NHL STENDEN
computer vision
& data science

Tourism analysis using object detection with deep neural networks

NHL Stenden Centre of Expertise in Computer Vision & Data Science

Alpay Vodenicharov

Supervisors: Ioannis Katramados, Martin Dijkstra

Abstract—Tourism is a crucial sector for every country. It is connected with several other important sectors, such as finance and transportation. Namely in certain countries it could be a key component in financial growth, therefore it is important to be able to measure tourism. Currently, there are no reliable methods to do so. This paper will concern if the measurement of tourism using deep-learning techniques is possible. The first steps of this goal is to detect objects of interests, such as cars and marinas in open space areas using satellite/aerial images. After experimenting with different algorithms, models and datasets, it is concluded that SCRDet-R^{2}CNN is the best performing CNN with rotated bounding boxes, with an F1 score of 77.99 for small vehicles and 87.89 for large. An honorable mention is plain Mask-RCNN, which gave highly accurate results despite being trained on a much smaller dataset, accomplishing an F1 score of 67.59.

1 Introduction

Tourism is an important social arrangement which has an impact on many fields, most importantly in the economy, society and in the environment. Tourism does not have its own sector - it consists of several other sectors, which are also factors for its success, such as transportation, accommodation, entertainment, finance, agriculture, catering, etc. The extension of tourism into those sectors has given it many definitions, and furthermore its steady growth has made the task of analysing and measuring the general tourism rate more sophisticated than in the past. To encourage tourism growth, it is important to improve the analysis of tourism.

With the current available technology, the task of measurement and calculation is mostly done automatically with computers, however with tourism that is still a challenge. Few potential key methods of tourism measurement are the following:

- Surveys
- Tracking visitations to attractions, accommodation establishments, tourism information centres, events, etc.
- Open-space occupancy (e.g. parking lots, marinas)

This paper concerns the quantification of tourism by calculating the occupancy of open spaces such as car parks and marinas. This will be achieved by gathering a dataset of satellite images zoomed into open spaces (e.g. parking lots) and the development of a convolutional neural network to detect the objects of interest (e.g. cars or boats). Once sufficiently trained with the dataset, the CNN must be able to detect and count objects in a given image. A state-of-the-art satellite image dataset called DOTA by Zhen Zhu et. al[1], which consists of more than 10 000 cropped images, could also be made use of.

Several other researchers have similar work, such as the articles "Segment-before-Detect: Vehicle Detection and Classification through Semantic Segmentation of Aerial Images"[2] by authors Audebert N., Le Saux B., Lefevre S., which is making use of a 3-step architecture - semantic segmentation to create pixel-level class masks with a convolutional neural network, vehicle detection by regressing the bounding boxes of the elements and lastly - object detection with a traditional classification CNN, and "Deep Learning Approach for Car Detection in UAV Imagery"[3] by Ammour N., Alhichri H., Bazi Y., using over-segmentation to decrease the amount of analysis to be done, feature extraction and region classification. Further object-detection algorithms and model structures could also be effectively used, such as YOLOv3 and Mask-RCNN. YOLOv3, (Joseph Redmon, 2018)[4] an improved version of the Yolo algorithm, is one of the most precise and fast object-detection algorithms. The initial version, Yolo, processes images at 45 frames per second and a smaller version - 125 frames per second, outperforming most state-of-the-art solutions to object detection at the time. YOLOv3 introduces a bigger network with higher accuracy and roughly the same speed. The YOLO network's architecture consists of 24 convolutional layers followed by 2 fully connected layers. The convolutional layers of the network are used for feature extraction on the images. Lastly, the fully connected layers give an output of probabilities and coordinates. Furthermore, YOLO's detection method is able to make context out of the picture as a whole, which reduces the chances of mistaking the background as the desired object. Another object detection and instance segmentation algorithm, Mask-RCNN (Kaiming He et al. 2018)[6] referred by the paper as a conceptually simple, flexible and general framework for object instance segmentation and detection. Mask-RCNN extends Fast RCNN with the addition of a branch for the prediction of an object mask on top of the already existing bounding box recognition branch. This concludes that in comparison to Fast RCNN, which has two outputs - class label and an offset of bounding boxes, Mask-RCNN has an additional output of the object mask. Furthermore, one of the main differences is that it has pixel-to-pixel alignment, which is missing in both, Fast and Faster RCNN. Another key difference is in the detector - the detection in Faster RCNN consists of two stages - the first stage makes proposals of object bounding boxes using a Region Proposal Network. The second stage is performing feature extraction from the bounding boxes, classification of objects and finally - bounding box regression. Mask-RCNN is almost identical, the one key difference being in the second stage, where it predicts the class and box offset in parallel with outputting a binary mask for every Region of Interest. Mask-RCNN is also easily generalizable for other tasks, making it a desirable choice for different projects. Mask-RCNN and YOLOv3 are both great state-of-the-art choices, however, their efficiency depends on the use cases. The original YOLO algorithm's paper suggests that one of the limitations of the algorithm is the difficulty to find small objects, especially small objects which are positioned in a flock. This assumes that YOLOv3 could have disadvantages detecting tightly parked cars from satellite images or other objects that are

