Influence of Aerial Image Resolution on Vehicle Detection Accuracy

Donata Gliaubičiūtė [1], Rokas Janavičius [1], Aušra Gadeikytė [1] and Lukas Paulauskas [1]

[1] *Kaunas University of Technology, Studentų g. 50, Kaunas, 51368, Lithuania*

Abstract

Nowadays, the engineering application of vehicle detection from aerial images is a challenging task due to the particularity of perspective, the small size of the objects, and the complex background. This research aim is to investigate low-resolution aerial images of vehicles that can be utilized for vehicle detection using machine learning models. The research work was conducted using one-stage deep learning-based object detection algorithms YOLOv5, YOLOv7, and YOLOv8 on the two datasets (COWC and VEDAI) that addressed the task of small vehicle detection. For the training of the models, available pre-trained weights were used as a starting point, and then each model was trained by utilizing transfer learning. The obtained results of the study demonstrated that by reducing the image pixel ratio every 5 cm per pixel from 12.5x12.5 to 27.5x27.5 cm per pixel, the accuracy of the object detection models decreases by an average of 3.51%. When the pixel ratio varies from 30x30 to 32.5x32.5 cm per pixel, the accuracy of the models drops by an average of 2.33% on the COWC dataset and 42.4% on the VEDAI dataset.

Keywords

Vehicle detection, aerial images, convolutional neural networks, pixel ratio, YOLOv5, YOLOv7, YOLOv8

1. Introduction

Object detection involves finding the numerous objects in the images and identifying their locations. Recently object detection has been considered one of the most challenging tasks in computer vision due to the appearance of objects varying greatly depending on various circumstances, such as image capture technologies. One such technology is UAVs (Unmanned Aerial Vehicles) - a critical enabler for a wide range of applications including automated driving, crowd flow counting, topographic exploration, environmental pollution monitoring, etc.

Over the years, a lot of effort has been put into identifying vehicles and other small targets in the images that UAVs collect. According to B. Wang and B. Xu in 2021 [1] the most common difficulties of object detection in aerial images are:

- The particularity of perspective. Since aerial images are typically taken from above, the objects have less texture features [1]. As a result, the targets can be easily mistaken with other objects [2].
- The size of the objects. In aerial images, the objects are quite small (composed of only 15 to 30 pixels). Moreover, Convolutional Neural Networks (CNNs) down-sampling layers minimize the amount of information that each object has. For instance, after four down-sampling layers, a 24x24 pixels object maintains only around one pixel in feature maps, making it challenging to identify small objects from the background [1].
- The complexity of the background. Usually, aerial images might cover an area of several square kilometers. The presence of different backgrounds in this receptive field, such as the countryside, mountains, urban areas, etc., interfere with the object detection process [1].

28th Conference on Information Society and University Studies (IVUS'2023), May 12, 2023, Kaunas, Lithuania

EMAIL: donata.gliaubiciute@ktu.lt (A. 1); rokas.janavicius@ktu.edu (A. 2); ausra.gadeikyte@ktu.lt (A. 3); lukas.paulauskas@ktu.lt (A. 4)

Deep learning algorithms have enabled vehicle identification technologies to attain very high performance. The deep convolutional neural network may use the dataset to train and enhance its model independently. Deep learning-based object detection algorithms that are frequently utilized may be split into two categories: one-stage and two-stage detectors [3].

Two-stage object detection algorithms Faster R-CNN (Region-based Convolutional Neural Networks) divide the target detection into two stages, that is, first use the Region Proposal Network (RPN) to extract candidate target information, and then use the detection network to complete the location and category of candidate targets [3]. One-stage object detection algorithms such as YOLO (You Only Look Once) do not require to use RPN, and directly generate the location and category information of the target through the network, which is an end-to-end target detection algorithm. Therefore, the single-step target detection algorithms have a faster detection speed [3].

One of the first methods to use convolutional neural networks for object detection and to show off their impressive capabilities is region-based CNN (R-CNN) [4]. In R-CNN, a selective search algorithm selects image regions that could contain target objects, and then the CNN is used to map the target objects in the suggested region. Fast R-CNN used an SPP (Spatial Pyramid Pooling) layer and a RoI pooling layer to increase accuracy and runtime over R-CNN [4]. Unlike the R-CNN, which classifies each region proposal independently, Fast R-CNN computes a feature map from a full image only once and then categorizes region proposals by projecting each one onto that feature map. Moreover, the Fast R-CNN algorithm uses a time-consuming selective search method to look for region suggestions in a target image. In Faster R-CNN [4], the selective search is replaced with RPN, which calculates region proposals from an input image. Faster R-CNN is 900% faster than Fast R-CNN and is made up completely of deep learning networks. Directly connecting the RPNs and the classifier network would help Faster R-CNN to further advance [5]. In 2017, T. Tang et al. designed an improved Faster-RCNN to solve the difficulties of locating the positions of small vehicles and classifying the vehicle from the background [6].

