Detection of different types of vehicles from aerial imagery

Jonas Uus
Applied Informatics faculty
Vytautas Magnus University
Kaunas, Lithuania
Email: jonas.uus@bpti.lt

Tomas Krilavičius
Vytautas Magnus University
Kaunas, Lithuania
Baltic Institute of Advanced Technology
Vilnius, Lithuania
Email: tomas.krilavicius@bpti.lt

***Abstract*—Accurate detection of vehicles in large amounts of imagery is one of the harder objects' detection tasks as the image resolution can be as high as 16K or sometimes even higher. Difference in vehicles size and their position (direction, they face) is another challenge to overcome to achieve acceptable detection quality. The vehicles can also be partially obstructed, cut off or it may be hard to differentiate between object colour and its foreground. Small size of vehicles in high resolution images complicates the task of accurate detection even more. CNN is one of the most promising methods for image processing, hence, it was decided to use their implementation in YOLO V3. To deal with big high resolution images method for splitting/recombining images and augmenting them was developed. Proposed approach allowed to achieve 81.72% average precision of vehicles detection. Results show practical applicability of such approach for vehicles detection, yet to reach higher accuracy on tractor, off-road and van categories of the vehicles the count in different vehicle categories needs to be balanced, i.e. more examples of the mentioned vehicles are required.**

I. INTRODUCTION

Vehicles' detection from aerial photography is a very important and quite a difficult task, especially when it is performed in real time or high resolution aerial or satellite images are used for vehicle detection, such as 18000x18000 px. resolution images in COWC [1] dataset. As the drones are used in more and more sectors (according to "cbinsights", currently unmanned aerial vehicle (UAV) could be used in 38 different sectors [2]), for that reason the volume of video and photo material from drones is increasing, the need to create solution for making use of this unprecedented amount of data has become pronounced (at the moment of writing this paper YouTube returns more than 3.3 million results with "aerial footage" query). For human to annotate vehicles from videos or high resolution images it takes a lot of resources. Thus vehicle detection task needs to be automated.

In this paper we investigate applicability of Convolutional Neural Networks. Due to good performance [3], we use YOLO V3 (You only look once) [4] CNN as a tool to apply proposed splitting/merging images method.

Moreover, we split the image into fixed overlapping rectangular frames (*a sliding window method*).

Some results show that Yolo V2 performs quite well with aerial imagery only with applied modifications: "First making the net shallower to increase its output resolution. Second changing the net shape to more closer match the aspect ratio of the data." [5].

In another vehicles detection solution newer YOLO version was used [6]. Images were taken from 3 publicly available datasets: VEDAI, COWC and DOTA. The model had good test results for small objects, rotating objects, as well as compact and dense objects, with 76.7% mAP and 92% recall.

None of these solutions used splitting and remerging technique with images' overlapping. They used already presplitted images.

II. PROBLEM

As computing speed is increasing, and technology is advancing neural networks are being optimised, it had been decided to apply best image augmentation/splitting/remerging methods for vehicle detection. In the application of neural network the following set of problems becomes apparent:

1) Having a variety of different resolution images in dataset (HD, Full HD, 2K ...).
2) Uneven vehicles' sizes in a dataset which are influenced by different ground sample distances (GSD).
3) Uneven vehicles' count in categories by having more cars than other vehicle categories combined.
4) Almost all of the fully connected convolutional neural networks have a fixed-size first layer and all images should be resized to fit the first layer.
5) Vehicles can be partially obstructed (only part of the vehicle could be seen).
6) Hard to differentiate vehicles from foreground (for example, black car parked in a shadow).
7) Vehicles may be facing multiple directions depending on the camera flight direction and its rotation.
8) Available vehicle detection solutions are limited to detecting a small number of features.
9) After re-merging splitted images, the same vehicle may be detected multiple times.