Alpay Vodenicharov is a Computing Science student at the NHL Stenden University of Applied Sciences, E-mail: alpay.vodenicharov@student.nhlstenden.com.
Ioannis Katramados is a researcher at the NHL Stenden Centre of Expertise in Computer Vision & Data Science, E-mail: ioannis.katramados@nhlstenden.com.
Martin Dijkstra is a researcher at the NHL Stenden Centre of Expertise in Computer Vision & Data Science, E-mail: martin.dijkstra@nhlstenden.com.

closely placed in open-spaces. Despite Mask-RCNN having higher correct localization predictions than YOLO, it makes more errors in the image background, mistaking it for the object, whereas one of YOLO's specialties is discovering the image background. In this project's case, making use of Mask-RCNN is assumed to result in many false-positives, but those could potentially be reduced by eliminating the background with YOLOv3 or YOLO, achieved by combining the two models. Certain aspects of those algorithms, such as the combination of Mask-RCNN and YOLO with the image processing from Ammour N. et al and Audebert N. et al, could be implemented in this proof of concept.

Once the PoC is complete, the following main research question will be answered: **Is it feasible to detect objects in satellite pictures of open-space areas and measure their occupancy?**

2 MATERIALS AND METHODS

2.1 Dataset

One of the defined classes which the final neural network must detect is cars from satellite images. The dataset that was used for the training of car detection was built from scratch. A further large-scale dataset, called DOTA, has also been used in regards of more complex networks. The networks in this paper will make use of the custom dataset, consisting of Google Maps images in satellite view and the DOTA dataset, which is assumed to be sufficient for this PoC.

2.1.1 Custom sample collection

The process of gathering the samples for the custom dataset is described in the process diagram in Appendix A:

The samples can be very easily collected with the use of a web interface that accepts the name of a city as a parameter. On confirmation, a request to the Google Maps API is made to fetch all parking lots in that particular city, furthermore picture of the parking lots in satellite view are downloaded in PNG format and lastly - they are ran through the classifier to detect if a picture contains a parking lot or is a false positive and store it in the appropriate folder. The classifier is a basic CNN created with the goal to detect false positive or pictures of indoor parking lots fetched from Google Maps. Since those images had no visible parking lots, they were excluded from the dataset. The CNN used has a basic architecture pictured in Appendix B.

Sample images of the custom dataset are shown below.

Fig. 1. Custom dataset sample image 1

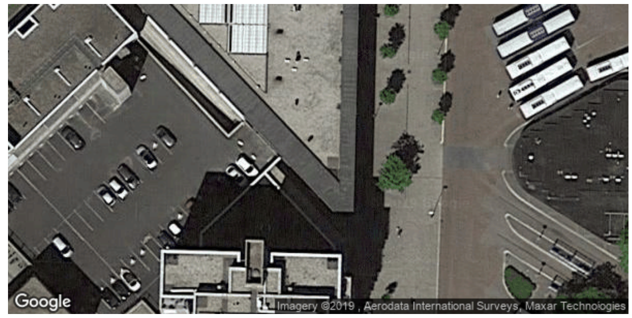

Fig. 2. Custom dataset sample image 2

2.1.2 DOTA

The DOTA dataset is a large-scale dataset built for the detection of objects or areas from aerial/satellite images, most notably vehicles, airplanes, ships and sports fields, sea fields (swimming pools, reservoirs, etc.) and specific building types. It is one of the biggest datasets available on aerial and satellite images, consisting of 2806 images with dimensions of 4000 x 4000 pixels. The images contain more than 188,282 instances of objects, annotated using arbitrary quadrilaterals, as stated by Zhen Zhu et. al.