In one pass, YOLO predicts and categorizes bounding boxes of objects. An image is initially divided into non-overlapping grids through YOLO. For each cell in the grids, YOLO fore-casts the likelihood that an object will be present, the coordinates of the anticipated box, and the object's class. Each cell's bounding boxes and their confidence scores are predicted by the network. The network then determines the classes' probabilities for each cell [7]. The first version of YOLO, coined YOLOv1, reportedly achieves a faster inference time, but lower accuracy compared to a single-shot detector [8]. In order to increase the speed and accuracy of detection, YOLOv2 was suggested. Anchor boxes are used in YOLOv2 together with convolutional layers that are not fully connected [8]. The accuracy of the network is further increased by YOLOv2 using batch normalization (BN) and a high-resolution classifier. YOLOv3 [9] uses three detection levels and predicts three box anchors for each cell. To extract feature maps, YOLOv3 adds a deeper backbone network (Darknet-53) to the system. Due to the addition of more layers, the prediction is slower than with YOLOv2. Many technical improvements were made in YOLOv4 while maintaining its computational efficiency. The improvements slightly affected the inference time but significantly increased accuracy [10].

According to A. Ammar et al. in 2021 [2], vehicle detection is possible for different data sets with an accuracy from 85.3% to 98%. However, vehicle detection is still challenging when aerial images are small in size and contain a large number of objects. It might cause information loss when convolution operations are performed [2].

There are different aerial image data sets such as OIRDS [11], PUCPR [13], COWC[14], and VEDAI [15] that might be used for the investigation of vehicle detection. The overhead imagery research data set (OIRDS) project produced a data set with almost 1,000 labeled images suitable for developing automated vehicle detection algorithms [11]. “Overhead imagery research data set” contains approximately 1,800 labeled targets. For each target, there are over 30 annotations and over 60 statistics, that describe the target within the context of the image. Images sizes range from 256×256 pixels to 512×512 pixels. The dataset contains five classes of vehicles (“truck”, “pickup”, “car”, “van” and “unknown”). Annotations give information such as color and distance to the ground [11]. On the other hand, this database is hard to apply to benchmark target detection algorithms because there is no defined evaluation protocol, the dataset is obtained by aggregating multiple sources of images (20 different sources), and does not have sufficient statistical regularity. These issues make the results difficult to reproduce, preventing other researchers from making any comparisons with this dataset [11, 12]. It was tried to split this database (easy, medium, and hard). However, the precise set of images in each split was not defined, preventing the reproduction of results [12].

Approximately 17,000 photos in the Pontifical Catholic University of Parana Dataset (PUCPR) are devoted to car counting in settings of various parking lots. The dataset includes details about 16,456 vehicles. The aerial images in the collection were taken from a drone view at a height of about 40 meters. The image set is annotated by a bounding box per car. All labeled bounding boxes have been well recorded with the top-left and bottom-right points [13].

Nearly 90,000 automobiles were collected using a drone from 4 distinct parking lots for the Car Parking Lot Dataset (CARPK). This is a large dataset with an emphasis on automobile counting in various parking lots. The bounding box for each car is annotated in the image set. Top-left and bottom-right points have been accurately recorded for each labeled bounding box. It is supporting object counting, object localizing, and further investigations with the annotation format in bounding boxes [13].

The purpose of this study is to investigate the change in the accuracy of object detection models for detecting vehicles in aerial images when the resolution of the images fed to the models is reduced. The findings of this study might be useful when certain situations require quick and real-time decision-making regarding the distribution of vehicles in a geographic space if collecting aerial photographs of this space is available. When it is known what minimum resolution and object detection models are sufficient to obtain acceptable results from aerial images, it is possible to save time for flying UAVs, processing information, and presenting results. The models selected for this study are one of the best-performing one-stage detectors (YOLOv5, YOLOv7, YOLOv8) that reach high accuracy and speed when applied to object detection tasks.

