Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230892

Light-weight YOLO based object detection algorithm for unmanned aerial vehicle

Disen Hu

University of Bristol, Queens Road, Bristol, BS8 1TH, United Kingdom

disen.hu@outlook.com

Abstract. Object detection algorithms based on deep learning usually have good results in terms of speed and accuracy on GPU- based computing platforms. However, as this kind of algorithm is not perfectly supported for CPU-based Unmanned aerial vehicle(UAV), the object detection algorithm usually used in UAV has the problem of slow detection speed, which will lead to traffic accidents, traffic congestion, and other problems. To solve this problem, we proposed an object detection algorithm based on YOLOv5. Firstly, aiming at the problem of light- weight model architecture, mobilenetv3 was added to YOLOv5 to replace the original backbone. Secondly, in order to maintain a high detection accuracy, omni-dimensional dynamic convolution was added in the feature fusion part of the network as a replacement for stander convolution. Through the architecture analysis, the proposed algorithm solves the problem in the UAV traffic monitoring system.

Keywords: mobilenetv3, omni-dimensional dynamic convolution, traffic monitoring system, UAV, YOLO

1. Introduction

Due to the transportation sector's fast development in recent years, cars have been the primary tool of modern transportation, and car sales and ownership are also increasing yearly. According to hedges\&companies, the world now has a staggering 14.46 billion cars on roads by 2022 [1]. Cars greatly facilitate people's travel efficiency and improve production efficiency in People's Daily life. However, with the increase in the frequency of people using cars, it is accompanied by many problems. These problems are traffic management problems, such as traffic jams, frequent traffic accidents, etc [2]. Therefore, solving the negative impact of cars through more efficient traffic management is a significant problem facing society.

In order to solve the problems in traffic management, the traditional traffic management system mainly uses fixed cameras. It is laborious to perform management tasks, but this is time-consuming. Recently, state-of-the-art research has proposed an intelligent traffic management system composed of Unmanned Aerial Vehicles (UAVs) as an alternative to traditional traffic management systems [3]. This UAV traffic management system is not limited by the traditional fixed camera coverage is small, the need to install a stable power supply and network on the highway, etc. It realizes intelligent and flexible traffic management. The basis of this traffic monitoring system is the object detection algorithm based on deep learning. Currently, the most advanced object detection algorithms are divided into single-stage and two-stage target detection algorithms [4]. The advantage of the single-

© 2023 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/6/20230892

stage algorithm lies in the realization of the end-to-end model, which is usually several times faster than the two-stage image detection speed. Among the single-stage algorithms, the YOLO series algorithm [5-7] is representative of the single-stage algorithm due to its ultra-fast detection speed and detection accuracy no less than that of the two-stage algorithm. Although YOLO has achieved a fast detection speed on most GPU-based computing platforms, it cannot achieve a good detection speed on CPU-based UAVs due to its design and optimization based on GPU platform [8]. However, since most of the UAVs used in the UAV traffic management system use CPU to detect vehicles on the road, there is still the most advanced target detection algorithm cannot achieve the satisfactory effect of both accuracy and speed [9-10], which will make the UAV traffic management system respond slowly and cause danger to road traffic.

In this paper, in order to improve the object detection algorithm's inference speed applied on CPU-based UAV, we propose an improved object detection algorithm based on YOLO [5-7]. Firstly, mobilenetv3 [11-13]} is added to the algorithm to replace the backbone, which can significantly improve the detection speed. Then we use Omni-dimensional dynamic convolution [14] to replace the stander convolution in the last layer of the YOLO head to increase the feature extraction capability.

The rest of this paper is organized as follows. In section 2, related works are introduced. In section 3, the proposed architecture is realized. Section 4, there is the analysis of the proposed architecture. The conclusion is in section 5.

2. Related works

2.1. YOLOv5

YOLOv5 is the most advanced object detection network, and it is the product of integration and innovation on yolov3 [7], and yolov4 [15]. The second, YOLOv5 has achieved good results in PASCAL VOC [16] and Common Objects in Context (COCO) [17] object detection task, so this paper adopts YOLOv5 object detection network for UAVs object detection algorithm.

