proposal by Sparse R-CNN, and hence, the detection performance does not deteriorate.
- (2) The size of the learnable proposals on a single-scale feature map can cover most anchor-derived proposals on multi-scale feature maps. Observing this, we propose using a single-layer output neck (SLON) in replacement of FPN to speed the inference without the cost of the detection performance.
- (3) Various experiments on the Microsoft common objects in context (MS-COCO) dataset²² and PASCAL visual object classes (VOC) dataset²³ show that the proposed IHA and SLON together reduce the computation of Sparse R-CNN, decreasing the floating point operations (FLOPs) by about 30% while slightly improving the detection accuracy.

II. RELATED WORKS

Object detection methods based on deep neural networks can be classified into two categories: single-stage methods and two-stage methods. Both categories of methods have been deeply researched and achieved very intriguing detection results. This paper aims at speeding up Sparse R-CNN, one of the two-stage detection methods, and we will give an overview of them in this section. The speeding

strategies for the two-stage methods will also be reviewed and discussed.

In the literature, many two-stage detection methods or structures have been proposed, including the selective search,²⁴ R-CNN,²⁵ Fast R-CNN,¹² Faster R-CNN,¹⁴ FPN,²¹ Mask R-CNN,¹⁶ Cascade R-CNN,¹⁵ Sparse R-CNN,¹⁷ non max suppression²⁶ (NMS), and Soft NMS.²⁷ Their structures generally comprise two stages, the proposal-generation stage and the detection stage. Among the two-stage methods is R-CNN²⁵ that normally used the selective search²⁴ on the central processing unit (CPU) to produce 2000 proposals. To accelerate R-CNN, Girshick¹² proposed Fast R-CNN employing the RoI pooling to resize the proposals to the same size for batch processing them on the CPU. In order to reduce the computation on CPU, Ren proposed Faster R-CNN¹⁴ that replaced the selective search with the region proposal network (RPN) to generate 1000 proposals from anchors. Since RPN is a lightweight module and can be computed together with other network structures on GPU, Faster R-CNN greatly reduced the inference time.

Sparse R-CNN¹⁷ replaced the anchor-derived proposals with the learnable proposals and successfully reduced the number of proposals to 300. The learnable proposals are embedded as parameters in the networks and can be trained, so when proposals are fewer, they are easier to be regressed than non-learnable anchor-derived proposals. Nevertheless, the learnable proposals are too sparse to overlap with all objects and the non-overlap objects will be ignored in the training under the prevalent Max-IoU assigner. Thus, Sparse R-CNN applied the Hungarian assigner proposed in DETR²⁰ to assign each object with a single proposal. For well regressing the single proposal to the object, a finer regression structure of six heads is required. The speed of Sparse R-CNN was slowed by six iterative heads for each learnable proposal.

Since ResNet,²⁸ many heavy-weight models were proposed in order to extract more abstract image features as backbones and hence enhance the detection performance. The heavy-weight backbone structures include Hourglass,²⁹ ResNext,³⁰ CspNet,³¹ and Swin Transformer.³² To effectively fuse multi-scale feature maps, a feature pyramid network (FPN)²¹ was proposed and fed in the output feature maps of multiple scales from the backbone. FPN fuses them and outputs the feature maps of multiple scales for detecting objects of different sizes, as illustrated in Fig. 3(a). FPN can be formulated as

$$Out_i = Conv(F_i) = Conv(Conv(In_i) + F_{i+1} \uparrow), \quad (1)$$

where In_i , F_i , and Out_i denote the input, the indeed feature maps, and the output of the i_{th} layer of FPN, respectively, $Conv$ denotes the convolution function, and \uparrow denotes upsampling by a factor 2. FPN defined the backbone-neck-head structure for the two-stage methods, and a number of variants have been put forward to improve FPN, such as deep feature pyramid reconfiguration (DFPR),³³ path aggregation network³⁴ (PANet), neural architecture search FPN³⁵ (NAS-FPN), and EfficientDet.¹⁰ However, FPN and its variants increased the output stages (4–5) of the neck, and thus increased the corresponding computation. Also, the FPN employed by Sparse R-CNN was observed to be less effective for learnable proposals than for anchor-derived proposals. Seeing this, our paper considered speeding up Sparse R-CNN by simplifying the iterative heads and FPN.

III. METHODS

This section discusses speeding up Sparse R-CNN by simplifying the redundant structures of it. We first designed the iterative Hungarian assigner to reduce the number of iterative heads when using learnable proposals and then proposed a single-layer output neck to replace the multi-layer outputs in FPN. Finally, we discussed how the proposed methods speed up Sparse R-CNN.