2.1.3 Annotation

A total of 360 positive and 420 of negative samples have been collected, majority of which were done by the web interface, which is elaborated in the below sub-chapter. Those samples were used to train the above-mentioned classifier. 147 out of the 360 positive samples have been annotated. The annotation process was initially done by using an annotation tool called LabelImg. Once 80 images were annotated, the network that was trained was able to output some of the cars in the images. This allowed for the creation of a web tool that made the annotation process easier. The VOC annotation format was used. The DOTA dataset uses the same standard, however, instead of one x_{min} y_{min} and x_{max} y_{max} it takes a bounding box annotation of x_i and y_i where i is the vertices of the edges of the object.

2.2 Software and Hardware

2.2.1 Web Interface

To ease the process of data gathering, annotation and analysis, a web interface, currently hosted locally, has been developed using Node.js with the Express MVC framework. The reason behind the decision to use Node.js is due to the simplicity of an all-javascript web application, its speed, and the ease of executing python processes, which are vital for model inference and training. After the first batch of training, the network was able to output coordinates of bounding boxes wrapped around each car. The initial results were inaccurate, however, using HTML5's Canvas library, an online annotation tool was created. Every sample downloaded from Google Maps API to the web server was put through the neural network to output potential coordinates of bounding boxes and store the coordinates in a MongoDB database. Within the web interface, the users are able to change, delete and add new bounding boxes and store the new, corrected coordinates in the database. Lastly, the web interface gives a visual representation of the system to users. Currently the following actions can be done through the interface:

- Add custom dataset
- Create model with custom class for object detection
- Fetch parking lot samples based on city
- Track progress of the models in training
- Perform CRUD on the database
- Annotate samples

2.2.2 Software

The usage of the appropriate software was vital to achieve the end-goal, starting with Python version 3.7 and the Mask-RCNN[7] neural network library for the model training and inference. Further Python packages for assistance include NumPy, sklearn, matplotlib, OpenCV, and others. Behind Mask-RCNN lies the Keras High-Level neural network API library. To perform GPU training efficiently, a Cuda version of 10.1 was also made use of. The web application consists of the usage of Node.js version 10.15.3 and Express3 for the backend. The frontend consists of in-built technologies such as pure javascript and HTML5 Canvas for the annotations. Google Maps API is used for the retrieval of locations and samples.

2.2.3 Mask-RCNN

According to Ross Girschick et. al. (2018) Mask-RCNN takes the approach of detecting objects in an image and generating a mask for the objects simultaneously. This makes Mask-RCNN a simple and effective approach for initial testing of whether object detection from aerial or satellite images is feasible.

Stated by Xiang Zhang (2018)[8], Mask-RCNN's structure is based on two stages. The first stage consists of four convolutional layers, which extract feature maps of an input image. An additional four pooling layers are succeeding the convolutional layers in order to down sample the feature maps by summarizing the presence of features in patches of the feature maps, as explained by Jason Brownlee (2019)[9]. Lastly, the sample is passed to a Region Proposal Network, which outputs a binary class and proposed regions with potential bounding boxes. In the second stage, another network receives the ROI and defines the classes and draws bounding boxes and masks of the objects.