There are four editions of YOLOv5's official implementations: YOLOv5s, YOLOv5m, YOLOv51, and YOLOv5x. The three algorithms YOLOv5m, YOLOv51, and YOLOv5x are the results of deepening and broadening on the basis of YOLOv5s, which has the network with the lowest feature map width and the most minor depth in the YOLOv5 series.

YOLOv5 network structure is divided into input, Backbone, Neck, and Prediction. YOLOv5 enhanced Mosaic data in the data input part. Backbone mainly adopts the Focus structure and CSP (Cross Stage Partial network) structure. The FPN+PAN (Path Aggregation Network) structure was added to the Neck. In Prediction, Generalized Intersection over Union (GIoU) loss is used to construct the loss function of the border anchor box in place of Complete Intersection ratio over Union (CIoU) loss. Weighted Non-Maximum Suppression (NMS) operation is used by YOLOv5 in the target detection backward propagation to filter numerous target anchor boxes.

2.2. Light-weight network architecture

Since Alexnet was proposed in 2012 by A. Krizhevsky [18], the Convolutional neural network has been widely used in tasks of object detection, image classification, and instance segmentation. However, as the performance requirements of the model are becoming higher in practical application, the demand for light-weight models is also increasing, and many light-weight models have been proposed accordingly.

For example, light-weight models SqueezeNet [19], shufflenet [20], and Xception were proposed in 2016 [21],2017 and 2017, respectively. These models were designed to solve two problems in the model application. The first is the problem of model storage; hundreds of layers of networks contain a large number of weight parameters, and it is very difficult to save a large number of weight parameters to the satisfied device's random access memory (RAM). The second problem is the reasoning speed of the model. In practical application, it is often millisecond level. In order to meet the

DOI: 10.54254/2755-2721/6/20230892

practical application schema, the processor performance should be improved, or the computation amount should be reduced.

Mobilenet is state of the art light-weight network. Mobilenet is available in versions v1, v2, and v3. Mobilenetv1 divides traditional convolution into deep convolution and 1 × 1 point convolution through depth separable calculation [11-13]. In addition, width multiplication and resolution multiplication are introduced to control the number of model parameters, reducing the number of network parameters and improving CPU devices' computing speed. Mobilenetv2 implements the inverted residual structure to solve the problem of disappearing gradients during feature extraction. In Mobilenetv3, the inversed residual block, linear bottleneck, and squeeze-and-excitation (SE) attention mechanism from Mobilenetv2 are combined with the depth-separable convolution from Mobilenetv1. In addition, to search the network's setup and parameters, a neural architecture search (NAS) is performed. In order to decrease the amount of computation and obtain the effects of less calculation and greater accuracy, the swish activation function is enhanced to h-swish.

2.3. Detection precision improvement of convolution neural network

A deep convolution neural network is used to extract image features for image recognition. Convolution kernel is usually used to extract features of a small receptive field of the feature map. However, the information on each regional feature has different influences on whether the detection network can correctly recognize the image. Therefore, the attention mechanism, as a method to improve the detection accuracy, simplify the model, and accelerate the calculation [22], has been used in many computer vision tasks in recent years. Since Squeeze and Excitation Networks [23] published on CVPR in 2018, it is a representative application of attention mechanisms in computer vision. A lot of follow-up work based on this work has been generated. For example, Selective Kernel Networks [24], CBAM: Convolutional Block Attention Module etc [25]. The focus of these efforts is on adding attention mechanisms to feature maps in the feature extraction process. In addition, dynamic convolution is also widely used in computer vision as an efficient method to add attention to the convolution kernel in the process of feature extraction [26]. Omni-dimensional dynamic (ODConv) [14] was proposed in 2022. As an enhancement of dynamic convolution, Omni-dimensional dynamic convolution (ODConv) can be regarded as the continuation of dynamic convolution, which extends the dynamic characteristics in one dimension of dynamic convolution, and considers the dynamics in the spatial domain, input channel, output channel, and other dimensions, so it is called omni-dimensional dynamic convolution. ODConv uses a multi-dimensional attention mechanism to learn complementary attention along the four dimensions of the kernel space through the parallel strategy. It can be easily embedded into existing CNN networks. Experiments on ImageNet classification and COCO detection tasks verify the superiority of the proposed ODConv: it can improve the performance of large models and light-weight models [14]. Due to its improved feature extraction ability, ODConv with one convolution kernel can still achieve comparable or even better performance than the existing multi-kernel dynamic convolution.

