International Journal of Advanced Engineering Research and Science (IJAERS)
Peer-Reviewed Journal
ISSN: 2349-6495(P) | 2456-1908(O)
Vol-11, Issue-12; Dec, 2024
Journal Home Page Available: https://ijaers.com/
Article DOI: https://dx.doi.org/10.22161/ijaers.1112.10

Object Detection using ELAN

Sanjivani Sharma, Dr RK Sharma

Department of Computer Science, AKTU University, Lucknow, UP, India
sanjivanisharma1161@gmail.com, rajubrains@gmail.com

Received: 27 Nov 2024,

Receive in revised form: 20 Dec 2024,

Accepted: 25 Dec 2024,

Available online: 31 Dec 2024

***Keywords*— Object detection, ODUELAN framework, deployment strategies, optimization techniques, and training approaches.**

***Abstract*—** *The ODUELAN has long served as the de facto industry standard for effective object detection. The ODUELAN community has grown significantly, enhancing its application across a wide range of hardware platforms and situations. This technical report includes, we make an uncompromising effort to advance its limits to the next level. attitude for practical use in the workplace. We carefully review the most recent advances in object detection from academia or business, taking into account the various demands for accuracy in the actual environment. We substantially include concepts from contemporary data detect, export, training approaches, testing strategies, and optimisation techniques. Additionally, we combine our ideas and experience to provide a set of deployment focus.*

I. INTRODUCTION

present object identification is a crucial component problem inside computer vision since It is frequently a crucial part of computer vision systems. Examples include robots [35, 58], autonomous driving [40, 18], and multi-object tracking [94, 93]. analysis of medical images [34, 46], etc. A variety of neural processing units (NPUs), a portable CPU or GPU, and devices made by well-known firms, are frequently used for real-time object detection. NPUs include, for instance, the Intel Neural Computing Stick, the Kernel on AI SoCs, the MediaTek AI Processing Unit, the Qualcomm Neural Processing Engine, and the Apple Neural Engine, the Google Edge TPU, Nvidia's Jetson AI Boundary Modules, and the MediaTek AI Processing Unit. Some of the edge devices concentrate on accelerating processes, such as MLP operations, depth-wise convolution, and vanilla convolution. In this research, We suggest an on-demand image detection that primarily supports GPU machines and portable GPUs from the edge of the cloud.

the previous years, present object detectors have been created for a variety of edge devices. For instance, the development The focus of MCUNet's [49, 48] and NanoDet's [54] development was on creating lightweight individual chips devices and enhancing edge CPU speed comparison. Enhancing the speed comparision of diverse GPUs is the goal of approaches like YOLOX [21] and YOLOR [81]. The development of an efficient architecture for real-time object detection has recently taken central stage. Real-time object detectors on CPUs [54, 88, 84, 83] that can be used on MobileNet [28, 66, 27, ShuffleNet [92, 55], or GhostNet [25] largely depend on their architecture. Another well-known real-time object detector for the GPU is being developed [81, 21, 97], and it primarily makes use of ResNet [26], DarkNet [63], or DLA [87], after which the architecture is optimised using the CSPNet [80] technique. The real-time object detectors used in the present mainstream are different from those used in this article. Our suggested solutions will concentrate on improving the training procedure in addition to the architecture. We'll concentrate on a few modules and optimisation techniques that may to increase object recognition accuracy, Enhancing the deduction value without doing so, the training cost has to be increased. The recommended modules and optimisation methods are referred to as "trainable bags of free stuff."