A. Iterative Hungarian assigner

Learnable proposals proposed and used in Sparse R-CNN are sparsely distributed, and the number of them is small compared with the normal proposals. The sparse distribution and the small number of proposals will bring a challenge for existing assigners, forcing Sparse R-CNN to apply more iterative detection heads to better regress the proposals. The Max-IoU assigner shown in Fig. 2(a) assigns each object with the most overlapped proposal but small objects may be assigned with no or few overlapped proposals, which causes the missing and/or localization inaccuracy of the small objects. The Hungarian assigner shown in Fig. 2(b) assigns each object with an exclusive proposal and, thus, reduces the total number of positive proposals. As a result, the possibility of each proposal being positive decreases at the inference stage, increasing the missing rate. The uniform assigner shown in Fig. 2(c) assigns one proposal to multiple objects, causing coverage confusion of the proposals.

Building a specialized assigner for Sparse R-CNN can address the challenge and contribute to regressing learnable proposals with fewer iterative detection heads. This work proposes the iterative Hungarian assigner (IHA) that assigns each object with an equal number of non-repetitive proposals, as shown in Fig. 2(d). IHA includes K iterations. For every iteration, IHA assigns each object with a proposal that has not been assigned to any objects in previous iterations. Specifically, the assignment in each iteration follows the rule of ensuring that the sum of the loss values between the assigned proposals and objects is minimal and is implemented by the Hungarian algorithm. After the iteration process, each object will be assigned with K exclusive proposals. Soft-NMS²⁷ is applied in the post-processing to exclude redundant proposals if multiple proposals locate on the same object at the inference stage.

Supposing that a single target exists and multiple proposals are produced. On the condition of applying fewer iterative heads, the model trained by the Hungarian assigner will only select one proposal to have high confidence. Thus, the target will be missing if the selected proposal is far from the target since fewer iterative heads cannot regress such a long distance. However, the model trained by IHA will select K proposals. One of them is more likely closer to the target and can be successfully regressed with fewer iterative heads. IHA enables to effectively reduce the number of iterative heads to 2 even when only a small number of proposals are used. Experimental results showed that IHA outperformed the Hungarian assigner when no other data augmentation is applied.

Also, IHA has the advantage of assigning an equal number of proposals to objects of various sizes, making small objects better detected than the Max-IoU assigner. Moreover, the assigned proposals for each object are exclusive, avoiding disturbing the convergence of the network.

