CSA: Channel-wise Similarity Attention for Vehicle State Classification

Youlkyeong Lee, Jehwan Choi, Jinsu An, Kanghyun Jo
Dept. of Electrical, Electronic and Computer Engineering
University of Ulsan, Ulsan, Korea
{yklee, jehwan, jinsu5023}@islab.ulsan.ac.kr, acejo@ulsan.ac.kr

***Abstract*—Developed for specific missions, CNNs have gradually improved the performance of object classification networks by using various architectures. The weight of the convolutional layer is a crucial factor in feature extraction. However, as the number of layers increases, performance degradation can occur due to problems such as the vanishing gradient. To overcome this problem, networks have evolved to continuously incorporate information from previous feature maps using various attention mechanisms. In this study, a Channel-wise Similarity Attention (CSA) method is proposed to measure the similarity of feature maps between channels and enhance positive information by highlighting it. Additionally, a deformable convolutional kernel is embedded to apply a flexible receptive field around the object area in the image, replacing the fixed receptive field of the conventional CNN layer. The network is trained end-to-end to classify the condition of vehicles on the road using collected drone flight images. The proposed model achieves an accuracy of 86.13% and 302 frames per second with a number of parameters of 1,273,504.**

***Index Terms*—Vehicle state classification, Drone flight image, channel-wise similarity attention, deformable convolution**

I. Introduction

Drones have various applications, and the image data captured by their cameras is analyzed using convolutional neural networks (CNN). In the domain of traffic monitoring, conventional closed-circuit television (CCTV) cameras are only capable of capturing traffic conditions from fixed locations. However, drone images offer valuable insights into traffic conditions across large areas and from diverse perspectives. The analysis of image data collects from autonomous vehicles and CCTV cameras requires distinct approaches, necessitating research on CNN.

The field of computer vision is rapidly evolving due to the improvement of computing hardware performance. Learning models with hundreds of millions to billions of parameters can quickly achieve high performance. However, promising performance requires significant computing power, leading to high costs. Models like VGG [1] and ResNet [2] have significantly contributed to CNN research, although they have a large number of parameters (138M for VGG-16 and 39.32M for ResNet-34) for image classification. To address this challenge, the aim is to reduce the weight of learning strategies and models to enable efficient learning.

Convolutional neural network (CNN)-based applications [3], [4] that ensure performance in various areas can be combined for specific purposes, such as vehicle state classification. This process involves vehicle detection, data selection, and image classification, following a sequence of steps. The YOLOv5 learning model, from the YOLO series [5], [6], [7], [8], is utilized for real-time vehicle detection. This model is capable of predicting the location, size, and class of an object through a single neural network architecture, allowing for real-time object detection. Vehicles with short distances are selected based on the detected vehicles, and the image area containing all five vehicles is cut. Data augmentation techniques such as geometric transformation [9] and mixup [10], [11] are used to help the learning model learn consistently from various data and diversify the images. To install these kinds of systems into the mobile or application, the light model is an inevitable condition. Mobilenet [12], [13], [14] is a representative model that is an extremely light model than the previous classification model like VGG [1], ResNet [2], and DenseNet [15]. The other proposed method puts the module to squeeze the feature map. Squeeze and excitation [16] block adaptively recalibrates channel-wise feature responses by explicitly modeling interdependencies between channels. A noble model can also be developed by improving these models or adopting some modules. The proposed learning model utilizes the residual block from ResNet and applies it to the task. Object detection and data augmentation are preprocessed, and a classification model is developed to determine vehicle conditions. The channel-wise similarity attention-based vehicle state classification model is used in the proposed method, applied by an attention mechanism to address the vanishing gradient problem and pixel-to-pixel similarity in feature maps that occur as the deeper convolution layer. The proposed method explains detailed information about the vehicle state classification model.

II. Proposed Algorithm

The classification network requires the following preprocessing to understand the proposed vehicle state. First, vehicle detection and data preparation are performed through drone flight images. In the last, this work is proposed the channel-wise similarity maps attention-based vehicle state classification architecture.