Instead of focusing on The key objective of this project is to develop a quick object detector for use in

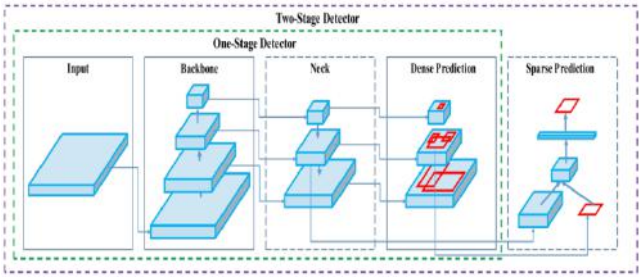
manufacturing systems and to optimise for parallel calculations, as indicated by The conceptual sign of limited processing capacity (BFLOP). We anticipate that using and training the intended item will be simple.

For instance, anybody who trains and tests on a typical GPU may provide real-time, excellent, and convincing object identification outcomes like the YOLOv4 findings seen in Figure 1. Anything we provide is summarised as follows:

1. We develop a reliable and successful System for identifying objects. Anyone may use a 1080 Ti or 2080 Ti GPU to train an extremely rapid and precise object detection.

2. During detector training, we evaluate the impact of cutting-edge Bag-of-Freebies and Bag-of-Specials item detection techniques.

3. We tweak cutting-edge techniques like CBN [89], PAN [49], SAM [85], and others to increase their efficiency and suited for training on a single GPU.



II. RELEVANT WORK

2.1 Methods to identify objects

A typical current detector has two components: One that looks like forecasts ite categories, a framework, and limits has already been equipped with Image Network. VGG [68], ResNet [26], ResNeXt [86], or DenseNet [30] may act as the main detectors when utilising GPU-based detectors. Regarding those devices that operate on CPU platforms, the backbone may be Mobile Network [28, 66, 27, 74], Squeeze Net [31], or Shuffle Net [97, 53]. The two primary varieties for the main element are one-stage detection of objects and multi-stage image analyzers. The The R-CNN [19] succession, which comprises the rapid R-CNN [18], quicker R-CNN [64], R-FCN [9], and Libra R-CNN [58] theories, is one of the most typical two-stage object sensor. Another method is to change an a two-stage image detection into a sensor for objects without links, such as RepPoints [87]. The most common choices for one-stage object detectors are YOLO [61, 62, 63], SSD [50], and RetinaNet [45]. lately anchorless one-stage object detection systems have been built. FCOS [78], CornerNet [37, 38], CenterNet [13], and CornerNet [37, 38] are a few of these monitors. between the skull and the vertebral column, Modern scanners for items generally have many layers, which are typically employed to collect Maps of features at various phases. It might be called the subject of the detect collarbone. A head often consists of a variety of top-down and bottom-up routes. The Path Aggregation Network (PAN) and the Feature Pyramid Network (FPN) are two networks that use this method [44]. Other researchers worked on the spot creating a brand-new skeleton (DetNet [43], DetNAS [7], HitDetector [20]), or a completely a fresh layout (SpineNet [12], for instance), in addition to the models already described.

In conclusion, the following elements make up an ordinary object detector:

• Input: Pyramid, Patches, and Image

• SpineNetwork[12], EfficientNetwork-B0/B7 [75], VGG16 [68], CSPResNeXt50 [81], CSPDarknet53 [81], ResNet-50 [26], and ResNet-50 [26] are the backbones.

• Neck:

SPP (25), ASPP (5), RFB (47), and SAM (85) are additional blocks.

Path-aggregation blocks include FPN [44], PAN [49], NAS-FPN [17], Full-connected FPN, BiFPN [77], ASFF [48], and SFAM [98].

