Vietnam Journal of Science and Technology ***61*** *(4) (2023) 708-722*

doi:10.15625/2525-2518/17595

Evaluating the influence of backbone network architectures for object detection in aerial images

Khang Nguyen[1, 2]

[1]*University of Information Technology, 1 Han Thuyen Street, Quarter 6, Linh Trung Ward, Thu Duc District, Ho Chi Minh City, VietNam*

[2]*Vietnam National University, km 20 Ha Noi Highway, Linh Trung Ward, Thu Duc City, Ho Chi Minh City, Viet Nam*

[*]Emails: *khangnttm@uit.edu.vn*

Received: 17 October 2022; Accepted for publication: 24 February 2023

Abstract. Drones are increasingly being used in surveillance, agriculture, and delivery tasks. However, the real-life application of images collected from drones in urban management is still limited. Although drone images have many advantages thanks to the flexibility of the latest devices, there are still new challenges, such as top-down views, small objects, arbitrary directions, and class imbalance. This paper presents the results of research, survey, and evaluation of the performance of CNN-based network architectures for object detection in aerial images. Experiments were conducted on seven deep learning network architectures: VGG, ResNet, ResNext, Res2Net, ResNeSt, HRNet, and RegNet to bring objective judgments and conclusions based onpractice, contributing to the development of solutions to be applied in determining the status of urban traffic. The baseline object detection method used to train object detection is Faster R-CNN, which is the standard method widely used nowadays. Experiments on UIT-Drone21 and XDUAV datasets were conducted to provide insightful analyses for further studies in the future.

Keywords: aerial images, object detection, vehicle detection, convolutional neural networks

Classification numbers: 4.7.4, 4.8.2

1. INTRODUCTION

Building Smart Cities is a trend in the 4.0 era, accompanied by a growing demand for traffic management in these urban areas. Currently, most roads in Viet Nam still use CCTV, and sensors as tools to monitor the traffic situation, but these devices are very passive in information collection since the shooting angles of a camera are always limited. In order to capture the entire traffic situation at a time, it is necessary to attach numerous cameras at different angles. But this puts pressure on the limited infrastructure in our country. On the other hand, the lack of information can lead to biased predictions that should not be made.

Drones are gaining more attention recently [1, 2] because the images collected from drones are comprehensive thanks to their ability to fly high over distances with a radius of up to several kilometers, flecxible expansion to all angles, decent quality of information collection, and high resolution (up to 4K) of the attached camera's device". With those advantages, unmanned aerial vehicles allow easy detection of traffic vehicles, including small objects such as cars and motorbikes. But this type of equipment has not been widely applied to urban management tasks in Viet Nam.

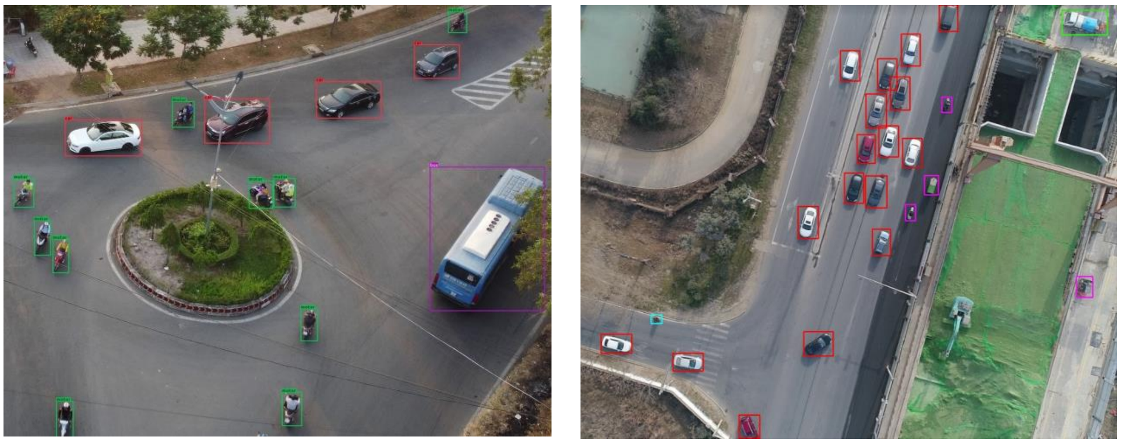
Figure 1. Results of object detection in aerial images.

The urban management problem does not stop at the information collection device because there is still a need for a solution where the information source is used more effectively and optimally. Advanced techniques can help analyze information sources, automatically make decisions, and reduce management pressure on the authorities. Currently, computer vision is one of the throne-problems in this field that are highly practical and easy to apply to real-life situations, typically the object detection problem. With input as image and output as location and identification of objects of interest, object detection is the ideal solution for traffic monitoring, vehicle estimation, convenient object tracking, and traffic analysis. Object detection in aerial images has been solved enthusiastically around the world. Many datasets proposed for this problem can be mentioned such as VisDrone-DET [3], UAVDT-DET [4], XDUAV [5], and AU-AIR [6]. In which, VisDrone-DET is considered the most standard benchmark for evaluating the detection performance of models.

On the other hand, for each object detection method, there is usually a convolutional network as the backbone architecture to extract essential features from the raw input information so that the model can learn better to make more accurate predictions. Furthermore, in this study it is hypothesized that the selection of which convolutional network for information processing greatly affects the performance and accuracy of the output model.

In this paper, the effectiveness of the recent architectures is surveyed, including VGG [7], ResNet [8], ResNeXt [9], Res2Net [10], ResNeSt [11], HRNet [12], and RegNet [13], which are very new backbone network architectures. The mentioned architectures are used to train the Faster R-CNN for detecting objects in aerial images. This study mainly focuses on the Vietnamese context, therefore, the datasets used to do the empirical study are UIT-Drone21 and XDUAV. UIT-Drone21 is the novel benchmark for capturing images from a DJI drone in urban Viet Nam, and XDUAV was collected using a DJI Phantom 2 at an average altitude of 100 m in Xi'an, China's rural and urban areas.