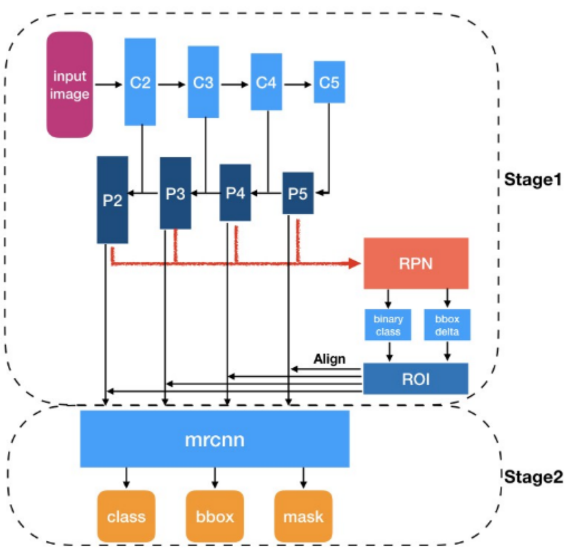

Fig. 3. Mask-RCNN Neural Network Structure (Xiang Zhan, 2018)

2.2.4 Rotated Bounding Boxes

In addition to Mask-RCNN, which outputs horizontal bounding boxes, other neural networks that output rotated bounding boxes will also be considered. Both methods will be tested and finally compared. The neural networks that will be used are the following:

- DRBox Algorithm
- SCRDet-R^{2}CNN

DRBox is a deep-learning object detection network which focuses on the detection of vehicles, airplanes and ships in remote sensing images. DRBox is capable of outputting rotated bounding boxes by making it learn not only the location and size as most generic networks, but also the angle of the target objects. This gives the network the awareness of orientation, which makes the results of DRBox rotation-invariant. The following figure showcases results originating from DRBox:

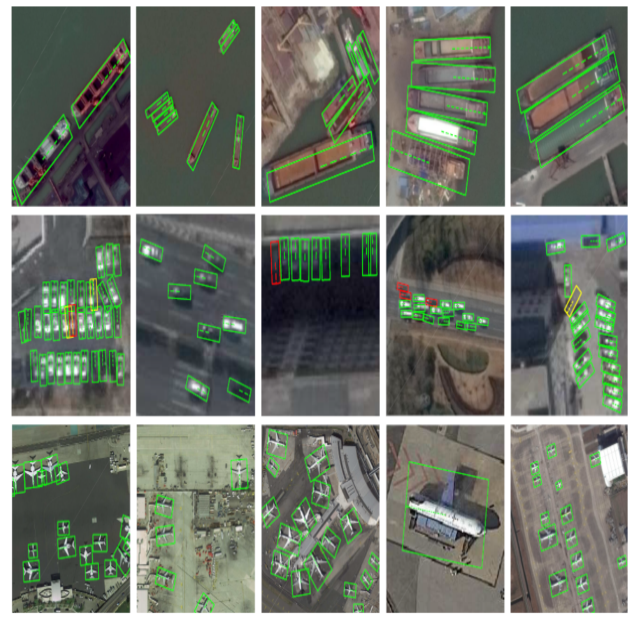

Fig. 4. DRBox official testing results on DOTA testing dataset

DRBox stems from the Caffe[10] deep-learning framework. Liu Lei et al[11] states that the input image is initially put through multi-layer convolutional layers to generate detection results. The final convolutional layer being for prediction and the others - for feature extraction. The sliding boxes that search for the object, in the case of DRBox, are also rotating. The boxes are rotated to different angles with different confidence levels and finally, the predicted bounding boxes are sorted by confidence level.

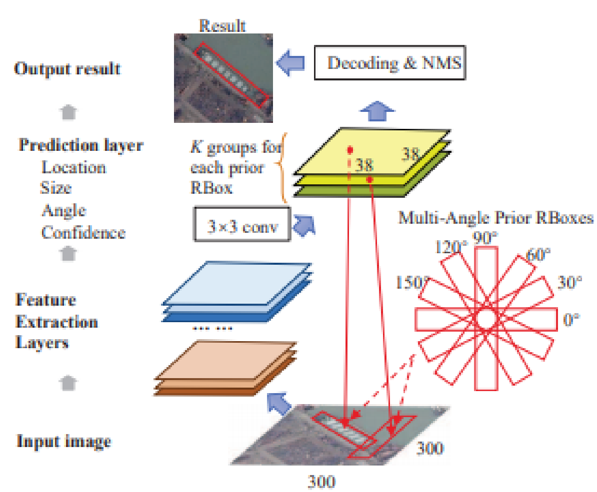

Fig. 5. DRBox Neural Network structure (Lei Liu, 2017)

SCRDet-R^{2}CNN is a robust detection network for small, cluttered and oriented objects, based on R^{2}CNN. This makes the network very effective for detecting objects from aerial or satellite images. Lastly, according to Xue Yang et al[12] the network has an added IoU constant factor to the LI loss, which addresses the boundary problem of the rotating bounding box. Lastly, the bounding box inputs that are given to the network are - x_0 y_0, x_1 y_1, x_2 y_2, x_3 y_3 - which represent the vertices of the 4 points that make up the bounding box, x_0 and y_0 always being the top left corner of the detected object.