Currently existing vehicles' detection solutions are subject to company trade secrets and companies do not openly discuss technical specifications and application results (for example

web platform Supervisely [7]). That is why it is difficult to adapt or even sometimes impossible to add additional functionality, some solutions are based on older versions of neural networks (for as long as they are functional) and they detect few vehicle categories. For example, one of the vehicle detection solutions [8] detects vehicle features based only on their size (either a small or a large vehicle). Also, currently available solutions which uses CNNs mostly work with fixed size input images or rescale them to fixed size as existing deep convolutional neural networks (CNNs) require a fixed-size (e.g. 224x224) input images [9]. As rescaling images is detrimental for small objects features within images, the images thus are split into smaller pieces, then after the process of individual detection of vehicles in each piece, every image is remerged into full sized image. For example, if a high resolution image such as 4K is rescaled to 608 by 608 pixels, then a rear glass of a car is about 20 by 10 px. after rescaling, and thus the window width becomes about 6 times smaller and its height about 3.5 times smaller and the size of a window decreases to about 6 by 6 px., as a result it becomes harder to differentiate between a van and a car and the probability of misidentification increases. In case of multiple detections in the overlapping image pieces, the NMS (Non-Maximum Suppression) [10] is used to remove duplicate detections as NMS retains only the overlapping bounding box with highest probability (if its area overlaps more than preset value). The herein discussed practice of YOLO application encompasses the attempt to solve all of the above problems.

III. Dataset

MAFAT tournament [11] provided images which were used for training, validation and csv file with boxes and classes, but the csv file was created with classification task in mind and it was not used. The images were adapted for object detection task as the original dataset was initially created for classification task and not every object was annotated, or *false positives* [12] (also called a false detection, vehicle is annotated where there is none) were assigned. Every image was manually annotated and some of them were removed. Those images that were removed were not taken orthogonal to ground, they were taken at an angle. Only images with top-down view were kept. For image augmentation horizontal and vertical flipping and rotation at $45°$ intervals was used.

Following is the count of dataset images:

1) 1712 images were chosen as training images, about 80% of original training dataset images.
2) After splitting training images into 500x500 pixel pieces, images count rose to 9141.
3) 1986 images were chosen for validation, about 78% of original validation dataset images.
4) 12 227 vehicles were annotated manually by me in the training dataset, Fig. 1.
5) 10 914 vehicles were annotated manually by me in the validation dataset, Fig. 2.

The number of vehicles used in the training images is presented in Fig. 1 and the validation datasets are presented in Fig. 2.

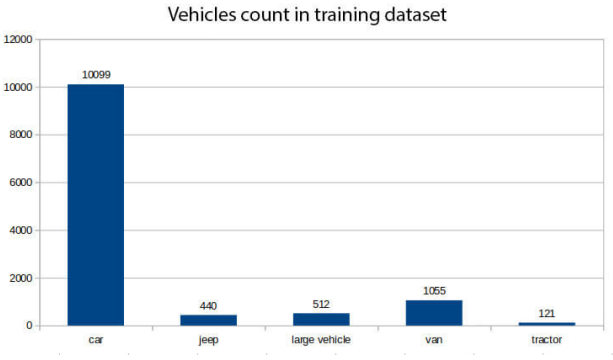


Fig. 1: Vehicles count in training dataset

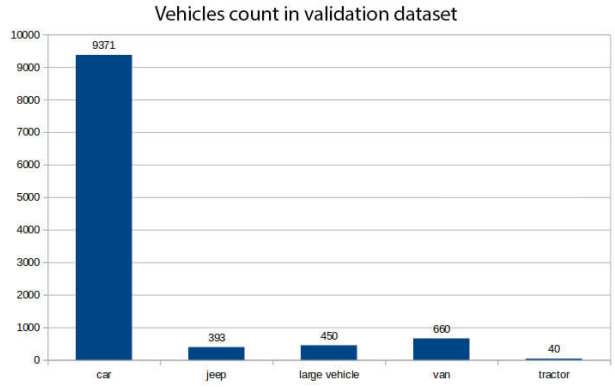


Fig. 2: Vehicles count in validation dataset

The characteristics of dataset images:

1) Images were taken from a variety of locations, some were taken in cities, others in rural areas.
2) Images were taken at a different time of a day.
3) Vehicles were lit from different sides.
4) The resolution of images were different from 900x600 px. to 4010x3668px.
5) Some parts of images were darkened out (for example one half of image was made completely black, while another half of image has picture).
6) GSD (Ground sample distance) of images varied between 5 and 15 cm.
7) Objects in images might have been obstructed by trees or cut off, only part of vehicle might have been seen (for example, a car parked in a garage, a car near the edge of the image).

