Hyneter:Hybrid Network Transformer for Multiple Computer Vision Tasks

Dong Chen, Duoqian Miao, and Xuerong Zhao

***Abstract*—In this article, we point out that the essential differences between convolutional neural network (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 Hybrid Network Transformer (Hyneter), after preexperiments that indicate the gap causes CNN-based and transformer-based methods to increase size-different objects results unevenly. Different from the divide-and-conquer strategy in previous methods, Hyneters consist of hybrid network backbone (HNB) and dual switching (DS) module, 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 in parallel, and DS adjusts excessive reliance on global dependencies outside the patch. Ablation studies illustrate that Hyneters achieve the state-of-the-art performance by a large margin of $+2.1 \sim 13.2$AP on COCO, and $+3.1 \sim 6.5$mIoU on VisDrone with lighter model size and lower computational cost in object detection. Furthermore, Hyneters achieve the state-of-the-art results on multiple computer vision tasks, such as object detection (60.1AP on COCO and 46.1AP on VisDrone), semantic segmentation (54.3AP on ADE20K), and instance segmentation (48.5AP^{mask} on COCO), and surpass previous best methods. The code will be publicly available later.**

***Index Terms*—Convolutional neural network (CNN), hybrid network, object detection, transformer.**

I. Introduction

CONVOLUTIONAL neural networks (CNN) 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 kernels, and more sophisticated structures of networks, which lead to a less progress of general performance with disproportionate huge model sizes.

On the other hand, transformer has made tremendous progress in vision tasks, which originates from natural language processing (NLP). Designed for sequence modeling and transduction tasks, the transformer is notable for its use of attention to model global dependencies in the feature. The tremendous success of NLP has led researchers to investigate its adaptation to computer vision, where it has recently demonstrated promising results on certain tasks. Compared with CNN-based methods, vision transformer and its follow-ups (including hybrid methods) expose the difference in size-sensitive performance, for they adopt different strategies for local information and global dependencies [1].

The essential differences between CNN-based and transformer-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 article, we devote to finding the answer and proposing a new vision transformer.

The object detection exploration begins with an unexpected experiment, as shown in Fig. 1. We manually restructure thousands of objects in multiple-class images with diverse backgrounds, such as grassland, sky, and indoor environment. A human, for example, is restructured as horse, bird/kite, and cow.[1] Fig. 1(d)–(f) is supposed to be detected as *unrecognized label*, but as *pseudolabel* (horse, bird/kite, and cow) by transformer-based detectors. However, CNN-based detectors show much better performance. This rate of being detected as pseudolabels (*pseudorate*, refer to Appendix I in the Supplementary Material) demonstrates that transformer-based methods are reliant on global dependencies and obtain inadequate local information of feature in details [1]. However, the CNN-based methods are just the opposite (see Fig. 2).

Manuscript received 28 June 2023; revised 12 November 2023; accepted 7 February 2024. Date of publication 25 March 2024; date of current version 5 June 2024. This work was supported in part by the National Key Research and Development Program Key Special Project for Cyberspace Security Governance under Grant 2022YFB3104700 and in part by the National Natural Science Foundation of China under Grant 61976158, Grant 62376198, and Grant 62006172. Paper no. TII-23-2345. *(Corresponding author: Duoqian Miao.)*

Dong Chen is with the Key Laboratory of Embedded System and Service Computing Ministry of Education, Tongji University, Shanghai 200092, China (e-mail: alan_chen@tongji.edu.cn).

Duoqian Miao is with Tongji University, Shanghai 200092, China (e-mail: dqmiao@tongji.edu.cn).

Xuerong Zhao is with the Computer Science and Technology School, Shanghai Normal University, Shanghai 201418, China (e-mail: xrzhao@shnu.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TII.2024.3367043.

Digital Object Identifier 10.1109/TII.2024.3367043

[1]Following the spirit of classic datasets (initial ground truths of COCO, PASCAL VOC, and VisDrone were also manually annotated, and later automatically algorithm driven), we will restructure objects later by algorithm driven. The manual process will not influence the result for CNN/transformer.

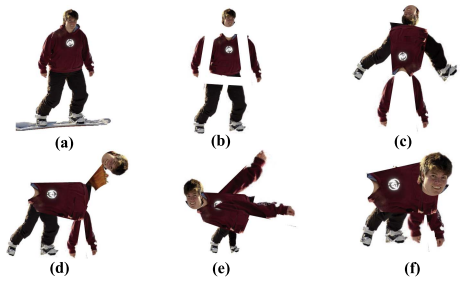

Fig. 1. Illustration of restructured objects. We restructure thousands of objects in multiple-class images of COCO. For example, there are three objects [(d)–(f)] supposed to be detected as *unrecognized label*, but as *pseudolabel* (horse, bird/kite, and cow) by transformer-based detectors. Transformer-based detectors should detect (b) and (c) as *unrecognized label*, but as *true label* (person/people). (a) Human. (b) Separated human. (c) Inverted human. (d) Horse. (e) Bird/kite. (f) Cow.

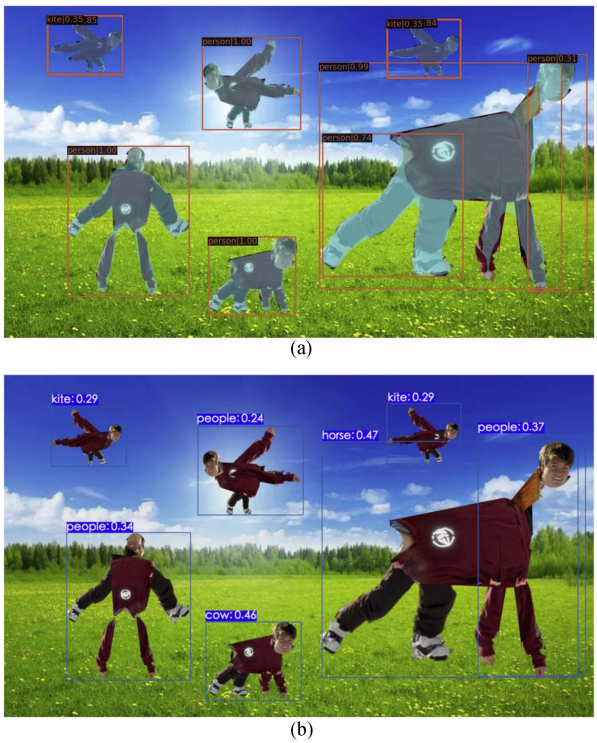

Fig. 2. Comparison of (a) CNN-based and (b) transformer-based methods on restructured objects. Objects are supposed to detected be as *unrecognized label*, but as *pseudolabel* by transformer-based detectors. However, CNN-based detectors show much better performance. The detection explorations get the same results with diverse backgrounds (ocean, grassland, sky, indoor environment, snow, playground, desert, and forest, etc).

The CNN-based methods extract feature with rich local information by convolution layers [2], [3]. Whereas transformer-based methods extract feature by providing the capability to decode and encode global dependencies in transformer blocks (TB) [4], [5]. Compared with CNN-based methods, transformer-based methods have worse performance in small objects (see AP, AP_s, and AP/AP_s in Tables XI and XII).

In this article, we demonstrate that the essential difference between CNN-based and transformer-based detectors is the gap between local information and global dependencies in feature extraction and propagation. First, We screen four influence factors: the number of CNN layer (CL), the number of TB, the number of token (NT), and the attention score scaler (δ). Preexperiments are conducted on COCO object detection under the influence of four factors on evaluation criterion (AP, AP_s,[2] *pseudorate*). Then, the preexperiments indicate that local information tends to help improve AP by increasing AP_s, and global dependencies tend to achieve the same effect by increasing AP_m and AP_l, which cause the essential difference between CNN-based and transformer-based detectors. Meanwhile, both of them will interfere with each other (see Tables I–V).

Given the above conclusions, we propose a new vision transformer, called Hybrid Network Transformer (Hyneter), which consists of hybrid network backbone (HNB) and dual switching (DS) module. The HNB is presented with equivalent positions of intertwined distribution of convolution and self-attention in parallel. Our backbone extends the range of local information by embedding convolution layers into TB in stages, so that

TABLE I
COMPARISON(%) ON DETR WITH RESNET-X BACKBONES ON COCO 2017 VAL SET

Backbone	#param.	AP	AP_s	AP/AP_s	Pseudo	True	Unre
R-34	27M	38.6	18.6	2.08	67.0	30.0	3.0
R-50	41M	42.0	20.5	2.05	60.4	33.9	5.7
R50-DC5	41M	43.3	22.5	1.92	55.7	36.1	8.2
R-101	60M	43.5	21.9	1.99	51.2	38.4	10.4
R101-DC5	60M	44.9	23.7	1.89	42.6	45.2	12.2
R-152	92M	45.4	24.3	1.86	40.0	47.7	12.3

We train DETR with setting as technical details in [17]. R50-DC5 means ResNet-50 with dilated C5 stage. Pseudo means pseudorate; true means unrecognized label rate.