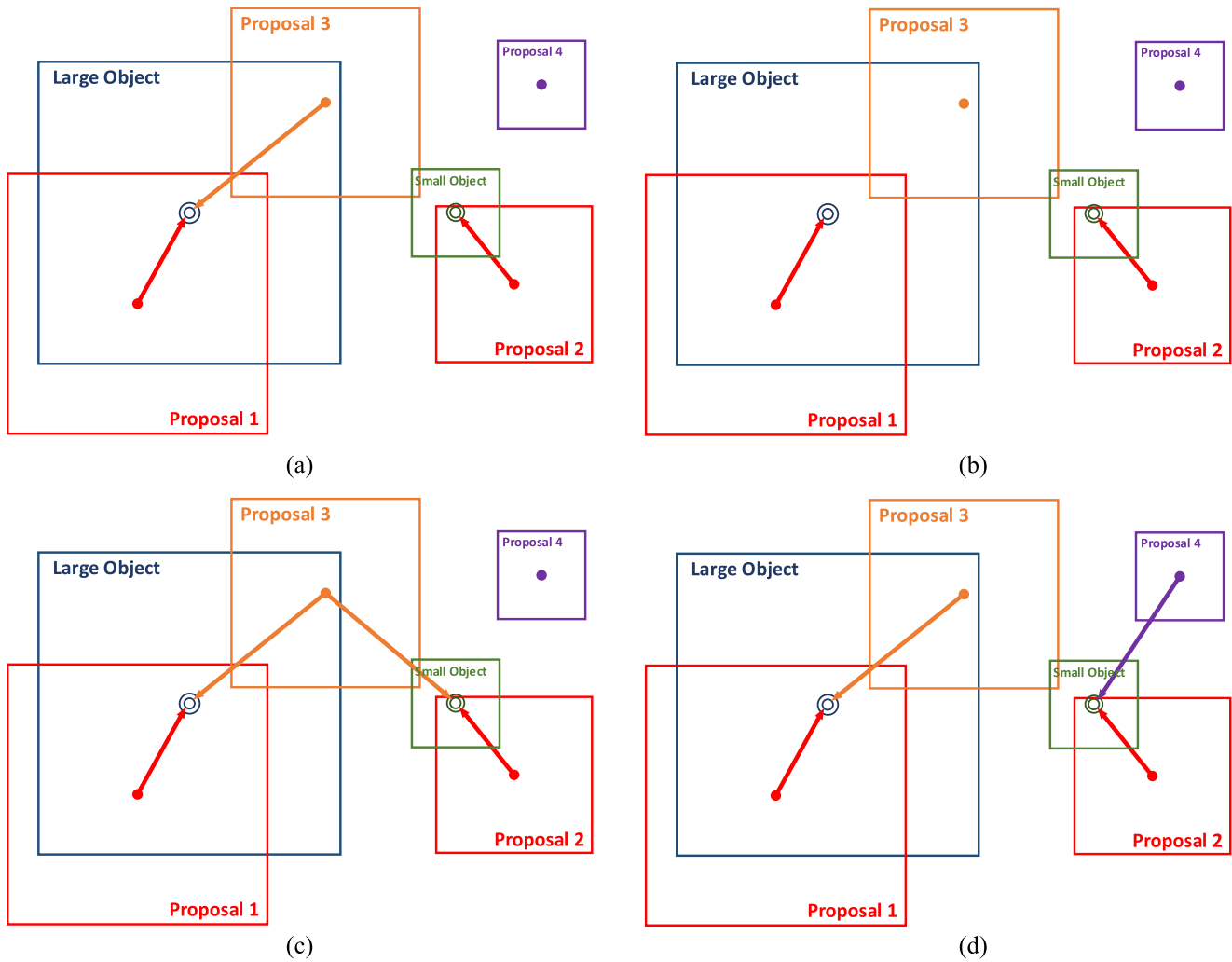


FIG. 2. The process of assigning proposals to objects. The green box denotes the smaller object, the blue box denotes the larger object, boxes of other colors denote proposals, and the arrows denote the assigning processing. (a) The Max-IoU assigner assigns more proposals to large objects and fewer proposals to small objects. (b) The Hungarian assigner assigns each object with one single proposal, decreasing the number of positive proposals. (c) The uniform assigner may assign one proposal to multiple objects, especially when a small number of proposals are used, e.g., the orange proposal is assigned to two objects. (d) The iterative Hungarian assigner assigns multiple exclusive proposals to each object, facilitating the detection of small objects and making the localization more accurate.

B. Single-layer output neck

FPN formulated as Eq. (1) has two functionalities: multi-scale feature fusion and various-scale detection. Multi-scale feature fusion is able to encode abundant context information from feature maps on various scales. Various-scale detection divides feature maps into multiple scales and conducts the detection on them, which has the advantage of assigning a proposal of a similar size to the target.

We investigated the contribution of two functionalities in Sparse R-CNN and found that keeping only multi-scale feature fusion negligibly affects the average precision (AP) (39.6 vs 39.7 as given in Table IV) but keeping only various-scale detection decreases the AP from 39.6 to 31.7. This says that various-scale detection plays

a much less important role. The reason may be that the size of the proposals is also learnable in Sparse R-CNN, and the learnable size can cover most anchor-derived proposals. Typically, anchor-derived proposals are yielded from the anchors of fixed scales on a single feature map, and thus, various-scale detection is needed to map feature maps to various scales so that the proposals can be aligned better with the objects of different scales. Unlike anchor-derived proposals, learnable proposals of varying sizes can be aligned with objects on a single-layer feature map. This motivates us to design the single-layer output neck (SLON) in Fig. 3(b). Formally,

$$\begin{aligned} F_i &= \text{Conv}(In_i) + F_{i+1} \uparrow, i = m - 1, m - 2, \\ Out &= \text{Conv}(F_{m-2}), \end{aligned} \quad (2)$$

