ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10096922

HYNETER: HYBRID NETWORK TRANSFORMER FOR OBJECT DETECTION

Dong Chen *Duoqian Miao** *Xuerong Zhao*

Department of Computer Science and Technology, Tongji University

ABSTRACT

In this paper, we point out that the essential differences between CNN-based and Transformer-based detectors, which cause worse performance of small object in Transformer-based methods, are the gap between local information and global dependencies in feature extraction and propagation. To address these differences, we propose a new vision Transformer, called **Hy**brid **Net**work Transform**er** (Hyneter). Different from the divide and conquer strategy in previous methods, Hyneters consist of Hybrid Network Backbone (HNB) and Dual Switching module (DS), which integrate local information and global dependencies, and transfer them simultaneously. Based on the balance strategy, HNB extends the range of local information by embedding convolution layers into Transformer blocks, and DS adjusts excessive reliance on global dependencies outside the patch. Ablation studies illustrate that Hyneters surpass the state-of-the-art results on multiple vision tasks.

Index Terms— Object Detection, Transformer, Hybrid Network, CNN

1. INTRODUCTION

Convolutional neural networks (CNNs) have dominated computer vision modeling for years. With the help of increasingly large neural networks and progressively complex convolution structures, the performance has seen significant improvement in recent time. However, scholars have focused on greater model size, more diverse convolution kernel, and more sophisticated structures of network, which lead to a less progress of general performance with disproportionate huge model size.

On the other hand, Transformer has made tremendous progress in vision tasks, which originates from natural language processing. Designed for sequence modeling and transduction tasks, the Transformer is notable for its use of attention to model global dependencies. Compared to CNN-based methods, vision Transformer and its follow-ups [1] expose the difference in size-sensitive performance, for they adopt different strategies for local information and global dependencies [2].

*Corresponding author

The essential differences between Transformer-based and CNN-based detectors are derived from the gap between local information and global dependencies in feature extraction and propagation. However, we have not found enough studies on these differences. In this paper, we devote to find the answer and propose a new vision Transformer.

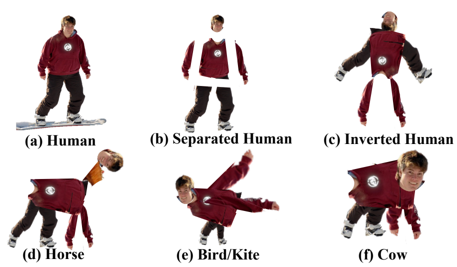

Fig. 1. An illustration of restructured objects. We restructure thousands of objects in multiple-class images of COCO. $(d) \sim (f)$ are supposed to detected as *unrecognized labels*, but as *Pseudo labels* (horse, bird/kite, and cow) by Transformer-based detectors.The Transformer-based should detect (b) and (c) as *unrecognized labels*, but *True label* (human).

The exploration begins with an unexpected experiment shown in Figure 1. We restructure thousands of objects with diverse backgrounds. A human, for example, is restructured as horse, bird/kite, cow, etc. in Figure 1. However, CNN-based detectors show much better performance. This rate of being detected as *pseudo labels* (*Pseudo Rate*) demonstrates that Transformer-based methods are reliant on global dependencies and obtain inadequate local information of feature in details [2, 3]. However, the CNN-based ones are just the opposite.

The CNN-based methods extract feature with rich local information by convolution layers [4, 5, 6]. While Transformer-based methods extract feature by providing the capability to decode and encode global dependencies in Transformer blocks [7, 8] (see Figure 2). Compared to CNN-based methods, Transformer-based methods have worse performance in small objects.

In this paper, we propose a new vision Transformer, called **Hy**brid **Net**work Transform**er** (Hyneter), which consists of Hybrid Network Backbone (HNB) and Dual Switching mod-

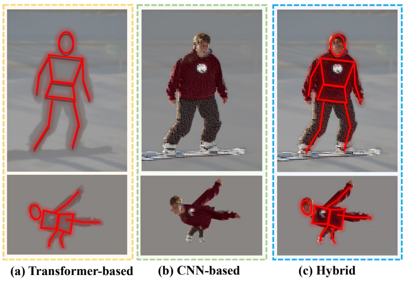

Fig. 2. An illustration of feature maps on Transformer-based, CNN-based and Hybrid methods. Hybrid feature map (c) integrates the characteristics of global dependencies (a) and local information (b), which is beneficial to objects of all sizes.

ule (DS). Hybrid network backbone is presented with equivalent position of intertwined distribution of convolution and self-attention. Our backbone extends the range of local information by embedding convolution layers into Transformer blocks in stages, so that local information and global dependencies will be passed to *Neck* or *Head* simultaneously. The Dual Switching module establishes cross-window connections in order to maintain local information inside the patch, while weakening excessive reliance on global dependencies outside the patch.

