Lightweight Bird Eye View Detection Network with Bridge Block Based on YOLOv5

Jehwan Choi
Grad. School of Electrical and Computer Engineering
University of Ulsan, Korea
jhchoi@islab.ulsan.ac.kr

Kanghyun Jo
School of Electrical Eng.
University of Ulsan, Korea
acejo@ulsan.ac.kr

***Abstract*—In this paper, The network with a faster detection speed than the original YOLOv5 nano model is proposed. The network defined as a bridge module reduced the number of channels and changed the speed quickly by applying pixel-wise operation instead of using a convolution layer. Especially, element-wise addition operation of each output feature maps is the main method. As a result, the detection speed is faster than the original detection method about 30 35%. On the other hand, mAP (mean average precision) is recorded at 50.7%, which is 1.4% lower than the original detection method. However, the original detection method showed good results in 3 classes and the proposed method showed good results in 5 classes. And the proposed method detected more objects in a detection result image. Therefore, the proposed method is a more efficient object detection network.**

***Index Terms*—Lightweight network, object detection, drone dataset, ghost module, inception module**

I. Introduction

Nowadays, many researchers are interested in designing a lightweight CNN structure to show good performance even in environments with insufficient computational power, such as mobile devices or low-spec computers, or when it is difficult to drive high-performance devices such as automobiles and drones. The lightweight CNN structure design is an essential part for tasks such as smartphone camera applications, vision-based artificial intelligence for autonomous driving, and search for missing persons using drones. The dataset used in this paper was also filmed with a drone as shown in Fig. 1, and the drone is the most necessary means of a Lightweight Network. Because, it is difficult to mount a high-performance computational processing unit, and there may be problems with flight performance and safety due to the nature of the drone if it is mounted.

Representatively, MobileNet [1]–[3] proposed Depthwise Separable Convolution and the internal structures such as linear bottleneck and inverted residual structures were effectively changed to significantly lighten the model without sacrificing performance. Depthwise Separable Convolution was proposed as one of the main techniques in ShuffleNet [4], [5] and GhostNet [6]. In addition, ShuffleNet [4], [5] proposed Pointwise Group Convolution to reduce the computational amount of 1x1 Pointwise Convolution along with a method of mixing channels. However, the speed to make a result should be fast, but the performance should not be too bad than other things or the accuracy is not good. In order to maintain accuracy,

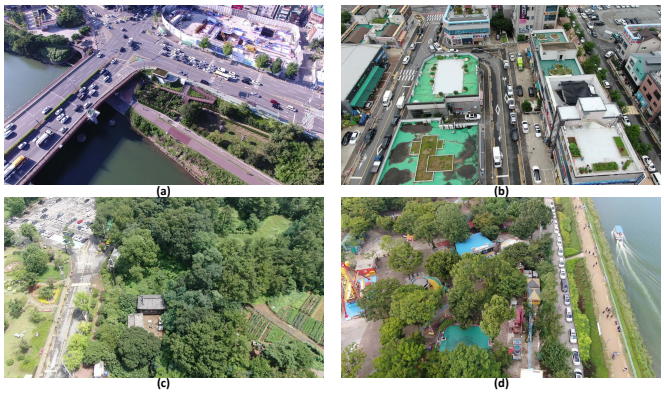


Fig. 1: The example image of the BEV drone dataset used in the experiment (a) Downtown_Taehwa-Bridge_80m_50°, (b) Downtown_Daegu_Geumho-District_60m_45°, (c) Tourist-area_Daegu_Suseong-pond_60m_45°, (d) Tourist-area_Daegu_Hwawon-Amusement-Park_60m_45°.

Inception [7], [8] and Exception [9] generated feature maps using filters of various kernels and presented a method of using a 3x3 kernel twice in succession instead of a 5x5 kernel.

