Baltic J. Modern Computing, Vol. 8 (2020), No. 2, 347-358
https://doi.org/10.22364/bjmc.2020.8.2.10

Vehicle Detection Using Different Deep Learning Algorithms from Image Sequence

Sumeyye CEPNI, Muhammed Enes ATIK, Zaide DURAN

Istanbul Technical University, Faculty of Civil Engineering, Department of Geomatics, Istanbul, Turkey

{cepnis,atikm,duranza}@itu.edu.tr

ORCID 0000-0002-4874-8400, ORCID 0000-0003-2273-7751, ORCID 0000-0002-1608-0119

Abstract. Image processing has become a very popular topic in recent years with images obtained by photogrammetry, remote sensing and computer vision. Deep learning practices are progressing rapidly with this innovation. Object detection, one of the new subjects of deep learning, is applied to high resolution aerial or remote sensing images to ex.tract information from these images. Traditional convolutional neural network (CNN) methods perform estimates in two stages but remain slow in terms of speed performance. You Only Look Once (YOLO) method that is used for real-time object detection, is quickly performed using a single convolutional neural network. In this study, YOLO-v3, YOLO-v3-spp and YOLO-v3-tiny models were applied in the Google Colab environment using python programming language. The comparison of YOLO models trained on COCO data was performed on the video obtained separately from a UAV and the terrestrial camera to identify the vehicles. As a result of the study, the highest results were obtained in Yolov3-spp method with average IoU 84,88% and precision value as 72,02%.

Keywords: Deep Learning, YOLO, Object Detection, Neural Network

1. Introduction

The detection of vehicles ahead and the traffic conditions while driving are important factors for safe driving, accidental cruising and automatic driving and tracking (Jazayeri et al. 2011). Especially real-time perception plays an important role in the development of autonomous vehicles. Therefore, image processing can be used for these purposes. Image processing that has become a popular topic in photogrammetry is used for a variety of applications. However, it is a method used to extract some useful information from the original images obtained. The advancing technology provides many possibilities to develop methods to obtain some useful information from the original images, such as the detection, identification and classification of the objects. In particular, the detection and classification of objects in image content have been one of the main subjects in the literature on computer vision and photogrammetry (Yang et al., 2019).

Convolutional neural networks (CNN) have become a standard tool for many applications in computer vision and machine learning. CNN-based object sensors have made successful and important steps in object detection with the advancement of technology. CNN-based approaches consist of two classes, two-stage and one-stage detection methods (Dai, 2019). Two-stage classes include object detection methods such as regions with convolutional neural networks (R-CNN), Fast R-CNN and Faster R-CNN. In R-CNN, the image is first divided into region proposals and then CNN is applied for each region respectively. The size of the zones is determined and the correct zone is placed in the artificial neural network (Girshick et al., 2014). Although these methods have more accurate than single-stage methods, prediction of the features makes them slow. You Only Look Once (YOLO) is a single-stage object detection method. Additionally, classifying the image by CNN used for object detection, especially for the desired objects, YOLO solves the object perception as a regression problem. It can obtain the position of the object, category and corresponding confidence score, further increase the detection speed and detect the object in the real-time target (Redmon et al., 2016). In the study of Tashiev et al. (2017), real-time classification of intelligent transport systems to provide solutions to traffic systems was performed. Classification is made using the convolutional network YOLO model on the BIT-Vehicle data set. Images are obtained from two different cameras with 1600x1200 size and 1920x1080 size. The model works with 90.35% accuracy. Han P. et al. (2019), the DRoINs network model was created for the analysis of changes occurring at different times over aerial images obtained by Unmanned Aerial Vehicles (UAV). To prevent seasonal errors, the dataset contains 1246 pairs of images. The data set includes 8 categories of cars, boats, motorcycles, aircraft, tractors, trucks and pickups. The experiment was carried out with a computer with 12 Xeon 2.5 GHz CPUs core and 128 GB memory and GeForce GTX 1080 Ti GPU and 44G memory. The model was compared with YOLOS and RPNS models. The network outperformed other models with a 94.9 mAP ratio. Another study (Carlet and Abayowa, 2017) innovations were made to improve the performance of the YOLOv2 model designed for real-time object detection. The YOLOv2 model, which was developed for better detection of vehicles from aerial images, was compared in the data sets of VEDAI, DLR3k, NeoVision2-Helicopter, AFVID 1, AFVID 2, AFVID3, AFVID4 and AF Building Camera. The proposed improved YOLOv2 model yielded successful results.