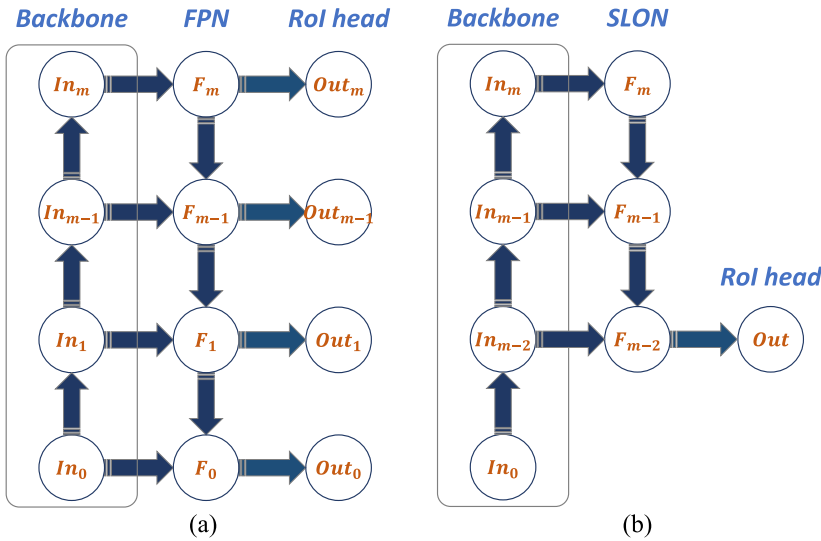


FIG. 3. The illustration of different neck structures. (a) FPN takes multiple feature maps as input and outputs multi-scale feature maps. (b) The single-layer output neck fuses feature maps across scales but outputs single-scale feature maps.

where m denotes the top layer of FPN and $m - 2$ denotes the third high layer of FPN. SLON only has a single output layer and needs one RoI head for generating proposals, reducing the computation of other output layers without decreasing performance.

C. Speeding up Sparse R-CNN

The major factor in speeding up Sparse R-CNN of the proposed methods is reducing the times of processing learnable proposals. Since every proposal will be serially processed by all iterative heads in a single layer, the times can be calculated by

$$T_p = N_{layer} \times N_{pro} \times N_{iter}, \tag{3}$$

where T_p denotes the total processing times, N_{layer} denotes the number of layers of the neck network, N_{pro} denotes the number of learnable proposals produced in each layer, and N_{iter} denotes the number of iterative heads in each layer. SLON reduces N_{layer} from 4 to 1 and IHA reduces N_{iter} from 6 to 4, decreasing T_p to $\frac{1}{4} \times \frac{4}{6} = \frac{1}{6}$ times. Hence, $\frac{5}{6}$ time of running the detection head is saved.

Also, the reduction of both N_{iter} and N_{layer} means that the memory storing the parameters of the detection heads and FPN will be accordingly reduced. This can slightly reduce the required memory for the whole Sparse R-CNN model and hence is the minor factor in speeding up Sparse R-CNN.

IV. EXPERIMENTS

IHA was employed to reduce the number of iterative heads of Sparse R-CNN, while SLON was employed to replace FPN in Sparse R-CNN. We used Compact Sparse R-CNN to denote Sparse R-CNN equipped with the proposed methods and abbreviated it as CS R-CNN. We used the challenging MS-COCO benchmark²² and the commonly used metrics for object detection to evaluate the built method. Indices including mAP and AP_{50} show the detection accuracy of the model and are the higher, the better; indices including Parameters, $FLOPs_{total}$, and $FLOPs_{det}$ show the detection speed of the model and are the lower, the better. All methods were trained

on the COCO train2017 split containing over 118 000 images, evaluated on the val2017 split containing 5000 images, and tested on the test2017 split containing 41 000 images.

The size of input images of CS R-CNN was 1333×800 . All models were trained with only common data augmentation like image resize and flip without special data augmentation like random crop and mosaic.

For the PASCAL VOC benchmark,²³ all methods were trained on the trainval07+12 split and tested on the test07 split.

A. Comparison with the state-of-the-art

This section first presents the comparison results between CS R-CNN and the detection methods when employing ResNet-50²⁸ as the backbone. The CS R-CNN model was trained by following the 36-epoch training way in Sparse R-CNN. K for the IHA was set to 3, and all inferences were conducted on the NVIDIA 1080Ti graphics processing units (GPUs).

In Table I, the CS R-CNN using 300 learnable proposals and four iterative heads achieved 44.7 AP and 64.8 AP_{50} , respectively. The detection performance was higher than that of Sparse R-CNN using 300 learnable proposals and six iterative heads, e.g., 0.8 AP and 2.5 AP_{50} rise. Also, CS R-CNN had less computation than Sparse R-CNN. The FLOPs of CS R-CNN was 117G, smaller than that

TABLE I. Detection accuracy, FLOPs, and FPS of different models employing ResNet-50 as the backbone of the COCO validation dataset. Sparse R-CNN and CS R-CNN both used 300 learnable proposals.

Method	N_{iter}	AP	AP_{50}	FLOPs	FPS
RetinaNet ³	1	38.7	58.0	250G	12.0
Faster R-CNN ¹⁴	1	40.2	61.0	216G	12.8
Cascade R-CNN ¹⁵	3	44.3	62.2	243G	9.3
Sparse R-CNN ¹⁷	6	43.9	62.3	172G	11.3
CS R-CNN	4	44.7	64.8	117G	15.6

of Sparse R-CNN (172G). As a result, the inference speed of CS R-CNN was significantly higher than Sparse R-CNN (15.6 frames per second (FPS) vs 11.3 FPS). Compared with Cascade R-CNN, CS R-CNN provided higher detection accuracy, less than 0.5 times FLOPs, and about 1.7 times inference speed.