TABLE II
COMPARISON(%) ON DETR WITH VARIANT TB AND RESNET-50 BACKBONE ON COCO 2017 VAL SET

Blocks	#param.	AP	AP_s	AP/AP_s	Pseudo	True	Unre
×1.0	41M	42.0	20.5	2.05	60.4	33.9	5.7
×2.0	51M	44.1	21.0	2.10	66.5	28.7	4.8
×3.0	61M	45.4	20.4	2.23	68.7	25.4	5.9
×4.0	70M	46.1	20.3	2.27	72.9	23.1	4.0

×3.0 means that DETR with three TB and ResNet-50.

TABLE III
COMPARISON(%) ON DETR WITH VARIANT NT AND RESNET-50 BACKBONE ON COCO 2017 VAL SET

Tokens	#param.	AP	AP_s	AP/AP_s	Pseudo	True	Unre
×1.0	41M	42.0	20.5	2.05	60.4	33.9	5.7
×1.5	62M	43.5	20.8	2.10	58.5	35.7	5.8
×2.0	83M	44.1	21.2	2.09	57.0	36.4	6.6
×2.5	103M	45.0	21.6	2.08	55.4	37.1	7.5

×2.0 means that DETR with 2.0 × HW tokens and ResNet-50.

[2]AP denote the average precision of all categories, AP_s for small objects, AP_m for medium objects, and AP_l for large objects. AP/AP_s represents the gap between AP and AP_s. The closer the AP/AP_s is to one, the greater the contribution of AP_s. AP, AP_s, AP_m, and AP_l are evaluation indicators for multiple datasets.

TABLE IV
COMPARISON(%) ON DETR WITH VARIANT ATTENTION SCORE SCALER AND RESNET-50 BACKBONE ON COCO 2017 VAL SET

Scalers	#param.	AP	AP_s	AP/AP_s	Pseudo	True	Unre
×1.0	41M	42.0	20.5	2.05	60.4	33.9	5.7
×1.5	41M	42.9	20.6	2.06	65.0	32.5	2.5
×2.0	41M	43.5	21.0	2.07	67.0	30.7	2.3
×2.5	41M	44.7	21.5	2.08	71.1	27.0	1.9

×2.0 means that DETR with 2.0 × attention score to other score = $Q_i \cdot K_{i\neq I}$ and ResNet-50.

TABLE V
PEARSON CORRELATION COEFFICIENT (ρ) COMPARISON(%) ON FACTORS AND EVALUATING INDICATORS (AP, AP_s, AP/AP_s, AND PSEUDO)

ρ	AP	AP_s	AP/AP_s	Pseudo	True	Unre
CL	0.92	0.98	-0.98	-0.92	0.95	0.98
Trans blocks	0.99	-0.50	0.98	0.98	-0.96	-0.91
Tokens	0.98	1.00	0.48	0.98	0.85	0.79
Scaler δ	0.99	0.75	1.00	0.99	-0.89	-0.99

Gray indicates negative correlation and white indicates positive correlation.

TABLE VI
OBJECT DETECTION PERFORMANCE (%) ON HYNETER VARIANTS WITH MASK R-CNN FRAMEWORKS ON COCO 2017 TEST-DEV SET

Method	Originals	HNB	DS	AP	AP_s	AP/AP_s	#param.
Hyneter-base							
baseline	✓			52.3	21.5	2.43	85M
	✓	✓		55.0	25.3	2.17	87M
	✓	✓	✓	**57.1**	28.3	2.02	90M
Hyneter-plus							
baseline	✓			54.8	23.0	2.38	125M
	✓	✓		56.4	26.7	2.11	129M
	✓	✓	✓	**58.0**	27.9	2.08	134M
Hyneter-max							
baseline	✓			55.7	25.7	2.17	227M
	✓	✓		58.3	27.4	2.10	236M
	✓	✓	✓	**60.1**	**29.8**	2.07	247M

Originals means pure transformer baselines without HNB or DS, which is similar to Swin-T structurally.
The bold values mean the best performance.

TABLE VII
MODULE-LEVEL INFORMATION EXCHANGE CROSS WINDOWS COMPARISON (%) ON COCO 2017 TEST-DEV SET WITH CASCADE MASK R-CNN

Backbone	Module	AP	AP^{mask}	#param.	FLOPs	FPS
Swin Tranformer						
Swin-T	Shift window	50.5	43.7	86M	745G	15.3
Swin-S		51.8	44.7	107M	838G	12.0
Swin-B		51.9	45.0	145M	982G	11.6
Slide-Transformer [42]						
Slide-Swin-T	Deformed Shifting	51.1	44.3	86M	747G	–
Slide-Swin-S		52.5	45.4	107M	838G	–
Slide-Swin-B		52.7	45.5	145M	983G	–
MaxViT [40]						
MaxViT-T	Multi-axis Attention	52.1	44.6	69M	475G	–
MaxViT-S		53.1	45.4	107M	595G	–
MaxViT-B		53.4	45.7	157M	856G	–
Hyneter						
Hyneter-base	DS	57.1	45.1	90M	969G	12.5
Hyneter-plus		58.0	46.9	134M	1195G	7.8
Hyneter-max		**60.1**	**48.5**	247M	2250G	4.8

The bold values mean the best performance.

TABLE VIII
MODULE-LEVEL INFORMATION EXCHANGE CROSS WINDOWS COMPARISON (%) ON IMAGENET-1K, **NOT** PRETRAINED ON IMAGENET-22K

Method	Image size	#Param.	FLOPs	Throughput	Top-1
Swin Transformer—Shift windows					
Swin-T	224^2	29M	4.5G	755/s	81.3
Swin-S	224^2	50M	8.7G	437/s	83.0
Swin-B	224^2	88M	15.4G	278/s	83.3
Swin-B	384^2	88M	47.0G	85/s	84.2
CSWin Transformer [45]—windows expending					
CSWin-T	224^2	23M	4.3G	701/s	82.7
CSWin-S	224^2	35M	6.9G	437/s	83.6
CSWin-B	224^2	78M	15.0G	250/s	84.2
CSWin-B	384^2	78M	47.0G	–	85.4
Slide Transformer—windows sliding					
Slide-Swin-T	224^2	29M	4.6G	755/s	82.3
Slide-Swin-S	224^2	51M	8.9G	437/s	83.7
Slide-Swin-B	224^2	89M	15.5G	278/s	84.2
Hyneter—DS					
Hyneter-base	224^2	29M	6.9G	765/s	83.0
Hyneter-plus	224^2	51M	12.8G	477/s	83.9
Hyneter-max	224^2	90M	19.7G	301/s	84.9
Hyneter-max	384^2	90M	62.0G	105/s	**86.0**

The bold values mean the best performance.

local information and global dependencies will be passed to *neck* or *head* simultaneously. The DS module establishes cross-window connections in order to maintain local information inside patches, while restraining excessive reliance on global dependencies outside patches. Based on the balance strategy, Hyneters integrate and transfer local information and global dependencies simultaneously, so they are able to significantly improve performance.

Ablation studies illustrate that Hyneters with HNB and DS achieve the state-of-the-art performance by a large margin of $+2.1 \sim 13.2$AP on COCO, and $+3.1 \sim 6.5$mIoU on VisDrone in object detection. Furthermore, Hyneters surpass previous best methods on multiple tasks significantly (see Tables VI–XV), such as object detection (60.1AP on COCO and 46.1 on VisDrone), semantic segmentation (54.3AP on ADE20K), and instance segmentation (48.5AP^{mask} on COCO).

The contributions of this work are listed as follows.

1) This article explains the different mechanisms of feature extraction in CNN and transformer frameworks.
2) We propose a new vision transformer (Hyneter) with HNB and DS module.
3) Hyneters achieve excellent performance on multiple computer vision tasks with lighter model size and less computational cost in system-level comparisons.

II. RELATED WORK

A. CNN-Based Vision Backbones

The backbone networks of deep learning are evolving. LeNet (1998), AlexNet (2012), VGGNet (2014), GoogLeNet (2014), ResNet (2015), and MobileNet (2017) are preserved in development of deep learning [6]. EfficientNet (2019) proposes a more

TABLE IX
SYSTEM-LEVEL COMPARISON ON IMAGENET-1K, **NOT** PRETRAINED ON IMAGENET-22K

Method	Image size	#Param.	FLOPs	Throughput	Top-1
PVT-S [15]	224^2	25M	3.8G	820/s	79.8
PVT-M [15]	224^2	44M	6.7G	526/s	81.2
PVT-L [15]	224^2	61M	9.8G	367/s	81.7
T2T$_t$-14 [13]	224^2	22M	6.1G	–	81.7
T2T$_t$-19 [13]	224^2	39M	9.8G	–	82.2
T2T$_t$-24 [13]	224^2	64M	15.0G	–	82.6
CvT-13 [30]	224^2	20M	4.5G	–	81.6
CvT-21 [30]	224^2	32M	7.1G	–	82.5
CvT-21 [30]	384^2	32M	24.9G	–	83.3
Swin-T	224^2	29M	4.5G	755/s	81.3
Swin-S	224^2	50M	8.7G	437/s	83.0
Swin-B	224^2	88M	15.4G	278/s	83.3
Swin-B	384^2	88M	47.0G	85/s	84.2
CSWin-T	224^2	23M	4.3G	701/s	82.7
CSWin-S	224^2	35M	6.9G	437/s	83.6
CSWin-B	224^2	78M	15.0G	250/s	84.2
CSWin-B	384^2	78M	47.0G	–	85.4
Hyneter-base	224^2	30M	6.9G	765/s	83.5
Hyneter-plus	224^2	52M	12.7G	476/s	84.0
Hyneter-max	224^2	95M	19.7G	301/s	85.3
Hyneter-max	384^2	95M	62.0G	104/s	**86.8**

The bold values mean the best performance.