The scope of this study is to compare the methods of object detection methods, which are added every day, by using the aerial and terrestrial videos to find the appropriate methods according to the data group. In addition, the deficiencies and advantages of the methods are revealed and the results that will form the basis for new models are obtained. This study aims to compare object detection methods on aerial and terrestrial videos to find suitable methods according to the data group. In addition to the local video, the videos obtained from unmanned aerial vehicle (UAV) were also used in the study. Vehicle detection from UAV images has some difficulties due to the extremely high resolution of the images. These problems were also examined in the study. Especially, the performance of methods has been compared to find the car on video frames. Thus, precision and average Intersection over Union (IoU) of the car was investigated. In accordance with real-time applications, the methods were performed over real data. The results represented as tables. As a result of the study, inferences about the use of vehicle detection with deep learning in real time applications have been provided.

2. CNN Structure for Object Detection

CNNs can be defined as systems that implement the learning ability which is the basic function of the human brain for computers. It was originally inspired by the visual cortex in biology. The visual cortex consists of small cells that are sensitive to specific areas of the visual field (Oztemel, 2012). Inspired by the discovery that neurons systematically create a visual perception through a horizontal architecture, the foundations of CNNs have been formed. Convolutional neural networks consist of a plurality of convolution layers, non-linear, pooling and fully connected layers.

2.1. Convolutional Layer

Convolutional layer is the first layer of neural networks and it is composed of the pixel value sequence of the input image. The input image has size of m x m x r. r is the number of bands (r = 3 for RGB image), m is the height and width of the image. The first layer consists of k filters that have n x n x q dimensions. When finding the filter value array, select the q value that is the same as the variable n and, r which is smaller than the size of the image. The filters generate k interconnected activation (feature) maps (Ozkan and Ulker, 2017). The property map consists of regions where the individual properties of each filter are extracted. Depending on the intensity of these properties, it is determined which regions are more important. On the input image, a smaller-sized filter matrix is passed through the image (3x3 filter is applied to the 5x5 input image). At each step, the filter coefficients are multiplied by the values in the corresponding color channel and their totals are calculated. The activation map is created by collecting all three channels.

2.2. Rectified Linear Unit

The rectified linear unit (ReLu) follows the convolution layer and has the activation function. The objective in this layer is to apply the linear structure obtained from the previous layer to a non-linear structure. This layer increases the speed of learning in the network.

2.3. Pooling Layer

Pooling layer usually follows the ReLu layer. Its main purpose is to reduce the input size for the next layer (Oztemel, 2012). As with the convolution layer, certain filters are defined in the pooling layer, and by applying these filters in a specific order on the image, the pixel values can be calculated in two different ways; average pooling and maximum pooling. For maximum pooling, the highest values of the pixels in the image are selected. In the average pool, the average values of the pixels in the image are taken.

2.4. Fully Connected Layer

In the CNN structure, ReLu and the pooling layer are followed by a fully connected layer (FC). It connects all neurons from the convolution and pooling layer with different combinations to produce better properties (Redmon et al., 2016).

2.5. DropOut Layer

The DropOut layer improves the performance quality of the network by preventing the memorization of the neural network.