The rest of the paper is organized as follows: Chapter 2 delves into the work related to deep convolutional neural networks and experimental architectures, Chapter 3 presents and evaluates the results, and at the end, Chapter 4 will give conclusions and directions for further research.

2. MATERIALS AND EXPERIMENTAL NETWORKS

2.1. Components of the convolutional neural network

Convolutional layers: The goal of the convolutional layers is to extract high-level features from the input image. Convolutional neural networks are not limited to just one convolutional layer. Usually, the first convolutional layer is responsible for capturing low-level features such as angles, colors, gradient directions, etc. With the layers added later, the architecture will be designed to accommodate the gradual collection of higher-level features. Thanks to that, the network can learn information, and semantic insights in the image as humans can understand.

Pooling layer: The pooling layer is a downsampling operation (downsampling is the reduction of the sampling frequency), often used after the convolutional layer, which increases the spatial invariance, reduces the computation and training time, but retains important features. There are many types of pooling, such as Sum pooling, $\boldsymbol{L2}$ pooling, Max pooing, and Average pooling. Among them, Max pooling and Average pooling are the most popular types of pooling.

Activation function: Activation functions are non-linear functions that are applied to the output of neurons in the hidden layer of a network model, and the result of the function is used as the input for the next layer. The purpose of the activation function is to help the model learn the complex non-linear systems implicit in the data. The most common activation functions are Sigmoid, Relu, and Softmax.

Batch normalization: Batch normalization is a widely applied technique for deep neural networks (DNNs). Thanks to smoothing the loss function surface (optimization of landscape), batch normalization will help the model to be trained faster and more stable [14].

Dropout: Dropout is a technique where randomly selected neurons are ignored during training. Their contributions to the activation of downstream neurons are temporarily removed during the forward propagation, and any weight updates are not applied to those neurons in the backpropagation process. It helps the final model avoid overfitting.

2.2. Experimental Backbone Architectures

VGG (2014): VGGNet [7] is a simple traditional deep learning network with stacked convolutional layers. The input image after passing through the network will reduce the resolution and increase the depth, which helps to learn the shallow features and depth of the image. VGGNet uses a $\mathbf{3} \times \mathbf{3}$ filter in the convolutional layers and achieved high results in the 2014 ILSVRC competition. VGGNet is the foundation for future network studies. In this study, VGG-19 architecture is used as the test architecture which is visualized in Figure 2.

ResNet (2015): ResNet, short for Residual Networks, is a CNN architecture proposed by He *et al.* in 2015 [8] and won the top 1 in the 2015 ILSVRC image classification contest. Different from previous traditional CNN architectures, ResNet's success is using Residual blocks to avoid (Figure 3). This block consists of convolutional layers (usually 2 layers $\mathbf{1} \times \mathbf{1}$ and 1 layer Vanishing Gradient. The idea of ResNet is the network architecture based on Residual blocks $\mathbf{3} \times \mathbf{3}$ in between). The block input is a feature x, after going through the convolutional layers in the block it obtains the feature $\boldsymbol{f}(\boldsymbol{x})$, then the feature x will be added with $\boldsymbol{f}(\boldsymbol{x})$ and the final output of the Residual block is $\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{x}$. Using Residual block will keep the features from shallow to deep layers, avoiding the gradual loss of image feature information leading to Vanishing Gradient.

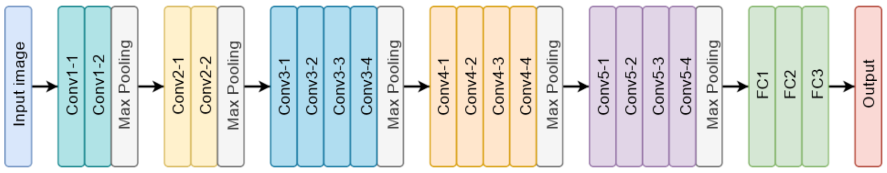

Figure 2. Architecture of VGG-19.

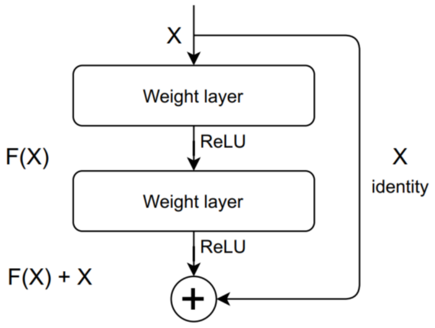

Figure 3. Illustration of Residual block in ResNet.

ResNeXt (2016): Proposed in the study [9], the ResNeXt architecture also uses the Residual block and skip connections; however, the authors split into multiple branches in the Residual block, then these branches are added together. The number of branches to be split is a hyperparameter called “cardinality”. The higher the “cardinality” will help reduce the error on the test set, which means that the overfitting will be solved. The dimension of each branch is denoted by d. Figure 4 illustrates the Residual block proposed by the authors.

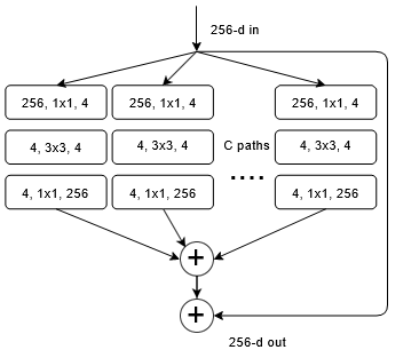

Figure 4. Illustration of Residual block in ResNeXt.

Res2Net (2019): Introduced in 2019 [10] with the desire to improve image representation with multidimensional features, Res2Net replaces the traditional 3×3 filter in the Residual block commonly found in ResNet or ResNeXt with a set of smaller filter groups. Specifically, after the convolutional layer 1×1, the authors divide the image feature into s sub-features, denoted by $\boldsymbol{x_i}$, where $i \in \{1,2,\dots,s\}$. Each sub-feature xi will have the same size but the depth is only equal to $\frac{1}{s}$ the input feature.