Looks: RPN [64], SSD [50], YOLO [61], RetinaNet;

- Dense Prediction (one-stage):

[45] (based on anchor)

FCOS [78] (anchor free), CornerNet [37], CenterNet [13], MatrixNet [60], Two-stage

- Sparse Prediction:

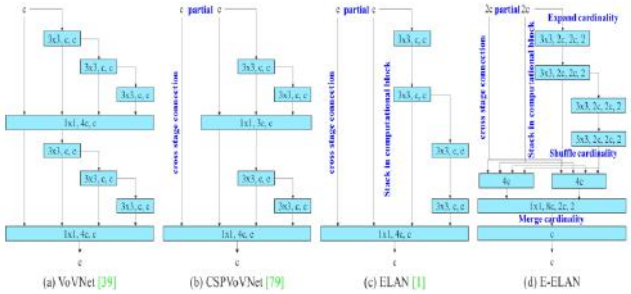
R-CNN [64], R-FCN [9], and Mask RCNN [23] (anchor based) are faster

RepPoints [87] (free anchor)

2.2 Model Re-parameterization

Several computing units are combined into one during the inference stage of Reparametrizing the equation processes [71, 31, 75, 19, 33, 11, 4, 24, 13, 12, 10, 29, 14, 78]. Model re-parameterization techniques may be divided into two groups: Groups at the module and model levels. You may consider it an ensemble method. Two popular model-level reparameterization methods are methods to arrive at the ultimate conclusion model. One approach is to average the model weights after training several the same models with various training datasets. A weighted average of model weights over different iteration counts can be calculated as an alternative. Module level re-parameterization has grown in popularity as a research

subject recently. This type of technique separates a module After instruction, modules may take similar or separate branches. A multi-branched module is consolidated into a single similar module when making a conclusion. But not every re-parameterized module that is offered can be exactly used to many architectural designs. As a result, we developed a newly created parameterization module and associated techniques for using different architectures in applications.



2.3 designing scale

designing scale allows a model to be scaled up or down to fit on different computing devices [72, 60, 74, 73, 15, 16, 2, 51]. The model scaling method typically employs a variety of In order to establish a suitable balancing of the number of connection variables, calculation, and inference efficiency and reliability, scaling factors including resolution (size of input picture), depth (number of layer), breadth (number of channel), and stage (number of feature pyramid) are used. Network architecture search (NAS) is a model scaling approach that is often used. Without creating too many difficult rules, The search space may be automatically searched by NAS for suitable scaling factors. The drawback of NAS is that finding model scaling factors needs a lot of expensive computing. The researcher makes an effort to study the connection comparing sizing variables and the number of processes and variables in [15] in order to determine the scaling variables and directly predict some laws needed by designing scale. According to our analysis of the literature, almost all model scaling strategies examine each scaling component separately, and even those that fall into the category of compound scaling also optimise each scaling factor separately. Following a study of the literature, we found that almost all scaling strategies for models examine each scaling component independently, and even compound scaling techniques also optimise each scaling factor. Considering the majority commonly employed NAS designs consider aspects of scaling that are not closely connected. We discovered that every scheme based on combining, including DenseNet [32] and VoVNet [39], would As the depth of these kinds of models changes, certain layers' input width was scaled. The preferred architecture for this model is concatenation-based, hence a new compound scaling method must be developed.

III. CONSTRUCTION

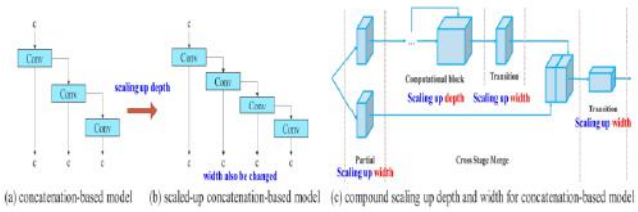
3.1 Extended efficient layer aggregation network

Just The key factors taken into account in the bulk of studies on developing effective platforms are the quantity of variables, the volume of the process, and the level of computation. Using the characteristics of memory access cost as a starting point,

The architecture of the transition layer is unaffected by ODUELAN; only the design of the computational block is changed. We propose the group distortion to be used to extend the commonality and stream of the calculation elements. The exact same stream factor and grouping attribute will be applied to each processing block in a processing layer. The feature chart generated by each processing block will then be concatenated after being split into collections based on the designated g group variable g. Currently, how many channels are incorporated in each set of features maps will be the same as it was in the primary building. at last, by including g feature map collections, we combine the a cardinality Keeping the initial ELAN project design in mind, E-ELAN has the ability to instruct other steps of processing together to offer new, more diversified functionality.