3. Improved model architecture

Based on the YOLOv5 object detection algorithm, this paper replaced the original backbone network by modifying the backbone network framework and using the feature extraction network of mobilenetv3. Furthermore, omni-dimensional dynamic convolution is used to introduce an attention mechanism in the detection layer, thus improving the precision of target detection. The mobilenetv3 backbone network is used to implement the migration of the algorithm to the mobile terminal and improve the algorithm's target detection speed on the UAV device. As shown in Fig. 1, the proposed model is described in the following.

DOI: 10.54254/2755-2721/6/20230892

3.1. Input end

3.1.1. Mosaic data augmentation. In the input end of the network, the Mosaic data augmentation method was used to process on data set. In this method, four images are randomly cropped and then spliced into one image as training data. This policy's advantage is that the images' background is enriched, and the batch size is increased when the four images are used together. In the process of batch normalization, four images are calculated. Therefore, this method does not rely much on the computing resources of the training data set.

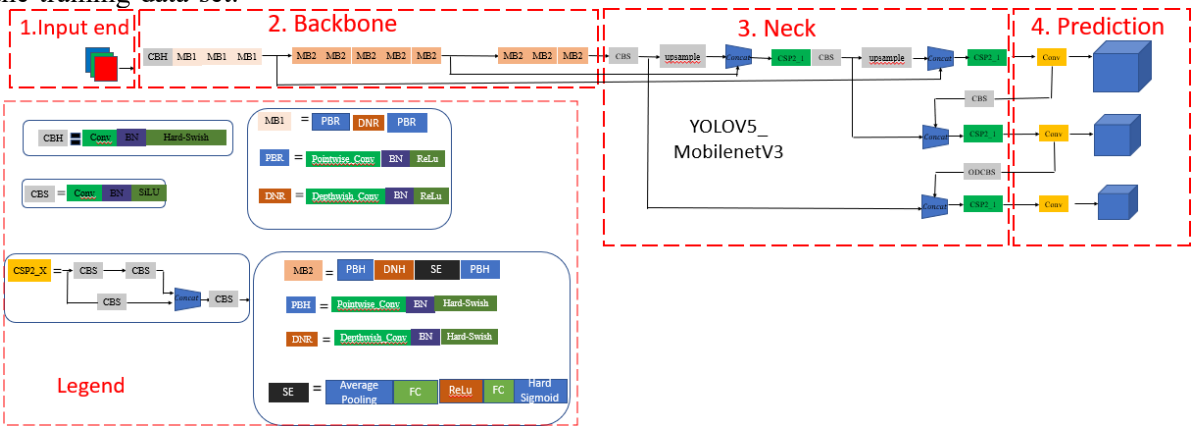

Figure 1. Light-weight YOLO.

3.1.2. Adaptive picture scaling. Various pictures have different lengths and widths when using standard target detection techniques. The conventional approach is to feed the original picture to the detection network after evenly scaling it to a standard size.

This technique involves adding a few light black borders to the source image. The amount of the black edge at both ends will vary after scaling and filling since many of the real images used with this approach have varied aspect ratios. However, if there is more filling, information redundancy will occur, which will slow down reasoning.

3.2. Backbone

The proposed model uses the bottleneck structure from mobilenetv3 Fig.2 to replace the original backbone Darknet53 in YOLOv5.

