





Sea Turtle Detection Using Faster R-CNN for Conservation Purpose

Mohamed Badawy  and Cem Direkoglu ^(✉) 

Center for Sustainability, Department of Electrical and Electronics Engineering,
Middle East Technical University - Northern Cyprus Campus, Kalkanli,
Guzelyurt, North Cyprus, via Mersin 10, Turkey
mu.badawy@gmail.com, cemdir@metu.edu.tr

Abstract. Automatically monitoring sea turtles over extensive coastlines is an important task for environmental research and conservation nowadays. Unfortunately, some of the sea turtle species have become endangered today and this is why there is a need for search-and-rescue. Computer vision algorithms can be used for sea turtle detection and monitoring. Recently, due to the powerful Convolutional Neural Networks (CNNs), computer vision crucial applications came to reality. Although such algorithms are computationally expensive, they have proved promising results where real-time applications can be feasibly implemented given high-capability GPUs. In this paper, we present a system of sea turtles detection using a Faster R-CNN algorithm. This system performs the sea turtles' detection on a cloud (off-board). Our detection algorithm can be performed using a static camera, or a moving camera that is mounted on UAVs for surveillance, search-and-rescue purposes.

Keywords: Computer vision · Object detection · Sea turtles · Faster R-CNN

1 Introduction

Most of the sea turtle species have become endangered. There is a need for automatic coastal detection and monitoring for sea turtles. Recently, UAVs have emerged strongly and have been widely used in a lot of different applications; mainly they are used in object-detection based tasks such as surveillance, search-and-rescue and similar tasks. However, using drones to monitor and detect moving objects is a pretty difficult and tough process. It is really challenging due to the fact that the on-board camera resolution is quite low and thus most of these images are blurry and noisy. Since the target objects are quite small, the detection becomes more difficult to be achieved. The need of real-time detection also adds another challenge to the detection process. In this paper, we propose a computer vision based sea turtle detection system using a static camera, or a camera mounted on a Drone. Detecting sea turtles is never easy as it might sound due to the scarcity of sea turtles' data set as well as their diversity. Taking all these factors in consideration, we built our approach to create this system, incorporating the technical standards and trying to minimize the gap between the application requirements and the technical capabilities.

Sea turtle detection is an object detection problem. Recently, the Convolutional Neural Networks (CNNs) has been employed for object detection. Convolutional neural networks have two main key features: First, these networks take the advantage of local receptive fields, since image data within local spatial regions is likely to be related [1, 2]. Second, these networks are deep that means they comprise many layers to perform better performance in comparison to the traditional artificial neural networks. Thus, they are robust and prove reliability in object detection. In our work, we use the Faster R-CNN algorithm [7] to build a sea turtle detection system on a cloud (off-board). Our detection algorithm can be performed using a static camera, or a moving camera for search-and-rescue purposes. The proposed system is fast and effective.

2 Faster R-CNN

Evaluated on the VOC2007 [9] dataset, the Faster R-CNN algorithm achieves the highest accuracy, denoted as mAP, mean average precision, which is 73.2%. The main motivation beyond using faster R-CNN is its high precision in comparison to other object detection algorithms. Although the main downside of Faster R-CNN is its low FPS rate compared to YOLO [8] and SSD [12], Faster R-CNN is better when it comes to detecting objects of small size. In general, objects captured by the drone's camera have a small size and the YOLO algorithm potentially fails to detect objects of small size. We decided to use the Faster R-CNN because of its high precision which greatly surpasses other algorithms, and ability to detect objects of small size. Our main objective is correctly detecting sea turtles with the highest precision possible. Thus, we compromised the FPS rate as we are satisfied by the Faster R-CNN rate which is 7 FPS [8]. Table 1 also illustrates the performance of object detection algorithms on aerial images collected by a drone where we can see the highest precision is solely achieved by the Faster R-CNN algorithm [4].

Table 1. Object detection results on aerial images collected by a Drone [4].