2. Methods

2.1. Data Preprocessing

The research was conducted using object detection algorithms on the COWC and VEDAI datasets that address the task of small vehicle detection. The cars overhead with context (COWC) dataset contains many unique cars (32,716) from six different image sets, each covering a different geographical location and produced by different images [14]. The images cover regions from Toronto (Canada), Selwyn (New Zealand), Potsdam and Vaihingen (Germany), Columbus, and Utah (The United States). The COWC dataset provides data from overhead at 15 cm (about 5.91 in) per pixel resolution at ground (all data is EO) and is designed to be challenging for detection models. Furthermore, it contains 58,247 usable negative targets, many of which have been hand-picked objects similar to cars such as boats, trailers, bushes, and A/C units. To compensate for the additional difficulty, the context was included around targets. Context can help identify something that may not be a car or confirm it is a car. In general, the idea is to allow a deep learner to decide the weight between context and appearance such that something that looks very much like a car is detected even if it is in an unusual place.

The vehicle detection in aerial imagery (VEDAI) database includes various back-grounds such as woods, cities, roads, parking lots, construction sites, or fields. In addition, the vehicles to be detected have different orientations and can be altered by specular spots, occluded, or masked. Each image is available in several spectral bands and resolutions. VEDAI set has 2950 cars in 512x512 and 1024x1024 images. The dataset with 1024x1024 resolution images has a resolution of 12.5cm×12.5cm per pixel. Likewise, 512x512 resolution images have a resolution of 25cm×25cm per pixel. The images were taken during the spring of 2012. Raw images have 4 uncompressed color channels. The dataset has nine different classes of vehicles: “plane”, “boat”, “camping car”, “car”, “pick-up”, “tractor”, “truck”, “van”, and “other”. Two meta-classes are also defined and considered in the experiments. The “small land vehicles” class has the “car”, “pick-up”, “tractor”, and “van” classes included, and the “large land vehicles” class contains the “truck” and “camping car” classes [15].

When preparing the VEDAI dataset for the experiments, these classes have been dropped: “plane”, “boat”, “camping car”, “tractor”, “truck”, and “other”. These changes were done in order to have comparable visual data about the vehicles. Only one class "Car" was left in both datasets. In the COWC dataset, a class of negative samples was removed. Figure 1. depicts a histogram of different numbers of cars in the VEDAI dataset.

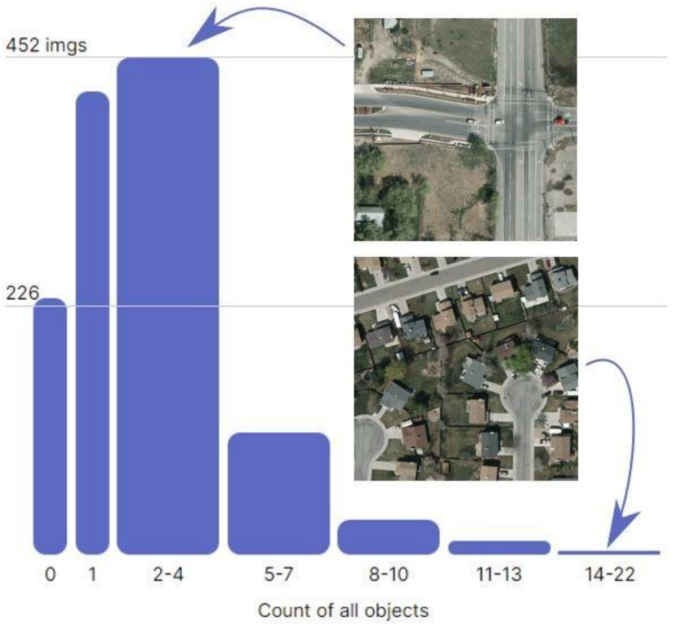

Figure 1: Histogram of "Car” class count by the image in the VEDAI dataset

The Roboflow [16] platform was used to manage the data sets. For each evaluation sample, the dataset images were proportionally reduced in size to achieve the desired centimeters per pixel ratio. With the use of the Roboflow platform, an analysis of the data was performed, resulting in the following findings:

- COWC dataset contains 32810 annotations. The average number of bounding boxes per image is 18.8. There are 800 images in the dataset with null examples.
- VEDAI dataset contains 2807 annotations. The average number of bounding boxes per image is 2.2. There are 233 images in the dataset with null examples.
- Figure 1. demonstrates that the VEDAI dataset has mostly from 2 to 4 vehicles per image and the COWC dataset has from 2 to 49 vehicles per image.