3. Methodology

3.1. Data Used

In the application, video obtained from UAV (van Es, 2017) with 1280x720 resolution and video obtained with terrestrial resolution of 1080x1920 were used. Both of the videos have a frame rate of 24 frames per second and the length of the videos is about a minute. The video obtained from the UAV was produced obliquely instead of nadir. Because the YOLO algorithm does not give results in the images obtained from the nadir due to the training set. The videos were used directly without any pre-processing steps.

3.2. COCO Dataset

YOLO model uses COCO dataset. COCO is a data set consisting of 80 separate classes such as cars, dogs, planes, bicycles, bags, and phones (Figure 1.) (Radovic et al., 2017). The data set contains 80,000 training images and 40,000 verification images.

Fig. 1. Sample images of car class from COCO dataset (Lin et al., 2014)

3.3. YOLO Object Detection

YOLO that is an open-source object detection and classification algorithm based on the CNN network. Conventional CNN networks generate regional predictions to suggest bounding boxes. This is followed by the step of grading, correcting, and removing duplicates of the bounding boxes. It re-scores all bounding boxes based on the objects found. Finally, the region with the highest score on the image is considered as detected (Figure 2.). In YOLO, it can predict which objects are present in an image and their positions at first glance. It performs object detection by spatially separated bounding boxes and predicting the entire image with a single neural network. The main advantage of this approach is that the whole image is evaluated by a single neural network and object detection according to the predicted regions (Radovic et al., 2017).

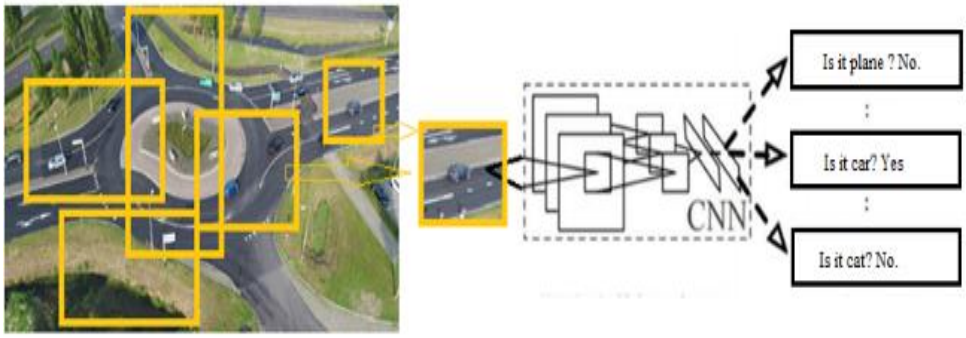

Fig. 2. YOLO workflow diagram.

The structure of YOLO algorithm consists of conventional neural networks (Figure 3.). YOLO starts detecting the object by dividing the input image into SxS gray. In general operation, only one object is estimated with each grid cell. Bounding boxes are created for each grid cell and assign trust points to these boxes. Each boundary box contains 5 variables: x, y, w, h and C. For the bounding box width, w is calculated, and h for the height. X and y represent the center of the box. For each grid, there are 80 conditional class probabilities (C), with the probability that the detected object belongs to a particular class (Redmon and Farhadi, 2017). When constructing a trust score, it measures both the classification and the trust of the object in its location. The trust score is zero when the grid does not contain any objects. The final output of the YOLO estimate is the tensor S x S x (B x 5 + C). The main concept of YOLO (7,7,30) is to create a single CNN network to estimate its tensor. It uses those with a confidence score higher than the 0.25 threshold when estimating with bounding boxes (Radovic et al., 2017).