Couple of images examples taken from dataset Fig 3.

See variation in image resolutions in table I.

The categories of vehicles that were being detected:

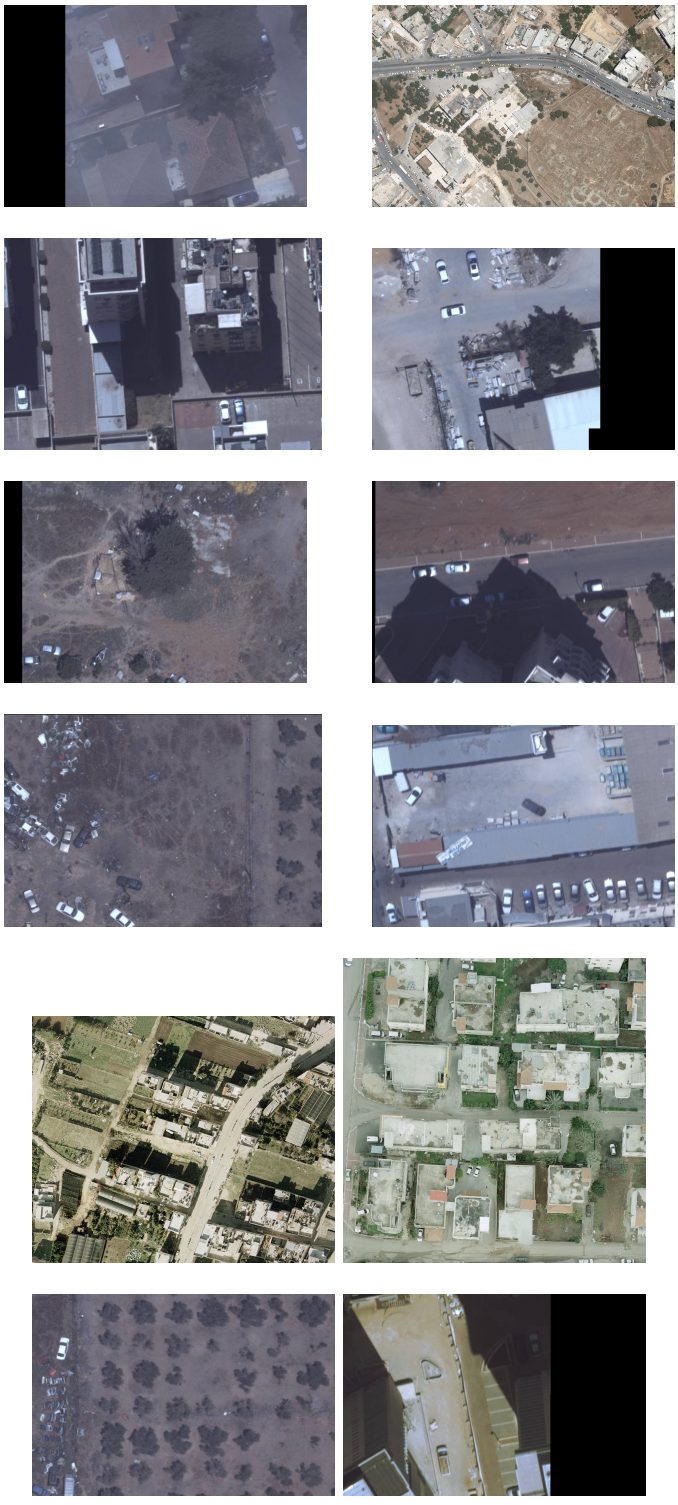


Fig. 3: Examples of images in dataset

TABLE I: Distribution of images with different resolution in dataset

Image resolution (px)	Validation dataset	In Training dataset
900 x 600	1975	1592
1057 x 800	2	3
1332 x 1283	1	0
2026 x 1649	6	37
4010 x 2668	2	40

1) Car,
2) Off-road vehicle,
3) Large Vehicle,
4) Van,
5) Tractor.

The above dataset was considered sufficient for the evaluation of developed method.

IV. Proposed solution

The objective was to develop a method for identification of diverse vehicles.