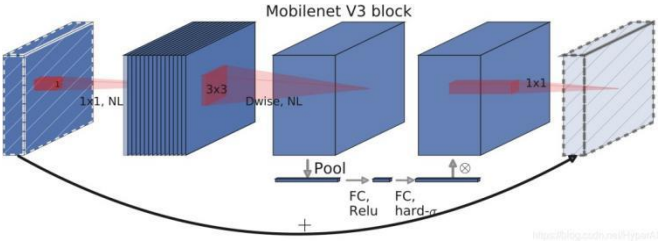

Figure 2. Mobiklenet V3 block.

To reduce the number of parameters in the network structure, mobilenet's main structure is adopted as the core convolutional layer of the backbone network. In comparison to the YOLOv5 backbone, it not only keeps a more reliable feature extraction while also drastically shrinking the size of the model and improving the ease of deployment in the mobile terminal. It also has a shallower network than the YOLOv5 network, which may better extract regionally specific fine characteristics and enhance detection performance on busy roadways. Therefore, mobilenetv3 replaces YOLOv5's backbone as it will spend less to extract features. Fig. 2 shows the Architecture of mobilenetv3. In the proposed model, we used a total of 9 bottleneck structures to replace the original backbone.

DOI: 10.54254/2755-2721/6/20230892

3.2.1. bottleneck structure. The bottleneck structure is com- posed of the convolution layer and the residual edge part. The central part first raises the dimension through 1×1 point-wise convolution to expand the number of channels in the input feature layer. Then, feature extraction is carried out through a 3×3 depth-wise convolution. The proportion occupied by each channel number is obtained by a global average pooling and two full connections of the obtained feature layer. Then the channel weight is adjusted to apply the attention mechanism. Finally, the dimension is reduced by 1×1 convolution, resulting in output. In the residual edge part, the input and output are directly connected to accomplish.

3.2.2. activation function. Mobilenetv3 employs the conventional ReLU function as the activation function in the top three layers of the network when applying the activation function. In the remaining six bottleneck structures, it employs the more efficient and effective h-swish activation function, reducing computation for the entire structure and enhancing performance.

3.3. Neck

The neck's purpose is to improve feature fusion and network feature extraction. PANet is adopted as the neck of this algorithm. Utilizing downsampling and upsampling techniques, PANet, which is based on feature pyramid networks (FPN), fuses feature maps of various scales at the same time, enhancing the features of the output layer after mapping and fusion and enhancing the network's capacity to express both shallow feature information and deep semantic information.

To improve the ability of feature detection, we introduce the attention mechanism on the last layer by adding ODConv in PANet to achieve better feature extraction. The ODConv can be defined as

$$y = (\alpha_{w1} \odot \alpha_{f1} \odot \alpha_{c1} \odot \alpha_{s1} \odot W_1 + \cdots + \alpha_{wi} \odot \alpha_{fi} \odot \alpha_{ci} \odot \alpha_{si} \odot W_i) * x \quad (1)$$

In equation (1), where $x \in R^{h \times w \times c_{in}}$ and $y \in R^{h \times w \times c_{out}}$ represent the input features and out features. w_i is the ith convolution kernel. The four attention parameters α_{wi}, α_{fi}, α_{ci} and α_{si}, represent the attention scalars of the convolution kernel, the space dimension of the convolution kernel, the input channel dimension and the output channel dimension respectively.

3.4. Prediction part

In the proposed model, CIOU_Loss is set as the loss function of the bounding box as equation (2)

$$CIoU = IoU - \frac{D_2^2}{D_C^2} - \alpha v \quad (2)$$

where α shows in equation (2) is an equilibrium parameter and does not participate in gradient calculation. The expression of α is as

$$\alpha = \frac{v}{(1 - IoU) + v} \quad (3)$$

The calculation process of parameter v is as

$$v = \frac{4}{\pi^2}(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h})^2 \quad (4)$$

where h^{gt} and w^{gt} in equation (4) are the height and width of the ground- truth, respectively. h and w are the height and width of the prediction result, respectively.