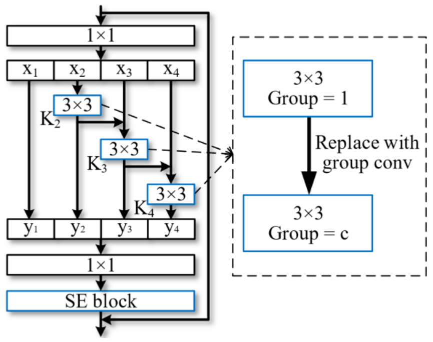

Figure 5. Res2Net Module [10].

In the Res2Net architecture, branches will be processed at a multi-scale with the desire to extract global and local features. To combine information at these rates, the authors use as low product class 1 × 1. The authors also show that the parameter s controls the number of dimensions. The larger s, the larger the learned filters will be with negligible computational cost.

HRNet (2019): The study [12] proposes an HRNet architecture to preserve high-resolution representations of features as they traverse the network. First, the authors start from a high-resolution convolutional stream, gradually add in lower-resolution streams, and then connect them. The authors' architecture consists of n stages, each with n streams corresponding to n resolution.

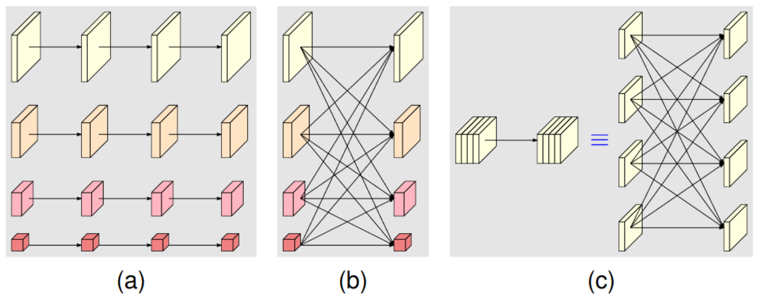

Figure 6. (a) Illustration of parallel multi-resolution streams; (b) Illustration of how to combine output at different resolutions; (c) Description of the combination as well as the fully connected layer on the convolutional layers [12].

Streams are denoted by N_{sr}, where s is the order of the stage and r is the resolution position of that stream. The resolution position of the next stream will be calculated as $\frac{1}{2^{r-1}}$. To combine convolutional streams, the authors use a transformation function $f_{xr}(x)$ which depends on the resolution position x and the degree position output resolution r. If $x = r$, the input and output have the same size, so there is no need to change. For $x < r$ the output is larger than the input, so $r - s$ convolutional layers 3×3whose with $strides = 2$. are needed. If $x > r$, the output is smaller than the input, so it is necessary to increase the output size by a bilinear sampling layer followed by a convolutional layer 1×1 to control and adjust the number of feature dimensions.

ResNeSt (2020): In the study [11], the authors propose a ResNest architecture that applies channel-wise on different branches to take advantage of this method's success in interacting between features and learning diverse semantics. But instead of using Residual blocks like ResNeXt, the authors use Split-Attention blocks. This architecture also divides the feature into K groups (K is the hyperparameter of the model) and proposes to add a hyperparameter R corresponding to the number of subnet branches. Hence, the total number of feature groups will be $G = KR$. The sub-branches in each block are connected by a Split-Attention block (Figure 7). In the Split-Attention block, the authors apply the shortcuts commonly found in the Residual block, for features that are not of the same size, a transformation function T will be applied to perform this.

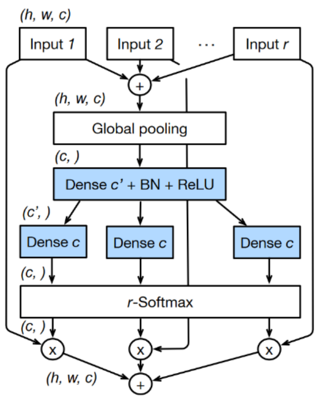

Figure 7. Split-Attention Block [11].

RegNet (2021): RegNet was introduced in early 2021 in the study [13], in which the authors propose a mnemonic mechanism to extract additional features, then feed them into the ResNet network. Specifically, this mnemonic includes multiple convolutional RNNs (LSTMs or GRUs). The authors suggest two types of Residual blocks in ResNet: non-bottleneck blocks and bottleneck blocks. Based on this, by applying RNN networks, the authors obtain the RNN-Regulated ResNet module (Figure 8a) and Bottleneck RNN-Regulated ResNet (Figure 8b).

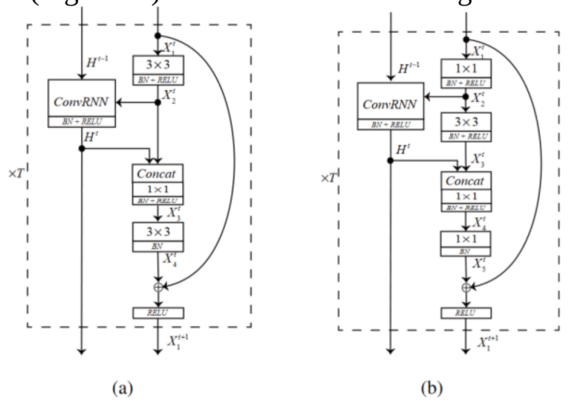

Figure 8. (a) RNN-Regulated ResNet; (b) Bottleneck RNN-Regulated ResNet [13].