Image resolution and sizes

The use of CNNs is complicated due to the dataset having a variety of different resolution images (HD, Full HD, 2K ...) and uneven vehicles' sizes in a dataset, see Sect. III. The different sizes in the images are influenced by different ground sample distances (GSD) [13]. As almost all of the convolutional neural networks have a fixed-size first layer [9], all images are resized to that layer size, so if an image resolution is as high as 16K and it is being resized to, for example, 608x608 px. all of the small vehicle features will disappear from the subsequent image. For this reason we propose to split the image into fixed overlapping rectangular frames (*a sliding window method*). This produces double detection problem as vehicles may be detected on both images. To remove duplicates, NMS (Non-Maximum Suppression) is used [10]. If two or more bounding boxes overlap with same vehicle category, then the box with highest detection probability is kept, while the others are removed. Amount of overlapping is determined by finding largest possible vehicle size in the dataset. This ensures that if the vehicle was cut off on one of the images, it would be fully visible in another image.

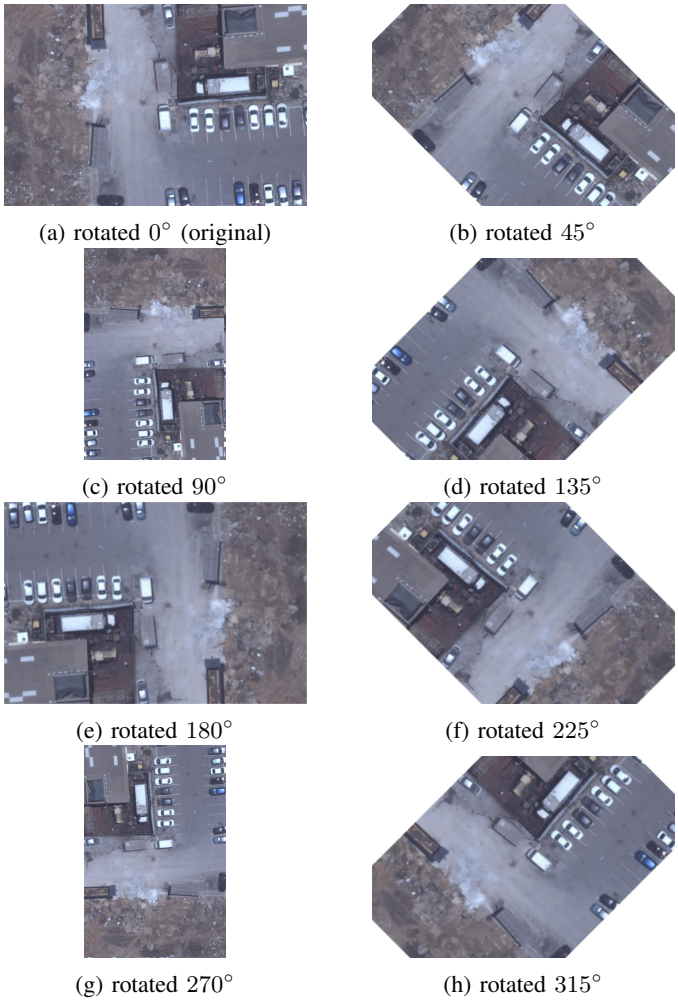
Image obstruction

One more problem with vehicle detection in images is that the vehicles can be partially obstructed (only part of the vehicle could be seen) for example when car are half parked in garage, or when car are parked alongside tree ant tree branches obstruct car features, or when car is on the edge of image.

Orientation

As vehicles orientation in images are not constant they may be facing multiple directions depending on the camera flight direction and its rotation. To solve different vehicles

orientation problem, the images are augmented with random rotation at $45°$ intervals Fig. 4.