TABLE X
OBJECT DETECTION PERFORMANCE (%) WITH VARIOUS FRAMEWORKS ON COCO 2017 VAL SET

Method	Backbone	AP	AP$_s$	AP/AP$_s$	#param.	FLOPs	FPS
Mask R-CNN	R-50	42.3	24.7	1.71	82M	739G	18.0
	Hyneter-plus	58.0	27.9	2.07	134M	1195G	7.8
ATSS	R-50	43.5	25.7	1.69	32M	205G	28.3
	Hyneter-plus	56.0	27.4	2.04	53M	605G	8.5
DETR	R-50 + trans	42.0	20.5	2.05	41M	86G	28
	Hyneter-plus	47.0	24.7	1.90	93M	807G	10.7

R50 + trans means R50 and TB as DETR backbone.

TABLE XI
OBJECT DETECTION (WITH MASK R-CNN) PERFORMANCE (%) WITH VARIOUS BACKBONES ON COCO 2017 VAL SET

Backbone	AP	AP$_s$	AP/AP$_s$	#params.	FLOPs	FPS
R-50	42.3	24.7	1.71	82M	739G	18.0
R-101	44.5	25.5	1.74	101M	819G	12.8
Swin-T	49.8	21.4	2.33	86M	745G	15.3
Swin-S	51.4	25.1	2.05	107M	838G	12.0
Swin-B	51.5	25.0	2.06	145M	982G	11.6
Swin-L	57.1	26.7	2.14	284M	1470G	–
Hyneter-base	57.1	28.3	2.02	90M	969G	12.5
Hyneter-plus	58.0	27.4	2.08	134M	1195G	7.8
Hyneter-max	**60.1**	**29.8**	2.07	247M	2250G	4.8

The bold values mean the best performance.

TABLE XII
SYSTEM-LEVEL COMPARISON (%) ON COCO 2017 TEST-DEV SET

Method	AP	AP$_s$	AP/ AP$_s$	#param.	FLOPs	FPS
CNN: anchor-based two stage						
CoupleNet(ResNet-101)	34.4	13.4	2.57	–	–	–
FitnessNMS(DeNet-101)	39.5	18.9	2.09	–	–	–
DetNet(DetNet-59)	40.3	23.6	1.71	–	–	–
Cascade(ResNet-101)	42.8	23.7	1.81	–	–	–
CNN: anchor-based one stage						
YOLOv2(DarkNet-19)	21.6	5.0	4.32	–	–	–
DSSD(ResNet-101)	33.2	13.0	2.55	–	–	–
RefineDet512(ResNet-101)	36.4	16.6	2.19	–	–	–
RetinaNet(ResNet-101)	39.1	21.8	1.79	–	–	–
CNN: anchor-free keypoint based						
CornerNet(Hourglass-104)	40.5	19.4	2.09	–	–	–
CenterNet(Hourglass-104)	44.9	25.6	1.75	–	–	–
RepPoints(ResNet-101-DCN)	45.0	26.6	1.69	–	–	–
CNN: anchor-free center based						
GA-RPN(ResNet-50)	39.8	21.8	1.83	–	–	–
FCOS(ResNeXt-64x4d-101)	43.2	26.5	1.63	–	–	–
CNN: others						
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	86G	28.0
UP-DETR	42.8	20.8	2.06	–	–	–
Deformable DETR	46.9	27.7	1.69	63M	262G	–
Conditional DETR	45.1	25.3	1.78	44M	195G	–
Swin Transformer with Cascade Mask R-CNN						
Swin-B (HTC++)	56.4	25.1	2.25	160M	1043G	–
Swin-L (HTC++)	57.1	25.6	2.23	284M	1470G	–
Swin-L (HTC++)*	58.0	26.0	2.23	284M	–	–
Hybrid methods						
GC ViT [27] with Cascade Mask R-CNN 3 × schedule						
GC ViT-T(Mask R-CNN)	47.9	–	–	48M	291G	–
GC ViT-T	51.6	–	–	85M	770G	–
GC ViT-S	52.4	–	–	108M	866G	–
GC ViT-B	52.9	–	–	146M	1018G	–
Mobile-Former [29] with End-to-end object detector for 300 epochs						
E2E-MF-508M	43.3	24.6	1.76	26.3M	–	–
E2E-MF-294M	40.5	20.6	1.97	24.9M	–	–
E2E-MF-214M	39.3	19.9	1.97	20.1M	–	–
E2E-MF-151M	37.2	17.4	2.14	14.8M	–	–
MixFormer [28] with Mask R-CNN 1 × schedule						
MixFormer-B1	40.6	–	–	26M	183G	–
MixFormer-B2	41.5	–	–	28M	187G	–
MixFormer-B3	42.8	–	–	35M	207G	–
MixFormer-B4	45.1	–	–	53M	243G	–
MixFormer [28] with Mask R-CNN 3 × schedule						
MixFormer-B1	43.9	–	–	26M	183G	–
MixFormer-B2	45.1	–	–	28M	187G	–
MixFormer-B3	46.2	–	–	35M	207G	–
MixFormer-B4	47.6	–	–	53M	243G	–
MixFormer [28] with Cascade Mask R-CNN 3 × schedule						
MixFormer-B4	51.6	–	–	91M	721G	–
CNN + transformer						
Conformer-S/32 [47](FPN)	43.1	26.8	1.61	55.4M	288.4G	13.5
Conformer-S/32(Mask R-C)	43.6	27.5	1.59	58.1M	341.4G	10.9
CMT-S [48](RetinaNet 1x)	44.3	–	–	44.3M	231.0B	–
CMT-S(RetinaNet 3x+MS)	46.9	–	–	55.0M	–	–
Ours with Cascade Mask R-CNN						
Hyneter-base	57.1	28.3	2.02	90M	969G	12.5
Hyneter-plus	58.0	27.9	2.08	134M	1195G	7.8
Hyneter-max	**60.1**	**29.8**	2.07	247M	2250G	4.8

*Indicates multi-scale testing. The frameworks in swin-transformer [18] is cascade mask R-CNN. EfficientDet-D7x(1537) [46].
The bold values mean the best performance.

generalized idea on the optimization of current classification networks, arguing that the three common ways of enhancing network metrics, namely, widening the network, deepening the network, and increasing the resolution, should not be independent of each other [7], [8].

Along with the backbone evolving, convolution kernels are also changing. Deformable convolution adds an offset variable to the position of each sampled point in the convolution kernel, enabling random sampling around the current position without being restricted to the previous regular grid points. Dilated convolution can effectively focus on the semantic information of the local pixel blocks, instead of letting each pixel rub

TABLE XIII
SYSTEM-LEVEL COMPARISON (%) OF HYNETERS ON VISDRONE-DET 2020 [50] AND 2021 [51]

Method	AP	AP_{50}	AP_{75}
VisDrone-DET 2020 [50]			
DroneEye2020 (A.4)	34.57	58.21	35.74
TAUN (A.5)	34.54	59.42	34.97
CDNet (A.6)	34.19	57.52	35.13
CascadeAdapt (A.7)	34.16	58.42	34.5
HR-Cascade++ (A.9)	32.47	55.06	33.34
MSC-CenterNet (A.11)	31.13	54.13	31.41
CenterNet+ (A.12)	30.94	52.82	31.13
ASNet (A.13)	29.57	52.25	29.37
CN-FaDhSa (A.14)	28.52	49.5	28.86
HRNet (A.15)	27.39	49.9	26.71
DMNet (A.16)	27.33	48.44	27.31
HRD-Net (A.17)	26.93	45.45	27.77
PG-YOLO (A.18)	26.05	49.63	24.15
EFPN (A.19)	25.27	48.18	23.37
CRENet (A.20)	25.16	44.38	24.57
Cascade R-CNN++ (A.21)	24.66	43.53	24.71
HR-ATSS (A.22)	24.23	41.84	24.43
CFPN (A.23)	22.85	42.33	21.88
Center-ClusterNet (A.24)	22.72	41.45	22.13
HRC (A.26)	21.23	43.56	18.39
IterDet (A.27)	20.42	36.73	20.25
GabA-Cascade (A.29)	18.85	33.60	18.66
VisDrone-DET2021 [51]			
DBNet	39.4	65.4	41.0
SOLOer	39.4	63.9	40.8
Swin-T	39.4	63.9	40.8
TPH-YOLOv5	39.1	62.8	41.3
VistrongerDet	38.7	64.2	40.2
EfficientDet	38.5	63.2	39.5
DroneEye2020	34.5	58.2	35.7
Cascade R-CNN	16.0	31.9	15.0
DroneEye2020	34.57	58.21	35.74
DPNet-ensemble	37.3	62.0	39.1
Ours			
Hyneter-base	41.9	65.8	43.7
Hyneter-plus	43.7	70.1	45.8
Hyneter-max	**46.1**	**73.9**	**47.0**

The bold values mean the best performance.