Ablation studies illustrate that Hyneters with HNB and DS achieve the state-of-the-art performance by a large margin of $+2.1 \sim 13.2AP$ on COCO in object detection. Furthermore, Hyneters surpass previous best performance on multiple tasks, such as object detection($60.1AP$ on COCO), semantic segmentation ($54.3AP$ on ADE20K), and instance segmentation ($48.5AP^{mask}$ on COCO) in Tables 1 $\sim$ 5.

2. HYBRID NETWORK TRANSFORMER

In this section, we propose a new vision Transformer, called *Hybrid Network Transformer*, that capably serves as a backbone for multiple computer vision tasks. An overview of Hyneter is presented in Figure 3 (a). Data is preprocessed as Method in [9].

2.1. Hybrid Network Backbone

Many hybrid backbones [10] are presented in previous works, which put convolution and self-attention in the non equivalent position. Previous methods employ self-attention within the CNN backbone architecture or use them outside, which completely cleavage the relation of local information and global dependencies by separated distribution of convolution and self-attention. Hybrid network backbone is presented with equivalent position of intertwined distribution of convolution and self-attention, which extends the range of local information, so that local information and global dependencies will be passed to *Neck* or *Head* simultaneously.

There are 4 stages in our backbone, starting with a convolution layer of 3 multi-granularity kernels. The number of tokens is reduced by this multi-granularity convolution layer, and dimension is multiplied. The data feature $S\,(C' \times \frac{H}{4} \times \frac{W}{4})$ will be sent into convolution layers and Transformer blocks.

As shown in Figure 3 (b), the Transformer blocks extract feature maps of global dependencies and CNN layers extract feature maps of local information in the Stage 1 and 2. The output $(C \times \frac{H \times W}{4 \times 4})$ of the final Transformer block in Stage 1 will be re-viewed and permuted as $X(C \times \frac{H}{4} \times \frac{W}{4})$. After the convolution layers, the S turns into S_1 with the same size $(C \times \frac{H}{4} \times \frac{W}{4})$. The dot product between S_1 and X is the key operation of combination for global dependencies and local information. The X_1 ($X_1 = S_1 \cdot X$) after dot product operation, will go to activation function $X_2 = tanh(X_1)$. The addition of X_2 and X copy will be the output of Stage 1. After being re-viewed and permuted twice, the addition turns to the input (X') of Stage 2.

With hybrid network approach, consecutive self-attention Transformer blocks are computed as

$$\begin{aligned} X &= \text{Re-view}(\text{ GMSA } (S)) \\ S_1 &= \text{Conv}_1(S) \oplus \text{Conv}_2(S) \oplus \text{Conv}_3(S) \\ X' &= \text{Re-view}(X \oplus \tanh{(X \cdot S_1)}) \end{aligned} \tag{1}$$

2.2. Dual Switching

The Dual Switching module will be implemented in Stage 3 and 4, in order to maintain local information while weakening excessive reliance on global dependencies. Global dependencies from global self-attention are conducted in Transformer blocks. For efficiency, the global multi-head self-attention (GMSA) will be implemented within local windows in a non-overlapping manner.

As illustrated in Figure 3 (c), the ouput of Transformer block will be re-viewed and permuted as $X(C \times \frac{H}{4} \times \frac{W}{4})$. Then, adjacent columns in the feature map will switch with each other. After the column switching, adjacent rows in the feature map will switch with each other, too. The solo-switching is finished. Finally, the interlaced columns/rows in solo-switched feature map will switch with each other, again.

The Dual Switching module establishes cross-window connections while maintaining local information in the patch, which is followed by layerNorms (LN), Transformer blocks, and multi-layer perceptions (MLP) with residual connection modules. Dual Switching suspends the procedure of establishing excessive global dependencies, meanwhile, retaining local information for small object performance (AP_S). With Dual Switching module, the process is computed as

$$\begin{aligned} X_{l+1} &= \text{ GMSA } (\text{LN}(X_l)) + X_l \\ X'_{l+1} &= \text{MLP}(\text{LN}(X_{l+1})) + X_{l+1} \end{aligned} \tag{2}$$