4. Model analysis

The main aim of this section is to demonstrate and analyze the performance between YOLOv5 and the proposed architecture on CPU-based UAV devices. In this section, we analyzed our proposed architecture from two aspects of algorithm's time complexity and space complexity, respectively. In addition, detection result in other experiment is analyzed.

DOI: 10.54254/2755-2721/6/20230892

4.1. Time complexity and space complexity

The time complexity of the proposed model was evaluated by floating point operations (FLOPs). The mathematical definition of FLOPs can be described as following

$$FLOPs = [(C_i \times k_w \times k_h) + (C_i \times k_w \times k_h - 1) + 1] \times C_0 \times W \times H \quad (5)$$

In the equation (5), where the value in middle brackets represents the amount of computation (multiplication and addition) required by the convolution operation to calculate a point in the feature map; $C_i \times k_w \times k_h$ represents the amount of multiplication in a convolution operation. $(C_i \times k_w \times k_h - 1)$ represents the amount of addition in a convolution operation. +1 represents bias. W and H represent the length and width of the feature map, respectively. $C_o \times W \times H$ represents the number of all elements of the feature map.

The space complexity of the proposed model was evaluated by the number of model parameters.

$$params = C_o \times (k_w \times k_h \times C_i + 1) \quad (5)$$

Where C_o denotes the number of output channels, C_i denotes the number of input channels, k_w denotes the width of the convolution kernel, and $k_h$denotes the height of the convolution kernel. The value in $(k_w \times k_h \times C_i + 1)$ parentheses represent the number of weights in a convolution kernel, +1 represents bias, parentheses represent the number of parameters in a convolution kernel, and C_0 indicates that there are 0 convolution cores in this layer.

Table 1. FLOPs.

YOLOv5			light-weight YOLO		
Name	GFOLPs	parameters	Name	GFLOPs	parameters
Conv	1.47	73984	conv bn hswish	0.1	464
Conv	3.80	156928	MobileNet Block	0.03	612
C3	8.10	295424	MobileNet Block	0.11	3864
Conv	3.79	1118208	MobileNet Block	0.07	5416
C3	14.37	1180672	MobileNet Block	0.06	13736
Conv	3.78	6433792	MobileNet Block	0.09	55340
C3	20.62	4720640	MobileNet Block	0.09	21486
Conv	3.78	9971712	MobileNet Block	0.05	28644
C3	7.99	2624512	MobileNet Block	0.06	91848
SPPF	2.10	525312	MobileNet Block	0.08	91848
Conv	0.42	0.00	MobileNet Block	0.10	294096
Upsample	0.00	0.00	MobileNet Block	0.10	294096
Concat	0.00	2757632	Conv	0.02	25088
C3	8.84	131584	Upsample	0.00	0.00
Conv	0.42	0.00	Concat	0.00	0.00
Upsample	0.00	0.00	C3	0.99	308736
Concat	0.00	690688	Conv	0.11	33024
C3	8.87	590336	Upsample	0.00	0.00
Conv	1.89	0.00	Concat	0.00	0.00
Concat	0.00	2495488	C3	1.00	77568
C3	8.87	2360320	Conv	0.47	147712
Conv	1.89	0.00	Concat	0.00	0.00
Concat	0.00	9971712	C3	0.95	296448
C3	7.99	457725	ODConv	0.00	603353
Prediction	1.46	46563709	Concat	0.00	0.00
Total	109.6	73984	C3	0.95	1182720
/	/		Prediction	0.22	67425
/	/		Total	6.0	3607016

DOI: 10.54254/2755-2721/6/20230892

The table 1 shows the FLOPs and parameters in each layer of the two network models when the input image size is the size. It is shown that the FLOPs of YOLOv5 are about 19 times that of the proposed architecture, which proves that the proposed model has a massive advantage over the original algorithm in time complexity. And it is also shown that the number of parameters of YOLOv5 is about 13 times that of the proposed model, which proves that the proposed model has a huge advantage over the original algorithm in space complexity.