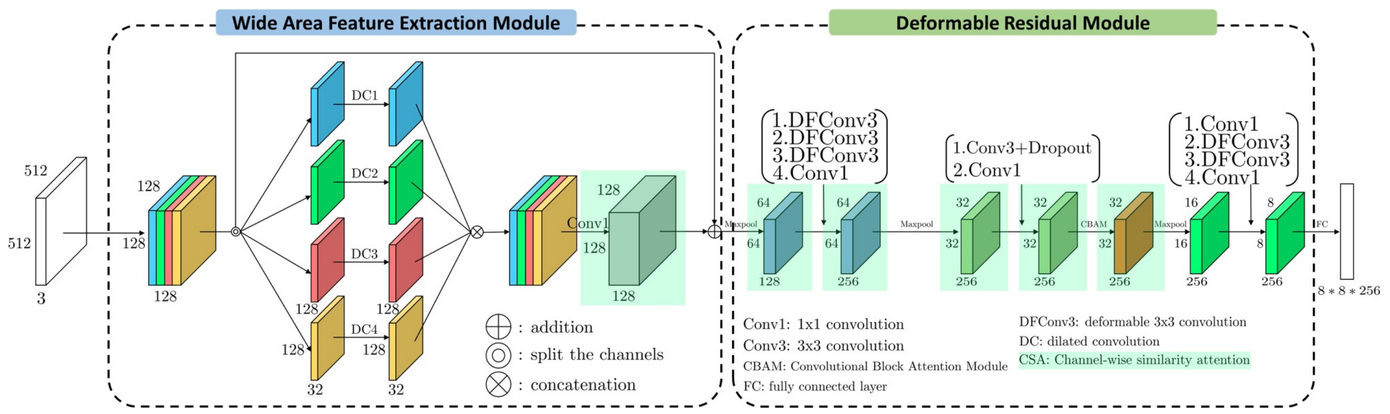

Fig. 1: Overall architecture for vehicle state classification. There are two parts, WAFE module, wide area feature extraction, and DR module, deformable residual module. CSA: Channel-wise similarity attention module in the green box.

A. Proposed architecture

1) Wide area feature extraction module: In Fig 4(b), the distance between vehicle positions in the input image is large, which may include unnecessary area information. To address this issue, a Wide area feature extraction module is proposed in this work. The initial convolutional layer of the module has a kernel size of 5×5, 64 channels, a stride of 4, and a dilated ratio of 3. The subsequent 1x1 layer increases the number of channels from 64 to 128, allowing for the sharing of information between channels on the reduced feature map. Gaussian Error Linear Units (GELU) [17] is used as the activation function in the proposed convolutional layer. Maxpooling is employed to reduce the size of the input feature map by half. The feature map is then divided into four groups, and each group is subjected to a 3x3 convolutional layer with dilated ratios of [1,3,5,7]. The resulting feature maps in the four groups are concatenated, and a 1x1 kernel is used to generate a feature map that extracts features from flexible regions. Like a residual block, the output adds the feature map from the initial convolutional layer.

2) Deformable residual module: The Deformable residual module is designed on the concept of a deformable convolutional layer introduced by Dai et al. [18]. Unlike the traditional convolutional layer has a fixed receptive field. The deformable convolutional layer allows for modifications in the position corresponding to the feature map and kernel value. In a regular grid, $\mathcal{R} = \{(-1,-1),(-1,0),\ldots,(0,1),(1,1)\}$ describes the position of kernel location. Let $\mathbf{x}$ be the input feature map, $\mathbf{p}_0$ be the pixel location, and $\triangle\mathbf{p}_n$ be the offset. These are added to obtain $\mathbf{p}_0 + \mathbf{p}_n + \triangle\mathbf{p}_n$ for the traditional convolution.

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \triangle\mathbf{p}_n) \qquad (1)$$

The offset, $\triangle\mathbf{p}_n$ is trained with learnable parameters and adopted as a method to effectively extract features, as the position value of $\mathbf{p}$ gradually moves around the features of the object during training.