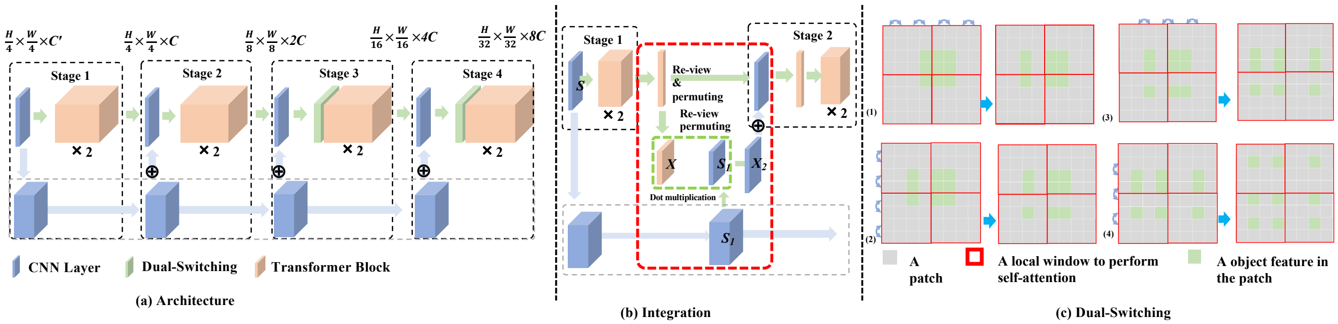

Fig. 3. (a) The architecture of Hyneter 1.0. In one stage, there are 2 Transformer blocks in Transformer part (top) and 2-layer multi-granularity convolution layers in CNN layers part (bottom). Positional encoding is only in the first Transformer block. (b) An illustration of unidirectional feature integration between Transformer block (top) and CNN layer (bottom). (c) An illustration of Dual Switching. The process is implementing as (1)→(4).

where X_l and X'_{l+1} denote the the feature in Stage l and the input of Stage $l+1$.

Architecture Variants. We establish basic model, called Hyneter 1.0, to have of size and computation complexity similar to DETR-DC5-R101. This paper also presents Hyneter Plus and Max, which are 2 versions of around $2.0\times$ and $4.0\times$ the model size and computation complexity, respectively.

3. EXPERIMENTS

In this section, we first ablate the important design elements of Hyneter. Then, we conduct experiments on multiple datasets in several vision tasks.

3.1. Ablation studies

Settings. The following experiments were conducted on MS COCO 2017 dataset using two GeForce RTX 3090 GPUs and 2 Tesla V100 PCIe 32GB GPUs. For the ablation study and comparisons, we consider four typical object detection frameworks: Swin Transformer (V1, V2)[9, 11], and DETRs (DETR[12], UP-DETR[13], Conditional DETR[14]).

Dataset. We perform experiments on COCO 2017 detection datasets, containing 118k training images, 5k validation images and 20K test-dev images. The ablation study is performed using the validation set, and a system-level comparison is reported on test-dev. Each image is annotated with bounding boxes and panoptic segmentation.

We conduct ablation studies on COCO 2017 object detection in Table 1. Hyneter brings consistent $+3.2 \sim 4.8$ AP and $+4.1 \sim 6.8$ AP_S gains over pure Transformer detectors. Furthermore, HNB brings $+1.6 \sim 2.7$ AP and $+1.7 \sim 3.8$ AP_S gains over original detectors, just with slightly larger model size. Meanwhile, DS gets $+1.6 \sim 2.1$ AP and $+1.2 \sim 3.0$ AP_S gains over original ones, with the same model size.

Method	Originals	HNB	DS	AP	AP_s	AP/AP_s	#param.
Hyneter 1.0							
baseline	✓			52.3	21.5	2.43	85M
	✓	✓		55.0	25.3	2.17	90M
	✓	✓	✓	**57.1**	**28.3**	**2.02**	90M
Hyneter Plus							
baseline	✓			54.8	23.0	2.38	125M
	✓	✓		56.4	26.7	2.11	134M
	✓	✓	✓	**58.0**	**27.9**	**2.08**	134M
Hyneter Max							
baseline	✓			55.7	25.7	2.17	227M
	✓	✓		58.3	27.4	2.10	247M
	✓	✓	✓	**60.1**	**29.8**	**2.07**	247M

Table 1. Object detection performance (%) on Hyneter vatiants with Mask R-CNN frameworks on MS COCO `test-dev` set. *Originals* means pure Transformer baselines without HNB or DS, which is similar to Swin-T structurally.