We observed that CS R-CNN improved AP₅₀ more than AP. The reason may be that the iterative Hungarian assigner improves the ratio of the predicted boxes having IoU > 0.5 over the total predicted boxes, but the ratio of the predicted boxes having larger IoU than other IoU thresholds (e.g., 0.8) may not be equally improved.

Table II shows that CS R-CNN can be used together with other stronger backbone structures. Fed in the test images of a single scale on the COCO test dataset, CS R-CNN using Swin-S³² as the backbone and four iterative heads gave 50.0 AP and 70.7 AP₅₀, respectively. CS R-CNN gave detection accuracy slightly inferior to the state-of-the-art methods, such as EfficientDet¹⁰ employing EfficientNet-D5 and Cascade Mask R-CNN employing Swin-S. In terms of AP₅₀, CS R-CNN yielded a comparable performance to Cascade Mask R-CNN when both employed Swin-S as the backbone while running faster.

B. Module analysis

This section investigates the contribution of IHA and SLON to the built model. All comparing models are trained by following the 12-epoch training way of MMDetection.³⁶ K for the IHA was set to 3, and all inferences were conducted on the NVIDIA 1080Ti GPUs. Also, we continued using 300 learnable proposals and using the ResNet-50 as the backbone.

To investigate the contribution of IHA, we used FPN as the neck and replaced the Hungarian assigner with IHA in Sparse R-CNN. Table III gives the comparison result between the Hungarian assigner and IHA. When $N_{iter} = 6$, the proposed IHA improved the detection accuracy from 39.6% AP to 40.9% AP, and from 59.6% AP₅₀ to 60.8% AP₅₀; when $N_{iter} = 4$, the AP was significantly improved from 37.4 to 39.9 and the AP₅₀ was improved from 56.8 to 59.3. IHA improved the Hungarian by a larger margin for $N_{iter} = 4$ than $N_{iter} = 6$, which indicates that $N_{iter} = 4$ iterative heads may be unable to well regress the detection boxes and the proposed IHA can reduce the missing caused by the imprecise regression. This proves

TABLE II. Comparison of detection accuracy between CS R-CNN using four iterative heads and the state-of-the-art detection methods on the COCO test-dev dataset. All models were fed in single-scale images of the same size.

Method	Backbone	AP	AP ₅₀	FLOPs
CornerNet ⁴	Hourglass-104	40.6	56.4	1849G
RepPoint ⁶	ResNet-101-DCN	45.0	66.1	210G
FCOS ⁷	ResNet-101-DCN	46.6	65.9	221G
ATSS ⁸	ResNet-101-DCN	46.3	64.7	226G
EfficientDet ¹⁰	EfficientNet-D5	51.5	70.5	126G
YOLOF ¹¹	ResNeXt-101	44.7	64.1	354G
Cascade mask R-CNN	Swin-S ³²	51.8	70.4	356G
Sparse R-CNN ¹⁷	ResNeXt-101-DCN	48.9	68.3	258G
CS R-CNN	Swin-S ³²	50.0	70.7	212G

TABLE III. The effectiveness of applying IHA to Sparse R-CNN-R50 compared with Hungarian assigner.

Assigner	N_{iter}	AP	AP ₅₀	AP ₇₅	FLOPs	FPS
Hungarian	6	39.6	59.6	42.0	172G	11.3
IHA	6	40.9	60.8	43.6	172G	11.3
Hungarian	4	37.4	56.8	39.4	164G	13.3
IHA	4	39.9	59.3	42.6	164G	13.3

that the multiple positive proposals assigned by IHA can effectively reduce the missing rate at the inference stage. It can also be observed that the increase of N_{iter} had a greater impact on FPS than FLOPs. Specifically, when N_{iter} increased from 4 to 6, the corresponding FLOPs only increased from 164G to 172G, less than 5%, but the FPS decreased from 13.3 to 11.3, more than 15%.

To investigate the contribution of SLON, we used the Hungarian assigner and replaced FPN with SLON. Table IV gives the detection results when FPN or SLON was used. When $N_{iter} = 6$, SLON yielded higher detection accuracy than FPN and improved the inference speed from 11.3 FPS to 13.1 FPS. When $N_{iter} = 4$, SLON also yielded higher detection accuracy than FPN and improved the inference speed from 13.3 FPS to 15.6 FPS. In both cases, SLON reduced over 40G FLOPs compared with FPN.