TABLE XIV
INSTANCE SEGMENTATION (WITH CASCADE MASK R-CNN) PERFORMANCE (%) WITH VARIOUS BACKBONES ON COCO 2017 TEST-DEV SET

Backbone	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param.	FLOPs
ResNet-50-C4	30.3	51.2	31.5	–	–
ResNet-101-C4	32.7	54.2	34.3	–	–
ResNet-50-FPN	32.5	55.4	31.7	82M	–
ResNet-101-FPN	35.9	60.7	36.8	101M	–
ResNeXt-101-FPN	36.7	59.5	38.9	–	–
Swin-T	43.7	66.6	47.1	86M	745G
Swin-S	44.7	67.9	48.5	107M	838G
Swin-B	45.0	68.4	48.7	145M	982G
E2E-MF-508M	43.3	61.8	46.8	26.3M	–
E2E-MF-294M	40.5	58.8	43.5	24.9M	–
E2E-MF-214M	39.3	57.3	42.1	20.1M	–
E2E-MF-151M	37.2	54.5	39.9	14.8M	–
Hybrid methods					
GC ViT-T(Mask R-CNN)	43.2	67.0	46.7	48M	291G
GC ViT-T	44.6	67.8	48.3	85M	770G
GC ViT-S	45.4	68.5	49.3	108M	866G
GC ViT-B	45.8	69.2	49.8	146M	1018G
MixFormer-B1(1x)	37.5	59.7	40	26M	183G
MixFormer-B2(1x)	38.3	60.6	41.2	28M	187G
MixFormer-B3(1x)	39.3	61.8	42.2	35M	207G
MixFormer-B4(1x)	41.2	64.3	44.1	53M	243G
MixFormer-B1(3x + MS)	40.0	62.9	42.9	26M	183G
MixFormer-B2(3x + MS)	40.8	64.1	43.6	28M	187G
MixFormer-B3(3x + MS)	41.9	65.6	45.0	35M	207G
MixFormer-B4(3x + MS)	43.0	66.7	46.4	53M	243G
MixFormer-B4	44.9	67.9	48.7	91M	721G
Ours					
Hyneter-base	45.1	78.3	42.2	90M	969G
Hyneter-plus	46.9	79.9	45.0	134M	1195G
Hyneter-max	**48.5**	**82.1**	**46.7**	247M	2250G

The bold values mean the best performance.

together with the surrounding blocks, which affects the detail of segmentation [9], [10].

B. Transformer-Based Vision Backbones

The pioneering work of ViT [11] directly applies a transformer architecture on nonoverlapping image patches for image classification. ViT and its follow-ups [12], [13], [14], [15] achieve an impressive speed-accuracy tradeoff on image classification compared with convolution networks. The results of ViT on image classification are encouraging, but its architecture is unsuitable for use as a general-purpose backbone network on dense vision tasks or when the input image resolution is high, due to its low-resolution feature maps and the quadratic increase in complexity with image size [16].

DETR [17] and Swin Transformer [18], following ViT and variants, are representative methods in computer vision. DETR and its follow-ups (UP-DETR [19], conditional DETR [20], OW-DETR [21], and Deformable DETR [22]) demonstrate excellent plasticity and flexibility in computer vision tasks. Meanwhile, Swin Transformer is effective, achieving state-of-the-art accuracy on both object detection and semantic segmentation with huge model size and heavy computational cost (see in ablation studies and Table XI–XV).

C. Hybrid Network Vision Backbones

Many hybrid backbones [23] are presented in previous works, which put convolution and self-attention in the nonequivalent position. Previous methods employ self-attention blocks within or outside the CNN backbone architecture. Furthermore, representative hybrid methods completely cleave the relation of local information and global dependencies by separated distribution of convolution and self-attention.

Next-ViT [24] is designed to stack Next Convolution Block and Next TB in an efficient hybrid paradigm, which boosts performance in various downstream tasks. Li et al. [25] revisited the design choices of ViTs and proposed an improved supernet with low latency and high parameter efficiency. They further introduce a fine-grained joint search strategy that can find efficient architectures by optimizing latency and number of parameters simultaneously. By combining CNN and Transformer, HBCT [26] extracts deep features beneficial for superresolution reconstruction in consideration of both local and nonlocal priors, while being lightweight and flexible enough.

TABLE XV
SYSTEM-LEVEL COMPARISON (%) OF SEMANTIC SEGMENTATION ON ADE20K VAL AND TEST SET

Method	Backbone	val mIoU	test score	#param.	FLOPs	FPS
DANet	ResNet-101	45.2	–	69M	1119G	15.2
Dlab.v3+	ResNet-101	44.1	–	63M	1021G	16.0
ACNet	ResNet-101	45.9	38.5	–	–	–
DNL	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet	ResNet-101	44.9	–	86M	1029G	20.1
OCRNet	HRNet-w48	45.7	–	71M	664G	12.5
Dlab.v3+	ResNeSt-101	46.9	55.1	66M	1051G	11.9
Dlab.v3+	ResNeSt-200	48.4	–	88M	1381G	8.1
SETR	T-Large	50.3	61.7	308M	–	–
Swin Transformer [18]						
UperNet	Swin-S	49.3	–	81M	1038G	15.2
UperNet	Swin-B	51.6	–	121M	1841G	8.7
UperNet	Swin-L	53.5	62.8	234M	3230G	6.2
ConvNet [53]						
UperNet	ConvNeXt-B	53.1	–	122M	1828G	–
UperNet	ConvNeXt-L	53.7	–	235M	2458G	–
UperNet	ConvNeXt-XL	54.0	–	391M	3335G	–
Hybrid methods						
UperNet	MixFormer-B1	43.5	–	35M	854G	–
UperNet	MixFormer-B2	43.9	–	37M	859G	–
UperNet	MixFormer-B3	45.5	–	44M	880G	–
UperNet	MixFormer-B4	48.0	–	63M	918G	–
UperNet	GC ViT-T	47.0	–	58M	947G	–
UperNet	GC ViT-S	48.3	–	84M	1163G	–
UperNet	GC ViT-B	49.2	–	125M	1348G	–
Ours						
UperNet	Hyneter-base	50.6	62.0	82M	862G	15.0
UperNet	Hyneter-plus	53.0	63.4	125M	1605G	8.9
UperNet	Hyneter-max	**54.3**	**65.9**	231M	2905G	6.8

The comparison data and setting are from [18, Appendix], [53], and UperNet [52].
The bold values mean the best performance.

The cores of GC ViT [27] are global context self-attention modules, jointly with standard local self-attention. In addition, GC ViTs address the lack of inductive bias in ViTs and improve the modeling of interchannel dependencies by proposing a novel downsampler, which leverages a parameter-efficient fused inverted residual block. Chen et al. [28] proposed bidirectional interactions across branches to provide complementary clues in the channel and spatial dimensions. Mobile-Former [29] leverages the advantages of MobileNet at local processing and transformer at global interaction, and enables bidirectional fusion of local and global features. Mainstream hybrid models typically unify CL and TB into a single pipeline in serialization [see in Fig. 4(c)]. However, the work in [28] and [29], and CvT [30] provide new insights for subsequent work.

Different from pure attention models (such as SASA [31], LRNet [32], SANet [33], Axial-SASA [34] and ViT), VideoBERT [35], VILBERT [36], and CCNet [8] employ self-attention on the top of backbone architecture. AA-ResNet [37] also attempted to replace a fraction of spatial convolution channels with self-attention. But hybrid network methods proved to be imbalanced in size-sensitive performance, as they utilize local information and global dependencies unequally [38].

D. Information Exchange Cross Windows

In Swin Transformer, cyclic shifted window is used to exchange information cross windows that complements the computation of self-attention inside each isolated window. CrossFormer [39], MaxViT [40], and DiNA [41] use the group or dilated strategy to replace cyclic shift operation in Swin Transformer or stand-alone self-attention in SASA [31], since two of these operations are unfriendly latency. MaxViT [40] allows global–local spatial interactions on arbitrary input resolutions with only linear complexity.

Slide-Transformer [42] employs efficient depthwise convolution as sliding windows to communicate information cross windows. Twins-PCPVT and Twins-SVT [43] use global self-attention layers that subsample a set of key and value matrices. These methods provide novel perspectives for improvement and directions for follow-ups (see in ablation studies and Table VII).

However, the hybrid methods can also lead to disadvantages, such as high computational costs, a large number of parameters, and poor interpretability. Therefore, when designing hybrid models based on architectural reference, it is still necessary to make improvements to the corresponding details in practical research.

III. ANALYSIS OF INFLUENCE FACTORS

In this section, preexperiments are conducted to analyze the qualitative and quantitative influence of four factors to local information and global dependencies in transformer-based detectors. The comparisons will provide foundations to study feature extraction mechanism in CNN/transformer framework and create a new transformer.