3.2. Object Detection and Instance Segmentatio on MS COCO

Setting. For the ablation study, we consider 4 typical object detection frameworks: Mask R-CNN, ATSS, DETR, and Swin Transformer with the same setting (multi-scale training, ADamW optimizer with initial learning rate of 0.00001 and weight decay of 0.05) in mmdetection [15]. We adopt ImageNet-22K pre-trained model as initialization for system-level comparison.

Dataset is mentioned in **Ablation studies**.

Comparison to ResNet. Our Hyneter architecture brings consistent $+5.0 \sim 15.7$ AP and $+1.7 \sim 4.2$ AP_S gains over ResNet-50, with acceptable larger model size. All Hyneters achieve significant gains of $+14.8 \sim 15.6AP$ and $+3.6 \sim 4.3AP_S$ over ResNet-50 or ResNet-101 (see Table 3).

Comparison to Swin Transformer. The comparison of Hyneter and Swin Transformer under different backbones with Mask R-CNN is showed in Table 3. Hyneters achieve a high detection accuracy of $60.1AP$ and $29.8AP_S$, which are

Method	Backbone	AP	AP_s	AP/AP_s	#param.
Mask R-CNN	R-50	42.3	24.7	1.71	82M
	Hyneter-plus	58.0	27.9	2.07	134M
ATSS	R-50	43.5	25.7	1.69	32M
	Hyneter-plus	56.0	27.4	2.04	53M
DETR	R-50 + trans	42.0	20.5	2.05	41M
	Hyneter-plus	47.0	24.7	1.90	93M

Table 2. Object detection performance (%) with various frameworks on MS COCO val set. *R50 + trans* means that R50 and Transformer Blocks as DETR Backbone.

Backbone	AP	AP_s	AP/AP_s	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#params.
R-50	42.3	24.7	1.71	32.5	55.4	31.7	82M
R-101	44.5	25.5	1.74	35.9	60.7	36.8	101M
Swin-T	49.8	21.4	2.33	41.5	70.1	42.0	86M
Swin-S	51.4	25.1	2.05	41.5	70.1	42.0	107M
Swin-B	51.5	25.0	2.06	42.0	74.0	42.6	145M
Swin-L	57.8	26.7	2.16	–	–	–	284M
Hyneter-1.0	57.1	28.3	**2.02**	45.1	78.3	42.2	90M
Hyneter-plus	58.0	27.4	2.08	46.9	79.9	45.0	134M
Hyneter-Max	**60.1**	**29.8**	2.07	**48.5**	**82.1**	**46.7**	247M

Table 3. Object detection (with Mask R-CNN) performance (%) with various backbones on COCO val set.

significant improvement of $+2.3 \sim 7.3$ AP and $+3.1 \sim 6.9$ AP_S over Swin series methods with lighter model size.

Comparison to previous state-of-the-arts. Table 4 lists the comparison of our best results with precious state-of-the-art methods. Hyneter method achieves $+60.1AP$ and $29.8AP_S$on COCO *test-dev* set, surpassing the previous best performances by $+9.4AP$ (ATSS [17]), $+5.0AP$ (EfficientDet-D7x [16]), $+13.2AP$ (Deformable DETR [18]), and $+2.1AP$ (Swin-L [9] with HTC++ and multi-scale testing). Furthermore, Hyneters greatly improve AP_S, comparing with Swin Transformer series. Our best model (Hyneter Max) achieves $48.5AP^{mask}$, $82.1AP^{mask}_{50}$, and $46.7AP^{mask}_{75}$ with competitive model size, surpassing all previous best results.

3.3. Semantic Segmentation on ADE20K

Setting. In training, we employ the AdamW optimizer with an initial learning rate of 1.0×10^{-5}, a weight decay of 0.01, a scheduler that uses linear learning rate decay, and a linear warmup of 1,500 iterations. Models are trained on 2 GPUs with 4 images per GPU for 140K iterations

Dataset. ADE20K has more than 25K images of complex daily scenes, including various objects in natural space environment (20.2k for training, 2K for validation, 3K for test).

Table 5 lists the mIoU, and model size (#param) for different method/backbone pairs. From these results, it can be seen that Hyneter Max is $+4.3mIoU$ higher than SETR with much lighter model size. It is also $+6.0mIoU$ higher than ResNeS200, and $+9.4mIoU$ higher than ResNeSt-101. Our Hyneter series with UperNet achieve $50.6mIoU, 53.0mIoU$, and $54.3mIoU$ on the val set, surpassing the previous Swin Transformer series by $+0.8 \sim 1.4mIoU$.