Method	Car	Cat	Dog	Person	mAP
Yolo	87.2	100.0	81.0	88.7	79.4
SSD 300	100.0	100.0	66.7	92.9	21.6
SSD 500	93.2	100.0	69.6	81.7	82.6
Faster R-CNN	89.7	100.0	77.3	81.7	83.9

Faster R-CNN is mainly comprised of two modules. The First module is the region proposal network (RPN). The second module is the detector that makes use of the proposed regions by the first module. Thus the entire system forms a single, unified network for object detection. Thus the RPN module directs the Fast R-CNN module where to look for the desired objects [6, 7, 10].

3 Implementation

Using Tensorflow [5] and Anaconda python 3.6, we developed the code and started the training. This process includes three important steps. First step is building the code to implement the proposed algorithm. We have used python as our developing environment and we could develop codes for building the neural network taking in consideration the weights function and the loss function as well as the number of layers. Thus, our project can substantially detect see turtles in images or videos as well as in live webcam. Second step is gathering a huge and considerable amount of data and starting the training process. For the training part, we have done a lot of experiments to optimize our training. It is of a great importance to mention that the model should be trained enough, but at the same time it shouldn't be over trained. Finally, third step is testing the module.

3.1 Training the Faster R-CNN for Sea Turtle Detection

The process of collecting datasets was never easy. There is a scarcity of sea turtles' datasets on known datasets websites such as Kaggle and Google Images. However, we could collect datasets of green and loggerhead sea turtles of approximately 1000 images of which we used around 500. It is worthy to mention that some of these images was retrieved from ImageNet [3]. We then labelled these images and annotated them one by one using LabelImg software [11]. Then, we generate the .xml files which contain the dimensions of the label box (Xmin, Xmax, Ymin, Ymax). We used around 20% of our data to test our network and 80% to train the CNN. Our first training wasn't successful since the training period was more than the required period, thus we over trained the CNN. The training reaches a number of epochs when the loss function is ultimately reduced to its minimum value after which the loss function start to rise again which in this case produces potentially wrong training and thus no detection was observed.

The next training of the iteration was done carefully by closely observing the loss function such that the neural network is not over trained. The loss function is expected to have a graph where the loss function decrements to reach levels below 0.1 where we have our CNN training completed and saturated. Thus, we shouldn't exceed the training epochs of this saturation stage. Accordingly, we did the training in a proficient manner where the loss function decrements till it reaches a value below 0.05 and that is when we shall stop the training and proceed to the next step which is testing our model.

4 Testing the Faster R-CNN for Sea Turtle Detection

Since we have built three different programs to work with our detector which are image, video, and webcam test models, we tried three of them to detect the sea turtles which worked perfectly fine. We tested our module also on different images and challenging ones and it showed promising results. Although there were some inaccuracies in detecting some of the sea turtles provided in a given photo, the network showed impressive result overall. Below, in Figs. 1 and 2, the network has been tested by giving images, videos and by also using live webcam. Images in Fig. 2 have been captured by Drones.