Without loss of generality, the representative DETR is adopted to dissect the influence of CL, TB, NT, and attention score scaler (δ). The δ is a parameter that controls the calculation of attention score in transformer

$$\text{attention score} = \begin{cases} q_i \cdot k_l, i = l \\ \delta q_i \cdot k_l, i \neq l. \end{cases} \tag{1}$$

As given in Tables I–IV, comparisons on DETRs with different factors demonstrate the qualitative and quantitative relationships among local information, global dependencies, four factors, and detector performance. With the help of Pearson correlation coefficient[3] (ρ) in Table V, the rules are summarized as follows.

1) With the increase of CL, the detectors will pay more attention to local information, reducing the reliance on global dependencies, and will gradually improve AP and AP_s.
2) The increase of TB will promote detectors to rely more on global dependencies, thereby improving performance, but hurt AP_s.

[3]The Pearson correlation coefficient is widely used to measure the degree of correlation between two variables. The changes in factors and evaluating indicators follow the property of linear relationships. *Pseudorate*, *true label rate*, and *unrecognized label rate* are just for explaining the differences between CNN and transformer methods, not evaluation indicators.

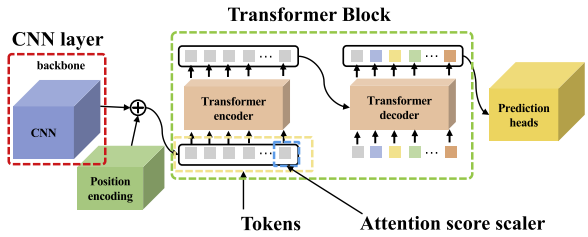

Fig. 3. Illustration of DETR stressing on four factors: CL, TB, tokens, and attention score scaler.

3) The increase of token will restrain the methods' reliance on global dependencies, and simultaneously improve the methods AP and AP_s, but increase model size.
4) The increase of the attention score scaler simultaneously improves AP, increasing the reliance on global dependencies, nor does it increase model size.

Conclusion: Local information tends to increase AP_s to improve AP, and global dependencies tend to increase AP_m and AP_l in order to improve AP. Meanwhile, both of them will interfere with each other. The gap between local information and global dependencies in feature extraction and propagation causes worse performance of small object in transformer-based methods. Computing self-attention impedes extraction of local information feature, while convolution layers prevent extracting feature of global dependencies.

IV. Hybrid Network Transformer

In view of the above conclusions and inspired by the authors in [28] and [29], we propose a new vision transformer, called *Hyneter*, that capably serves as a backbone for multiple computer vision tasks, which consists of HNB and DS module. Hyneter will unify CL and TB into pipelines in parallel. DS can establish cross-window connections while maintaining local information in the patch, functionally similar to *shifted window-based self-attention*, but with less computational cost.

An overview of the Hyneter architecture is presented in Fig. 4(a), which illustrates the basic version. Data are preprocessed as method in [18].

A. Hybrid Network Backbone

Many hybrid backbones [23] are presented in previous works, which put convolution and self-attention in the nonequivalent position. Previous methods employ self-attention within the CNN backbone architecture or use them outside. Furthermore, representative hybrid methods (such as DETR, see Fig. 3) completely cleave the relation of local information and global dependencies by separated distribution of convolution and self-attention. HNB is presented with equivalent position of intertwined distribution of convolution and self-attention. Our backbone extends the range of local information, so that local information and global dependencies will be passed to *neck* or *head* simultaneously.

There are four stages in our backbone, starting with a convolution layer of three multigranularity kernels. The NT are reduced by this multigranularity 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 TB.

As shown in Fig. 4(b), the TB extract feature maps of global dependencies, and CL extract feature maps of local information in Stages 1 and 2. The output ($C \times \frac{H \times W}{4 \times 4}$) of the final TB in Stage 1 will be reviewed 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 reviewed and permuted twice, the addition turns to the input (X') of Stage 2.

With a hybrid network approach, consecutive self-attention TB are computed as

$$\begin{aligned} X &= \text{Re-view}(\text{GMSA}(S)) \\ S_1 &= \text{Conv}_1(S) \oplus \text{Conv}_2(S) \oplus \text{Conv}_3(S) \\ X_2 &= \tanh(X \cdot S_1) \\ X' &= \text{Re-view}(X \oplus X_2) \end{aligned} \tag{2}$$

where GMSA means global multihead self-attention. The TB in Stages 1 and 2 are pure self-attention with maintaining the NT, and together with interfaces for convolution layer output. The blocks in Stages 3 and 4 will be implemented with DS. Forcing concatenation of different types of feature maps leads to confusion in feature information. The experiments proved that the integration in Fig. 4(b) achieved the best results.

B. Dual Switching

The DS module will be implemented in Stages 3 and 4, in order to maintain local information while restraining excessive reliance on global dependencies.

Global dependencies from global self-attention are conducted in TB, where the dependencies among tokens are computed. With respect to NT, the computation results in quadratic complexity, which is inadequate for many vision tasks with huge NT. For efficiency, the GMSA will be implemented within local windows in a nonoverlapping manner.

As illustrated in Fig. 5, the output of TB 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 DS module establishes cross-window connections while maintaining local information in the patch, which is followed by LayerNorms (LN), TB, and multilayer perceptions (MLP) with residual connection modules.

After Stages 1 and 2 in our backbone, the feature in a patch with abundant local information has established considerable global dependencies with surrounding patches. DS suspends the

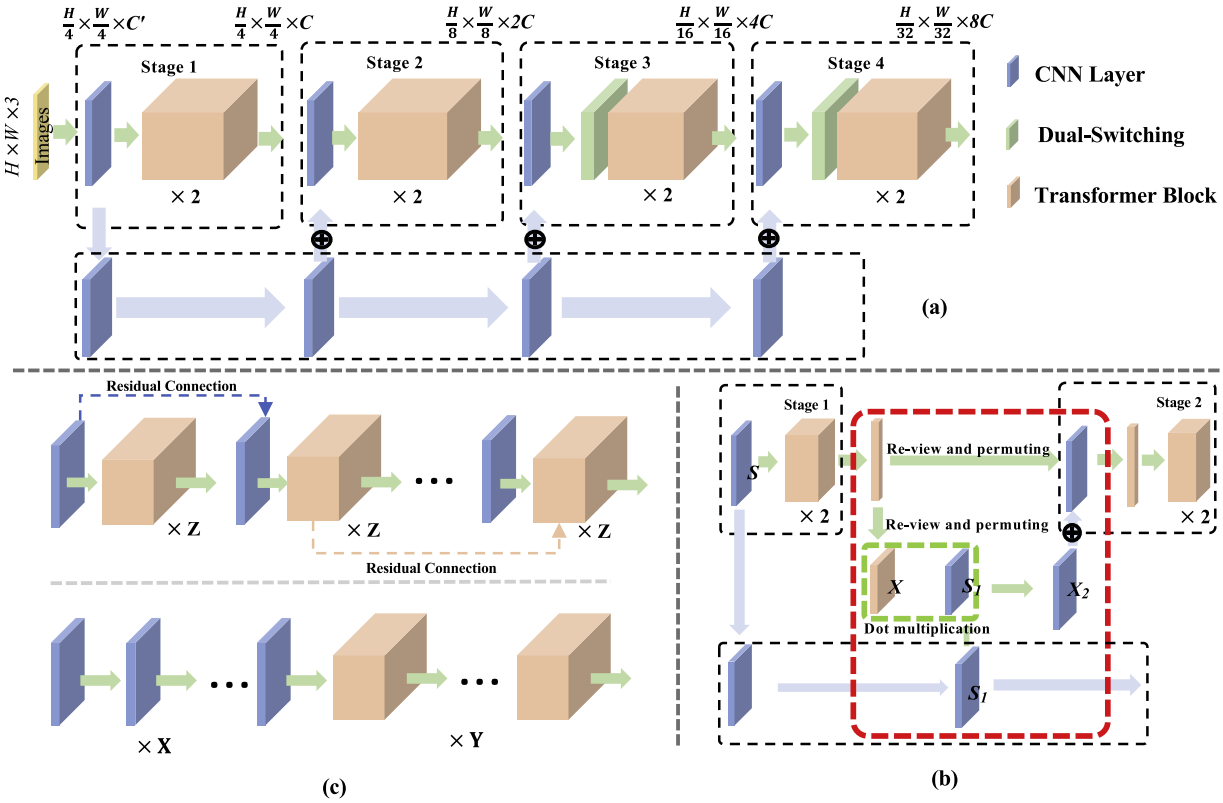

Fig. 4. (a) Architecture of Hyneter-base. There are two TB in one stage of transformer pipeline (top) and two-layer multigranularity convolution layers in one stage of CNN pipeline (bottom). Positional encoding, patch partition, and self-attention in the first TB, but patch partition and self-attention in others. (b) Illustration of unidirectional feature integration between TB (top) and CL (bottom). (c) Two kinds of traditional hybrid models, imitating MixFormer [28], CvT [30], [upper half in (c)], and DETR [17] [lower half in (c)]. (a) Architecture of Hyneter 1.0. (b) Integration. (c) Traditional Hybrid Models.