2.2. Object Detection Algorithms

The YOLOv5 adopted the concept of anchor boxes to speed up the R-CNN algorithm and abandoned the use of manually chosen anchor boxes was released in 2020. To get a better prior value, K-means clustering was done on the bounding box dimensions [17].

Introduced in 2022 YOLOv7 surpassed all known object detectors created before in both speed and accuracy in the range from 5 FPS to 160 FPS and had the highest accuracy 56.8% AP among in that time all known real-time object detectors with 30 FPS or higher on GPU V100. YOLOv7 was trained on the MS COCO dataset from scratch without using any other datasets or pre-trained weights. The YOLOv7 model preprocessing method is integrated with YOLOv5, and the use of Mosaic data augmentation is suitable for small object detection [17].

The most recent group of YOLO-based object detection models is called YOLOv8. For detection, segmentation, and classification, there are five models (Nano, Small, Medium, Large, and Xtra Large) in each category of the YOLOv8. The fastest and smallest is YOLOv8 Nano, and the slowest and most accurate is YOLOv8 Extra Large (YOLOv8x). All of the YOLOv8 models had improved throughput when compared to other YOLO models trained at 640 image resolution while using around the same amount of parameters [17]. The effectiveness of object detection on 640 image size between YOLOv8 and YOLOv5 is summarized in Table 1.

Table 1
Object detection performance comparison between YOLOv8 and YOLOv5

Model Size	YOLOv5	YOLOv8	Difference
Nano	28	37.3	+33.21%
Small	37.4	44.9	+20.05%
Medium	45.4	50.2	+10.57%
Large	49	52.9	+7.96%
Xtra Large	50.7	53.9	+6.31%

3. Evaluation metrics

For the evaluation of the performance of the models, the following evaluation metrics [18] were used in this study:

- Total (pre-process + inference + NMS) detection speed in milliseconds;
- Precision;
- Recall;
- Mean average precision (mAP) calculated at Intersection over Union (IoU) [19] threshold 0.5 (mAP@0.5);
- mAP over different IoU thresholds from 0.5 to 0.95 with step 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95);
- F1 score;
- Confusion Matrix.

The precision metric Precision stands for the proportion of positive samples in the samples with positive prediction results (see calculation formula (1)).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1}$$

where TP denotes true positive samples, FP – false positive samples.

The recall represents the prediction result as the proportion of the actual positive samples in the positive samples to the positive samples in the whole sample. The calculation formula (2), where FN stands for false negative samples can be defined as

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

The F1 score is the weighted average of precision and recall, calculated (3) as follows:

$$F1 = \left(\frac{2}{Recall^{-1} + Precision^{-1}}\right) = 2 \cdot \frac{Precision \cdot Recall}{\mathrm{Precision} + \mathrm{Recall}} \tag{3}$$

Precision reflects the model’s ability to distinguish negative samples. The higher the precision, the stronger the model’s ability to distinguish negative samples. Recall reflects the model’s ability to identify positive samples. The higher the recall, the stronger the model’s ability to detect positive samples. The F1 score is a combination of the two. The higher the F1 score, the more robust the model.

For object detection tasks, the most common way to determine if a single object proposal is correct is by using the Intersection over Union (IoU) metric [19]. It takes the set A of proposed object pixels and set of true object pixels B and calculates the intersection area. The calculation formula (4) is as follows:

$$IoU(A, B) = \frac{A \cap B}{\mathrm{A} \cup \mathrm{B}} \tag{4}$$

In most cases, an IoU of over 0.5 means that the object was detected, otherwise it was a failure .

$$mA\mathrm{P} = \frac{1}{\mathrm{n}} \sum_{k=1}^{k=n} AP_k \; Recall = \frac{TP}{\mathrm{TP} + \mathrm{FN}} \tag{5}$$

The mean of average precision (AP) values are calculated over recall values from 0 to 1. Average precision is calculated as the weighted mean of precisions at each threshold and the weight is the increase in recall from the prior threshold. The mean of average precision is the average AP of each class. The mAP is evaluated (5) by finding each class‘s average precision (AP) and then averaging over all specified classes.

4. Experiments and results

During the experiments, the processes indicated in Fiure. 2. were carried out, which consisted of obtaining images with their original pixel ratio, resizing images, dividing the data sets into training, validation, and testing sets, running training sets to models, custom training models using pre-trained weights, evaluating models with testing sets.