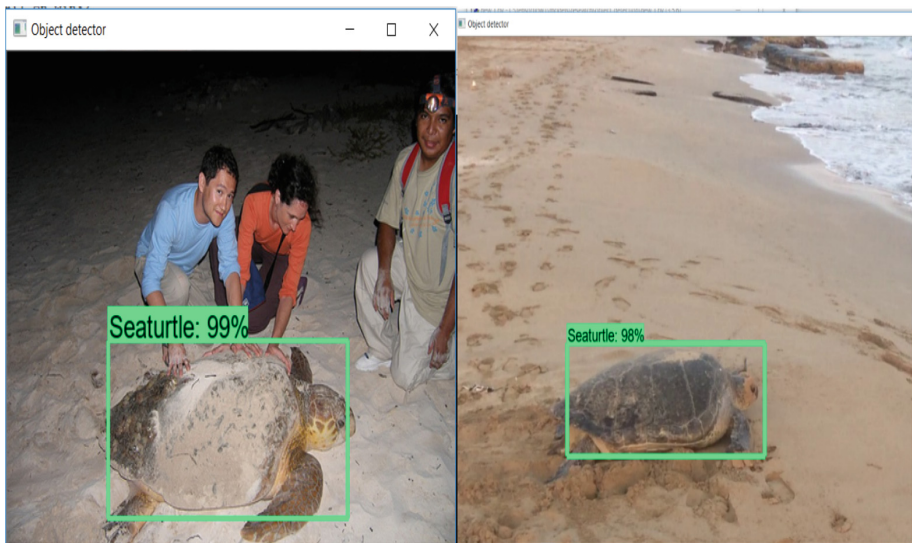


Fig. 1. Sea turtle detected with high confidence.

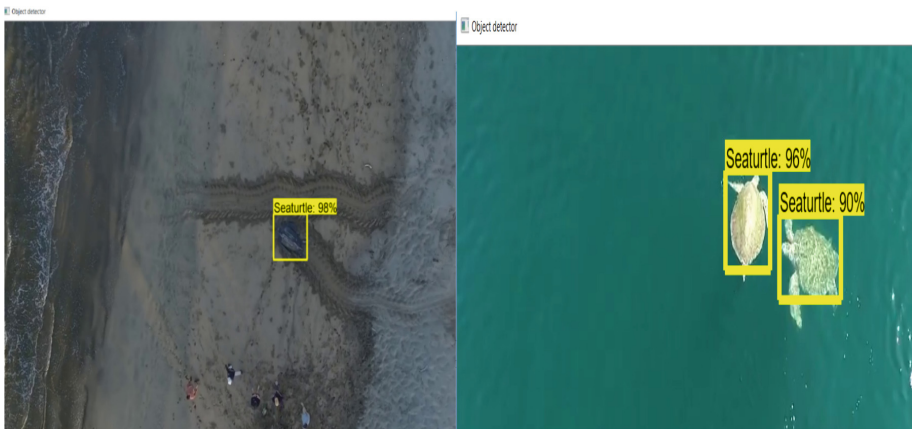


Fig. 2. Sea turtle is detected from drone video.

5 Evaluations

In this section the specifications of our machine, the precision calculations and the timing analysis are explained in detail. So we have implemented our sea turtles training and detection on a Lenovo ideapad 520 laptop with a Nvidia Card Geforce MX150 (4 GB) and intel core i7 8th generation processor. As for the hardware requirements as well as the R-CNN algorithm code can be found in <https://github.com/rbgirshick/py-faster-rcnn>.

5.1 Precision, Recall and F-Score Values

Precision can be defined as how accurate the predications made by the detection algorithm are. Thus, precision is defined by Eq. (1) as the ratio between the true positive results and the actual positive results of the algorithm. On the other side, recall is more meant to express with the total number of misses made by the algorithm, so as shown by Eq. (2), recall is the ratio between the true positive results and the summation of the misses and hits represented by true positive and false negative. Finally, F-score is a measure of accuracy that combines precision and recall results as given in Eq. (3). In Table 2, precision and recall related variables are listed. Table 3 shows the confusion matrix after evaluation, and Table 3 shows the computed Precision, Recall and F-Score values at threshold = 0.8 (confidence factor).

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP}) * 100\% \tag{1}$$

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN}) * 100\% \tag{2}$$

$$\text{F1score} = 2 * \text{Precision} * \text{Recall}/(\text{Precision} + \text{Recall}) \tag{3}$$

Table 2. Precision variables definition.

Variable	Definition	Explanation
TP	True Positive	Equivalent with hit
TN	True Negative	Equivalent with correct rejection
FP	False Positive	Equivalent with false error
FN	False Negative	Equivalent with miss

Table 3. Confusion matrix at threshold = 0.8.