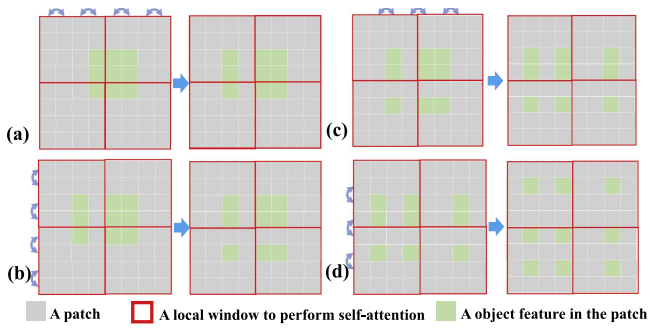

Fig. 5. Illustration of DS. The process is implementing as (a)→(b)→(c)→(d). Hyneters maintain that the number of patch is even in the pipeline.

procedure of establishing excessive global dependencies, meanwhile, retaining local information for small object performance (AP_s). With the DS module, the process is computed as

$$\begin{aligned} X_l &= \text{Dual-Switch}\left(X_l\right) \\ X_{l+1} &= \text{GMSA}\left(\text{LN}\left(X_l\right)\right) + X_l \\ X'_{l+1} &= \text{MLP}\left(\text{LN}\left(X_{l+1}\right)\right) + X_{l+1} \end{aligned} \tag{3}$$

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

Patches can be exchanged on adjacent local windows, which restrain excessive reliance on global dependencies within local windows, patches change its relative position and retain its internal characteristics. So, DS maintain local information within patches.

C. Architecture Variants

We establish basic model, called Hyneter-base, of model size similar to DETR-DC5-R101. This article also presents Hyneter-plus and Hyneter-max, which are two versions of around 2.0× and 4.0× the model size and computation complexity, respectively. The architecture hyperparameters of these model variants are as follows:

1) Hyneter-base : $d = 96$, CL = {2, 2, 2, 2}, TB = {2, 2, 2, 2};
2) Hyneter-plus : $d = 96$, CL = {2, 2, 3, 2}, TB = {2, 2, 6, 2};
3) Hyneter-max : $d = 128$, CL = {2, 2, 6, 2}, TB = {2, 2, 18, 2};

where d is the channel number of the TB in the first stage.

V. Experiments

In this section, we conduct experiments on multiple datasets in several vision tasks. In the following, we first ablate the important design elements of Hyneter. Then, we compare the proposed Hyneter architecture with the previous state-of-the-art methods on the three tasks.

A. Ablation Studies

Settings: The following experiments were conducted on COCO 2017 dataset using two GeForce RTX 3090 GPUs and two Tesla V100 PCIe 32 GB GPUs. All models under PyTorch framework are standard models without using any tricks. For the ablation study and comparisons, we consider four typical object detection frameworks: Swin Transformers (V1, V2) [18], [44] and DETRs (DETR [17], UP-DETR [19], conditional DETR [20]).

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. There are seven instances per image on average, up to 63 instances in a single image in training set, ranging from small to large on the same images.

Training: Hyneter is trained with AdamW and stochastic gradient descent (SGD) optimizers, changing AdamW to SGD until very final stage. We adopt Hyneter models with the learning rate (2^{-5}) for backbone. The backbone is the ImageNet 22k-pretrained model with batchnorm layers fixed (ImageNet 22k-pretrained Swin Transformer is the main comparative model), and the transformer parameters are initialized using the Xavier initialization scheme. The weight decay is set to be 10^{-4}.

Ablation Study I: We conduct ablation studies on COCO 2017 object detection. Table VI lists the results of Hyneter variants with Mask R-CNN. Our architecture with HNB or DS 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 detectors, with the same model size.

HNB extends the range of local information, retaining and transforming local information and global dependencies to *neck* simultaneously, which greatly increases the proportion of small object performance (AP/AP_s : $2.43 \rightarrow 2.17$; $2.38 \rightarrow 2.11$; $2.17 \rightarrow 2.10$), thereby improve general performance (AP : $52.3 \rightarrow 55.0$; $54.8 \rightarrow 56.4$; $55.7 \rightarrow 58.3$). With the deepening of stages, self-attention will constantly restrain local information and increase the role of global dependencies. Meanwhile, DS will retain local information in the patch, and restrain the excessive strengthening of existing global dependencies, which improve AP and AP_s concurrently (see Table VI).

Ablation Study II: Ablation studies on *information exchange cross windows module* are conducted on COCO 2017 with Cascade Mask R-CNN. SwinTransformer, Slide Transformer, and Hyneter use the same backbone with different information exchange modules. MaxViT uses variants of Swin Transformer's backbone with multi-axis attention. Hyneter works with DS modules to achieve the state-of-the-art performance even with fewer parameters and floating-point operations (FLOPs) (see Table VII).

Ablation studies on *information exchange cross windows module* are also conducted on ImageNet-1K. SwinTransformer, CSWin Transformer, Slide Transformer, and Hyneter use the same backbone with different information exchange modules. Hyneter works with DS modules to achieve the state-of-the-art performance even (see Table VIII).

We utilized a variety of exchange modules to validate the effectiveness and computational cost after similarly structured backbones. DS model achieved optimal performance (60.1AP, $48.5AP^{mask}$ in Table VII; 86.0 Top $-$ 1 in Table VIII), even with similar or less computational cost. Information exchange operations require a delicate balance between complexity and computational cost. MaxViT [40] and Slide-Transformer [42] indicate that we still need to carefully consider the degree of exchange, and it is not that the more complex the operation, the more sufficient the exchange, the better the results will be achieved.

B. Image Classification on ImageNet-1K

1) Setting And Dataset: For image classification, we benchmark Hyneter on ImageNet-1K, which contains 1.28M training images and 50K validation images from 1000 classes. The detailed implementation details are fully in accordance with [45], and we *do not* pretrained on ImageNet-22K.

Hyneter achieves the state-of-the-art performance (86.8% Top-1) in Table IX with the proposed HNB and DS, which presents comparisons with other backbones. Compared with the state-of-the-art Swin Transformer and CSWin Transformer, Hyneter achieves a slightly better speed-accuracy tradeoff (30 M Param, 6.9 G FLOPs, 765/s throughput in Hyneter-base 224^2 image size; 95 M Param, 62.0 G FLOPs, 104/s throughput in Hyneter-max 384^2 image size).

C. Object Detection on COCO 2017

1) Setting: For the ablation study, we consider four typical object detection frameworks: Mask R-CNN, adaptive training sample selection (ATSS), DETR, and Swin Transformer with the same setting (multiscale training, and the AdamW optimizer with an initial learning rate of 0.00001 and a weight decay of 0.05) in mmdetection (see Tables X and XI). We adopt ImageNet-22K pretrained model (*not* on ImageNet-1K) as initialization for system-level comparison. *We trained and tested methods on COCO 2017* (see in Fig. 6).

Dataset is mentioned in *ablation studies*.

2) Comparison With ResNet: The results of Hyneter-plus and ResNet-50 are listed in Table X. Our Hyneter-plus architecture brings consistent $+5.0 \sim 15.7$ AP and $+1.7 \sim 4.2$ AP_s gains over ResNet-50, with an acceptable larger model size. All Hyneters achieve significant gains of $+14.8 \sim 15.6$AP and $+3.6 \sim 4.3AP_s$ over ResNet-50 or ResNet-101, which have a similar or lighter model size (see Table XI).

3) Comparison With Swin Transformer: The comparison of Hyneter and Swin Transformer under different backbones with Mask R-CNN is given in Table XI. Hyneters achieve high detection accuracies of 60.1AP and $29.8AP_s$, which are significant improvement of $+2.3 \sim 7.3$ AP and $+3.1 \sim 6.9$ AP_s over Swin Transformers with lighter model size.