2.3. Baseline Object Detection Model: Faster R-CNN

To train the object detection task, the Faster R-CNN method, a standard two-stage object detector proven to perform well on aerial images, is chosen. Faster R-CNN [15] is an enhanced version of Fast R-CNN [16] proposed by Ren et al. in 2015, and it is a two-stage detector. This model is a single, unified network for object detection consisting of 2 modules: the Region Proposal Network module (RPN module [15]) and the Fast R-CNN detector. The Region Proposal Network will serve as the "attention" of the system, telling the Fast R-CNN detector where to look. This module is also the main alteration of Faster R-CNN compared to its predecessor. The RPN module not only improves the system's speed but also helps Faster R-CNN achieve state-of-the-art detection accuracy on PASCAL VOC 2007 [17], 2012 [18], and MS COCO datasets. In addition, the Faster R-CNN model and RPN are the foundation of the first-place winning entries in ILSVRC [19] and COCO 2015 competitions.

For better illustration, Faster R-CNN can be formulated as follows:

$$\boldsymbol{f} = \boldsymbol{CNNBackbone}(\boldsymbol{I})$$
$$\boldsymbol{R^c} = \boldsymbol{RegionalProposalNetwork}(\boldsymbol{I})$$
$$\boldsymbol{R^f} = \boldsymbol{Pooling}(\boldsymbol{f}, \boldsymbol{R^c})$$
$$\boldsymbol{B} = \boldsymbol{FullyConnectedLayer^b}(\boldsymbol{R^f})$$
$$\boldsymbol{C} = \boldsymbol{FullyConnectedLayer^c}(\boldsymbol{R^f})$$

where $\boldsymbol{CNNBackbone}$ is the convolutional neural network that is used to extract features $\boldsymbol{f}$ for the input image $\boldsymbol{I}$. $\boldsymbol{Regional\ Proposal\ Network}$ is the Regional Proposal Network which is used to extract coordinates of regions of interest in the image based on feature $\boldsymbol{f}$. $\boldsymbol{R^f}$ is the set of fixed-size feature vectors of regions of interest extracted from $\boldsymbol{f}$. $\boldsymbol{B}$ is the set of predicted boxes which is inferenced from $\boldsymbol{Fully\ Connected\ Layer^b}$. $\boldsymbol{C}$ is the set of predicted categories of corresponding boxes in boxes inference from $\boldsymbol{Fully\ Connected\ Layer^c}$.

3. RESULTS AND DISCUSSION

3.1. Dataset

UIT-Drone21. Experiments were conducted on the dataset UIT-Drone21[1], consisting of 15,370 images. UIT-Drone21 is the largest dataset that includes images captured in Vietnamese traffic contexts and served for tasks such as object detection and object tracking.

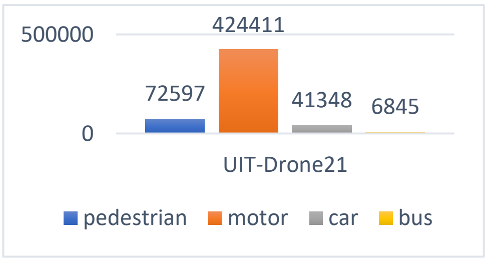

Figure 9. Statistics of the number of objects per vehicle category in the UIT-Drone21 dataset

The dataset is built on four classes: Pedestrian, Motor, Car, and Bus. However, due to the specificity of Viet Nam's traffic, the motor class will account for most of the samples in the

[1] https://github.com/nguyenvd-uit/uit-together-dataset/blob/main/UIT-Drone21.md

dataset. The dataset is divided into three subsets, in which 8,580 images are for training, 1,061 images are for testing, and 5,729 for model evaluation. Notably, for challenging the object detection models, scenarios in the test set do not appear in the train set. These can evaluate the generalization of models. The statistics of the number of objects per vehicle category are illustrated in Figure 9. Some examples of UIT-Drone21 dataset are shown in Figure 10.

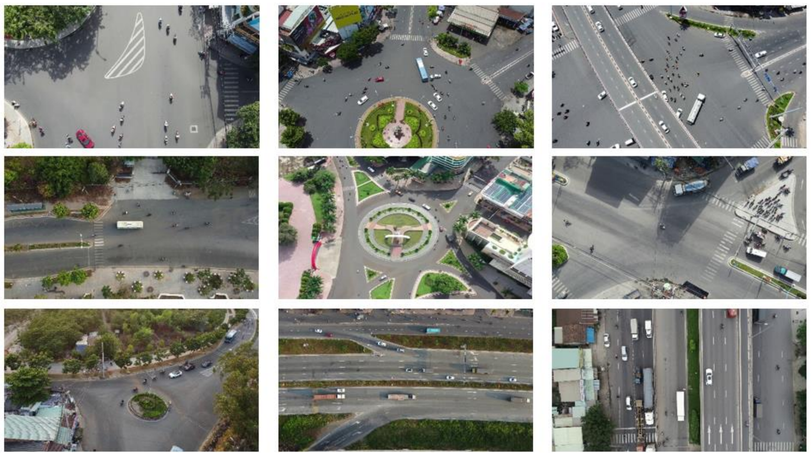

Figure 10. Scenes appearing in the UIT-Drone21 dataset.

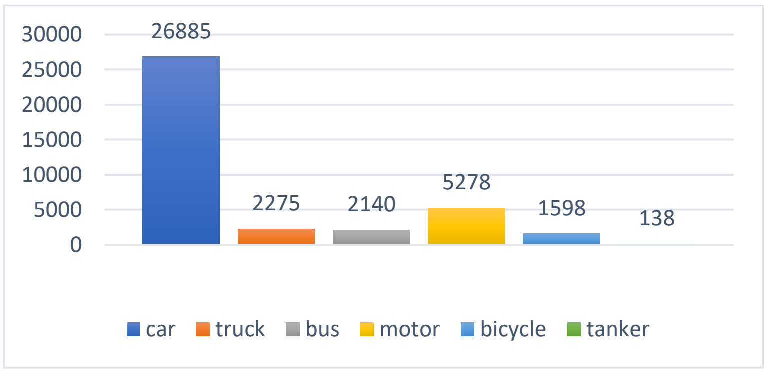

Figure 11. Statistics of the number of objects per vehicle category in the XDUAV dataset.