Method	AP	APs	AP/APs	#param.
ATSS(ResNeXt-101-DCN)	50.7	33.2	1.53	–
EfficientDet-D7x(1537)	55.1	–	–	77M
DETR series Backbone: DC5-R50 or R50				
DETR	43.3	22.5	1.92	41M
UP-DETR	42.8	20.8	2.06	–
Deformable DETR	46.9	27.7	1.69	–
Conditional DETR	45.1	25.3	1.78	44M
Swin Transformer with Cascade Mask R-CNN				
Swin-B (HTC++)	56.4	25.1	2.25	160M
Swin-L (HTC++)	57.1	25.6	2.23	284M
Swin-L (HTC++)*	58.0	26.0	2.23	284M
Ours with Mask R-CNN				
Hyneter-1.0	57.1	28.3	**2.02**	90M
Hyneter-plus	58.0	27.9	2.08	134M
Hyneter-Max	**60.1**	**29.8**	2.07	247M

Table 4. System-level comparison (%) on MS COCO `test-dev` set. * indicates multi-scale testing. The frameworks in Swin Trans (Swin-Transformer [9]) is Cascade Mask R-CNN. EfficientDet-D7x(1537)[16]

Method	Backbone	val mIoU	test score	#param.
DANet	ResNet-101	45.2	–	69M
Dlab.v3+	ResNet-101	44.1	–	63M
OCRNet	ResNet-101	45.3	56.0	56M
UperNet	ResNet-101	44.9	–	86M
OCRNet	HRNet-w48	45.7	–	71M
Dlab.v3+	ResNeSt-101	46.9	55.1	66M
Dlab.v3+	ResNeSt-200	48.4	–	88M
SETR	T-Large	50.3	61.7	308M
UperNet	Swin-S	49.3	–	81M
UperNet	Swin-B	51.6	–	121M
UperNet	Swin-L	53.5	62.8	234M
UperNet	Hyneter 1.0	50.6	62.0	82M
UperNet	Hyneter Plus	53.0	63.4	125M
UperNet	Hyneter Max	**54.3**	**65.9**	231M

Table 5. Results of semantic segmentation on the ADE20K val and test set. The comparison data is from Appendix A2.3 in [9].

4. CONCLUSION

In this work, we propose a new vision Transformer, called Hyneter, to address the differences between CNN-based and Transformer-based detectors by integrating and transfering local information and global dependencies simultaneously in feature extraction and propagation. Hyneters achieve the state-of-the-art performance on multiple tasks significantly, and surpass previous best methods. Although Hyneters have made significant achievements, the method cannot reduce the model size. We will continue to research around this goal. We do hope that Hynerters will play a role of cornerstone to encourage balancing methods between local information and global dependencies in computer vision.

5. REFERENCES

[1] Xinyue Xu and Xiaolu Zheng, “Hybrid model for network anomaly detection with gradient boosting decision trees and tabtransformer,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 8538–8542.

[2] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas, “Attention is not all you need: Pure attention loses rank doubly exponentially with depth,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 2793–2803.

[3] Kristian Fischer, Felix Fleckenstein, Christian Herglotz, and André Kaup, “Saliency-driven versatile video coding for neural object detection,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 1505–1509.

[4] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun, “You only look one-level feature,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13039–13048.

[5] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy, “Do vision transformers see like convolutional neural networks?,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 12116–12128, 2021.

[6] Hyomin Choi and Ivan V Bajic, “High efficiency compression for object detection,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1792–1796.

[7] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He, “Exploring plain vision transformer backbones for object detection,” *arXiv preprint arXiv:2203.16527*, 2022.

[8] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang, “Dynamic head: Unifying object detection heads with attentions,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 7373–7382.

[9] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10012–10022.

[10] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani, “Bottleneck transformers for visual recognition,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16519–16529.

[11] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al., “Swin transformer v2: Scaling up capacity and resolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12009–12019.

[12] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko, “End-to-end object detection with transformers,” in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[13] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen, “Up-detr: Unsupervised pre-training for object detection with transformers,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1601–1610.

[14] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang, “Conditional detr for fast training convergence,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3651–3660.

[15] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al., “Mmdetection: Open mmlab detection toolbox and benchmark,” *arXiv preprint arXiv:1906.07155*, 2019.

[16] Mingxing Tan, Ruoming Pang, and Quoc V. Le, “Efficientdet: Scalable and efficient object detection,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li, “Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection,” in *CVPR*, 2020.

[18] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai, “Deformable detr: Deformable transformers for end-to-end object detection,” *arXiv preprint arXiv:2010.04159*, 2020.