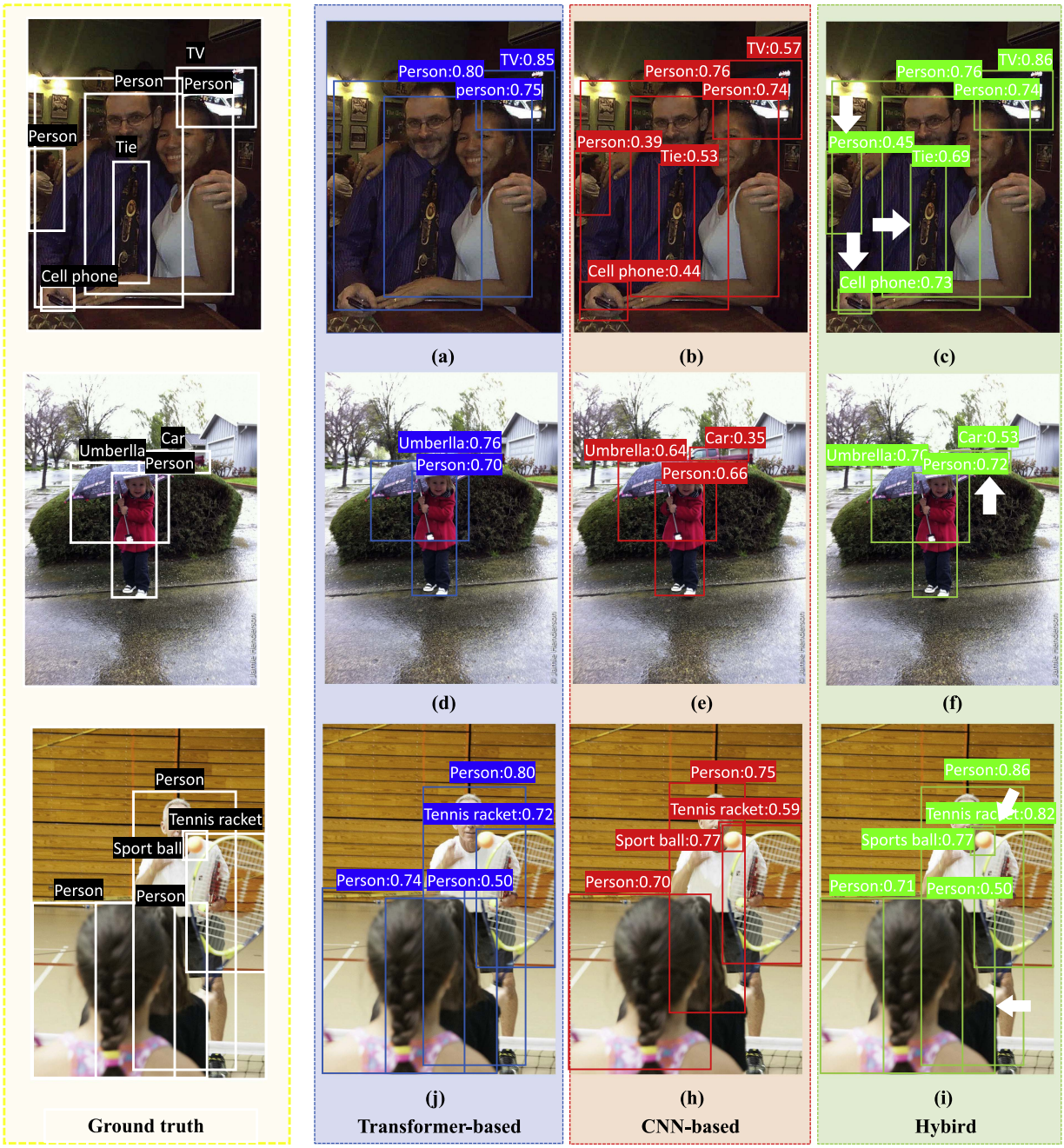

Fig. 6. Comparison of transformer-based, CNN-based, and hybrid methods on COCO 2017. Comparing with the hybrid method, the transformer-based method ignores a person, a cell phone, a tie in (c), a car in (f), and a sportball in (i), meanwhile the CNN-based method ignores a person in (i). At the same time, the hybrid method gets better performance in accurate positioning [see a person, a cell phone, a tie in (c), a car in (f), and a person in white in (i)].

4) Comparison With Previous State of the Art (SOTA) and Hybrid Ones: Table XII 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 [49]), +5.0AP (EfficientDet-D7x [46]), +13.2AP (Deformable DETR [22]), and +2.1AP (Swin-L [18] with HTC++ and multiscale testing). Furthermore, Hyneters surpass the previous best performances and *greatly improve* AP$_s$, comparing with Swin Transformers, GC ViT [27], MixFormer [28], Mobile-Former [29], Conformer [47], and CMT-S [48] in system-level comparison.

D. Object Detection on VisDrone

1) Setting: For comparison, we consider methods with the same setting (multiscale training, and the AdamW optimizer with an initial learning rate of 0.00001 and a weight decay

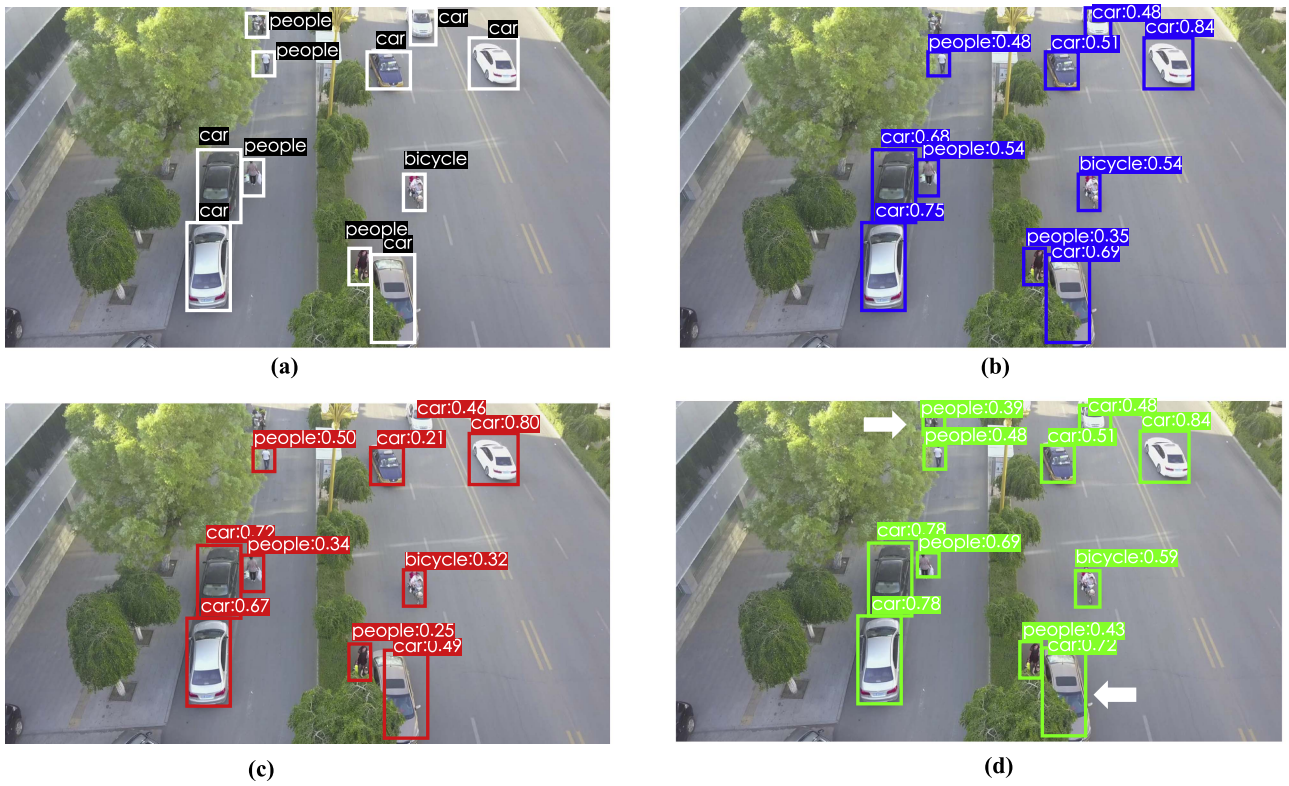

Fig. 7. Comparison of transformer-based, CNN-based, and hybrid methods in VisDrone-DET 2020 and 2021. Comparing with the hybrid method, the transformer-based and the CNN-based methods ignore a person [see a person in (d)], meanwhile the hybrid method gets better performance in accurate positioning [see a car in (d)]. (a) Ground truth. (b) Transformer-based. (c) CNN-based. (d) Hybird.

of 0.05) in mmdetection. We adopt ImageNet-22K pretrained model as initialization for system-level comparison.

2) Dataset: The VisDrone dataset consists of 400 video clips formed by 265 228 frames and 10 209 static images, captured by various drone-mounted cameras, covering a wide range of aspects, including location, environment, and objects (ten classes). These frames are manually annotated with more than 2.6 million bounding boxes or object points of interests, such as pedestrians, cars, bicycles, and tricycles.

Table XIII compares our best results with those of previous state-of-the-art models in VisDrone-DET 2020 and 2021 challenge (see Fig. 7). Our best model (Hyneter-max) achieves 46.1AP, 73.9AP_{50}, and 47.0AP_{75} on VisDrone, surpassing all previous best results in Table XIII. Comparing with the hybrid method, the transformer-based and the CNN-based methods ignore a person [see a person in Fig. 7(d)]. Meanwhile the hybrid method gets better performance in accurate positioning in small objects [see a car in Fig. 7(d)], due to the integration of local information and global dependencies. For more information on VisDrone-DET, refer to Appendix II in the Supplementary Material.

E. Instance Segmentation on COCO 2017

Setting and *Dataset* are mentioned in *ablation studies*. We strictly followed the implementation details in Cascade Mask R-CNN.

Table XIV compares our best instance segmentation results with those of previous state-of-the-art models on COCO 2017. Our best model (Hyneter-max) achieves 48.5AP^{mask}, 82.1AP^{mask}_{50}, and 46.7AP^{mask}_{75} with competitive model size and computational cost, surpassing all previous best results (see Table XIV).

F. Semantic Segmentation on ADE20K

1) 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 1500 iterations. Models are trained on two GPUs with four images per GPU for 140K iterations. We strictly followed the implementation details in UperNet [52].