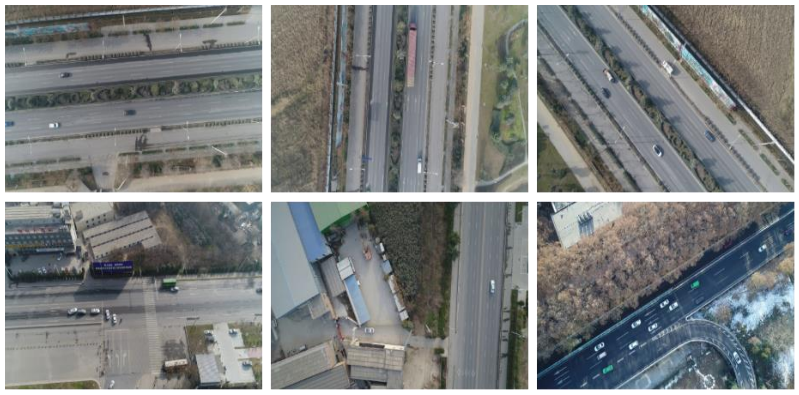

Figure 12. Scenes appearing in the XDUAV dataset.

XDUAV [26]. Extensive experiments were conducted with the XDUAV dataset. XDUAV was collected using a DJI Phantom 2 at an average altitude of 100 m in Xi'an, China's rural and urban areas. The dataset consists of 11 videos which are recorded in resolution 1920×1080, 30 fps. There are 4,344 images, including 3,475 training images and 869 testing images. The categories include car, truck, bus, motor, bicycle, and tanker. The statistics of the number of objects per vehicle category are illustrated in Figure 11. Some examples of the XDUAV dataset are shown in Figure 12.

3.2. Metrics

The test was evaluated on the mean Average Precision (mAP) measure according to the standard of the MS COCO dataset [20]. The mAP is a metric that aggregates AP results across multiple thresholds. Specifically, the mean of 10 IOUs from 50 % to 95 % (the difference is 5 %) is calculated. On the other hand, the accuracy of a specified IoU value (AP at IoU 50 % and AP at IoU 75 %) is also evaluated.

To illustrate better, the mAP metric can be formulated as follows:

$$\mathbf{IoU} = \frac{\boldsymbol{AreaOverlap}(\boldsymbol{b^{gt}}, \boldsymbol{b^{*}})}{\boldsymbol{AreaUnion}(\boldsymbol{b^{gt}}, \boldsymbol{b^{*}})} \#(\mathbf{1})$$

$$\mathbf{AP} = \frac{\boldsymbol{TP_c}}{\boldsymbol{TP_c} + \boldsymbol{FP_c}} \#(\mathbf{2})$$

$$\mathbf{AP_c} = \frac{\mathbf{1}}{\#\mathbf{T}} \sum_{\mathbf{IoU} \in \mathbf{T}} \mathbf{AP}[\mathbf{c}, \mathbf{IoU}] \ \#(\mathbf{3})$$

$$\mathbf{mAP} = \frac{\mathbf{1}}{\#\mathbf{C}} \sum_{\mathbf{c} \in \mathbf{C}} \mathbf{AP_c} \ \#(\mathbf{4})$$

where $\mathbf{IoU}$ is the ratio that indicates the overlap of predicted boxes and ground-truth boxes. AreaOverlap and AreaUnion calculate the overlap area and union area, respectively. $\mathbf{AP}$ is the average precision of category $\boldsymbol{C}$, calculated by the ratio of true positive predictions and the sum of true positive predictions $\boldsymbol{TP_c}$ and false positive predictions $\boldsymbol{FP_c}$. A predicted box that is detected as a $\boldsymbol{TP}$ prediction must satisfy two conditions: 1) its $\mathbf{IoU}$ with ground-truth boxes must be higher than a defined threshold; 2) the predicted category must be accurate with the ground-truth box. $\mathbf{AP}_{\boldsymbol{c}}$ is the average of the list of $\mathbf{AP}$ scores calculated by 10 IoU thresholeds from 50 % to 95 %. An $\mathbf{mAP}$ is the mean average precision of a list of $\mathbf{AP}$ scores per class.

3.3. Experimental Details

To investigate the performance of the architectures outlined Section 2.2 on the UIT-Drone21 dataset, Faster R-CNN [21] provided in MMDetection toolbox [22] was used as the standard method. For each architecture, 24 loops (epochs) with batch size = 1 on 4 Nvidia RTX 2080ti GPUs are trained. The images before training will be resized to 1333 × 800, using random horizontal data augmentation with $\boldsymbol{p} = \mathbf{0.5}$. Other configurations are kept as default. The results are reported in Table 1.

Table 1. Results on vehicle categories on the UIT-Drone21 dataset (%).

Architecture	**Pedestrian**	**Motor**	**Car**	**Bus**
VGG-19	1.1	28.4	41.6	11.8
ResNet-101	2.8	**34.0**	**57.7**	**37.1**
ResNeXt-101	2.2	32.1	53.9	36.4
ResNest-50	**6.6**	**35.1**	**58.0**	29.6
Res2Net-101	3.3	32.9	55.9	**40.6**
HRNetV2_W32	**4.2**	33.0	56.4	32.4
RegNetx_1.6gf	2.7	32.8	56.1	31.2

Table 2. Results on different types of AP metrics on the UIT-Drone21 dataset (%).

Architecture	**AP**	**AP @50**	**AP @75**	AP_s	AP_m	AP_l
VGG-19	20.7	36.5	20.2	2.3	15.2	27.8
ResNet-101	**32.9**	46.5	**38.7**	2.2	20.4	**44.3**
ResNeXt-101	31.2	47.4	35.8	2.8	19.6	41.8
ResNest-50	32.3	**51.7**	34.4	2.6	**23.3**	41.5
Res2Net-101	**33.2**	**48.9**	**38.8**	2.9	**21.0**	**43.5**
HRNetV2_W32	31.5	46.7	37.2	**3.5**	20.4	43.3
RegNetx_1.6gf	30.7	47.5	36.7	**3.0**	20.3	40.3

Table 3. Results on vehicle categories on the XDUAV dataset (%).