In this paper, we set the YOLOv5 nano version as the baseline network and modify the core parts like bottleneck CSP module and C3 module to propose a bridge block that is faster than the existing model and has little sacrifice in accuracy. In the proposed part, it is designed to obtain advantages in terms of speed and parameters by replacing the convolution operation in the bottleneck module with element-wise addition and to obtain various feature maps by configuring it like an Inception module. Therefore, the proposed method in this paper can be summarized as follows:

- Instead of the BottleneckCSP module or C3 module that contains the bottleneck algorithm, we propose dividing a total of 4 feature maps, calculating each feature map, and then concatenating it.
- To prevent the feature of image is disappeared because of the object in Bird Eye View(BEV) image is very small and quickly lost, each feature map performs element-wise addition before concatenating.
- Our method achieve similar accuracy with YOLOv5 nano models with lower number of parameters.

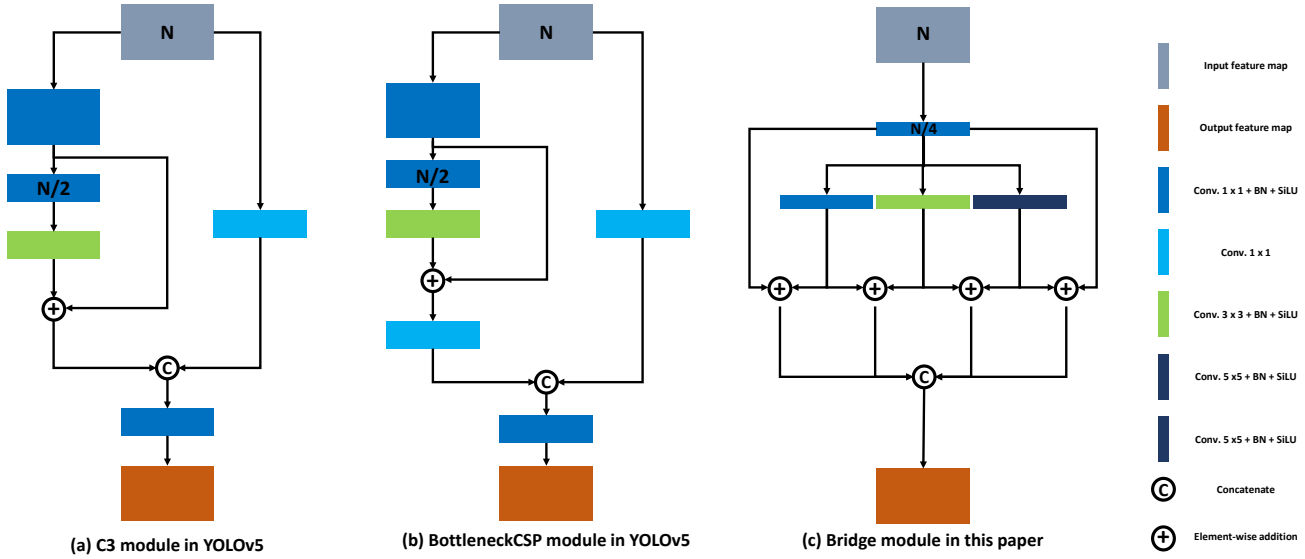


Fig. 2: Illustration of the proposed Bridge module with comparing module in YOLOv5.

II. Proposed Method

The base-line network of the method proposed in this paper is the YOLOv5 nano model, and we propose the Bridge module which is a core area for faster and more efficient results instead of the BottleneckCSP module and the C3 module.

A. Bridge Module

To reduce the computation cost the bridge module has a small number of convolution layers as much as possible and concatenates the channels after calculating the number of channels used. As you can see in Fig. 2, the number of channels of the input feature map was reduced by half in the original model(YOLOv5). But it was reduced to a quarter in the proposed method. Each feature map was calculated and then concatenated. Also, element-wise addition of each image before concatenating prevents loss of image features. As the number of channels is reduced, the kernel size also decreases, so the number of operations decreases.