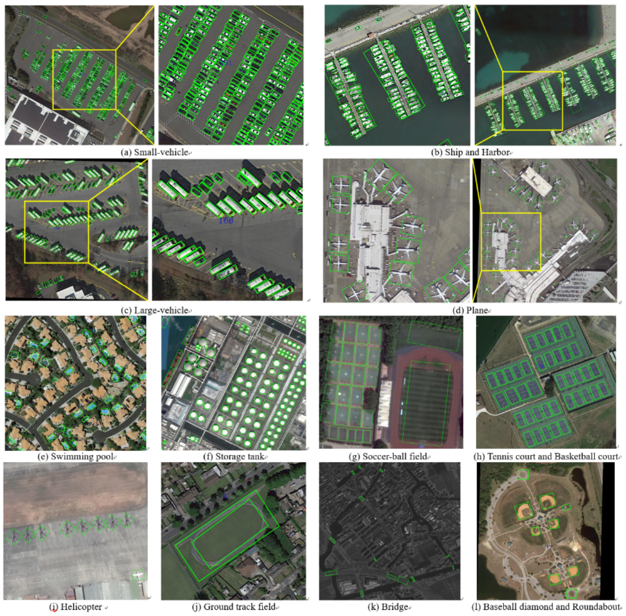

Fig. 6. SCRDet-R^{2}CNN official Results on the DOTA testing dataset

The structure and process of SCRDet-R^{2}CNN consists of two stages - at first, the feature map has reduced noise and more feature information by adding SF-Net and MD-Net. This stage still regresses a horizontal box. With the required parameters and the rotation non-maximum suppression (R-NMS) operation for the proposals, a final oriented result is received. The structure is visualized in Appendix C.

2.2.5 Hardware

The following hardware described in table 1. were used to train the neural networks:

Table 1. Hardware used to train the neural networks.

Hardware	Specification
GPU	NVIDIA RTX2070
GPU memory	8GB
RAM	14.75GB
CPU	12 Cores
OS	DEbian 10.1
CUDA versions	10.1 and 9

3 Experiments

The end goal is to achieve high accuracy in counting objects of interest while maintaining a good performance, hence why multiple networks and structures are going to be experimented. The initial experiments will be performed on an implementation of Mask-RCNN using Keras to estimate the feasibility of object detection of aerial and satellite images. Since Mask-RCNN outputs horizontal bounding boxes, other neural networks with support for rotational bounding boxes will also be experimented. The latter is assumed to have less background clutter for easier counting. The results from the networks that tackle rotating boundig boxes - DRBox and SCRDet-R^{2}CNN, will be visualized, analyzed and compared. The final results of those networks will be directly assisting in the measurement of open-space area occupancy by calculating how busy certain areas are.

3.1 Experiment: Mask-RCNN

The experiments to be performed with Mask-RCNN will ascertain the feasibility of object detection in satellite images. A plain Mask-RCNN network will be trained on the custom dataset that was gathered using the Google Maps API. The training of the model will be done with the parameters described below in table 2.: The dataset is split into

Table 2. Mask-RCNN training parameters and values.

Parameter	Value
Epochs	900
Epoch Steps	1000
Learning Rate	0.001
Learning Momentum	0.9
Batch Size	2
Backbone	Resnet101
Pool Size	7
Mask Pool Size	14

300 images for training, 100 for validation and 50 for testing. Once trained, inference will be done on a set of sample images from Google Maps that were not included in the training dataset.