Architecture	**Car**	**Truck**	**Bus**	**Motor**	**Bicycle**	**Tanker**
VGG-19	78.9	76.7	71.3	50.4	44.3	55.3
ResNet-101	77.3	77.3	77.6	51.0	43.4	66.4
ResNeXt-101	79.2	79.4	77.6	52.9	**47.6**	**73.8**
ResNest-50	**80.9**	79.9	79.4	**55.1**	**49.5**	68.2
Res2Net-101	79.4	**80.2**	**79.8**	52.2	46.7	**71.1**
HRNetV2_W32	**81.2**	**81.5**	**81.8**	**54.3**	47.5	69.2
RegNetx_1.6gf	77.1	78.0	75.2	52.7	40.7	65.7

Table 4. Results on different types of AP metrics on the XDUAVdataset (%).

Architecture	**AP**	**AP @50**	**AP @75**	AP_s	AP_m	AP_l
VGG-19	62.8	89.7	76.1	34.1	58.7	74.0
ResNet-101	65.5	92.1	77.5	33.0	61.6	75.7
ResNeXt-101	68.4	**94.3**	80.1	35.5	**64.4**	78.0
ResNest-50	**68.8**	93.4	**80.9**	**36.7**	**64.4**	78.1
Res2Net-101	68.2	**94.0**	79.7	35.6	64.3	**79.0**
HRNetV2_W32	**69.3**	93.0	**81.5**	**36.8**	**64.6**	**79.8**
RegNetx_1.6gf	64.9	92.8	77.6	33.3	61.3	75.6

Table 5. Number of parameters of experimental backbones and their corresponding AP scores.

Backbone	**Params (Millions)**	**AP on XDUAV (%)**	**AP on UIT-Drone21 (%)**
VGG-19	37.27	62.8	20.7
ResNet-101	60.14	65.5	**32.9**
ResNeXt-101	98.87	68.4	31.2
ResNest-50	44.35	**68.8**	32.3
Res2Net-101	61.04	68.2	**33.2**
HRNetV2_W32	**27.10**	**69.3**	31.5
RegNetx_1.6gf	**31.49**	64.9	30.7

3.4. Discussion

To thoroughly evaluate the performance of 07 different backbones, AP scores of detection performances of categories are reported and the average AP among them is taken. With the test set of UIT-Drone21, the results are reported in Tables 1 and 2. Detection performances on the test set of XDUAV are reported in Tables 3 and 4. The number of learned parameters of Faster R-CNN is also reported using different backbones in Table 5.

UIT-Drone21. Through the experimental results reported in Table 1 and Table 2, the VGG-19 architecture gives the worst performance on all metrics, which is predictable because large number of parameters in VGG network can increase the risk of overfitting for large training data, so the features extracted from VGG-19 does not include useful information for drone images. The Res2Net-101 architecture gives the best harmonic accuracy when the average AP recorded is $\mathbf{33.2}$ %, this architecture also records the best results on small objects with APs of $\mathbf{2.9}$ %. However, for specific classes, ResNet-101 records better results, when AP on three data categories Pedestrian ($\mathbf{6.6}$ %), Motor ($\mathbf{35.1}$ %) and Car ($\mathbf{58}$ %) gets the best results, in contrast, for the Bus class, AP only reaches 29.6 %, much lower than ResNest-50 ($\mathbf{-11}$ %).Moreover, in the Pedestrian class, ResNest-50 gives the best results with AP of 6.6 %, far ahead of the second highest architecture in this measure, HRNetV2_W32 ($\mathbf{+2.4}$ %). Thereby it can be seen that, to a certain extent, ResNest-50 is quite good at detecting small to medium objects but uite bad for large objects (Bus). On the other hand, in the RegNet architecture, the results are quite low compared to our expectations, when the average AP of this architecture is lower than the original ResNet-101version ($\mathbf{-2.2}$ %). In general, the remaining architectures (except VGG-19 and Res2Net-101) give AP results in the range from $\mathbf{30.7}$ % to $\mathbf{32.9}$ %. But in general, the architectures give poor results on too small feature classes (box area $< \mathbf{32^2}$ pixels). Specifically, when comparing the APs of the pedestrian class, the results are in the range from $\mathbf{1.1}$ % to $\mathbf{6.6}$ %, while the APs of the classes also give results from 2.2 % to 3.5 % only.

XDUAV. In the test set of the XDUAV dataset, VGG19 continues to perform as the worst (62.8 % AP). However, different from the results on the UIT-Drone21 dataset, HRNet performs the best in the XDUAV dataset (69.3 %). In UIT-Drone21, HRNet achieves the fourth-highest results. Its success lies in the multi-branch features learning; it will explore the low-to-high features, in parallel which capture the information high-to-low resolution of images. This is necessary with aerial images because the size of vehicle objects captured from the DJI varies. Therefore, the backbone should explore the pattern from low-to-high features adaptively, and HRNet does this well in the XDUAV dataset. The runner-up is ResNest architecture (68.8 %).

The learning scheme that explores the channel-wise information in deep features helps ResNest explore more patterns in images by depth. Another aspect that can be observed is XDUAV has fewer samples than UIT-Drone21, which is the reason for the poor performance of ResNet-101. Note that the biggest difference between ResNet and VGG is the skip connection and design of the residual block, and they are all deep convolutional neural networks that explore the patterns from low-to-high-level features. Therefore, it also needs a large number of samples to learn. HRNet and ResNest are the networks that explore more aspects of deep features, and it helps them in learning with a small number of samples.