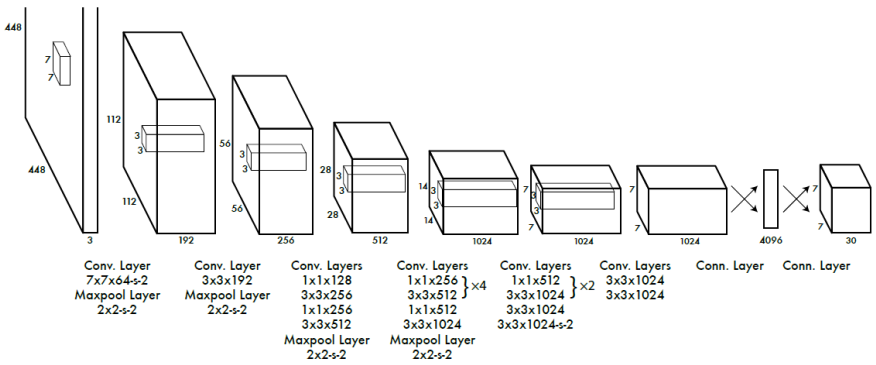

Fig. 3. The architecture of YOLO (Redmon et al., 2016)

3.4. YOLO Mathematical Model

3.4.1. Lost Function. YOLO makes its prediction with bounding boxes. To calculate the true positive (TP), it must find the object. As shown in Eq. (1), YOLO uses the total squared error on the real plot to estimate the losses that occur when finding the real object (Redmon et al., 2016). The loss function consists of loss of classification (classError), loss of localization (coordError) and loss of position confidence (iouError).

$$\text{loss} = \sum_{i=0}^{s^2} \text{classError} + \text{coordError} + \text{iouError} \tag{1}$$

where s is number of grids, classError is classification error, coordError is localization error, iouError is confidence error.

3.4.2. Classification Error. The classError is the classification error value (Eq. (2)). The classification function results in the correctness of the estimated object.

$$\sum_{i=0}^{s^2} 1_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{2}$$

where 1_{ij}^{obj}=1 if an object appears, otherwise 0, $\hat{p}_i(c)$ - conditional class probability for class c.

3.4.3. Localization Error. With the loss of localization, the accuracy of the estimated bounding box is calculated. The equation is measured by errors in the position and dimensions of the bounding box as shown in Eq. (3) (Redmon et al., 2016).

$$\lambda_{coord}\sum_{i=0}^{s^2}\sum_{j=0}^{B}1_{ij}^{obj}\left[(x_{i-\hat{x}i})^2+(y_{i-\hat{y}_i})^2\right] + \lambda_{coord}\sum_{i=0}^{s^2}\sum_{j=0}^{B}1_{ij}^{obj}\left[\left(\sqrt{w_i}-\sqrt{\hat{w}_i}\right)^2+\left(\sqrt{h_i}-\sqrt{\hat{h}_i}\right)^2\right] \quad (3)$$

where B is boundary box, x and y are offsets to the corresponding cell, w is width of image, h is height of images, λ_{coord} increase the weight for the loss in the box coordinates, 1_{ij}^{obj}=1 if the j_{th} boundary box in cell *i* is responsible for detecting the object, otherwise 0.

3.4.4. Confidence Error. Eq. (4) which is loss of confidence is calculated separately depending on the presence or absence of the bounding box and the object. If an object is detected in the boundary box;

$$\sum_{i=0}^{s^2}\sum_{j=0}^{B}1_{ij}^{obj}(C_i-\hat{C}_i)^2 \quad (4')$$

where $\hat{C}_i$ is confidence score.

If an object is not detected in the box, the confidence error is;

$$\lambda_{noobj}\sum_{i=0}^{s^2}\sum_{j=0}^{B}1_{ij}^{noobj}(C_i-\hat{C}_i)^2 \quad (4'')$$

where $1_{ij}^{noobj}$is the supplement of 1_{ij}^{obj}, $\lambda_{noobj}$is weights to prevent class unbalance because of most boxes do not contain objects.

In the application, the video obtained from the UAV with the video obtained from the UAV with sky tools traffic monitoring DJI INSPIRE 2 + POWERLINE Cam X5S 15mm was used distance of 50 m from ground. The other video that was obtained as terrestrial with a resolution of 1080x1920 was used. Terrestrial video was obtained manually by 1080p with 12 MP camera. The algorithms were run on a virtual computer that has 12 GB of RAM and an NVIDIA Tesla K80 GPU graphics card.