An ablation study was conducted on Sparse R-CNN using 300 learnable proposals, four iterative heads, and ResNet-50 as the backbone. The result showed that IHA and SLON worked well when they were employed independently and gave higher detection accuracy and speed when they were employed together, as shown in Table V.

C. Module settings

This section investigates the influences of parameter settings on iterative heads, IHA, and SLON. Baseline, training, and testing settings are the same as in Sec. IV B.

TABLE IV. The effectiveness of applying SLON to Sparse R-CNN-R50 compared with FPN.

Neck	N_{iter}	AP	AP ₅₀	AP ₇₅	FLOPs	FPS
FPN	6	39.6	59.6	42.0	172G	11.3
SLON	6	39.7	59.8	42.0	125G	13.1
FPN	4	37.4	56.8	39.4	164G	13.3
SLON	4	37.5	57.1	39.4	117G	15.6

TABLE V. The effectiveness of applying both IHA and SLON to Sparse R-CNN with four iterative heads.

IHA	SLON	AP	AP ₅₀	FLOPs	FPS
		37.4	56.8	164G	13.3
✓		39.9	59.3	164G	13.3
	✓	37.5	57.1	117G	15.6
✓	✓	39.9	59.2	117G	15.6

The influence of changing the number of iterative heads is first studied and the result is given in Table VI. The iteration must be greater than 0 because equaling 1 means that the structure just has an output structure without any iteration. The result shows that no iteration led to the terrible performance of 7.2 AP and 15.8 AP₅₀, which was just a rough guess of the object location. However, only adding one iteration greatly increased AP to 30.4 and AP₅₀ to 50.3. The trend of significant growth continued until iteration increased to 4 as shown in Fig. 4. The increase in the iteration no undoubtedly decreased the FPS. Thus, employing four iterations is a reasonable choice.

The influence of changing the K in the iterative Hungarian assigner is second studied. Likely, $K = 1$ means that the iterative Hungarian assigner degrades to the vanilla Hungarian assigner. The result is given in Table VII. The result shows that the detection performance increases when K increased from 1 to 4, and the increase of K reduced FPS too small to be observed. When $K > 4$, proposals that have large losses with the object will be taken into consideration, hindering the convergence, as shown in Fig. 5.

The influence of changing the depth of SLON, $m - \alpha$, is third studied. The result in Table VIII shows that the speed monotonically decreased from 17.0 FPS to 13.2 FPS when α increased from 0 to 3. This was not only caused by the increase in convolutional layers but also caused by the increase in the feature map size. The proposal that

TABLE VI. Effect of the number of iterative heads on the COCO validation dataset. $N_{iter} = 4$ is a good trade-off between accuracy and running speed.

N_{iter}	AP	AP ₅₀	AP ₇₅	FLOPs	FPS
1	7.2	15.8	5.7	106G	20.1
2	30.4	50.3	31.3	110G	18.3
3	37.4	57.1	39.4	113G	16.8
4	39.7	59.3	41.9	117G	15.6
5	40.6	60.7	43.3	121G	14.4
6	41.0	61.1	43.5	125G	13.1

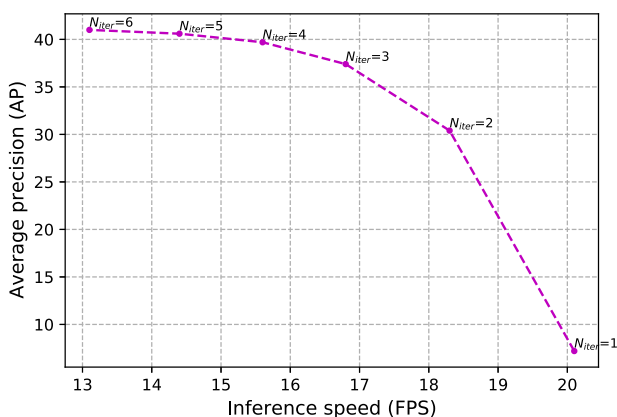


FIG. 4. The relationship between the inference speed (FPS) and the average precision (AP) for different N_{iter} . AP increases with N_{iter} while FPS decreases with N_{iter} .

TABLE VII. Effect of the K in the iterative Hungarian assigner on the COCO validation dataset.