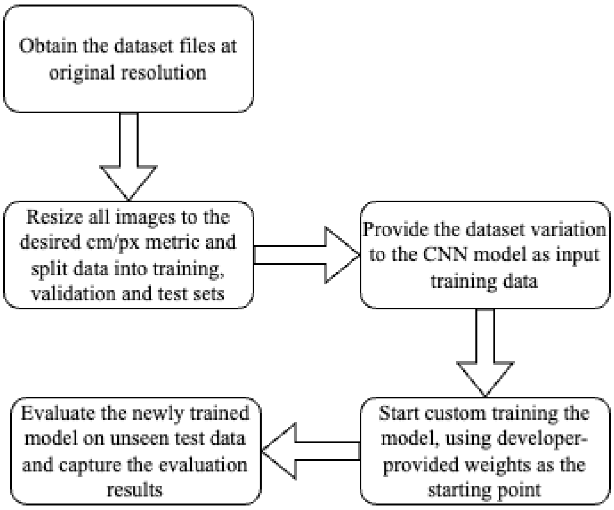
Obtain the dataset files at original resolution
Resize all images to the desired cm/px metric and split data into training, validation and test sets
Provide the dataset variation to the CNN model as input training data
Start custom training the model, using developer-provided weights as the starting point
Evaluate the newly trained model on unseen test data and capture the evaluation results

Figure 2: Process flow diagram of experiments

For the training of the models, available pre-trained weights were used as a starting point for each variation of the datasets („yolov5s.pt“ for the YOLOv5, „yolov7.pt“ for the YOLOv7, and „yolov8s.pt“ for YOLOv8). Afterward, each model was trained by utilizing transfer learning for 100 epochs. After training, testing was performed, by providing models with unseen images and capturing their inference time, as well as available precision metrics. Regarding the dataset splits, both datasets featured the same data split: 70% training, 20% validation, and 10% testing. The hardware used for training and testing each variation is provided in the Table 2.

Table 2
Hardware used for the experiments

Hardware	Specification
CPU model	Intel(R) Xeon(R)
CPU frequency	2.30GHz
GPU model	Nvidia T4
GPU VRAM	16GB
GPU Memory Clock	1.59GHz

The following issues were observed during models training:

- YOLOv8 training on COWC at 15x15 cm per pixel variation would display uncommon behavior in several repeated runs, where model training performance drops in the middle of training, and the precision drops instantly.

- YOLOv7 training on VEDAI at 20x20 cm per pixel variation also exhibits inconsistent behavior, as after the first 20 training epochs the training metrics fluctuate and drop, resetting the training progress.

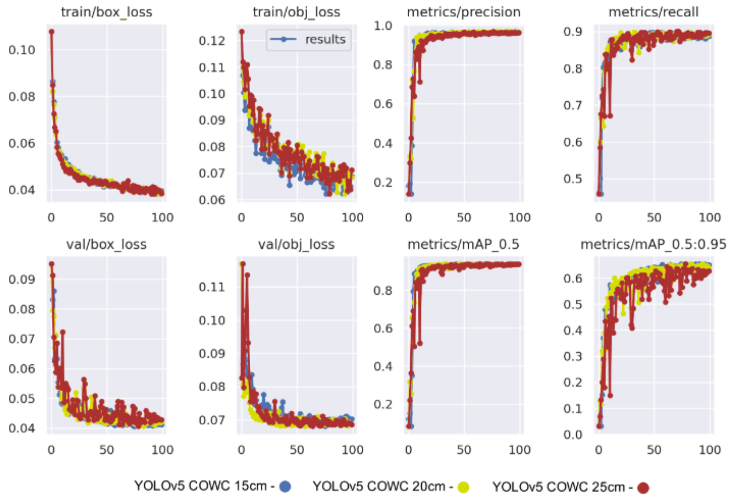

Figure 3: Training YOLOv5 on COWC results

The results of training YOLOv5 on the COWC dataset are provided in Figure 3. The testing results (see Table 3.) demonstrate the performance of YOLOv5, YOLOv7, and YOLOv8 models when trained and tested at various image pixel ratios. Specifically, the models were evaluated at 15, 20, 25, 27.5, 30, and 32.5 cm per image pixel resolution. However, it should be noted that the original pixel ratio of the VEDAI dataset is 12.5 cm per pixel, while the initial pixel ratio of the COWC dataset is 15 cm per pixel, meaning that the original resolution of the datasets did not match in the starting evaluation phase. As a result, the findings of the VEDAI and COWC datasets were not directly comparable during the experiments when the pixel ratio was 12.5 cm per pixel.

It was found that on the VEDAI dataset P, R, and mAP indicators values are smaller than on the COWC dataset (see Table 3). However, the testing speed was higher. When the datasets had the lowest pixel ratio, which is 32.5 cm per pixel, the YOLOv7 model obtained better test results on the COWC dataset than the YOLOv5. However, YOLOv5 showed significantly higher values on the VEDAI dataset than YOLOv7. For the COWC dataset, YOLOv7 achieved mAP score of 0.93, while YOLOv5 achieved 0.889 under the same conditions. For the VEDAI dataset, YOLOv7 achieved mAP score of 0.029, while YOLOv5 achieved 0.248. YOLOv7 had a faster total detection speed across all datasets and their variations. Figure 4. display the change of accuracy metric of all the models changed with the COWC or VEDAI datasets when the image pixel ratio was reduced by 5 or 2.5 cm, while Figure 5. graphs the detection speed results for the same evaluation task.

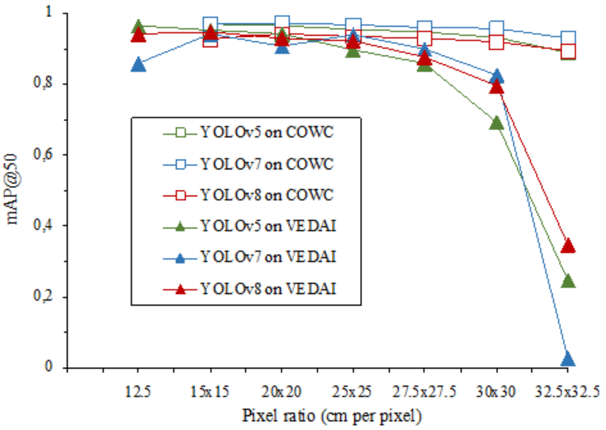

Figure 4: Graph of dependence of mAP@50 indicator and image pixel ratio

Table 3
Testing YOLOv5 YOLOv7 and YOLOv8 models on COWC and VEDAI datasets results

Model	Dataset	Centimeters per pixel	Detection Speed (ms)	P	R	mAP@ 50	mAP@ 50-95
v5	COWC	32.5 x 32.5 cm	10.9	0.87	0.79	0.88	0.45
	VEDAI		9.5	0.41	0.31	0.24	0.07
v7	COWC		9.7	0.93	0.84	0.93	0.56
	VEDAI		5.3	0.11	0.11	0.03	0.01
v8	COWC		6.6	0.89	0.83	0.89	0.53
	VEDAI		5.1	0.42	0.41	0.34	0.11
v5	COWC	30 x 30 cm	14.8	0.91	0.87	0.93	0.55
	VEDAI		9.8	0.71	0.64	0.69	0.31
v7	COWC		12.8	0.94	0.91	0.95	0.61
	VEDAI		8.8	0.83	0.74	0.82	0.35
v8	COWC		7.6	0.93	0.85	0.91	0.60
	VEDAI		5.7	0.72	0.73	0.79	0.39
v5	COWC	27.5 x 27.5 cm	18.2	0.95	0.88	0.94	0.58
	VEDAI		13.9	0.83	0.77	0.86	0.42
v7	COWC		14	0.95	0.90	0.96	0.62
	VEDAI		9.8	0.8	0.86	0.89	0.48
v8	COWC		10.4	0.94	0.87	0.93	0.62
	VEDAI		6	0.87	0.78	0.87	0.49
v5	COWC	25 x 25 cm	23.6	0.96	0.9	0.95	0.61
	VEDAI		15.3	0.88	0.78	0.89	0.44
v7	COWC		18	0.95	0.93	0.96	0.64
	VEDAI		11.7	0.90	0.88	0.94	0.5
v8	COWC		12.5	0.94	0.88	0.93	0.63
	VEDAI		9.2	0.88	0.86	0.92	0.55
v5	COWC	20 x 20 cm	30.2	0.96	0.92	0.96	0.65
	VEDAI		20.9	0.87	0.87	0.94	0.54
v7	COWC		30.5	0.96	0.94	0.97	0.67
	VEDAI		20.6	0.88	0.82	0.90	0.50
v8	COWC		18.1	0.95	0.89	0.94	0.66
	VEDAI		14.4	0.94	0.85	0.93	0.58
v5	COWC	15 x 15 cm	46.3	0.96	0.93	0.97	0.67
	VEDAI		36.8	0.96	0.87	0.95	0.55
v7	COWC		40.5	0.96	0.94	0.97	0.65
	VEDAI		32.6	0.95	0.84	0.94	0.55
v8	COWC		27.3	0.93	0.88	0.92	0.60
	VEDAI		25.9	0.91	0.90	0.94	0.61
v5	VEDAI	12.5 x 12.5 cm	41.9	0.95	0.89	0.96	0.57
v7			38.5	0.81	0.84	0.86	0.47
v8			30.1	0.91	0.90	0.94	0.63

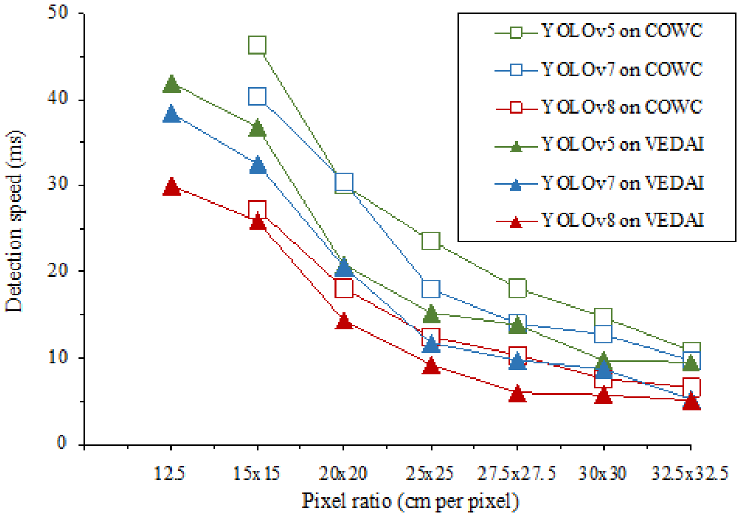

Figure 6: Graph of dependence of detection speed and image pixel ratio

5. Conclusion

In this paper, the change in the accuracy of vehicle detection using reduced pixel ratio aerial images was investigated. When models object detection on aerial images mAP indicator results from 0.86 to 0.97 needed to be reached, the usage of a pixel ratio of 12.5x12.5 to 27.5x27.5 cm per pixel, and YOLOv5, YOLOv7, and YOLOv8 object detection algorithms were proposed. When investigating the dependence of aerial image resolution on the performance of object detection models, it was observed that one-stage object detection algorithms such as YOLOv5, YOLOv7, and YOLOv8 achieve an average of 3.51% lower mAP scores when the image pixel ratio is reduced every 5 cm per pixel from 12.5x12.5 to 27.5x27.5 cm per pixel.

The YOLOv8 model had the most stable results among other models, decreasing by an average of 0.24% when tested on the COWC dataset from 12.5x12.5 to 27.5x27.5 cm per pixel image pixel ratio. The YOLOv5 model achieved an average mAP reduction of 9.6% with images from 12.5x12.5 to 27.5x27.5 cm per pixel image pixel ratio from the COWC dataset, significantly lagging behind the other tested models. All models performed better on the COWC and not on the VEDAI dataset during testing. The VEDAI dataset only had 2807 annotated vehicles, while the COWC dataset contained 32810 annotated vehicles, which may have influenced the testing results. Additionally, vehicles in the VEDAI dataset were labeled, even if some of them were partially hidden by other objects or just partially visible at the edges of the image. When models were tested with the COWC dataset, results dropped by an average of 0.68% and by an average of 6.35% with the VEDAI dataset when using image pixel ratios between 12.5x12.5 and 27.5x27.5 cm per pixel. More pronounced changes in the mean average accuracy of the models are noticeable when the pixel ratio varies from 30x30 to 32.5x32.5 cm per pixel. When the pixel ratio was 30x30 cm per pixel then the accuracy dropped by an average of 1.42% on the COWC dataset and 11.96% on the VEDAI dataset. When the pixel ratio was 32.5x32.5 cm per pixel then the accuracy dropped by an average of 3.23% on the COWC dataset and 72.84% on the VEDAI dataset.

Future research should include other one-stage and two-stage deep learning-based object detection algorithms and experiments with more image-pixel ratio options in order to collect more data on the change in accuracy of the object detection models.

6. Acknowledgements

The analysis was carried out as a group work of the subject P170M113 Applied Research Project in Kaunas University of Technology Masters' programme Artificial Intelligence in Computer Science. We also thank dr. Andrius Kriščiūnas for support, useful suggestions, and mentoring process.