3.2 Model scalling for concatenation-based model

Model scaling is typically used to change certain model characteristics and create models of different sizes to support different inference rates. The Efficient Net scaling model, for instance, [72] takes into account the extent, breadth, and clarity. The scaling model's goal with respect to the scaled-YOLOv4 is to alter the amount of phases [79]. Dollar' et al. [15] investigated the impacts of both group and plain convolution on the number of parameters and calculations while increasing in both size and dimension, and they used the findings to construct the suitable designing scale approach.



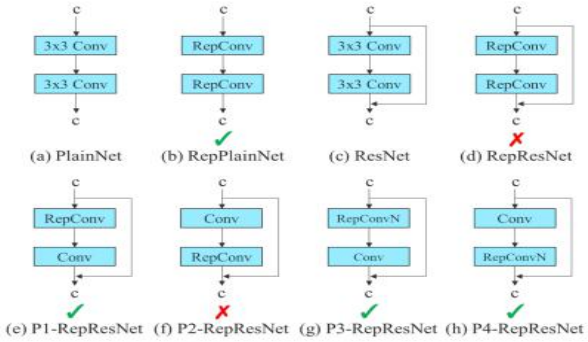
The aforementioned occurrence shows that we are unable to study various scaling variables individually for a model

using combination, but rather that they must be taken into consideration jointly. For instance, enlarged thickness will alter the ratio between a The middle layer input and output channels, which can lower the model's hardware needs. Consequently, we must present the appropriate formula for scaling a complicated model for a model using combination. Scaling the an algorithmic block's depth factor requires calculating the modification to that block's export stream. The outcome is displayed in Figure 3(c). after applying the same amount of modification to the transition layers' width factor scaling. We propose a compound scaling technique that preserves the model's ideal structure and its original properties.

IV. TRAINABLE BAG-OF-FREEBIES

4.1.Re-parameterized generation is anticipated

RepConv [13] has done well on the VGG [68], but the precision will be greatly diminished when it is used straight to ResNet [26], DenseNet [32], and other models. We look into the transmission of gradient flows. channels should be used to combine re-parameterized convolution with different networks. In addition, Cordingly, our anticipated redesigned convolution, was constructed.



Identity connection, 3 3 convolution, and 1 1 convolution are all included in the convolutional layer known as RepConv. The link to identification in RepConv eliminates the concatenation in DenseNet and the residual in ResNet, resulting in a broader diversity of gradients for various feature maps, according to our analysis of RepConv's performance when used in conjunction with other architectures. For these factors, we create the proposed re-parameterized convolution's structure using RepConv without same relationship (RepConvN). In our perspective, There shouldn't be a unique link when a re-parameterized layer of convolution replaces a layer of convolution with residual or mixture activation. Figure 4 shows a plainnet and resnet version of our "planned re-parameterized inversion". A re-parameterized The convergence research using concatenation- and residual-based method models will be described in relation to the overall goal of the ablation research session.

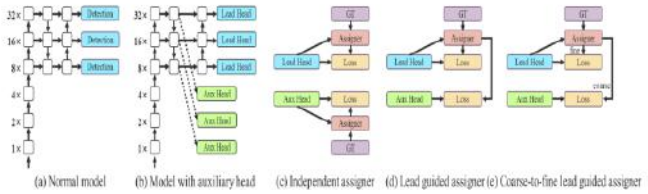


Figure 5: Coarse for auxiliary and fine for lead head label assigner. Compare with normal model (a), the schema in (b) has auxiliary head. Different from the usual independent label assigner (c), we propose (d) lead head guided label assigner and (e) coarse-to-fine lead head guided label assigner. The proposed label assigner is optimized by lead head prediction and the ground truth to get the labels of training lead head and auxiliary head at the same time. The detailed coarse-to-fine implementation method and constraint design details will be elaborated in Apendix.