4.2. Model precision analysis

In order to analyze the impact of the mobilenet replacement backbone on detection accuracy, we analyzed it by referring to the model implemented in this paper [27]. In the [27], mobilenetv1 was used to replace the backbone of the proposed model, which is established based on YOLOv4. The experiments and tests were carried out on the VOC data set. The experimental results show that compared with the YOLOv4 algorithm, the map50 of the proposed model decreases from 92.98% to 86.48% when using map50 as the test criteria.

This is a negligible drop in accuracy. Moreover, in order to analyze the precision impact of ODConv implemented in our proposed model, we refer to the experiment in paper [14]. In the experiment of this paper [14], mobilenetV2 and FasterRcnn were used as the baseline of the experiment. The test result of this experiment shows that When using an ODConv to replace a stander convolution at the last layer of the backbone, the map50 test accuracy was increased from 57.2% to 60.7%.

5. Conclusion

In this paper, there are two improvements have been achieved. The first is the light-weight construction of the model, which reduces the time complexity and space complexity of the model by 19 times and 13 times respectively, compared with the original model. Theoretically, the algorithm inference speed can be 19 times faster for the UAV equipment running based on the CPU. Secondly, by adding an attention mechanism to the model, the detection precision of the proposed architecture will remain relatively high when the proposed architecture has been light-weighted at the same time. By achieving these two goals of architecture improvement, the object detection performance of CPU-based UAV is greatly improved, while the accuracy of the model is not significantly reduced.

References

[1] “Vehicles in use. — International Organization of Motor Vehicle Man- ufacturers” Scribbr. https://www.oica.net/category/vehicles-in-use/ (ac- cessed Nov. 12, 2022).

[2] Q. Quan, K. Cai, ‘Low altitude UAV traffic management:An introductory overview and proposal’, Acta Aeronautica et Astronautica Sinica, vol. 41, no. 1, p. 23238, 2020, doi: 10.7527/S1000-6893.2019.23238.

[3] A. Alioua, H. Djeghri, M. E. T. Cherif, S.-M. Senouci, and H. Sed- jelmaci, ‘UAVs for traffic monitoring: A sequential game-based com- putation offloading/sharing approach’, Computer Networks, vol. 177, p. 107273, 2020, doi: https://doi.org/10.1016/j.comnet.2020.107273.

[4] Z.-Q. Zhao, P. Zheng, S. Xu, and X. Wu, ‘Object detection with deep learning: A review’, IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.

[5] J. Redmon and A. Farhadi, ‘YOLOv3: An Incremental Improvement’. arXiv, Apr. 08, 2018. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/1804.02767

[6] J. Redmon and A. Farhadi, ‘YOLO9000: Better, Faster, Stronger’. arXiv, Dec. 25, 2016. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/1612.08242

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ‘You Only Look Once: Unified, Real-Time Object Detection’. arXiv, May 09, 2016. Accessed: Nov. 12, 2022.

DOI: 10.54254/2755-2721/6/20230892

[Online]. Available: http://arxiv.org/abs/1506.02640

[8] Y. E. Wang, G.-Y. Wei, and D. Brooks, ‘Benchmarking TPU, GPU, and CPU Platforms for Deep Learning’. arXiv, Oct. 22, 2019. Accessed: Nov. 19, 2022. [Online]. Available: http://arxiv.org/abs/1907.10701

[9] F. Outay, H. A. Mengash, and M. Adnan, ‘Applications of unmanned aerial vehicle (UAV) in road safety, traffic and highway infrastructure management: Recent advances and challenges’, Transportation Research Part A: Policy and Practice, vol. 141, pp. 116–129, Nov. 2020, doi: 10.1016/j.tra.2020.09.018.

[10] N. A. Khan, N. Z. Jhanjhi, S. N. Brohi, R. S. A. Usmani, and A. Nayyar, ‘Smart traffic monitoring system using Unmanned Aerial Vehicles (UAVs)’, Computer Communications, vol. 157, pp. 434–443, May 2020, doi: 10.1016/j.comcom.2020.04.049.