The Convolutional Block Attention Module (CBAM) [19] is applied to the previously calculated feature map, allowing for channel-wise and spatial-wise attention to be applied. CBAM helps to improve model performance by enhancing important channel and spatial information while suppressing unnecessary noise. It can be attached in units of channels and spatial locations.

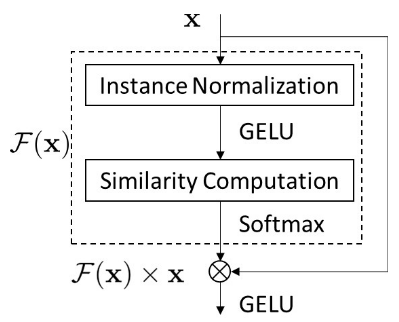

Fig. 2: Channel-wise Similarity Attention Module

3) Channel-wise similarity attention: Traditional convolutional layers in current systems use shared weights for input channels. They combine the filter value with input values to compute pixel values in the receptive field of the convolution kernel. This extraction process identifies edges and textures of objects in the feature map, and the classifier outputs probability values based on the number of proposed classes. However, the outline of an object is an important feature that provides continuous information about feature points and is essential for determining the shape of the object. The conventional convolution operation has limitations due to a small kernel area, making it difficult to capture long-range relationships between pixels. To overcome this limitation, Channel-wise similarity attention (CSA) module is proposed, which generates a similarity map between extracted features in the feature map. This mechanism continuously conveys pixel similarity, thereby capturing long-range relationships. CSA contains instance normalization [20], GELU [17], Softmax and similarity computation. Let $x \in \mathbb{R}^{R\times C\times W\times H}$ be an input feature map. x_{kij} denotes its kij-th element, where k is the index of channels and (i, j) is the index of pixel coordinate in k-th channel. $\mathbf{y}_{kij}$ is the normalized feature value at (i, j). μ_k and σ_k^2 are the mean and variance value in k-th channel. ϵ is small number, 10^{-5} for preventing zero value of denominator. In Eq. 2, instance normalization follows as:

$$\mathbf{y}_{kij} = \frac{x_{kij} - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}}, \mu_k = \frac{1}{HW} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} x_{kij}$$

$$, \sigma_k^2 = \frac{1}{HW} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} (x_{kij} - \mu_k)^2 \quad (2)$$

Normalized feature map transfers to calculate the similarity map in Eq. 3. $A_k, \in \mathbb{R}^{B \times C \times WH}$ is feature map in batch size, B, channel, C, size of image, WH at k-th channel. $A_k^T, \in \mathbb{R}^{B \times WH \times C}$ is the transpose of the feature map, A_k. $\mathbf{S}_k$ is the similarity map:

$$\mathbf{S}_k = \frac{A_k \cdot A_k^T}{\|A_k\|^2} \quad (3)$$

The similarity map has been converted into a probability value by utilizing the softmax function. This probability value, $\mathcal{F}(\mathbf{x})$ is then multiplied by $\mathbf{x}$ in Eq. 4, the input feature map, as illustrated in Fig. 2.

$$\mathbf{Out}(x) = \mathcal{F}(\mathbf{x}) \times \mathbf{x} \quad (4)$$

The output of attention, $\mathbf{Out}(x)$ highlights the probability value of the pixel-to-pixel similarity in the feature map, as well as the probability generated by the softmax, for the existing feature map.

B. YOLOv5 for Vehicle Detection

The detection of vehicles is critical for evaluating their safety and condition. Recent advancements in computer vision and deep learning have enabled the automation of this task using artificial intelligence. The study utilizes the widely-used YOLOv5 [8] algorithm for vehicle detection. The YOLOv5 detector is used by the vehicle status classifier in real-time vehicle detection for proposed works. The detector was trained using a combination of self-processed drone flight images and the VisDrone [21] dataset, a comprehensive benchmark for object detection in intricate environments. This method allowed for the development of a robust detector that can precisely identify vehicles in various scenarios. Additionally, the detector can be fine-tuned to identify specific vehicle types, such as cars, trucks, buses, or motorcycles. In Fig. 3, it shows the result of YOLOv5 with drone image.