3.5. Evaluation Metrics

In the study, average IoU (Eq. (5)) and precision (Eq. (6)) values were calculated for comparison. In order to calculate these, True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) values should be determined.

TP: True Positive refers to plane points which are included inside the detected model.

TN: True Negative refers to non-plane points which are outside the detected model.
FP: False Positive refers to plane points which are not included inside the detected model.
FN: False Negative refers to non-plane points which are included inside the detected model (Alpaydin, 2010).

$$Precision = \frac{TP}{TP+FP} \tag{5}$$

$$IoU = \frac{TP}{TP+FP+FN} \tag{6}$$

4. Results and Discussion

The study was carried out on the Google Colab cloud service. The algorithm, taken from the COCO dataset of weights that are trained on 80,000 training images, was run using the open source artificial neural network called as Darknet library. Only the vehicles are focused as objects to be determined.

As a result of the study, vehicles from video images were detected by YOLOv3, YOLOv3-spp and YOLOv3-tiny. The object type and IoU value for each detected vehicle are displayed by the algorithm on the bounding box. For classification assessment of the algorithm, it is necessary to compare the model with the actual results. In this study, from the simple and fast methods working in easy and homogeneous backgrounds, the deep learning methods used against the complex and difficult problems were evaluated. In order for the specified object to be considered correct, the IoU value must be 0.5 and above, similar to previous studies (Redmon et al., 2017). Even if the object is correctly named in aerial video, since the IoU value is less than 0.5, this is not considered an accurate object. YOLOv3-spp model was the most successful method with 84.88% on the ground accuracy of the methods on the COCO dataset (Table 1). Average IoU was reached in the terrestrial video (Figure 5.) with 88.56% and UAV videos (Figure 4.) with 81.21%. The YOLOv3-tiny method fails to detect small objects (Yi et al., 2019). As the UAV video videos are obtained from a distance, the object cannot be detected with YOLOv3-tiny. Therefore, the YOLOv3-tiny method did not yield any results in the video obtained by UAV in the real ground value and accuracy comparisons. YOLOv3-spp model is more successful with a precision of 72.02% in accuracy results; 63.53% precision was achieved in UAV and 80.49% in the terrestrial video (Table 2). In both model comparisons, both models yielded results in terms of model estimation and accuracy factor, and it was found to be more suitable with high performance on terrestrial videos. With the data set suitable for the targeted study, successful determinations can be made on the vehicles in UAV images. To increase the accuracy factor, training and verification procedures are important in the data set suitable for the data used as input. With the data set to be used, more successful detections can be realized in parallel with the targeted study purpose. The most important reason for the low accuracy in aerial video is that the data set used is more suitable for terrestrial images. However, with stronger training, success rates will increase confidence in deep learning methods.