B. Bridge Connection

In this module, input feature maps are extracted as many as possible using kernels of 1x1, 3x3, and 5x5 sizes. A total of 4 blocks are created including input feature map as shown in Fig. 3 and Table I. Finally, a total of 4 element-wise additions are performed. The four pairs consist of 1) the input feature map and the output of the 1x1 convolution layer, 2) the output of the 1x1 convolution layer and the output of the 3x3 convolution layer, 3) the output of the 3x3 convolution layer and the output of the 5x5 convolution layer, 4) the output of the 5x5 convolution layer and input feature map. The reason for grouping the output values using similar kernel sizes in pairs is the limiting of diversity. If the kernel size is different, there is also a difference between the outputs. At this time, if element-wise addition is performed between feature maps with too much difference, the error may become larger. After element-wise addition, the generated four feature maps are concatenated and proceeded to the next network. As shown in Fig. 4, the result of the computation makes more clear than the normal output feature map.

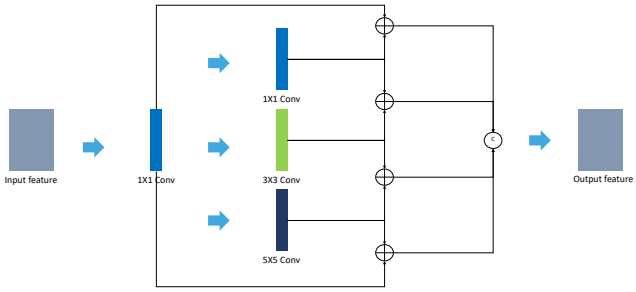


Fig. 3: Illustration of Bridge connection.

TABLE I: The detail information of proposed bridge module.

Input	Output	Module Config	Channel
Input	Conv1	1x1 Conv, BN, SiLU	N/4
Conv1	Conv2-1	1x1 Conv, BN, SiLU	N/4
Conv1	Conv2-2	3x3 Conv, BN, SiLU	N/4
Conv1	Conv2-3	5x5 Conv, BN, SiLU	N/4
Conv1 Conv2-1	Conv3-1	Element-wise addition	N/4
Conv2-1 Conv2-2	Conv3-2	Element-wise addition	N/4
Conv2-2 Conv2-3	Conv3-3	Element-wise addition	N/4
Conv2-3 Conv1	Conv3-4	Element-wise addition	N/4
Conv3-1 Conv3-2 Conv3-3 Conv3-4	Output	Concatenate, BN, SiLU	N