Complexity-Accuracy trade off. This study reports on the number of parameters of models trained with different backbones and their accuracies on the XDUAV dataset and UIT-Drone21 calculated by AP in Table 5. Because HRNet explores low-to-high deep features in parallel, the number of parameters reduces significantly. As can be seen in Table 5, HRNet only includes 27.1 million parameters, while the detection performance is the highest in XDUAV and fourth highest in UIT-Drone21. RegNet is the runner-up in terms of the number of parameters, but the results are lower than ResNet, which is a classical low-to-high convolutional network on both XDUAV and UIT-Drone21. All of these observations are important; they will help the next researchers choose a suitable backbone network to perform object detection tasks in aerial images according to their need.

3.5. Detections Visualization

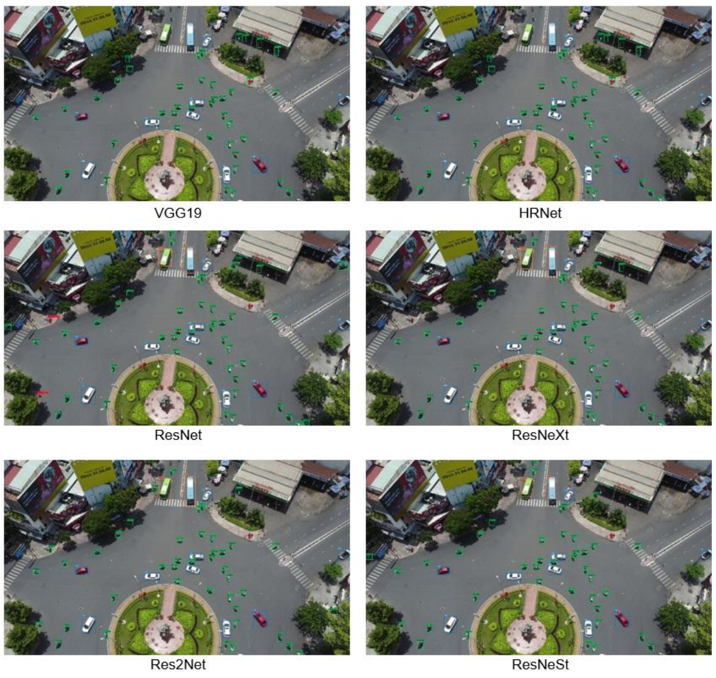

Figure 14. Visualization of detection results on a sample image from UIT-Drone21 dataset of Faster R-CNN models trained using six backbone networks (**8** × zoom in for better illustration).

This study also provides the visualization of detection results of Faster R-CNN models trained using different backbones in Figure 14. Based on the visualizations, ResNeXt, and Res2Net seem to do the bounding boxes regression task better. Indeed, the predicted coordinates from models trained using ResNeXt and Res2Net cover the objects more accurately than the others. However, ResNet appears to have two false positive predictions, which misrecognize the two motor objects as "pedestrians". Besides, HRNet and VGG seem to recognize the pattern better, while the objects under shadow are predicted correctly; however, large objects such as "bus" are misrecognized. Overall, ResNeSt detects quite well; while it does not misclassify the objects, it only misses the objects under the shadow or with no person on them.

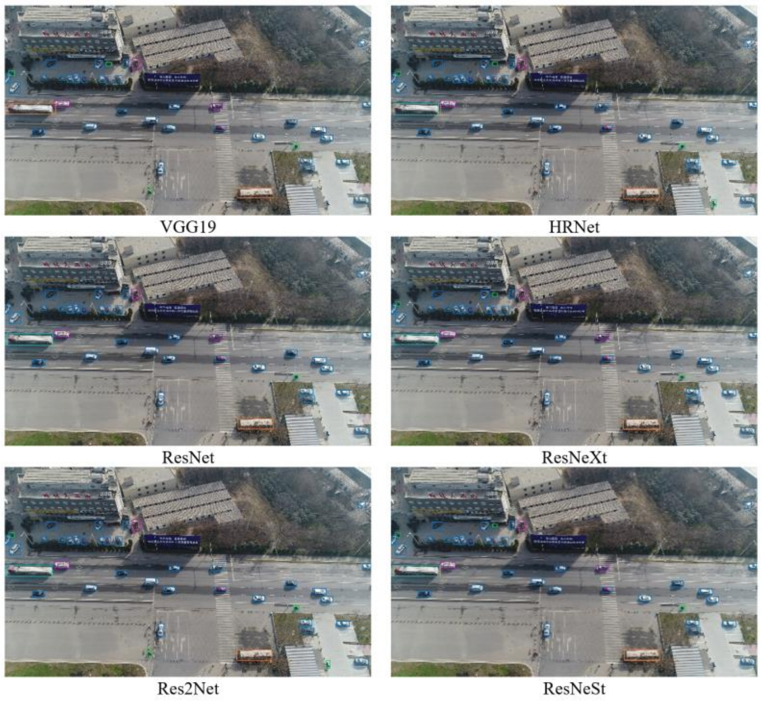

Figure 15. Visualization of detection results on a sample image from XDUAV dataset of Faster R-CNN models trained using six backbone networks ($\mathbf{8}\ \times$ zoom in for better illustration).

4. CONCLUSIONS

In conclusion, our study investigated seven different deep learning network architectures for object detection using the Faster R-CNN method on the UIT-Drone21 and XDUAV datasets. Our findings revealed that while simple network architectures like VGG-19 were prone to overfitting on large datasets like UIT-Drone21, more complex architectures resulted in improved performance, with a 10 % to 12 % increase in results. Notably, ResNest-50 demonstrated good performance in detecting small and medium objects, but showed lower performance on larger objects compared to other architectures. Furthermore, while Res2Net-101 did not achieve high results for individual classes, the overall results were the most consistent, with an average AP of 33.2 %, the highest among all the architectures evaluated. On the XDUAV dataset, which had fewer samples compared to UIT-Drone21, HRNet and ResNest architectures performed well, despite having lower numbers of learned parameters. HRNet emerged as the best-performing architecture with the highest AP of 69.3 % on the XDUAV dataset. These findings provide insights into the strengths and weaknesses of different deep learning network architectures for object detection in the context of UIT-Drone21 and XDUAV datasets. Further research could explore optimization strategies and fine-tuning techniques to improve the performance of these architectures for specific application scenarios.