(a) rotated $0°$ (original) (b) rotated $45°$ (c) rotated $90°$ (d) rotated $135°$ (e) rotated $180°$ (f) rotated $225°$ (g) rotated $270°$ (h) rotated $315°$

Fig. 4: Images augmentation by rotating

Image augmentation

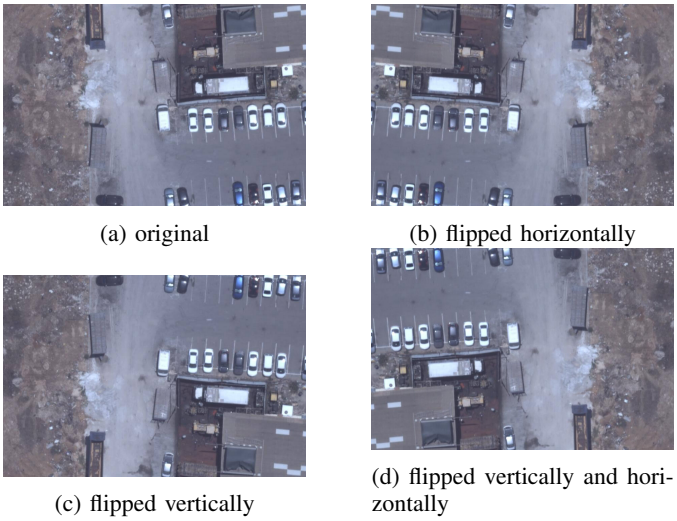
To increase images' count, images were augmented by rotating them at 45 degrees intervals. Additionally, dataset images were augmented by flipping them vertically, horizontally and by flipping both horizontally and vertically Fig. 5.

V. Experiments

A. Tools

For experiments, convolutional neural network YOLO V3 was used on Darknet framework. YOLO V3 architecture is presented in Fig. 6.

On original YOLO repository the problem was that while training, detection loss climbed to infinity, when any single parameter was changed, thus another forked repository [15] from github was used instead, as it does not have the same issue. For YOLO V3 to work with splitting/ merging workflow, original source code was modified. To know when the training had to be terminated, an average loss value was observed. It



(a) original (b) flipped horizontally (c) flipped vertically (d) flipped vertically and horizontally

Fig. 5: Images augmentation by flipping

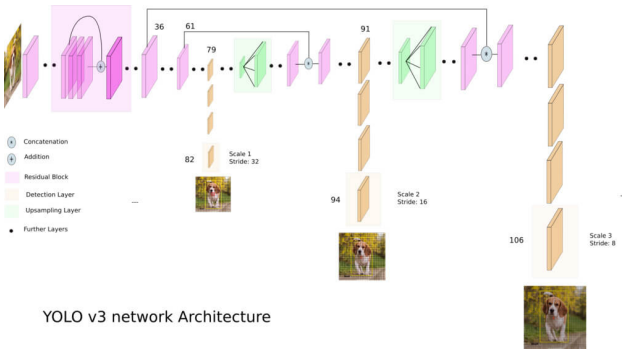


Fig. 6: YOLO V3 architecture [14]

was observed that if any bigger change was to be carried out on neural network, such as adding new object category, the neural network should be trained from previous weights in which neural network had been more generic at detections. Training after changing parameters from scratch would be even better, but that would take longer. It was observed that YOLO detects new class better when previous best weights are not used.

Also, it is hard to differentiate an off-road from a car when looking from above, as the body shape of an off-road may differ only slightly (for example, be wider), thus off-road was annotated as a car. Jeep category is hard too, as the only difference between a car and a jeep is that a jeep has a rear spare tire attached or it has a truck bed (like a pickup).

B. Dataset

Vehicle categories like cars, jeeps, large vehicles, vans and tractors need to be detected from the aerial photographs and their position needs to be marked by drawing bounding box around each the object. At first, cars' class had been divided into hatchbacks and sedans, but during manual objects' annotation it was observed, that if a car is half obstructed and only its front part can be seen, it is impossible to tell whether it is a sedan or a hatchback as the only differentiating factor

is the size of rear glass and only the trunk/ boot can be seen. For this reason, sedans and hatchbacks were merged into one vehicle class.

As the dataset contained mostly cars, YOLO learned that if unsure, it should ascribe an object to a car category, that way it could reach better mAP result in a long run than guessing rarer classes. This non-homogeneous dataset problem shows up, if dataset has different number of vehicles for given class in dataset. This non-homogeneous dataset problem could be solved by adding images in which rarer classes' vehicles are shown or by augmenting a larger number of rearer class images than images with other vehicles.