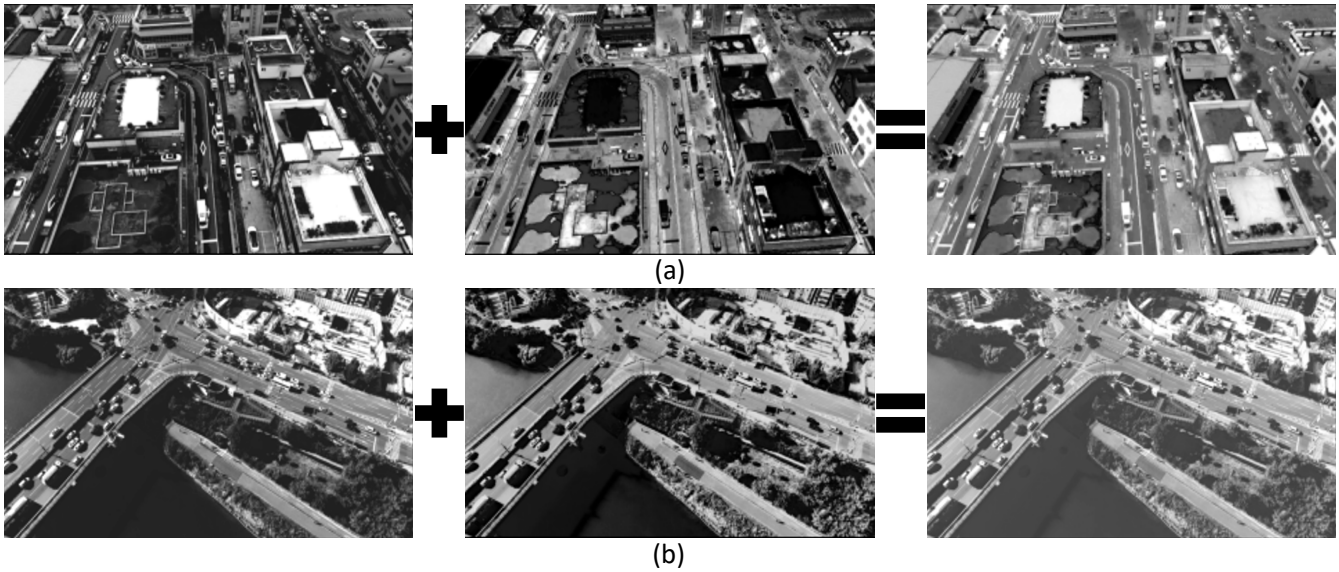


Fig. 4: Illustration of the result after element-wise addition with two feature maps. (a) The result image of Daegu_Geumho district, (b) The result image of Ulsan_Taehwa bridge

TABLE II: The information of data used in the experiment.

Category	Region_Place	Altitude	Angle	The number of image
Downtown	Ulsan_Taehwa bridge	80m	50°	1,025
	Daegu_Geumho district	60m	45°	967
Tourist area	Daegu_Suseong pond	60m	45°	1,058
	Daegu_Hwawon amusement park	80m	45°	982

III. Experiment

A. Dataset

The dataset used in this paper is a self-driving drone flight video constructed by the National Information Society Agency. The self-driving drone flight video consists of video images, images, and json files corresponding to 320 hours, and the data used for the experiment consisted of the number of 3,680 training images, the number of 320 testsets and validation sets each. The dataset was taken with a drone in downtown, tourist areas and forests in Korea. All the photographed areas are formed at various angles and altitudes as shown in Fig. 1. The information of data used in the experiment is shown in Table II. The original dataset consists of a total of 18 classes, but a total of 11 classes (trees, houses, buildings, traffic lights, traffic signs, telephone poles, buses, motorcycles, cars, trucks, people) was selected and the experiment was conducted since dataset of experiment is downtown and tourist area oriented.

B. Evaluation Metrics

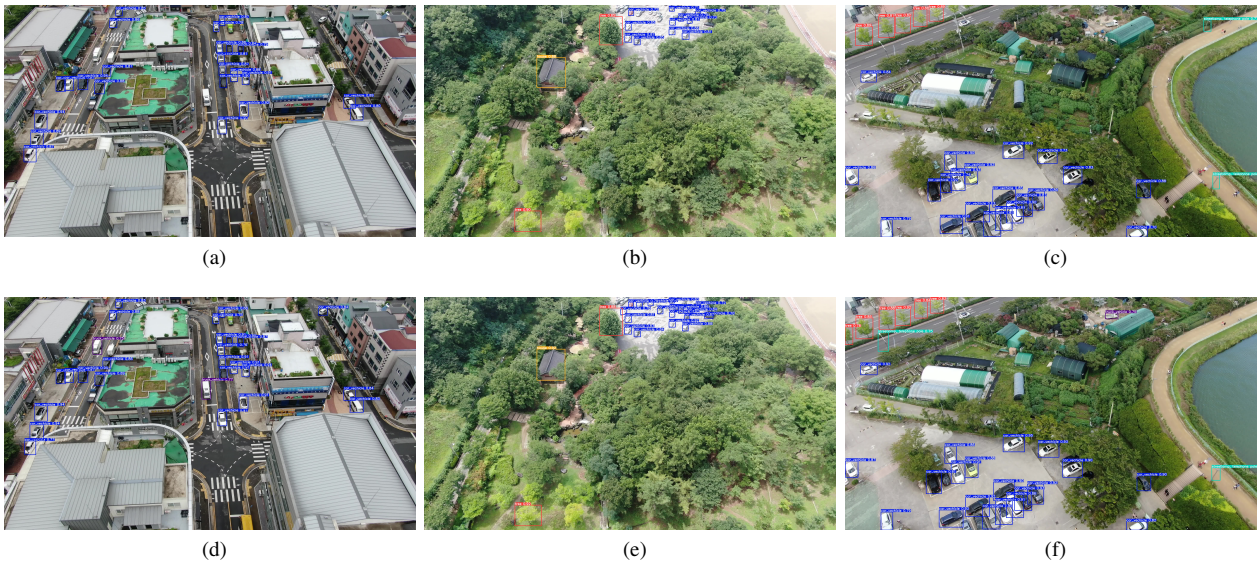
For performance evaluation, the number of parameters and GFLOS (GPU FLoating point Operations Per Second) were selected to compare the speed of the proposed method. In addition, mAP (Mean Average Precision), Precision, Recall, and F1 Score were used to compare accuracy with original model(YOLOv5 nano).