K	AP	AP ₅₀	AP ₇₅	FLOPs	FPS
1	37.3	57.0	59.0	117G	15.6
2	39.2	59.0	41.6	117G	15.6
3	39.7	59.3	41.9	117G	15.6
4	39.9	59.9	42.4	117G	15.6
5	39.5	59.2	42.1	117G	15.6

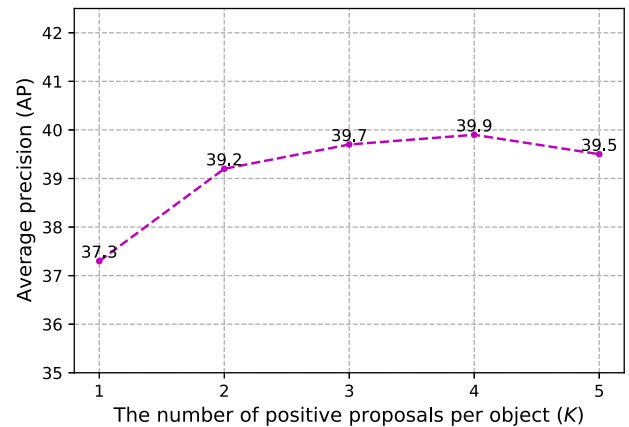


FIG. 5. The detection performance increases when K increases from 1 to 4 but decreases when $K > 4$. A too large K may generate hard proposals for training as positive samples, hindering the convergence of the network.

mapped the same region in the original image became larger with the increase of the feature map size, and thus, the RoI head needed to process the larger proposal and consumed more time. $\alpha = 2$ gave high detection accuracy with considerable speed.

D. Results on PASCAL VOC 2007

To examine the effectiveness of CS R-CNN on different datasets, we trained and tested it on the PASCAL VOC 2007 dataset.²³ All models were trained four epochs on the trainval07+12 split where all images were used for training three times in a single epoch. As shown in Table IX, CS R-CNN brought a 3.0 improvement in AP₅₀ over Sparse R-CNN with six iterative heads and a 4.3 improvement in AP₅₀ with four iterative heads, showing that CS R-CNN gave better results with a simpler structure.

TABLE VIII. Effect of the depth of SLON on the COCO validation dataset.

α	AP	AP ₅₀	AP ₇₅	FLOPs	FPS
0	36.7	57.5	38.1	105G	17.0
1	38.6	58.7	40.7	108G	16.4
2	39.9	59.9	42.4	117G	15.6
3	40.0	59.9	42.3	151G	13.2

TABLE IX. Results on the test split of the PASCAL VOC 2007 dataset.

Method	Assigner	Neck	N_{iter}	AP ₅₀	FLOPs
Sparse R-CNN	Hungarian	FPN	6	78.9	172G
Sparse R-CNN	Hungarian	FPN	4	77.6	164G
Sparse R-CNN	Hungarian	DE ^{11,18}	6	77.3	114G
CS R-CNN	Hungarian	SLON	6	79.6	125G
CS R-CNN	Hungarian	SLON	4	78.5	117G
CS R-CNN	IHA	FPN	4	81.7	164G
CS R-CNN	IHA	SLON	4	81.9	117G

The dilated encoder^{11,18} was applied to Sparse R-CNN with six iterative heads, and it reduces the FLOPs as well by only using F_m in Eq. (2) as the output layer. The reduction of FLOPs by the proposed SLON with four iterative heads (55G) is comparable to that by the dilated encoder (58G), but the SLON yielded a better detection performance (78.5) than the dilated encoder (77.3).

V. CONCLUSION

This work reduced the number of iterative detection heads and designed the iterative Hungarian assigner (IHA) that generated multiple proposals for each object. One or more of the multiple proposals generated by fewer detection heads may have a comparable regression accuracy to the single proposal generated by multiple detection heads. Also, this work designed the single-layer output neck (SLON) in replacement of FPN. The learnable proposals in a single layer cover a sufficiently large size variation and hence SLON retains a comparable detection accuracy to FPN. IHA combined with SLON significantly reduced the FLOPs and increased the inference speed, while slightly improving the detection accuracy.

Nevertheless, Sparse R-CNN embedded with the iterative Hungarian assigner still needed four iterative detection heads to give similar results. The inference speed can be further increased from 15.6 FPS to 20.1 FPS if only a single detection head is needed. In the future, we will explore why multiple iterative detection heads are needed for the learnable proposals and try to design the structures, giving similar results for the learnable proposals when fewer iterative detection heads are employed.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No. 62071060) and the Beijing Key Laboratory of Work Safety and Intelligent Monitoring Foundation.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Zihang He: Conceptualization (equal); Methodology (equal); Writing – original draft (equal); Writing – review & editing (equal).