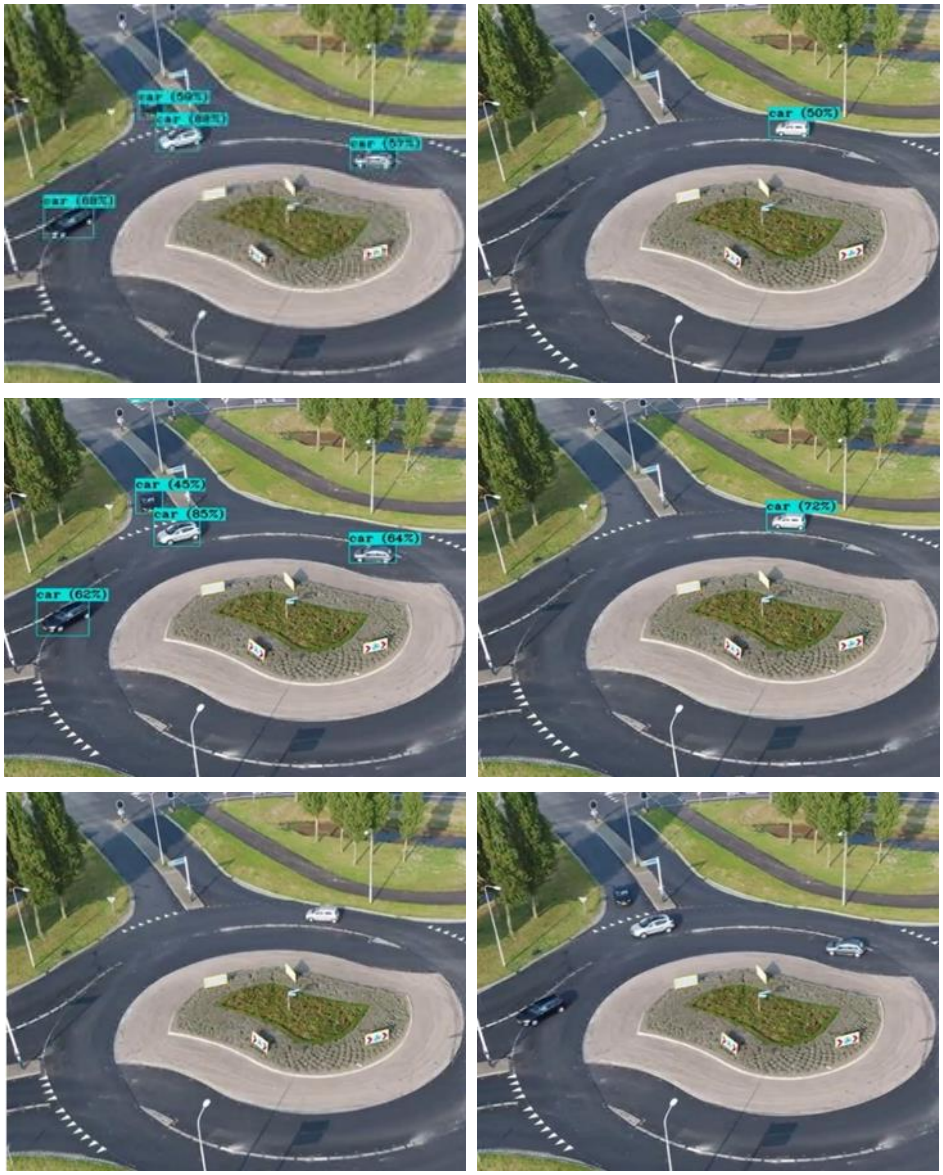

Fig. 4. Detection of vehicles by YOLOv3, YOLOv3-spp and YOLOv3-tiny method on UAV image (top to bottom)