C. Implementation Setup

All experiments were conducted for a total of 30 epochs, and the final mAP was calculated as the average of the accuracy obtained through 3 repetitions of the experiment. According to the number of datasets, the batch size was set to 40, and the learning rate was set to 0.0076. For accuracy and speed comparison, the YOLOv5 nano version was used by the baseline of this paper. The experiment was conducted on the specifications of an Intel Core i9-10900X CPU, GeForce RTX 3090 24GB, and 188GB of RAM memory.

IV. Result

The proposed method showed the number of parameters 1,276,385, 2.9 GFLPS, and mAP of 50.7%. It can be said that better results were achieved for three reasons, considering that the original detection method showed 1,825,327 parameters, 4.4 GFLOPS, and 52.1% of mAP. First, the detection speed was about 30 35% faster than the original detection model with the small number of parameters and the GFLOPS value. The second is the detection accuracy of individual classes. As shown in Table 3, the original detection model showed good performance in three classes (Person, Bus, Street Lamp), but the proposed method showed good performance in five classes (Car, Truck, Tree, House, and Building). The reason that the detection accuracy of Motorcycle, Traffic Light, and Traffic Sign was 0% is that the number of objects in the dataset was



(a) (b) (c)

(d) (e) (f)

Fig. 5: Example result image of original detection model[(a), (b), (c)] and proposed method model[(d), (e), (f)].

TABLE III: The accuracy of each classes by original detection method and proposed method.

Detection accuracy of original detection method											
Class	Person	Motorcycle	Car	Truck	Bus	Tree	Traffic Light	Traffic Sign	Street Lamp	House	Building
Accuracy	**17.4%**	0%	87.1%	40.7%	**55.6%**	96.6%	0%	0%	**81.3%**	88.9%	92.5%
Detection accuracy of proposed detection method											
Class	Person	Motorcycle	Car	Truck	Bus	Tree	Traffic Light	Traffic Sign	Street Lamp	House	Building
Accuracy	8.6%	0%	**88%**	**44.6%**	53.3%	**96.9%**	0%	0%	80.2%	**89.9%**	**93.1%**

not enough, and then learning was not done well. Third is more objects were detected in the same image as shown in Fig 4. Fig 4(a, b, c) shows the detection result by the original detection model, and Fig 4(d, e, f) shows the detection result by the proposed method. As you can see in each image, more objects are detected in the result of the proposed method.

V. Conclusion

In this paper, we propose a method to detect objects without significant loss in accuracy with fewer parameters and computational speed than the original detection models by using the Bridge module instead of the BottleneckCSP module and C3 module of YOLOv5. In order to reduce the number of parameters, the number of channels was reduced more than the original detection method. In addition, in order to speed up the computation, the method of element-wise addition and concatenation of each feature map instead of a convolution layer was applied. As a result, it increased the detection speed and showed good performance in individual class accuracy.

VI. Acknowledgement

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] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[2] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *CoRR*, vol. abs/1801.04381, 2018. [Online]. Available: http://arxiv.org/abs/1801.04381

[3] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019. [Online]. Available: http://arxiv.org/abs/1905.02244

[4] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," 2017. [Online]. Available: https://arxiv.org/abs/1707.01083

[5] N. Ma, X. Zhang, H. Zheng, and J. Sun, "Shufflenet V2: practical guidelines for efficient CNN architecture design," *CoRR*, vol. abs/1807.11164, 2018. [Online]. Available: http://arxiv.org/abs/1807.11164

[6] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," *CoRR*, vol. abs/1911.11907, 2019. [Online]. Available: http://arxiv.org/abs/1911.11907

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[8] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[9] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2017.