3.2 Experiment: DRBox

The experiments with DRBox are intended to achieve detection with rotated bounding boxes. The DRBox model, which was pre-trained on the DOTA dataset, will be ran on a small set of samples from the custom-gathered Google Maps dataset and later the results analyzed and compared to other solutions. The following prerequisites must be satisfied before the experiments can take place:

- Caffe installation and configuration
- MATLAB
- VGGNet
- gcc 6.0
- Cuda 9.0
- NVIDIA GPU with atleast 12GB of memory

3.3 Experiment: SCRDet-R^{2}CNN

SCRDet-R^{2}CNN, similarly to DRBox, will be experimented to achieve detection with rotated bounding boxes. This network will be trained on a modified DOTA dataset, where images that do not contain the objects of interest will be removed, resulting in a total of 7600 images. The pictures used for validation are an additional part of the DOTA dataset - a total of 6200 images, and lastly the testing will be performed on a small set of images of parking lots from Leeuwarden. The network will have the task to detect two classes - large and small vehicles. The training will be done with the following configuration parameters described in table 3.:

Furthermore, the following RPN configurations are found in table 4.:

Lastly, the Fast-RCNN parameters are presented in table 5.:

The requirements to run SCRDet-R^{2}CNN can be found in Appendix D.

4 Results

The final results from the experiments performed and designed using different algorithms and tools will be elaborated in detail in this chapter.

Table 3. SCRDet-R^{2}CNN training parameters and values.

Parameter	Value
Fixed blocks	2
RPN location loss weight	1 / 7.0
RPN classification loss weight	2.0
Fast-RCNN location loss weight	4.0
Fast-RCNN classification loss weight	2
RPN sigma	3.0
Backbone	Resnet101
Momentum	2
Learning rate	0.0003
Epoch iterations	21 000
Batch size	1

Table 4. SCRDet-R^{2}CNN RPN configurations.

Parameter	Value
Kernel size	3
RPN IOU positive threshold	0.7
RPN IOU negative threshold	0.3
RPN minibatch size	512
RPN positive rate	0.5
RPN sigma	3.0

4.1 SCRDet-R^{2}CNN

SCRDet-R^{2}CNN has shown to have the highest F1 score of the networks compared. This algorithm surrounds the objects of interest with a rotated bounding box with the least possible amount of background noise. The outputs are 2 images - one with the objects surrounded with a horizontal bounding box and another - with a rotated bounding box The following images are inferenced pictures with horizontal bounding boxes:

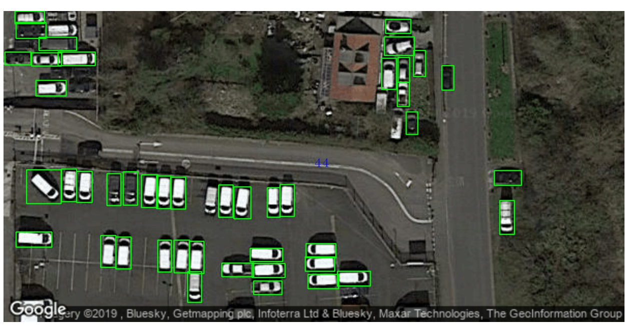

Fig. 7. SCRDet-R^{2}CNN Horizontal inference resulting image

Fig. 8. SCRDet-R^{2}CNN Horizontal inference resulting image

Table 5. SCRDet-R^{2}CNN parameters.

Parameter	Value
ROI size	14
ROI pool kernel size	2
Keep probability	1.0
Score threshold	0.1
Fast-RCNN Horizontal NMS IOU threshold	0.4
Fast-RCNN Rotated NMS IOU threshold	0.1
Fast-RCNN IOU positive threshold	0.4
Fast-RCNN IOU negative threshold	0.0
Fast-RCNN positive rate	0.4

In the images below are the rotated versions of the bounding boxes, which remove more noise and give a clearer representation of the object.