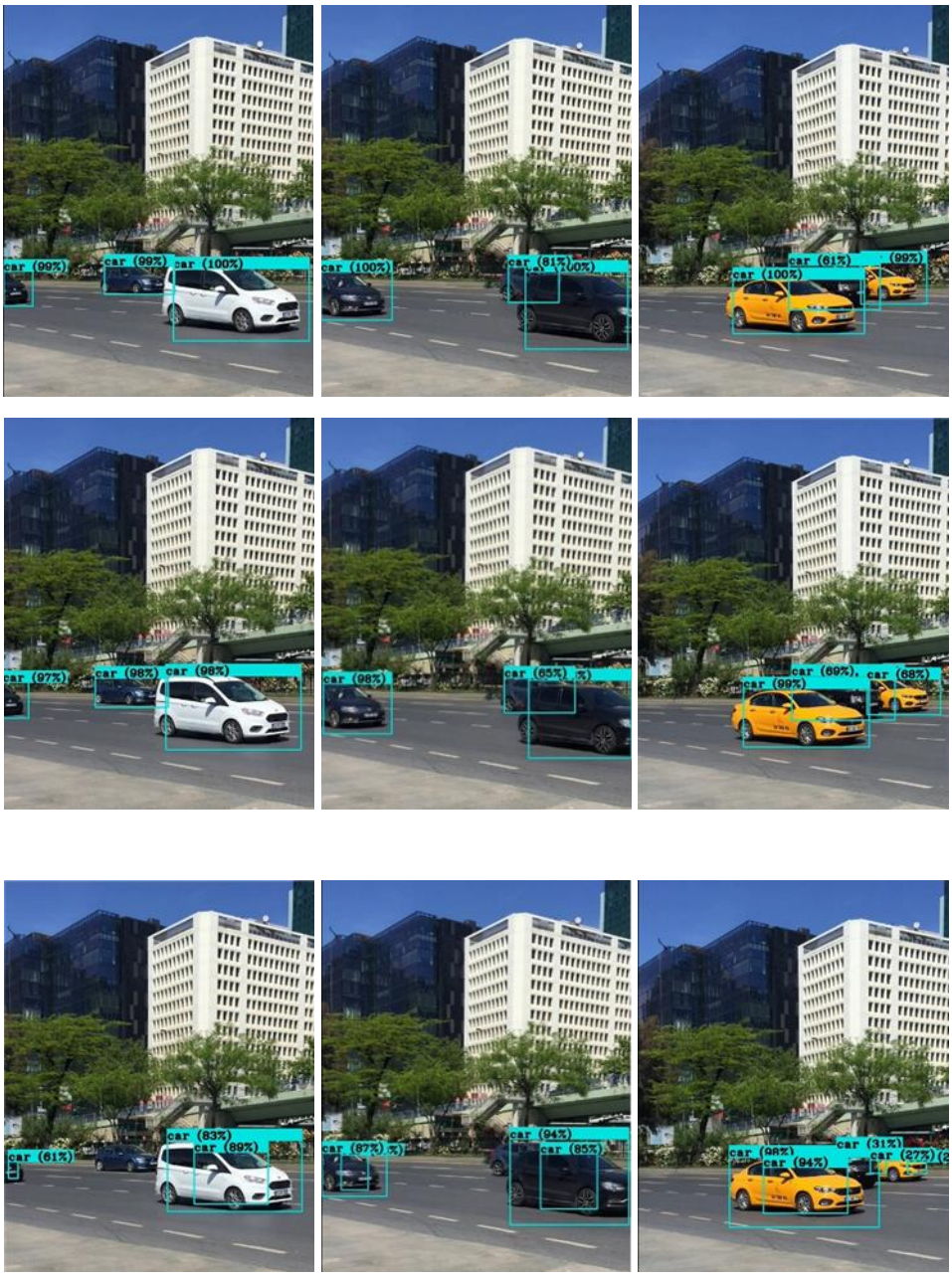

Fig. 5: Detection of vehicles YOLOv3 YOLOv3-spp and YOLOv3-tiny methods on terrestrial videos (top to bottom).

Table 1. Average IoU comparison of videos obtained by UAV and terrestrial methods for IoU ≥ 0.5.

Methods	Average IoU	UAV	Terrestrial
YOLOv3	83,70%	79,52%	87,89%
YOLOv3-spp		**81,21%**	**88,56%**
YOLOv3-tiny	-	-	78,40%

Table 2. Precision comparison of videos obtained by UAV and terrestrial methods.

Methods	Precision	UAV	Terrestrial
YOLOv3	70,30%	60,25%	80,36%
YOLOv3-spp	**72,02%**	**63,53%**	**80,49%**
YOLOv3-tiny	-	-	60,78%

It is seen that our study has high accuracy values compared to other studies with car detection with YOLO algorithms. For the purpose of comparison, studies aiming to detect objects over video or in real time were selected. Our study has produced successful results in vehicle detection from both aerial and terrestrial video images (Table 3). Although lower precision is obtained in aerial video, 80.49% precision value was obtained in terrestrial video.