Acknowledgements. This research is funded by Vietnam National University HoChiMinh City (VNU-HCM) under grant number B2023-26-01**.**

CRediT authorship contribution statement. Khang Nguyen: Methodology, Experiments, Investigation, Formal analysis, Writing paper.

Declaration of competing interest. The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

1. Chung Q. M., Le T. D., Dang T. V., Vo N. D., Nguyen T. V., and Nguyen K. - Data Augmentation Analysis in Vehicle Detection from Aerial Videos, Proceedings of International Conference on Computing and Communication Technologies, Ho Chi Minh City, Viet Nam, 2020, pp. 1-3.
2. Vinh Long Phan, Nguyen D. Vo, and Khang Nguyen, - Detecting objects in images that are limited in visibility by fog. The 23nd National Conference on Electronics, Communications and Information Technology, Ho Chi Minh City, Viet Nam, 2020, pp 44-49.
3. Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Haibin Ling, Qinghua Hu, Qinqin Nie, Hao Cheng, Chenfeng Liu, Xiaoyu Liu *et al*, - VisDroneDET2018: The Vision Meets Drone Object Detection in Image Challenge Results, Proceedings of European Conference on Computer Vision, Munich, Germany, 2018, pp. 437-468.
4. Hongyang Yu, Guorong Li, Weigang Zhang, Qingming Huang, Dawei Du, Qi Tian, and Nicu Sebe. - The Unmanned Aerial Vehicle Benchmark: Object Detection, Tracking and Baseline, Int J Comput Vis **128** (2020) 1141-1159. https://doi.org/10.1007/s11263-019-01266-1.
5. Yang J., Xie X., Shi G., Yang W. - A Feature-Enhanced Anchor-Free Network for UAV Vehicle Detection. Remote Sensing **12** (17) (2020) 2729-2809. https://doi.org/10.3390/rs12172729.
6. Ilker Bozcan and Erdal Kayacan. - Au-air: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance. In: IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 8504-8510.
7. Karen Simonyan and Andrew Zisserman. - Very deep convolutional networks for largescale image recognition. Proceedings of International Conference on Learning Representations, San Diego, CA, USA, 2015, pp. 1-14.
8. He K., Zhang X., Ren S., and Sun J. - Deep Residual Learning for Image Recognition, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016, pp. 770-778.
9. Xie S., Girshick R., Dollár P., Tu Z., and He K. - Aggregated Residual Transformations for Deep Neural Networks, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii,USA, 2017, pp. 5987-5995.
10. Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr - Res2Net: A New Multi-Scale Backbone Architecture, IEEE Trans. Pattern Anal. Mach. Intell. **43**, (2) (2021) 652-662. https://doi.org/10.1109/TPAMI.2019.2938758.

11. Zhang Hang, Wu Chongruo, Zhang Zhongyue, Zhu Yi, Lin Haibin, Zhang Zhi, Sun Yue He, Tong Mueller, Jonas Manmatha *et al*. - Resnest: Split-attention networks. Proceedings of the Conference on Computer Vision and Pattern Recognition, New Orleans, Louisiana, USA, 2022, pp. 2735-2745.
12. Wang Jingdong, Sun Ke Cheng, Tianheng Jiang, Borui Deng, Chaorui Zhao, Yang Liu, Dong Mu, Yadong Tan, Mingkui Wang, Xinggang *et al.* - Deep High-Resolution Representation Learning for Visual Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence **43** (10) (2021), 3349-3364, doi: 10.1109/TPAMI.2020.2983686.
13. Xu J., Pan Y., Pan X., Hoi S., Yi Z., and Xu Z. - RegNet: Self-Regulated Network for Image Classification, in IEEE Transactions on Neural Networks and Learning Systems **33** (3) (2022) 1-6. doi: 10.1109/TNNLS.2022.3158966.
14. Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Mądry - How does batch normalization help optimization?, Proceedings of the 32nd International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 2018, pp. 2488-2498.
15. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun - Faster R-CNN: towards real-time object detection with region proposal networks, Proceedings of the 28th International Conference on Neural Information Processing Systems, Cambridge, MA, USA, 2015, pp. 91-99.
16. Girshick R. - Fast R-CNN, Proceedings of IEEE International Conference on Computer Vision (ICCV), Las Condes, Chile, 2015, pp. 1440-1448.
17. Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn and Andrew Zisserman - The PASCAL Visual Object Classes (VOC) Challenge, Int J Comput Vis **88** (2010) 303-338, https://doi.org/10.1007/s11263-009-0275-4.
18. Mark Everingham, Ali Eslami S. M., Luc Van Gool, Christopher K. I. Williams, John Winn and Andrew Zisserman - The PASCAL Visual Object Classes Challenge: A Retrospective. Int J Comput Vis **111** (2015) 98-136 https://doi.org/10.1007/s11263-014-0733-5.
19. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei - ImageNet Large Scale Visual Recognition Challenge, Int J. Comput Vis. **115** (2015) 211-252, https://doi.org/10.1007/s11263-015-0816-y.
20. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and Lawrence Zitnick C. - Microsoft COCO: Common Objects in Context, European Conference on Computer Vision, Zurich, Switzerland, 2014, pp. 740-755.
21. Chen Kai, Wang Jiaqi, Pang Jiangmiao, Cao Yuhang, Xiong Yu, Li Xiaoxiao Sun, Shuyang Feng, Wansen Liu Ziwei, and Xu Jiarui *et al* - MMDetection: Open mmlab detection toolbox and benchmark, In:arXiv preprint arXiv:1906.07155, 2019, pp. 1-13.
22. Yang J., Xie X., Shi G., and Yang W. - A feature-enhanced anchor-free network for UAV vehicle detection. Remote Sensing **12** (17) (2020) 2729-2746, https://doi.org/10.3390/rs12172729