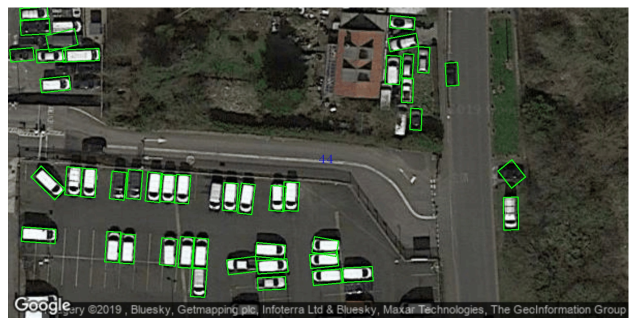

Fig. 9. SCRDet-R^{2}CNN Rotated inference resulting image

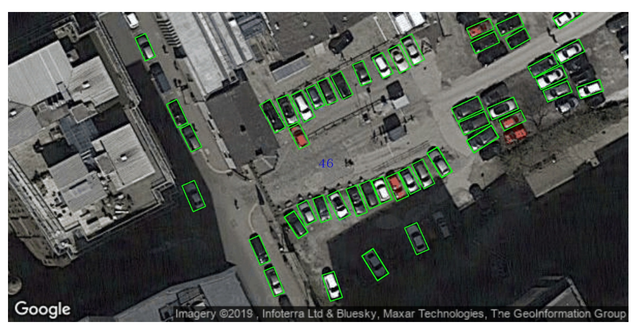

Fig. 10. SCRDet-R^{2}CNN Rotated inference resulting image

Several inconsistencies do exist in the case of pictures with smaller resolution, which portray the cars in the images smaller too, making it harder for the network to detect them. One of those examples is below:

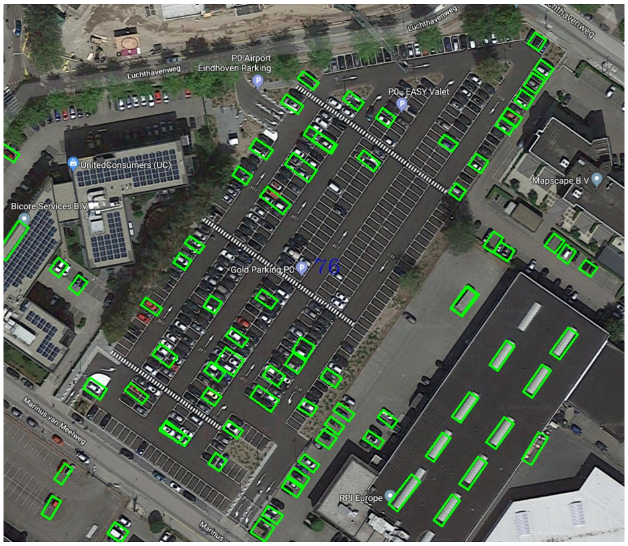

Fig. 11. SCRDet-R^{2}CNN Inconsistent inference image

The following table 6. represents the final results - F1 score, recall and precision of the network.

Table 6. SCRDet-R^{2}CNN F1, precision, recall and classes.

F1	Recall	Precision	Class
77.99	92	68	small-vehicle
87.89	88	86	large-vehicle

4.2 Mask-RCNN

Mask-RCNN also has produced desirable results, however it is less precise than SCRDet-R^{2}CNN and contains more false-positives. Another downside is the horizontal bounding boxes which create more noise and less clear representations of the object. The following plots showcase the results that were inferred:

Fig. 12. Mask-RCNN inference resulting image

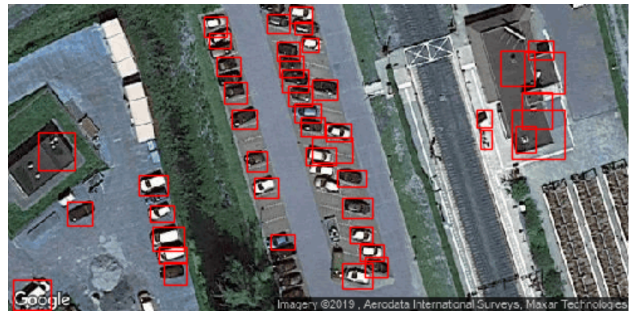

Fig. 13. Mask-RCNN inference resulting image

Fig. 14. Mask-RCNN inference resulting image

Finally, table 7. contains the evaluaton results for the Mask-RCNN network:

Table 7. Mask-RCNN F1, recall, precision and class.