III. Experiment

Drone AI Dataset: The dataset in this study uses self-capturing images of vehicles and generates a dataset by cropped road area from the original image through vehicle detection. The dataset is created by selecting five vehicles around the target vehicle based on their proximity. The area containing the selected vehicles is extracted from the original image. The total number of images in the dataset is shown in Table 1. The dataset contains 4,152 images and three vehicle status classes: lane_change, safety, and stop.

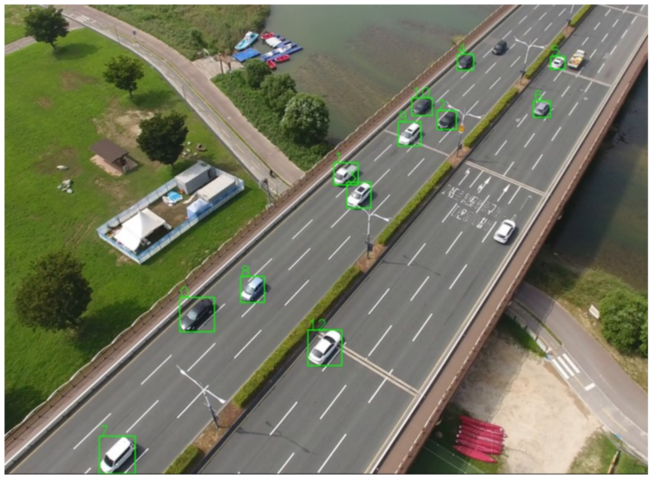

Fig. 3: Vehicle detection from YOLOv5 detector

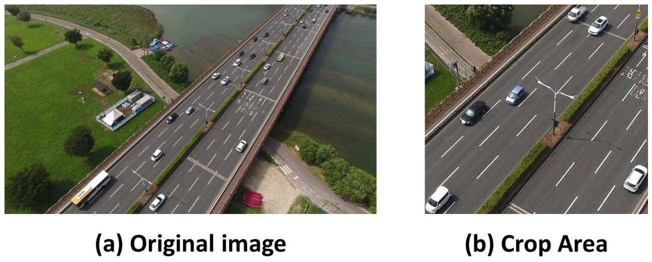

Fig. 4: (a) original image from drone view, (b) cropped area by object detector

TABLE 1: The number of datasets for vehicle state classification

Class	train	test	Total
lane_change	**860**	**214**	**1,074**
safe	**1,241**	**310**	**1,551**
stop	**1,222**	**305**	**1,525**
Total	**3,323**	**829**	**4,152**

Implementation Details: The vehicle classification model was developed using PyTorch [22]. The input image was resized to 512× 512 and the batch size was set to 32. The learning process was performed on an RTX 3090 GPU with 32GB of memory.

Object Detection: For this study, YOLOv5[8] is used for vehicle detection, and its results are used in the dataset generation. The training process uses 9,776 images, and 2,200 test images are used for evaluation. The vehicle detection performance is evaluated using mAP@50 and mAP@50:95 metrics, achieving 91.8% and 80.3%, respectively, as shown in Table 2.

TABLE 2: The mAP performance of YOLOv5 on drone train and test 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**

Vehicle State Classification: The proposed vehicle state classification model, based on the CSA module, illustrates the impact of applying the CSA module at various layers. In contrast, a dilated residual network (DRN) [23] serves as the baseline model. The DRN substitutes the convolutional layer in the ResNet [2] model with a dilated convolutional layer. Moreover, the DRN-A model eliminates the max-pooling layers from the ResNet structure, the DRN-B model adds four dilated layers at the end, and the DRN-C model removes the residual connection at the end. Unlike the DRN model, which utilizes a fixed receptive field location, the proposed classification model employs a flexible receptive field and integrates the CSA module. Table 3 presents experimental findings for comparison. DRN_D_22 indicates the model type

TABLE 3: Comparison of accuracy with the proposed method and DRN model. M is 10^6.