2) 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, and 3K for test). ADE20K covers various annotations of scenes, objects, and object parts, and each image has an average of 19.5 instances and 10.5 object classes.

Table XV lists mean intersection over union (mIoU), test score, and model size for different method/backbone pairs. From these results, it can be seen that Hyneter-max is +4.0mIoU higher than SETR with much lighter model size. It is also +5.9mIoU higher than ResNeSt-200, and +7.4mIoU higher than ResNeSt-101. Our Hyneters with UperNet [52] achieve 50.6mIoU, 53.0mIoU, and 54.3mIoU on val set, surpassing

Swin Transformers by $+0.8 \sim 1.4$mIoU, MixFormers by about $+6.3 \sim 10.8$mIoU, GC ViTs by about $+3.6 \sim 5.1$mIoU, and ConvNeXts by about $+1.0$mIoU with lighter model size and less computational cost.

VI. Conclusion

In this work, we point out that the essential differences between CNN-based and transformer-based detectors 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 Hyneter, which consists of HNB and DS. Based on the balance strategy, Hyneters integrate and transfer local information and global dependencies in parallel, so they are able to significantly improve performance. Ablation studies illustrate that Hyneters with HNB and DS achieve the state-of-the-art performance on multiple datasets for object detection. Furthermore, Hyneters achieve the state-of-the-art performance on multiple computer vision tasks (object detection, semantic segmentation, and instance segmentation) significantly, and surpass previous best methods.

More importantly, Hyneter's friendliness is toward small objects. The existing hybrid methods, due to the fragmentation of local information, result in a sharp decline in the performance of small objects. The runtime of Hyneter is much shorter than traditional hybrid models. We do hope that Hyneters will play a role of cornerstone to encourage balancing methods between local information and global dependencies in computer vision.

References

[1] Y. Dong, J.-B. Cordonnier, and A. Loukas, "Attention is not all you need: Pure attention loses rank doubly exponentially with depth," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 2793–2803.

[2] Q. Chen, Y. Wang, T. Yang, X. Zhang, J. Cheng, and J. Sun, "You only look one-level feature," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13039–13048.

[3] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do vision transformers see like convolutional neural networks?," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 12116–12128.

[4] Y. Li, H. Mao, R. Girshick, and K. He, "Exploring plain vision transformer backbones for object detection," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 280–296.

[5] X. Dai et al., "Dynamic head: Unifying object detection heads with attentions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7373–7382.

[6] T. Wang, Z. Zhang, and K.-L. Tsui, "A deep generative approach for rail foreign object detections via semisupervised learning," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 459–468, Jan. 2022.

[7] C. Dong and M. Duoqian, "Control distance IoU and control distance IoU loss for better bounding box regression," *Pattern Recognit.*, vol. 137, 2023, Art. no. 109256.

[8] J. Zhang, J. Chen, S. Chen, Z. Wang, and J. Zhang, "Detection and segmentation of unlearned objects in unknown environment," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 6211–6220, Sep. 2020.

[9] X.-T. Vo and K.-H. Jo, "Accurate bounding box prediction for single-shot object detection," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 5961–5971, Sep. 2022.

[10] X. Zhou et al., "Intelligent small object detection for digital twin in smart manufacturing with industrial cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1377–1386, Feb. 2021.

[11] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[12] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10347–10357.

[13] L. Yuan et al., "Tokens-to-Token ViT: Training vision transformers from scratch on ImageNet," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 558–567.

[14] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, "Transformer in Transformer," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 15908–15919.

[15] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 568–578.

[16] X. Zang, G. Li, and W. Gao, "Multidirection and multiscale pyramid in transformer for video-based pedestrian retrieval," *IEEE Trans. Ind. Informat.*, vol. 18, no. 12, pp. 8776–8785, Dec. 2022.

[17] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 213–229.

[18] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10012–10022.

[19] Z. Dai, B. Cai, Y. Lin, and J. Chen, "Up-DETR: Unsupervised pre-training for object detection with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1601–1610.

[20] D. Meng et al., "Conditional DETR for fast training convergence," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3651–3660.

[21] A. Gupta, S. Narayan, K. Joseph, S. Khan, F. S. Khan, and M. Shah, "Ow-DETR: Open-world detection transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 9235–9244.

[22] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "DD deformable transformers for end-to-end object detection," in *Proc. 9th Int. Conf. Learn. Representations*, 2021, pp. 3–7.

[23] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. pattern Recognit.*, 2021, pp. 16519–16529.

[24] J. Li et al., "Next-ViT: Next generation vision transformer for efficient deployment in realistic industrial scenarios," 2022, *arXiv:2207.05501*.

[25] Y. Li et al., "Rethinking vision transformers for MobileNet size and speed," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 16889–16900.

[26] J. Fang, H. Lin, X. Chen, and K. Zeng, "A hybrid network of CNN and transformer for lightweight image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1103–1112.

[27] A. Hatamizadeh, H. Yin, J. Kautz, and P. Molchanov, "Global context vision transformers," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 12633–12646.

[28] Q. Chen et al., "MixFormer: Mixing features across windows and dimensions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5249–5259.

[29] Y. Chen et al., "Mobile-Former: Bridging MobileNet and transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5270–5279.

[30] H. Wu et al., "CVT: Introducing convolutions to vision transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 22–31.

[31] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 68–80.

[32] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3464–3473.

[33] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10076–10085.

[34] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, "Axial-DeepLab: Stand-alone axial-attention for panoptic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 108–126.

[35] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "VideoBERT: A joint model for video and language representation learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7464–7473.

[36] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 13–23.

[37] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3286–3295.

[38] H. Fan, X. Wang, Q. Wang, S. Fu, and Y. Tang, "Skip connection aggregation transformer for occluded person reidentification," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 442–451, Jan. 2024.

[39] W. Wang et al., "Crossformer++: A versatile vision transformer based on cross-scale attention," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023.

[40] Z. Tu et al., "MaxVIT: Multi-axis vision transformer," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 459–479.
[41] A. Hassani and H. Shi, "Dilated neighborhood attention transformer," 2022, *arXiv:2209.15001*.
[42] X. Pan, T. Ye, Z. Xia, S. Song, and G. Huang, "Slide-transformer: Hierarchical vision transformer with local self-attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 2082–2091.
[43] X. Chu et al., "Twins: Revisiting the design of spatial attention in vision transformers," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 9355–9366.
[44] Z. Liu et al., "Swin transformer V2: Scaling up capacity and resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12009–12019.
[45] X. Dong et al., "CSWin transformer: A general vision transformer backbone with cross-shaped windows," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12124–12134.
[46] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10781–10790.
[47] Z. Peng et al., "Conformer: Local features coupling global representations for visual recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 367–376.
[48] J. Guo et al., "CMT: Convolutional neural networks meet vision transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12175–12185.
[49] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9759–9768.
[50] D. Du et al., "VisDrone-DET2020: The vision meets drone object detection in image challenge results," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 692–712.
[51] Y. Cao et al., "VisDrone-DET2021: The vision meets drone object detection challenge results," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 2847–2854.
[52] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 418–434.
[53] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11976–11986.

Dong Chen received the bachelor's degree information and computational science from Jinan University, Jinan, China, in 2015, and the master's degree in mechanical and electronic engineering from the Shanghai University of Applied Technology, Shanghai, China, in 2019. He is currently working toward the Ph.D degree with Tongji University, Shanghai.

He is currently with the Department of Computer Science and Technology, Tongji University, and Key Laboratory of Embedded System and Service Computing Ministry of Education as a Ph.D. Student, majoring in electronics and information. He has authored or coauthored more than four SCI papers and two EI papers in Ph.D. studying.His research interests include computer vision, artificial intelligence, deep learning and 2-D/3-D object detection. In the future, his research direction will focus on intelligent driving and 3-D object detection on mobile devices.

Duoqian Miao received bachelor's and master's degrees in mathematics from Shanxi University, Taiyuan, China, in 1985 and 1991, respectively, and the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1997.

He is currently a Professor of the College of Electronics and Information Engineering, Tongji University, Shanghai, China. He is also with the Department of Computer Science and Technology, Tongji University, and Key Laboratory of Embedded System and Service Computing Ministry of Education as the Vice Director. He has authored or coauthored about 180 scientific articles in IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON FUZZY SYSTEMS, *Pattern Recognition, Information Sciences*, and so on, and more than 100 articles have been cited by SCI or EI, eight of which are ESI papers. His research interests include artificial intelligence, machine learning, big data analysis, granular computing and rough sets, etc.

Dr. Miao is a Fellow of the International Rough Set Society (IRSS) and Chinese Association for Artificial Intelligence (CAAI).

Xuerong Zhao received the bachelor's degree in information and computational science from Jinan University, Jinan, China, in 2010, and the M.S. and Ph.D. degrees from the School of Mathematics and Statistics, Wuhan University, Wuhan, China, in 2012 and 2015, respectively.

She is currently a Lecturer of computer science and technology with Shanghai Normal University, Shanghai, China. Her research interests include three-way decision, granular computing, rough sets, formal concept analysis, and fuzzy sets.