Table 3: Precision comparison of videos obtained by UAV and terrestrial methods.

Study	Precision	Average IoU
Song et al. (2019)	**86.46%**	71.32%
Putra et al. (2018)	55.3%	-
Corovic et al. (2018)	63.00%	46.60%
Our Study	72,02%	**%84,88**

5. Conclusion

In both model comparisons, both models yielded results in terms of model estimation and accuracy factor, and it was found to be more suitable with high performance on terrestrial videos. Due to its structure, YOLO algorithm is suitable for real-time vehicle detection. With the data set suitable for the targeted study, successful determinations can be made on the vehicles in UAV images. To increase the accuracy factor, training and verification procedures are important in the data set suitable for the data used as input. With the data set to be used, more successful detections can be realized in parallel with the targeted study purpose. However, with stronger training, success rates will increase confidence in deep learning methods.

References

Carlet, J., Abayowa, B. (2017). Fast vehicle detection in aerial imagery. arXiv preprint arXiv:1709.08666.

Ćorović, A., Ilić, V., Durić, S., Marijan, M., Pavković, B. (2018). The real-time detection of traffic participants using yolo algorithm. In 2018 26th Telecommunications Forum (TELFOR) (pp. 1-4). IEEE.

Dai, X. (2019). HybridNet: A fast vehicle detection system for autonomous driving. Signal Processing: Image Communication, 70, 79-88.

Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

Han, P., Ma, C., Li, Q., Leng, P., Bu, S., Li, K. (2019). Aerial image change detection using dual regions of interest networks. Neurocomputing, 349, 190-201.

Jazayeri, A., Cai, H., Zheng, J. Y., Tuceryan, M. (2011). Vehicle detection and tracking in car video based on motion model. IEEE Transactions on Intelligent Transportation Systems, 12(2), 583-595.

Ozkan İ. and Ulker, E. Deep Learning and Deep Learning Models Used in Image Analysis. Gaziosmanpasa Journal of Scientific Research, 6(3)**.**

Oztemel, E. (2012). *Artificial Neural Networks*. 3rd. ed., DaisyScience International Publishing House, Istanbul.

Putra, M. H., Yussof, Z. M., Lim, K. C., Salim, S. I. (2018). Convolutional neural network for person and car detection using yolo framework. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 10(1-7), 67-71.

Radovic, M., Adarkwa, O., Wang, Q. (2017). Object recognition in aerial images using convolutional neural networks. Journal of Imaging, 3(2), 21.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

Redmon, J., Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

Redmon, J., Farhadi, A. (2016) "YOLO9000: Better, Faster, Stronger,":arXiv:1612.08242.

Song, H., Liang, H., Li, H., Dai, Z., Yun, X. (2019). Vision-based vehicle detection and counting system using deep learning in highway scenes. European Transport Research Review, 11(1), 51.

Tashiev, İ., Kul, S., Şentaş, A., Küçükayvaz, F., Eken, S., Sayar, A., Becerikli, Y. (2017). Real-Time Vehicle Type Classification Using Convolutional Neural Network. 1st National Cloud Computing and Big Data Symposium.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

van Es, M. (2017). https://youtu.be/8EjTUYek_Hw

Viola, P., and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. CVPR (1), 1, 511-518.

Yang, M. Y., Liao, W., Li, X., Cao, Y., Rosenhahn, B. (2019). Vehicle Detection in Aerial Images. Photogrammetric engineering and remote sensing: PEandRS, 85(4), 297-304.

Yi, Z., Yongliang, S., Jun, Z. (2019). An improved tiny-yolov3 pedestrian detection algorithm. Optik, 183, 17-23.

Received May 19, 2020, accepted May 29, 2020 as a paper from ISAG2019