Method	Layer with CSA module						#para	fps	Acc(%)
	1	2	3	4	5	6			
DRN_D_22	-	-	-	-	-	-	16.39M	217	**87.69**
DRN_D_38	-	-	-	-	-	-	26.50M	147	**86.49**
DRN_D_54	-	-	-	-	-	-	35.80M	107	**89.26**
DRN_C_26	-	-	-	-	-	-	21.13M	185	**89.62**
DRN_C_42	-	-	-	-	-	-	32.23M	136	**81.78**
Proposed	✓	-	-	-	-	-	1.27M	302	**86.13**
	✓	✓	-	-	-	-	1.27M	296	**72.13**
	✓	✓	✓	-	-	-	1.27M	283	**55.73**
	✓	✓	✓	✓	-	-	1.27M	283	**54.52**
	✓	✓	✓	✓	✓	-	1.27M	286	**54.76**
	✓	✓	✓	✓	✓	✓	1.27M	286	**53.31**
	-	✓	-	-	-	-	1.27M	294	**75.39**
	-	-	✓	-	-	-	1.27M	296	**76.38**
	-	-	-	✓	-	-	1.27M	294	**73.82**
	-	-	-	-	✓	-	1.27M	295	**65.74**
	-	-	-	-	-	✓	1.27M	295	**65.62**

(D or C) and the number of layers. It has 16.38M parameters and an accuracy of 87.69%. DRN_D_38 and DRN_D_54 have 26.5M and 35.8M parameters, respectively, with an accuracy of 86.49% and 87.69%. For type C, DRN_C_26 and DRN_C_42 have 21.13M and 32.23M parameters, respectively, with an accuracy of 89.62% and 81.78%. The proposed model has a total of 1.27 million parameters, which is 12 to 25 times smaller than DRN. In terms of accuracy, it is 3.49% lower than DRN_C_26 but has the advantage of reducing computation by approximately 95%. The results of applying the designed layer-specific CSA module are also presented. Among the proposed methods, applying CSA only to the last layer of WAFE achieves the highest accuracy of 86.13%. Applying CSA modules at multiple layers results in gradually decreasing accuracy. In the proposed network, the inference speed is observed to be approximately 280 to 300 frames per second in Table 3. This speed is not only associated with the total number of parameters but is also enhanced through the utilization of a dilated convolutional filter for feature extraction and the division of feature maps into groups. This efficient design allows for improved processing speed, demonstrating the effectiveness of the proposed network architecture. Furthermore, applying CSA modules for each layer shows higher performance than multi-layer. The performance of applying the CSA module decreases as the high-level feature map increases. This is because the CSA module has already focused on the information of objects in the high-level feature, and it lowers the performance instead. For low-level feature maps, high performance is achieved by adding a similarity of information between the normalization of feature maps and low-level pixels. The highest performance of 86.13% was achieved in the 1-st layer.

IV. Conclusion

In this study, a vehicle condition classification model based on channel-wise similarity attention is proposed. The model utilizes drone data that is designed for a specific mission. The proposed CSA module calculates the similarity of feature maps and activates low-level features through the attention mechanism. The conventional convolutional layer with a fixed receptive field densely extracts features, including unnecessary areas of the collected dataset. To address this issue, the deformable convolutional layer effectively extracts features around the object as learning continues through a flexible receptive field. The DRN model is used for comparison, which replaces ResNet layers with dilated convolutional layers to use a wide receptive field area. In the experiment, the proposed model achieves an accuracy of 86.13% by applying CSA to various layers. Although the proposed model is 3.49% lower than DRN_C_26, it saves 95% on the number of parameters.

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, 2015.
[2] 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.
[3] 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.
[4] Youlkyeong Lee, Qing Tang, Jehwan Choi, and Kanghyun Jo. Low computational vehicle re-identification for unlabeled drone flight images. *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, pages 1–6, 2022.
[5] 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.

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

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

[8] 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.

[9] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *Neural Information Processing Systems*, 2018.

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

[11] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2017.

[12] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861, 2017.

[13] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[14] Andrew G. Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019.

[15] 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.

[16] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2017.

[17] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv: Learning*, 2016.

[18] 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.

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

[20] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *ArXiv*, abs/1607.08022, 2016.

[21] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 2020.

[22] 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.

[23] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.