Xiang Ye: Methodology (supporting); Validation (equal); Writing – original draft (supporting); Writing – review & editing (supporting).
Yong Li: Conceptualization (equal); Methodology (equal); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

- C. Li, G. Cui, W. Zhang, S. Chen, and Y. Yang, "Defect detection in vehicle mirror nonplanar surfaces with multi-scale atrous single-shot detect mechanism," *AIP Adv.* **11**, 075202 (2021).
- Z. Wang, Z. Shi, J. Tong, W. Gong, and Z. Wu, "A detection method for impact point water columns based on improved YOLO X," *AIP Adv.* **12**, 065011 (2022).
- T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (IEEE, 2017)*, pp. 2980–2988.
- H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proceedings of the European Conference on Computer Vision (Springer, 2018)*, pp. 734–750.
- K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proceedings of the IEEE International Conference on Computer Vision (IEEE, 2019)*, pp. 6569–6578.
- Z. Yang, S. Liu, H. Hu, and L. Wang, "RepPoints: Point set representation for object detection," in *2019 IEEE International Conference on Computer Vision (ICCV 2019) (IEEE, 2019)*, pp. 9656–9665.
- Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," *2019 IEEE/CVF International Conference on Computer Vision (ICCV 2019) (IEEE, 2019)*, pp. 9626–9635.
- C. Wang, A. Bochkovskiy, and H. Liao, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *33rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2020) (IEEE, 2020)*, pp. 9756–9765.
- S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Li, "Scaled-YOLOv4: scaling cross stage partial network," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2021) (IEEE, 2021)*, pp. 13024–13033.
- M. Tan, R. Pang, and Q. Le, "EfficientDet: Scalable and efficient object detection," in *33rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2020) (IEEE, 2020)*, pp. 10781–10790.
- Q. Chen, Y. Wang, T. Yang, X. Zhang, J. Cheng, and J. Sun, "You only look one-level feature," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2021) (IEEE 2021)*, pp. 13034–13043.
- R. Girshick, "Fast R-CNN," *2015 IEEE International Conference on Computer Vision (ICCV) 1440–1448 (2015)*.
- K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.* **37**, 1904–1916 (2015).
- S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (IEEE Computer Soc, 2015)*, pp. 91–99.
- Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE, 2018)*, pp. 6154–6162.
- K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (IEEE, 2017)*, pp. 2961–2969.
- P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, and C. Wang, "Sparse R-CNN: End-to-end object detection with learnable proposals," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE, 2021)*, pp. 14454–14463.

- ¹⁸Z. Zheng, Z. Zhang, M. Fan, and L. Huang, "Improving the detection performance of sparse R-CNN with different necks," in *2022 6th International Conference on Advances in Image Processing* (Association for Computing Machinery, 2022), pp. 7–12.
- ¹⁹Q. Hong, F. Liu, D. Li, J. Liu, L. Tian, and Y. Shan, "Dynamic sparse R-CNN," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (IEEE, 2022), pp. 4723–4732.
- ²⁰N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *16th European Conference on Computer Vision (ECCV 2020)* (Springer Science, 2020), pp. 213–229.
- ²¹T. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)* (IEEE, 2017), pp. 936–944.
- ²²T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision* (Springer Science, 2014), pp. 740–755.
- ²³M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vision* **88**, 303–338 (2010).
- ²⁴J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vision* **104**, 154–171 (2013).
- ²⁵R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE, 2014), pp. 580–587.
- ²⁶A. Rosenfeld and M. Thurston, "Edge and curve detection for visual scene analysis," *IEEE Trans. Comput.* **C-20**, 562–569 (1971).
- ²⁷N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," in *Proceedings of the IEEE International Conference on Computer Vision* (IEEE, 2017), pp. 5561–5569.
- ²⁸K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2016), pp. 770–778.
- ²⁹A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European Conference on Computer Vision* (Springer-Verlag, Berlin, 2016), pp. 483–499.
- ³⁰S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)* (IEEE, 2017), pp. 5987–5995.
- ³¹C. Wang, H. Liao, Y. Wu, P. Chen, and I. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2021.
- ³²Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE International Conference on Computer Vision (ICCV 2021)* (IEEE, 2019), pp. 9656–9665.
- ³³T. Kong, F. Sun, C. Tan, H. Liu, and W. Huang, "Deep feature pyramid reconfiguration for object detection," in *European Conference on Computer Vision* (Springer-Verlag, Berlin, 2018), pp. 169–185.
- ³⁴S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *31st IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018)* (IEEE, 2018), pp. 8759–8768.
- ³⁵G. Ghiasi, T. Lin, and Q. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *32nd IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019)* (IEEE, 2019), pp. 7036–7045.
- ³⁶K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open MMLab detection toolbox and benchmark," *arXiv:1906.07155* (2019).