[11] A. Howard et al., ‘Searching for MobileNetV3’. arXiv, Nov. 20, 2019. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/1905.02244

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, ‘MobileNetV2: Inverted Residuals and Linear Bottlenecks’. arXiv, Mar. 21, 2019. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/1801.04381

[13] A. G. Howard et al., ‘MobileNets: Efficient Convolutional Neural Net- works for Mobile Vision Applications’. arXiv, Apr. 16, 2017. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/1704.04861

[14] C. Li, A. Zhou, and A. Yao, ‘Omni-Dimensional Dynamic Convolution’. arXiv, Sep. 16, 2022. Accessed: Nov. 12, 2022. [Online]. Available: http://arxiv.org/abs/2209.07947

[15] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ‘YOLOv4: Optimal Speed and Accuracy of Object Detection’. arXiv, Apr. 22, 2020. Accessed: Nov. 13, 2022. [Online]. Available: http://arxiv.org/abs/2004.10934

[16] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisser- man, ‘The Pascal Visual Object Classes (VOC) Challenge’, International Journal of Computer Vision, vol. 88, no. 2, pp. 303–338, Jun. 2010, doi: 10.1007/s11263-009-0275-4.

[17] T.-Y. Lin et al., ‘Microsoft COCO: Common Objects in Context’, in Computer Vision – ECCV 2014, Cham, 2014, pp. 740–755.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ‘ImageNet classification with deep convolutional neural networks’, Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[19] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, ‘SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and ¡0.5MB model size’. arXiv, Nov. 04, 2016. Accessed: Nov. 19, 2022. [Online]. Available: http://arxiv.org/abs/1602.07360

[20] X. Zhang, X. Zhou, M. Lin, and J. Sun, ‘ShuffleNet: An Ex- tremely Efficient Convolutional Neural Network for Mobile Devices’. arXiv, Dec. 07, 2017. Accessed: Nov. 19, 2022. [Online]. Available: http://arxiv.org/abs/1707.01083

[21] F. Chollet, ‘Xception: Deep Learning with Depthwise Separable Con- volutions’. arXiv, Apr. 04, 2017. Accessed: Nov. 19, 2022. [Online]. Available: http://arxiv.org/abs/1610.02357

[22] Z. Niu, G. Zhong, and H. Yu, ‘A review on the attention mechanism of deep learning’, Neurocomputing, vol. 452, pp. 48–62, Sep. 2021, doi: 10.1016/j.neucom.2021.03.091.

[23] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, ‘Squeeze-and-Excitation Networks’. arXiv, May 16, 2019. Accessed: Nov. 23, 2022. [Online]. Available: http://arxiv.org/abs/1709.01507

[24] X. Li, W. Wang, X. Hu, and J. Yang, ‘Selective Kernel Networks’, in 2019 IEEE/CVF

DOI: 10.54254/2755-2721/6/20230892

Conference on Computer Vision and Pattern Recog- nition (CVPR), Long Beach, CA, USA, Jun. 2019, pp. 510–519. doi: 10.1109/CVPR.2019.00060.

[25] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, 'CBAM: Convolutional Block Attention Module', in Computer Vision – ECCV 2018, vol. 11211, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 3–19. doi: 10.1007/978-3- 030-01234-2 1.

[26] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, 'Dy- namic Convolution: Attention Over Convolution Kernels', in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recogni- tion (CVPR), Seattle, WA, USA, Jun. 2020, pp. 11027–11036. doi: 10.1109/CVPR42600.2020.01104.

[27] Y. Liu, G. Shi, Y. Li, and Z. Zhao, 'M-YOLO based Detection and Recognition of Highway Surface Oil Filling with Unmanned aerial vehicle', in 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), Xi'an, China, Apr. 2022, pp. 1884–1887. doi: 10.1109/ICSP54964.2022.9778782.