F1	Recall	Precision	Class
67.59	81	58	small-vehicle

4.3 DRBox

DRBox has successfully produced rotated bounding boxes, however, the object detection is highly inaccurate. Furthermore, DRBox allows for the detection of one class only, which does not convenience the research requirements. In the majority of the cases, a high percentage of the bounding boxes are missplaced. The following image depicts the results from DRBox:

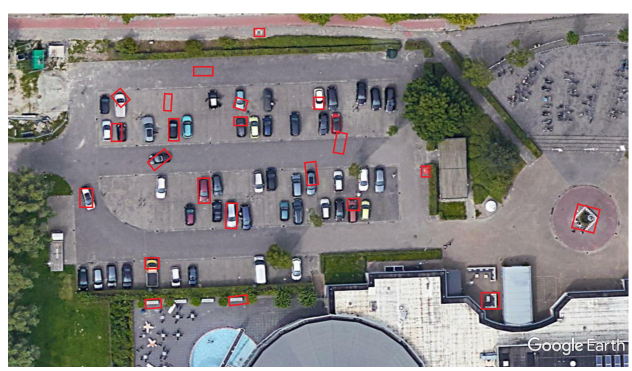

Fig. 15. DRBox resulting image

5 Discussion and Conclusion

After an extensive research has been conducted on the problem description of "how accurate can a neural network measure tourism and calculate objects of interest in an open area from satellite view?", a conclusion has been reached.

The first experiment, conducted using Mask-RCNN, had the goal to give an overview of whether it is feasible to detect objects of interest in satellite images. 450 samples have been fetched from Google Maps and annotated using the VOC PASCAL format and trained on 900 epochs. Despite the small dataset, the output of Mask-RCNN was surprisingly accurate, with certain false-positives. Mask-RCNN holds great potential for further experiments with a larger dataset, but is not viable for current usage. It has, however, presented a new issue - the problem of horizontal bounding boxes. Horizontal bounding boxes take additional space, increasing the noise inside the region of the object, which creates difficulties in flock of objects. Furthermore, the output image becomes less clear. DRBox is one of the solutions that tackles this problem by accepting a 5th variable in the annotation - the angle, which makes the model learn about the orientation of the objects. The DRBox model was trained on the DOTA dataset, containing over 14,000 aerial images. Despite the large-scale dataset, the results from DRBox were not sufficient for use. A substantial amount of the bounding boxes were placed wrong, however, the bounding boxes were rotated. It is concluded that DRBox is not viable for current and future use.

Finally, the SCRDet-R^{2}CNN network, which was also trained on a modified DOTA dataset, where images with only unnecessary objects such as sports fields, water tanks, etc.were removed, provided the best possible results from the experimented options with an F1 score of 77.99% for small vehicles and 87.89% for large vehicles. SCRDet-R^{2}CNN, unlike DRBox uses polygon bounding box annotation, which connects 4 points with their X and Y coordinate. This approach has proven to be successful by the experiments conducted and their results. SCRDet-R^{2}CNN has the highest precision rate, with few inconsistencies in smaller resolution pictures. This issue can be tackled by extending the DOTA dataset with smaller resolution pictures. In conclusion, SCRDet-R^{2}CNN creates desirable results and is usable presently.

In conclusion, Mask-RCNN shows promising results, however, horizontal bounding boxes are not as efficient. With an F1 score of 67.59 for small vehicles, Mask-RCNN is second in terms of suitability, DRBox being last with majority of its detections being wrong, as pictured in the results. For rotating bounding boxes and detection, SCRDet-R^{2}CN outperforms both networks with an outstanding F1 score of 87.89 for large vehicles, such as trucks, and 77.99 for small vehicles. Thus, object detection in satellite images is shown to be feasible and measurable.

6 Future Work

After extensive research, it has been concluded that object detection in aerial images is a feasible goal. This gives room for improvements on the already made progress in this paper.

6.1 DOTA dataset

Most of the false-positives or missed detections stem from the size of the input images. The DOTA dataset generally contains samples of 800x800/1200x1200 resolution. When a bigger/smaller input is given, difficulties will arise. Those could be avoided by further extending the dataset with more bigger and smaller samples. Due to time limitations in this project, enough amount of diverse pixel pictures could not be gathered.

6.2 Mask-RCNN

Mask-RCNN has produced promising results, thus it could be a subject for further extension. