DE-VSNet: Dense Efficient Vehicle State Classification Using Drone Traffic Dataset

Youlkyeong Lee, Jehwan Choi, Kanghyun Jo
Dept. of Electrical, Electronic and Computer Engineering
University of Ulsan, Ulsan, Korea
{yklee00815, choijh1897}@gmail.com, acejo@ulsan.ac.kr

2023 International Workshop on Intelligent Systems (IWIS) | 979-8-3503-0504-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/IWIS58789.2023.10284602

***Abstract*—This paper proposes a vehicle state classification model utilizing drones for achieving highly reliable autonomous driving vehicles. The model offers a wide field of view, providing improved data for the autonomous driving assistance system by considering the relationships with existing vehicles in the video. The object detection model, based on Convolutional Neural Networks, is trained using image data collected by drones. The YOLOv5 model is employed for detecting vehicles on the road. Detected vehicles are cropped into five candidate regions based on proximity, which serve as input data for the vehicle state classification model. The proposed model introduces a depthwise CSP block to effectively learn high-density layers and facilitate efficient weight learning. Additionally, a method that leverages deformable residual block to set receptive fields relying on features extracted from the fixed receptive field of the conventional convolutional kernel is proposed, enabling faster identification of critical features. Experiments are conducted using directly collected drone data to evaluate the performance of the trained model and compare it with a model based on dense layers. The approach aims to provide a more reliable and efficient vehicle state model for a robust autonomous driving system.**

***Index Terms*—Traffic analysis, drone flight image, efficient model, classification, object detection**

I. Introduction

The development of Autonomous vehicle technology has garnered significant interest worldwide. This technology requires the integration of various environmental information on the road, such as vehicle position, speed, obstacles, and traffic data, for seamless analysis. Furthermore, the integration with various next-generation mobility systems offers a promising approach to progressively advance reliable autonomous driving systems. Recently, image data collected through cameras installed in vehicles has gained attention for its high performance in developing algorithms for image classification, object detection, perception, and decision-making based on Convolutional Neural Networks (CNNs).

Popular models such as VGG [1], GoogLeNet [2], ResNet [3], DenseNet [4], SENet [5], and CSPNet [6] are widely used as backbone models to effectively extract features. Object detection, which involves locating and classifying objects in an image, is tackled by object detectors. Region-based detectors, such as R-CNN [7], Fast R-CNN [8], Faster R-CNN [9], and Mask R-CNN [10], are known for their high accuracy but are relatively slower due to the two-stage detection process. For real-time applications, single-stage object detectors like SSD [11], YOLO series [12]–[16] have been utilized in various applications. In particular, the You Only Look Once (YOLO) series overcomes the speed limitation and achieves performance comparable to two-stage models, making it widely adopted in commercial technologies.

By integrating these classification and decision-making algorithms, enhanced vehicle assistance systems have been developed to judge the state of the road. Previous research has proposed Long Short-Term Memory (LSTM)-based approaches that predict future vehicle positions using consecutive vehicle location information to determine the vehicle state [17]. Another study [18] proposes a classification model using drone aerial footage data to detect vehicles on the road and determine the current vehicle state taking into account the relationships between the target vehicle and surrounding vehicles.

This research is a continuation of the study presented in [18]. It focuses on designing an efficient model for vehicle state classification to improve reliability. The proposed approach consists of three components: 1) data generation; 2) object detection; and 3) vehicle state classification. Additionally, the study contributes to improving the accuracy of the classification model by modifying CSPNet [6].

II. Proposed Method

A. Vehicle Detection with YOLOv5

This research introduces a methodology for vehicle detection utilizing YOLOv5 [16], a sophisticated object detection model that has demonstrated superior performance across numerous visual recognition challenges. YOLOv5, an acronym for "You Only Look Once version 5", is an enhancement of the original YOLO framework, boasting improvements in computational speed and prediction accuracy. It employs a deep neural network structure that effectively extracts image features and concurrently predicts object bounding boxes and class probabilities in a single processing pass.

The backbone of YOLOv5, known as CSPDarknet53, is based on the Cross-Stage Partial (CSP) [6] network, enhancing the efficiency of feature extraction. It consists of two main components: CSPNet and Residual block. The CSPNet efficiently exchanges information between divided parts of the network, leading to faster and more accurate feature extraction compared to traditional architectures. The Residual block addresses the gradient vanishing problem, maintaining performance even with deeper networks. The head of YOLOv5

Fig. 1: Based on Algorithm 1, three figures illustrate cropped image, Input_j from original image, I.

is responsible for object detection, generating predicted bounding boxes, and class probabilities based on the predictions of network. It consists of three YOLO layers, each designed to detect objects of different sizes, ensuring effective detection of small and large objects. Each YOLO layer comprises convolutional operations for predicting bounding box coordinates and class probabilities, with dense layers performing the final transformation. By utilizing features at different scales, the head accurately detects objects of various sizes, making it a powerful and accurate object detection model.

B. Data Generation

Algorithm 1: Input Image for DE-VSNet

Data: Original Image: I, Vehicles: V, target bbox: tv
Result: Train Image for Classification: Input_j

begin
  for $v_j \in V$ **do**
    $tv = v_j$
    List of the distances between the target and others: LD
    for $v_i \in V$ **do**
      LD.append(d(tv, v_i))
    end
    Ascending order(LD)
    List of the shortest five distances: $[v_1, \ldots, v_5]$
    List of target and five vehicles: $SV = [tv, v_1, \ldots, v_5]$
    List of the x and y coordinate: X, Y
    for $v_k \in SV$ **do**
      $cx_k, cy_k, w_k, h_k = v_k$
      $ltx_k = cx_k - w_k/2$, $rbx_k = cx_k + w_k/2$
      $lty_k = cy_k - h_k/2$, $rby_k = cy_k + h_k/2$
      X.append(ltx_k), Y.append(lty_k)
      X.append(rbx_k), Y.append(rby_k)
    end
    min_x = Min(X), min_y = Min(Y)
    max_x = Max(X), max_y = Max(Y)
    $\text{Input}_j = I$[min_x:max_x, min_y:max_y]
  end
end

In Algorithm 1, data generation is a process for creating input images to train the vehicle state classification model. The input images are generated based on the detected vehicles from the vehicle detector. The Euclidean distance is calculated for all vehicles, V, surrounding the target vehicle, tv. The inter-vehicle distances with respect to the target vehicle are included in the LD list. LD is sorted in ascending order, and the five shortest distances are selected. The five selected vehicles and tv form the SV list. Each vehicle, v_k, in SV represents the location information of the vehicle. cx_k and cy_k are the x and y coordinates of the center of the vehicle, and w_k and h_k represent the width and height of the vehicle, respectively. Using this information, the upper left x, y coordinates, and the lower right x, y coordinates of the vehicle are generated and stored in the X and Y lists, respectively. X and Y lists are used to extract the minimum and maximum x, y coordinates, respectively, in order to crop the region encompassing the target vehicle and the surrounding five vehicles. Finally, as depicted in Fig. 1, an input image, Input_j, for the classification model is generated from the original image I.

C. Proposed DE-VSNet

The paper introduces Dense efficient vehicle state classification (DE-VSNet), a model comprising two modules: 1) Depthwise Cross-Stage Partial; 2) Deformable Residual. The overall structure of the proposed model is depicted in Fig. 2. By incorporating these two modules, DE-VSNet can effectively classify the motion state of detected vehicles, allowing the use of drone images for safe and autonomous driving assistance information.

1) Depthwise Cross-Stage Partial: The Depthwise Cross-Stage Partial is an improved block based on the Cross-Stage Partial Network (CSP) [6]. Instead of using 3×3 conv layers as employed in CSPNet, 3×3 depthwise layers are adopted to effectively construct the feature map. To address the issue of repetitive gradient learning, CSP Network proposes a method to increase the gradient path by dividing the feature map of the base layer into two parts, as illustrated in Fig. 3 (b). The input feature map of CSPNet, denoted as $x_0 = [x_{0'}, x_{0''}]$, is divided into two parts. The output of the first dense layer is represented as $[x_{0''}, x_1, \ldots, x_k]$, while the output of the transition layer is x_T, which is concatenated with $x_{0''}$. Subsequently, $x_{0'}$ is connected with the transition layer, and $x_{0''}$ is connected with the transition layer, forming the final output x_U.

$$\begin{aligned} \mathbf{x_k} &= \mathbf{w_k} * [\mathbf{x_{0''}}, \mathbf{x_1}, \ldots, \mathbf{x_{k-1}}] \\ \mathbf{x_T} &= \mathbf{w_T} * [\mathbf{x_{0''}}, \mathbf{x_1}, \ldots, \mathbf{x_k}] \\ \mathbf{x_U} &= \mathbf{w_U} * [\mathbf{x_{0'}}, \mathbf{x_T}] \end{aligned} \tag{1}$$

For the update of the weights $(w_{k'}, w_{T'}, w_{U'})$ using backpropagation, Eq. (1) is utilized, where f denotes the weight update function and g_i represents the gradient of the i-th layer. By inspecting the gradients of the Cross-Stage Partial Network, we can observe that the proposed method avoids the repetitive update of gradients in the dense layers of DenseNet. This

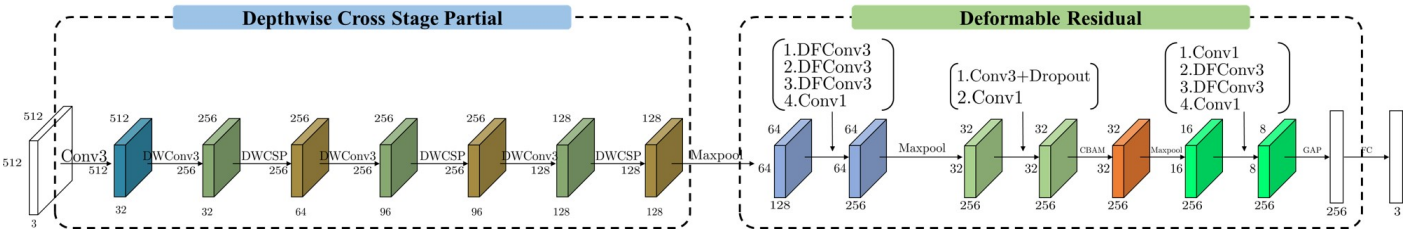


Fig. 2: This illustrates the overall architecture of DE-VSNet. The first component adopts the Depthwise Cross-Stage Partial (DWCSP) block, which is based on the Depthwise convolution and CSP block. The second component consists of Deformable Residual blocks, which are built on deformable convolutional operations.

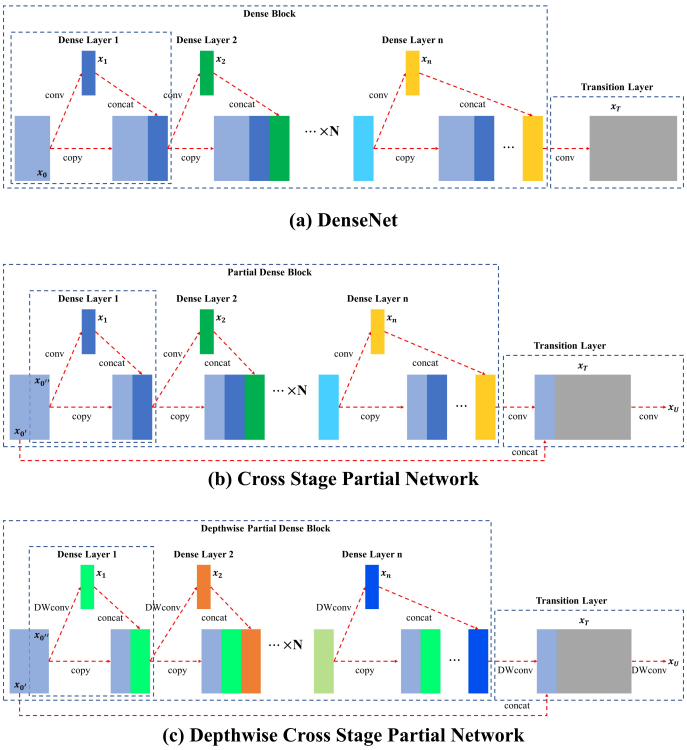
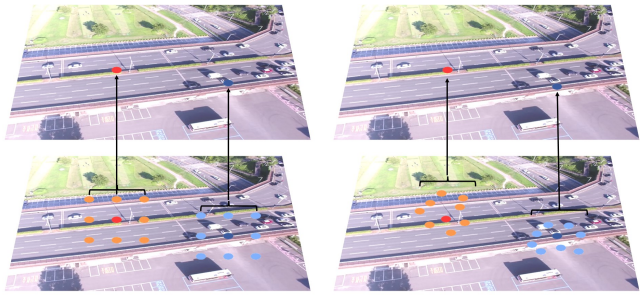


Fig. 3: A comparison of model structures for models constructed using Dense layers is shown in the figure. (a), (b), and (c) represent the Dense layer techniques used in DenseNet, CSPNet, and the proposed DWCSPNet, respectively.

reduces computational overhead and significantly contributes to computational efficiency.

$$\begin{aligned} \mathbf{w_{k'}} &= \mathbf{f}(\mathbf{w_k}, \mathbf{g_{0''}}, \mathbf{g_1}, \ldots, \mathbf{g_{k-1}}) \\ \mathbf{w_{T'}} &= \mathbf{f}(\mathbf{w_T}, \mathbf{g_{0''}}, \mathbf{g_1}, \ldots, \mathbf{g_k}) \\ \mathbf{w_{U'}} &= \mathbf{f}(\mathbf{w_U}, \mathbf{g_{0'}}, \mathbf{g_T}) \end{aligned} \tag{2}$$

To incorporate the computation reduction of CSP Network approach while effectively including additional filters to enhance the feature extraction block of the classification model, this work proposes the Depthwise Cross-Stage Partial Module. This module replaces the 1×1 conv layer used in the original CSP with a 3×3 depthwise conv layer. By applying the computation reduction method of CSP and simultaneously



(a) Conventional convolution (b) Deformable convolution

Fig. 4: (a) Represents the positions of kernel values in a conventional convolutional layer, (b) the positions of kernel values in a deformable convolutional layer.

incorporating effective filters, this module contributes to improving the performance of the classification model. Fig. 3 (c) illustrates the representation of the proposed Depthwise Cross-Stage Partial Module.

2) Deformable Residual: The features extracted from the DWCSPNet are further refined using the deformable residual module proposed in [18]. This module employs deformable convolutional layers, as introduced in [19], to flexibly search for features within the image and effectively extract discriminative feature maps. The deformable convolutional layers in the deformable residual part of Fig. 2 significantly enhance the capability of model to adapt to object deformations and spatial transformations, thus contributing to more accurate vehicle state classification results. Fig. 4 (a) demonstrates that the traditional 3×3 convolutional layer possesses a fixed receptive field over the image area, depicted by the red and blue dots. However, when dealing with image data for vehicle detection, vehicles are often distributed with spatial gaps between them. Consequently, using a fixed receptive field could lead to the extraction of feature information that includes unnecessary background details. In order to overcome this limitation and conduct more effective convolutional operations, the deformable convolution technique is adopted. As depicted in Fig. 4 (b), deformable convolution introduces an offset to the convolutional layer, enabling convolution operations based on the offset information. In this approach, the deformable

convolutional layer incorporates learnable offsets for each spatial location within the input feature map. These offsets instruct the sampling grid to adapt at each location during the convolution process. By incorporating these learnable offsets, the deformable convolutional layer dynamically adjusts its receptive field to the local context of each spatial location. This adaptability allows the network to effectively capture contextually relevant information, such as vehicle positions and spatial relationships, which results in the extraction of more informative and discriminative features for improved vehicle state classification within the proposed DE-VSNet model.

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \triangle \mathbf{p}_n) \tag{3}$$

Eq. (3) defines the offset used in deformable convolution [19], which determines the position of the kernel during the convolution process. The output feature map is denoted as $\mathbf{y}$. The kernel grid, $\mathcal{R}$, represents the receptive field and is defined as $\mathcal{R} = (-1,-1),(-1,0),\ldots,(0,1),(1,1)$. The convolution occurs at the pixel position of the input image $\mathbf{x}$, denoted as $\mathbf{p}_0$, and at the individual positions in $\mathcal{R}$, represented by $\mathbf{p}_n$, with corresponding offsets $\triangle \mathbf{p}_n$. Importantly, the offset value $\triangle \mathbf{p}_n$ is generated based on the convolutional layer's value and is updated in each iteration during training. Consequently, the position of the input value is determined by $\mathbf{p}_0 + \mathbf{p}_n + \triangle \mathbf{p}_n$, and the convolution operation is performed by multiplying this position with the convolution kernel weight, $\mathbf{w}(\mathbf{p}_n)$, at that particular location.

To enhance both channel-wise and spatial-wise information, the Convolutional Block Attention Module (CBAM) [20] is employed through three deformable convolutional layer operations. Subsequently, the fully connected layer receives the feature map that has been calculated by these two deformable convolutional layers.

III. Experiment

This research focuses on determining the state of vehicles using drone aerial footage data. The image data was collected through drones at a resolution of 3840×2160 (4K). For object detection, 9,776 training images and 2,200 test images were used from the collected dataset. The dataset contains labeled data for sedans and trucks, with a total of 309,470 objects. Object detection was performed using the PyTorch [21] framework, utilizing four NVIDIA A100 GPUs.

A. Object Detection

Table 1 presents the achieved object detection performance on the drone dataset. The YOLOv5 [16] framework was used for the object detection task. The overall training performance reaches 95.75% mAP@50 and 83.8% mAP@50:95. For testing, the performance achieved is 91.8% mAP@50 and 80.3% mAP@50:95. Using this object detection model, vehicles on the road are detected, and the detected vehicles serve as input images for the DE-VSNet.

TABLE 1: Result of object detection with drone dataset

Class	Images	Instance	mAP@50	mAP@50:95
all_train	9,776	309,470	95.75	83.8
car_vehicle	9,776	277,263	97.2	86.0
truck_vehicle	9,776	32,207	94.3	81.6
all_test	2,200	85,398	91.8	80.3
car_vehicle	2,200	78,765	96.1	85.3
truck_vehicle	2,200	6,633	87.5	75.4

B. DE-VSNet

The collected data for the vehicle state classification model is presented in Table 2. The dataset consists of a total of 5,854 images, categorized into three classes: lane change, safe, and stop. This dataset was collected and classified using Algorithm 1, resulting in the creation of the classification model dataset.

TABLE 2: Train and test dataset for Drone flight image

Class	train	test	Total
lane_change	1,350	214	1,564
safe	2,293	310	2,603
stop	1,382	305	1,687
Total	5,025	829	5,854

This paper proposes a Dense efficient Vehicle State Classification model compared to dense layer-based models. DenseNet121 [4] achieves 92.4% accuracy with 7.98 million parameters, 15.21 GFlops, and 121 connected layers. DenseNet169 satisfies 92.04% accuracy but has a two-fold increase in parameters and more about 30% GFlops. DarkNet53, with 53 convolutional layers used for feature extraction in YOLOv3 [14], performs at 81.18% accuracy. YOLOv4 [15], using CSPNet for feature extraction, improves speed through gradient dispersion but achieves 79.97% accuracy with a two-fold reduction in parameters and GFlops compared to DarkNet53. Our proposed model combines the enhanced DWCSPDarkNet and deformable residual network. The block value in Table 3 represents the number of DWCSP iterations. For b=[2,2,2], the model achieves 2.49 million parameters, 13.28 GFlops, and an accuracy of 88.54%. By varying the value of b experimentally, we find that b=[2,4,4] yields approximately 3.1 times fewer parameters than DenseNet121 with similar GFlops and 0.36% higher accuracy. This demonstrates the efficient reduction of parameters in models with dense layers while maintaining high performance. On the other hand, proposing a higher number of iterations with b=[4,8,8] results in 3.86% lower accuracy, indicating the negative impact of excessive iterations of efficient layers.

IV. Conclusion

This paper proposes efficiently high-density layers for the vehicle state classification model. The overall process is divided into vehicle detection, data generation, and vehicle state classification. YOLOv5 is used to detect vehicles, and based on the detection, target vehicles are selected. Candidate vehicles are then identified by calculating the distances between neighboring vehicles. This approach is used to generate input

TABLE 3: Comparison of dense layer kinds of models and proposed model, DE-VSNet

Method	Block	Params	GFlops	ACC(%)
DenseNet121	-	**7.98M**	**15.21**	**92.4**
DenseNet169	-	**14.15M**	**18.04**	**92.04**
DarkNet53	-	**40.59M**	**37.38**	**81.18**
CSPDarkNet53	-	**18.03M**	**17.27**	**79.97**
DWCSPDarkNet+DR	**b=[2,2,2]**	**2.49M**	**13.28**	**88.54**
DWCSPDarkNet+DR	**b=[2,3,2]**	**2.51M**	**14.06**	**88.30**
DWCSPDarkNet+DR	**b=[2,4,4]**	**2.56M**	**15.52**	**92.76**
DWCSPDarkNet+DR	**b=[4,8,8]**	**2.70M**	**20.72**	**88.90**

images for the vehicle state classification model. The proposed DE-VSNet is based on CSPNet and incorporates Depthwise convolutional layers to facilitate efficient gradient learning and effective feature extraction. Compared to the DenseNet121 and DenseNet169 models, which learn high-density layers, DE-VSNet achieves an improved accuracy of 0.36% and 0.72%, respectively. Additionally, efficient management of parameters and GFlops is presented. In conclusion, the proposed DE-VSNet demonstrates an efficient and superior performance for the vehicle state classification model by effectively transforming high-density layers.

ACKNOWLEDGMENT

This results was supported by "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(MOE)(2021RIS-003)

REFERENCES

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[2] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2014.

[3] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[4] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2016.

[5] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:2011–2023, 2017.

[6] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh. Cspnet: A new backbone that can enhance learning capability of cnn. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1571–1580, 2019.

[7] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2013.

[8] Ross B. Girshick. Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[9] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:386–397, 2020.

[11] W. Liu, Dragomir Anguelov, D. Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, 2015.

[12] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[13] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2017.

[14] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *ArXiv*, abs/1804.02767, 2018.

[15] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *ArXiv*, abs/2004.10934, 2020.

[16] Glenn R. Jocher, Alex Stoken, Jiří Borovec, NanoCode, Ayushi Chaurasia, TaoXie, Liu Changyu, Abhiram, Laughing, tkianai, yxNONG, Adam Hogan, lorenzomammana, AlexWang, Jan Hájek, Laurentiu Diaconu, Marc, Yonghye Kwon, Oleg, wanghaoyang, Yann Defretin, Aditya Lohia, ml ah, Ben Milanko, Ben Fineran, D. P. Khromov, Ding Yiwei, Doug, Durgesh, and Francisco Ingham. ultralytics/yolov5: v5.0 - yolov5-p6 1280 models, aws, supervise.ly and youtube integrations. 2021.

[17] Youlkyeong Lee, Qing Tang, Jehwan Choi, and Kanghyun Jo. Low computational vehicle lane changing prediction using drone traffic dataset. *2022 International Workshop on Intelligent Systems (IWIS)*, pages 1–4, 2022.

[18] Youlkyeong Lee, Seong Eun Kim, Jehwan Choi, and Kanghyun Jo. Vehicle state classification from drone image. *2023 IEEE International Conference on Industrial Technology (ICIT)*, pages 1–5, 2023.

[19] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017.

[20] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In-So Kweon. Cbam: Convolutional block attention module. In *European Conference on Computer Vision*, 2018.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.