Cross-validation statistical method was used during YOLO training, the dataset was divided into images for training and validation. The neural network can not see any of the validation images during training, it can only see them when its performance is validated. This method is used to prevent overfitting. The following modifications were performed for the purposes for training and validation images in a dataset:

1) Images' slicing/ overlapping parameter values modification.
2) Fixing wrongly annotated vehicle data and their bounding boxes' locations in the datasets.
3) Changing vehicles' count of classes by adding, merging existing, then reannotating dataset.
4) Choosing images from dataset for training/ validation.
5) Experimenting with images' manipulations (vertical/ horizontal flipping, image rotation), this drastically improved dataset size. These manipulations were manually coded as YOLO, unlike Tensorflow, does not have these image manipuliations integrated.

The following modifications which were done on YOLO:

1) Change of YOLO layer resolution (mostly first layer, as all images are resized to the same resolution as the first layer size).
2) Experiments with different YOLO configurations and different layers' count.
3) Change of network parameters (such as anchors, recalculating certain layer size after vehicles' classes modifications, learning speed).
4) Adding a module to darknet for easier work with split images and for external communication with other programs.

C. Experiments results

To evaluate performance PASCAL VOC evaluation metrics were used and the results were compared using AP (average precision) [16]. This metric uses Jaccard index [17] for calculating IOU (intersection over union) to compare between ground truth and detection boxes.

After training the YOLO V3 neural network it managed to detect cars with 78.69% average precision (AP) Fig. 7, large vehicles with 44.85% average precision (AP) Fig. 8. Other vehicle categories such as jeeps, vans and tractors were detected but they were wrongly categorised. That was the reason vehicle category detection average precision was very low. To solve this problem, the dataset needs to have more unified count of vehicles in every category.

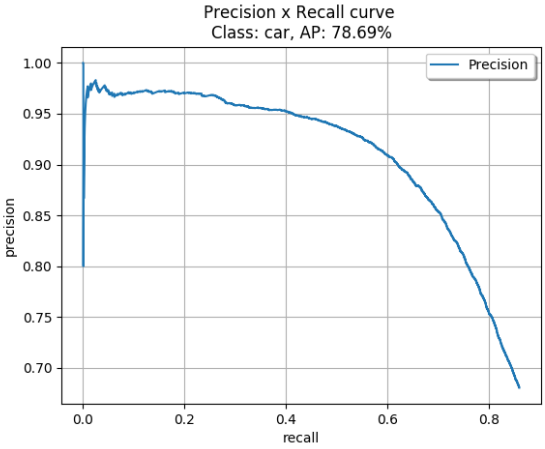


Fig. 7: Precision and recal curve for cars category

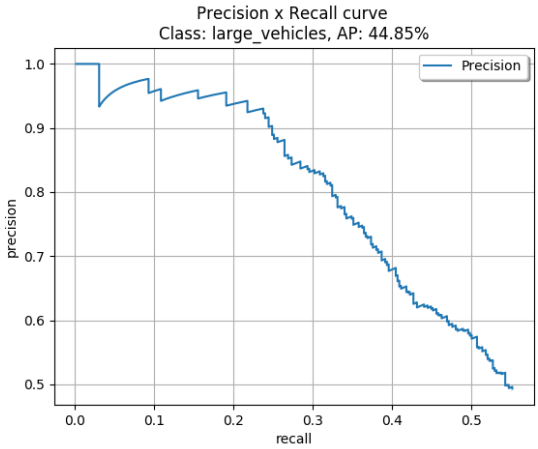


Fig. 8: Precision and recall curve for large vehicle category

The above figures show how precision and recall are corelated, for example, if we choose precision at 95%, then 45% of cars were detected in validation images at that level of precision. *F-Score* [18] at this precision level is equal to 0.61, if recall increases to 80% then the precision drops to 75%. *F-Score* at 75% is equal to 0.77. When all categories were merged into one and then results were validated again, average precision increased to 81.72% Fig. 9. This indicates that in order detection precision is increased, YOLO V3 needs to classify categories more accurately.