7. References

[1] B. Wang and B. Xu, “A feature fusion deep-projection convolution neural network for vehicle detection in aerial images,” PLoS One, vol. 16, no. 5, p. e0250782, May 2021, doi: 10.1371/journal.pone.0250782.

[2] A. Ammar, A. Koubaa, M. Ahmed, A. Saad, and B. Benjdira, “Vehicle Detection from Aerial Images Using Deep Learning: A Comparative Study,” Electronics (Basel), vol. 10, no. 7, p. 820, Mar. 2021, doi: 10.3390/electronics10070820.

[3] P. Soviany and R. T. Ionescu, “Optimizing the Trade-Off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction,” in 2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Sep. 2018, pp. 209–214. doi: 10.1109/SYNASC.2018.00041.

[4] R. Girshick, “Fast R-CNN,” in 2015 IEEE International Conference on Computer Vision (ICCV), Dec. 2015, pp. 1440–1448. doi: 10.1109/ICCV.2015.169.

[5] Y. Koga, H. Miyazaki, and R. Shibasaki, “A CNN-Based Method of Vehicle Detection from Aerial Images Using Hard Example Mining,” Remote Sens (Basel), vol. 10, no. 1, p. 124, Jan. 2018, doi: 10.3390/rs10010124.

[6] T. Tang, S. Zhou, Z. Deng, H. Zou, and L. Lei, “Vehicle Detection in Aerial Images Based on Region Convolutional Neural Networks and Hard Negative Example Mining,” Sensors, vol. 17, no. 2, p. 336, Feb. 2017, doi: 10.3390/s17020336.

[7] H. V. Koay, J. H. Chuah, C.-O. Chow, Y.-L. Chang, and K. K. Yong, “YOLO-RTUAV: Towards Real-Time Vehicle Detection through Aerial Images with Low-Cost Edge Devices,” Remote Sens (Basel), vol. 13, no. 21, p. 4196, Oct. 2021, doi: 10.3390/rs13214196.

[8] W. Liu et al., “SSD: Single Shot MultiBox Detector,” 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

[9] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Apr. 2018.

[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” Apr. 2020.

[11] F. Tanner et al., “Overhead imagery research data set &#x2014; an annotated data library &#x00026; tools to aid in the development of computer vision algorithms,” in 2009 IEEE Applied Imagery Pattern Recognition Workshop (AIPR 2009), Oct. 2009, pp. 1–8. doi: 10.1109/AIPR.2009.5466304.

[12] A. Kembhavi, D. Harwood, and L. S. Davis, “Vehicle Detection Using Partial Least Squares,” IEEE Trans Pattern Anal Mach Intell, vol. 33, no. 6, pp. 1250–1265, Jun. 2011, doi: 10.1109/TPAMI.2010.182.

[13] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, “Drone-Based Object Counting by Spatially Regularized Regional Proposal Network,” in 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 4165–4173. doi: 10.1109/ICCV.2017.446.

[14] T. N. Mundhenk, G. Konjevod, W. A. Sakla, and K. Boakye, “A Large Contextual Dataset for Classification, Detection and Counting of Cars with Deep Learning,” 2016, pp. 785–800. doi: 10.1007/978-3-319-46487-9_48.

[15] S. Razakarivony and F. Jurie, “Vehicle detection in aerial imagery : A small target detection benchmark,” J Vis Commun Image Represent, vol. 34, pp. 187–203, Jan. 2016, doi: 10.1016/j.jvcir.2015.11.002.

[16] B., N. J. (2022), S. J., et. al. Dwyer, “Roboflow (Version 1.0) [Software].” https://roboflow.com. computer vision., 2022.

[17] S. Rath, “YOLOv8 Ultralytics: State-of-the-Art YOLO Models,” https://learnopencv.com/ultralytics-yolov8/, Jan. 10, 2023.

[18] K. Jiang et al., “An Attention Mechanism-Improved YOLOv7 Object Detection Algorithm for Hemp Duck Count Estimation,” Agriculture, vol. 12, no. 10, p. 1659, Oct. 2022, doi: 10.3390/agriculture12101659.

[19] H. Rezatofighi, N. Tsoi, J. Y. Gwak, A. Sadeghian, I. Reid, and S. Savarese, “Generalized intersection over union: A metric and a loss for bounding box regression,” Apr. 2019, arXiv:1902.09630.