4.2 Fine for lead loss and coarse for auxiliary

Deep supervision is a technique that is widely used while training deep networks [38]. The assistance loss acts as the direction for the shallow network weights, and its basic notion is to supply more auxiliary heads to the network's intermediate tiers. Despite this, extensive oversight [70, 98, 67, 47, 82, 65, 86, 50] may dramatically enhance the model's the efficiency of various jobs, even for often convergent architectures like ResNet [26] and DenseNet [32]. Figure 5(a) and (b), which Demonstrate the subject's sensor structure both "without" and "along" vigorous oversight, respectively. The head in this investigation was that generates the final output is referred to as the lead head, while the head that assists in training is referred to as the Additional head.

Deep supervision must be concentrated on the targeted objectives whether the lead head is an auxiliary head or vice versa. We unintentionally encountered fresh variant problem, namely "How to assign gentle labels to the main lead and secondary heads," when exploring strategies related to soft label assigners. For the extent that we are aware, the relevant There isn't yet literature on this subject. Figure 5(c) shows the results of the currently most popular method, which separates the lead head and auxiliary head before utilising their own forecast findings and the basic truth about carry out labels are assigned. This study proposes a novel label assignment strategy that employs lead head prediction to guide both the lead head and the auxiliary head. To put it another way, we employ lead column prediction. to direct the development of coarse-to-fine layered categories to learn in the lead head and support heads. The two recommended allocation of the deep monitoring label methods are shown in Figures 5(d) and (e), accordingly.

prediction The primary head directed term assigner uses the actual fact and the leader head's predictions outputs as its two main inputs. which then uses optimisation to produce soft labels. This collection of soft labels will

serve as the lead head and auxiliary head's target training model. Since leader head has a moderate amount of acquiring capacity, the neutral label that emerges from it should be better in detecting the variation and connection between the source and target inputs. We may also think of this studying while a form of generalised residual learning. By allowing the less experienced assistant head to pay attention to the concepts that the head of leadership acquired, the a larger lead neck capable to focus on studying remaining knowledge that hasn't yet been taught.

The identify assigner with a coarse-to-fine lead point is utilised. both the ground truth and the point head's anticipated outcome to create soft labels. However, during the method, we create two separate sets of soft labels: coarse label and fine label, which are identical to the soft labels created by the lead head guided label assigner. Loosening the limitations on the positive sample assignment technique allows for the production of coarse labels by allowing more grids to be seen as positive targets. This is because an auxiliary head's learning capacity is lower than a point head's, thus in order to prevent losing the knowledge that has to be kept, we will focus on improving the auxiliary head's recall. As the resultant work for the lead head results, we may separate the high accuracy results from the high recall outcomes. A subpar prior might be produced by the final forecast. It is crucial to pay attention if the coarse label's increased weight is close to that of the fine label. In order to avoid good squares that are exceptionally sharp from creating excellent smooth label we placed restrictions on the decoder in order to lessen their impact. Through the use of the method previously stated, it is feasible to dynamically alter the relative weights of fine and coarse labels throughout the learning process, and it is also guaranteed that fine labels have a higher optimizable upper bound more coarse labels.

4.3 Various trainable bags-of-freebies

We shall mention a few trainable bag-offreebies in this area. We utilised several of these freebies in our training, but we didn't come up with the original ideas. The Appendix will elaborate on the training specifics for these bonuses, including: (1) Batch normalisation in the topology of conv-bn-activation: This section mostly joins the convolutional layer and batch normalisation layer. This integrates the leaning and frequency of the convolutional layer created during the deduction stage with the average and deviation of the whole batch normalisation. (1) Pre-computing at the judgement step in YOLOR allows hidden data to be transformed into a vector. (2) Convolution feature map multiplied by implicit knowledge in YOLOR [81]. The resulting vector may be coupled with the convolution layer's skew and intensity that comes before or after. EMA model, third: Only the EMA model is used as the final inference model in our system. Mean teachers employ the EMA approach [75].