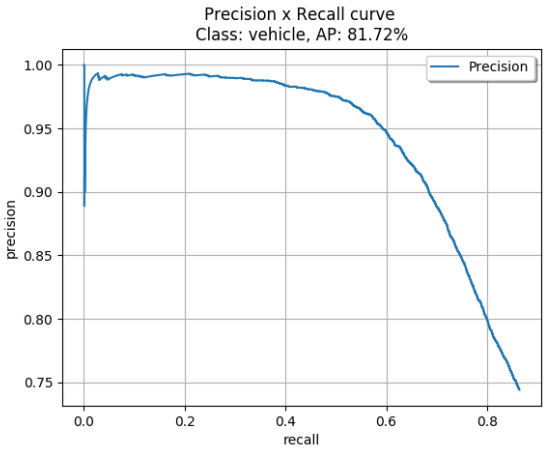


Fig. 9: Precision and recal graph when all vehicles are merged to one category

VI. Conclusions

This application could be used for statistics (to count how many vehicles are there in a given image), vehicles tracking, prediction of further vehicle movement direction and real-time vehicle detection from real time video feed. A vehicles' detection application was created so as users could easily configure it and make vehicles' detection task easier. The user only needs to input images and a couple of parameters to execute vehicles' detection with CNN.

Results:

1) Dataset was prepared for vehicles detection task by manually annotating all of the vehicles in dataset images.
2) Images' were augmented to increase dataset size.
3) Method for combining splitting and joining images and using convulutional neural network for vehicles detection was proposed.
4) Proposed method performance was tested by using YOLO V3 CNN

Conclusions:

1) When YOLO V3 is used together with proposed method is capable of detecting cars with 79% accuracy and large vehicles with 45% accuracy.
2) When proposed method is used, YOLO V3 CNN still has difficulty detecting characteristics of other vehicles, such as off-road, tractors and vans which makes the final detection result lower.
3) Proposed method helps to avoid losing vehicles and their features that would otherwise be lost by resizing high resolution images.
4) The dataset used for training and validation should have more unified count of vehicles categories (more photos with tractors, large vehicles and jeeps should be added to the dataset).

For future work R-CNN and SSD networks will be trained on Tensorflow framework as those networks are also widely used CNN's for object detection tasks and they will be tested using same proposed method. Also, as currently used images' dataset is relatively small, it needs to be increased from freely available datasets and photos taken from drones. As the dataset should have more unified count of vehicles categories, more photos with tractors, large vehicles and jeeps should be added to the dataset.

References

[1] Wesam A. Sakla Kofi Boakye T. Nathan Mundhenk, Goran Konjevod. A large contextual dataset for classification, detection and counting of cars with deep learning. *arXiv:1609.04453*, 2016.
[2] CBINSIGHTS. 38 ways drones will impact society: From fighting war to forecasting weather, uavs change everything. Accessed: 2019.02.22.
[3] Joseph Redmon and Ali Farhadi. Yolo: Real-time object detection. Accessed: 2019.02.22.
[4] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.
[5] Jennifer Carlet and Bernard Abayowa. Fast vehicle detection in aerial imagery. *CoRR*, abs/1709.08666, 2017.
[6] J. Lu, C. Ma, L. Li, X. Xing, Y. Zhang, Z. Wang, and J. Xu. A vehicle detection method for aerial image based on yolo. *Journal of Computer and Communications*, pages 98–107, 2018.
[7] Supervise. The leading platform for entire computer vision lifecycle. Accessed: 2019.02.22.
[8] Alexey. Object detection on satellite images. Accessed: 2019.02.22.
[9] Shaoqing Ren Jian Sun Kaiming He, Xiangyu Zhang. Spatial pyramid pooling in deep convolutional networks for visual recognition. *arXiv:1406.4729v1*, 2014.
[10] Adrian Rosebrock. Non-maximum suppression for object detection in python. Accessed: 2019.02.22.
[11] yuvalsh. Mafat challenge - fine-grained classification of objects from aerial imagery. Accessed: 2019.02.22.
[12] Google. Classification: True vs. false and positive vs. negative. Accessed: 2019.02.22.
[13] Wikipedia contributors. Ground sample distance. Accessed: 2019.02.22.
[14] Ayoosh Kathuria. What's new in yolo v3? Accessed: 2019.02.22.
[15] Alexey. Yolo-v3 and yolo-v2 for windows and linux. Accessed: 2019.02.22.
[16] Jonathan Hui. map (mean average precision) for object detection. Accessed: 2019.02.22.
[17] Wikipedia. Jaccard index. Accessed: 2019.02.22.
[18] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation. volume Vol. 4304, pages 1015–1021, 01 1970.