Actual	Predicted		
	Predicted = No	Predicted = Yes	
Actual = No	TN = 270	FP = 4	274
Actual = Yes	FN = 26	TP = 90	116
	296	94	

Precision, Recall and F-Score values at threshold = 0.8:

$$\text{Precision} = 95.7\%$$

$$\text{Recall} = 77.6\%$$

$$\text{F1score} = 85.7\%$$

5.2 Time Performance Analysis

As for timing, we have an overhead of nearly 10 s when all necessary libraries and packages are imported afterwards the detection starts. Apart from the overhead, once detection starts, we are able to process and record the result for more than 3 frames per second. As mentioned in [7], the frame per second rate of Faster R-CNN reaches 5 FPS on K40 GPU using VGG model. It also reaches 17 FPS using ZF model on K40 GPU. Thus it proves it is a reliable model especially if precision is more favored than the rate of detection which is the case for our work.

6 Conclusion

In this work, we have proposed an intelligent system for sea turtles detection where the Faster R-CNN algorithm is employed in an impressive manner, and giving promising results. We have used Tensorflow-GPU deep learning framework to train our Faster-RCNN. Our work is fast and effective and it is a practical solution to object detection for drone based images. This work is really deployable and scalable as it has also been integrated in a complete system of detecting and classifying sea turtles by retrieving the videos of sea turtles from an on-board camera mounted on drone, where the user gets notifications on finding turtles in a friendly user interface. In this project we implemented the detection system on a cloud which receives the videos from the drone and automatically detects the turtles and display them in UI to notify the user about the whereabouts of the turtle by giving the frame number and the bounding box dimensions where the turtle is detected. It effectively contributes to ecosystem solutions and environmental research in general and particularly sea turtle conservation projects as sea turtles are known to be endangered species. The complete project, with all necessary files to run it, is open source and can be accessed via this link <https://github.com/MuBadawy/Tensorflow-based-SeaTurtle-Detection>.

Acknowledgements. This work is sponsored by Cyprus Wildlife Research Institute (CWRI). Our project team would like to acknowledge CWRI for supporting us to buy the hardware components required in the project. We also would like to thank Chantal Kohl and Stefanie Kramer from Humboldt University for supplying us some sea turtle images and videos for experimentation in our project. This project has been implemented on a server which belongs to Hossam Ahmed and Mohamed Badawy, we would like to thank them as well.

References

1. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: CVPR14 Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2155–2162 (2014)
2. Girshick, R.: Fast R-CNN. <https://arxiv.org/abs/1504.08083> (2015)
3. Image-net.org: ImageNet Tree View. <http://image-net.org/synset?wnid=n01664065>. Accessed 11 June 2019

4. Lee, J., Wang, J., Crandall, D.J., Sabanovic, S., Fox, G.C.: Real-time, cloud-based object detection for unmanned aerial vehicles. In: 2017 First IEEE International Conference on Robotic Computing (IRC), pp. 36–43 (2017)
5. Abadi, M., Agarwal, A., et al.: Tensorflow: large-scale machine learning on heterogeneous distributed systems. [arXiv:1603.04467](https://arxiv.org/abs/1603.04467) (2016)
6. Gao, H.: Faster R-CNN explained. <https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8>. Accessed 3 Jan 2019
7. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. <https://arxiv.org/pdf/1506.01497.pdf> (2016)
8. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Redmon_You_Only_Look_CVPR_2016_paper.pdf (2016)
9. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks. <https://arxiv.org/pdf/1312.6229.pdf> (2014)
10. Szegedy, C., Reed, S., Erhan, D., Anguelov, D.: Scalable, high-quality object detection. <https://arxiv.org/abs/1412.1441> (2015)
11. Tzatalin LabelImg: Git code. <https://github.com/tzatalin/labelImg> (2015)
12. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.: SSD single shot multibox detector. <https://arxiv.org/abs/1512.02325> (2016)