Table 1: Comparison of baseline object detectors.

Model	#Param.	FLOPs	Size	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv4 [3]	64.4M	142.8G	640	49.7%	68.2%	54.3%	32.9%	54.8%	63.7%
YOLOR-u5 (r6.1) [81]	46.5M	109.1G	640	50.2%	68.7%	54.6%	33.2%	55.5%	63.7%
YOLOv4-CSP [79]	52.9M	120.4G	640	50.3%	68.6%	54.9%	34.2%	55.6%	65.1%
YOLOR-CSP [81]	52.9M	120.4G	640	50.8%	69.5%	55.3%	33.7%	56.0%	65.4%
YOLOv7	36.9M	104.7G	640	**51.2%**	**69.7%**	**55.5%**	**35.2%**	**56.0%**	**66.7%**
improvement	-43%	-15%	-	+0.4	+0.2	+0.2	+1.5	=	+1.3
YOLOR-CSP-X [81]	96.9M	226.8G	640	52.7%	**71.3%**	57.4%	36.3%	57.5%	68.3%
YOLOv7-X	71.3M	189.9G	640	**52.9%**	71.1%	**57.5%**	**36.9%**	**57.7%**	**68.6%**
improvement	-36%	-19%	-	+0.2	-0.2	+0.1	+0.6	+0.2	+0.3
YOLOv4-tiny [79]	6.1	6.9	416	24.9%	42.1%	25.7%	8.7%	28.4%	39.2%
YOLOv7-tiny	6.2	5.8	416	**35.2%**	**52.8%**	**37.3%**	**15.7%**	**38.0%**	**53.4%**
improvement	+2%	-19%	-	+10.3	+10.7	+11.6	+7.0	+9.6	+14.2
YOLOv4-tiny-3l [79]	8.7	5.2	320	30.8%	47.3%	32.2%	**10.9%**	31.9%	51.5%
YOLOv7-tiny	6.2	3.5	320	**30.8%**	**47.3%**	**32.2%**	10.0%	**31.9%**	**52.2%**
improvement	-39%	-49%	-	=	=	=	-0.9	=	+0.7
YOLOR-E6 [81]	115.8M	683.2G	1280	55.7%	73.2%	60.7%	40.1%	**60.4%**	69.2%
YOLOv7-E6	97.2M	515.2G	1280	**55.9%**	**73.5%**	**61.1%**	**40.6%**	60.3%	**70.0%**
improvement	-19%	-33%	-	+0.2	+0.3	+0.4	+0.5	-0.1	+0.8
YOLOR-D6 [81]	151.7M	935.6G	1280	56.1%	73.9%	61.2%	**42.4%**	60.5%	69.9%
YOLOv7-D6	154.7M	806.8G	1280	56.3%	73.8%	61.4%	41.3%	60.6%	70.1%
YOLOv7-E6E	151.7M	843.2G	1280	**56.8%**	**74.4%**	**62.1%**	40.8%	**62.1%**	**70.6%**
improvement	=	-11%	-	+0.7	+0.5	+0.9	-1.6	+1.6	+0.7

V. RESULT

A comparison of the results using several cutting-edge sensors for objects is show in Figure. Our ODUELAN on the Pare superiority test structure and are quicker and more accurate than The most rapid and precise sensors.

As several approaches employ GPUs with different designs to verify interpretation at runtime, we run ODUELAN on popular GPUs of the Designs of Maxwell, Pascal, and Volta are compared. it to other cutting-edge methodologies. Table 8 displays the frame rate comparison findings using either the Tesla M40 GPU or the GTX Titan X (Maxwell) GPU. The results of the Pascal GPU frame-per-second compare are displayed in Table 9 and include the TitanX (Pascal), TitanXp, GTX 1080 Ti, and Tesla P100 GPUs. Frame rates using a Volta GPU, which might be a Titan Volta or a Tesla Vol100 GPU, are compared in Table 10.

VI. CONCLUSION

We give a brand-new actual time element in this study identification architecture together with a model scaling method. We also find that new research ideas are generated by the process of building object identification systems. During the course of the investigation, we identified the replacement issue for the re-parameterized course and the assignment challenge for the allocating labels automatically. To address the issue and increase item identification accuracy, we recommend using the

trainable bag-of-gifts approach. We developed the cutting-edge ODUELAN of item detecting systems depending on the preceding provided information.

ACKNOWLEDGEMENTS

The National Centre for High-performance Computing (NCHC), which provided the computational and storage resources, is acknowledged by the authors.

REFERENCES

[1] Irwan Bello, William Fedus, Xianzhi Du, EkinDogusCubuk, Aravind Srinivas, Tsung-Yi Lin, Jonathon Shlens, and Barret Zoph. Revisiting ResNets: Improved training and scaling strategies. Advances in Neural Information Processing Systems (NeurIPS), 34, 2021. 2

[2] Alexey Bochkovskiy, Chien-Yao Wang, and HongYuan Mark Liao. YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020. 2, 6, 7

[3] Yue Cao, Thomas Andrew Geddes, Jean Yee Hwa Yang, and Pengyi Yang. Ensemble deep learning in bioinformatics. Nature Machine Intelligence, 2(9):500–508, 2020. 2

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision (ECCV), pages 213–229, 2020. 10

[5] Kean Chen, Weiyao Lin, Jianguo Li, John See, Ji Wang, and Junni Zou. AP-loss for accurate one-stage object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 43(11):3782–3798, 2020. 2

[6] Zhe Chen, YuchenDuan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. arXiv preprint arXiv:2205.08534, 2022. 10

[7] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 502–511, 2019. 5

[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 (CVPR), pages 7373–7382, 2021. 2

[9] Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Kaiqi Huang, Jungong Han, and Guiguang Ding. Reparameterizing your optimizers rather than architectures. arXiv preprint arXiv:2205.15242, 2022. 2

[10] Xiaohan Ding, YuchenGuo, Guiguang Ding, and Jungong Han. ACNet: Strengthening the kernel skeletons for powerful CNN via asymmetric convolution blocks. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 1911–1920, 2019. 2

[11] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse branch block: Building a convolution as an inception-like unit. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 10886–10895, 2021. 2

[12] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. RepVGG: Making VGG-style convnets great again. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13733–13742, 2021. 2, 4

[13] Xiaohan Ding, Xiangyu Zhang, Yizhuang Zhou, Jungong Han, Guiguang Ding, and Jian Sun. Scaling up your kernels to 31x31: Revisiting large kernel design in CNNs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 2

[14] Piotr Dollar, Mannat Singh, and Ross Girshick. Fast and ´ accurate model scaling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 924–932, 2021. 2, 3

[15] Xianzhi Du, Barret Zoph, Wei-Chih Hung, and Tsung-Yi Lin. Simple training strategies and model scaling for object detection. arXiv preprint arXiv:2107.00057, 2021. 2

[16] Chengjian Feng, YujieZhong, Yu Gao, Matthew R Scott, and Weilin Huang. TOOD: Task-aligned one-stage object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 3490–3499, 2021. 2, 5

[17] Di Feng, Christian Haase-Schutz, Lars Rosenbaum, Heinz ¨ Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. IEEE Transactions on Intelligent Transportation Systems, 22(3):1341– 1360, 2020. 1

[18] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of DNNs. Advances in Neural Information Processing Systems (NeurIPS), 31, 2018. 2

[19] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. OTA: Optimal transport assignment for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 303–312, 2021. 2, 5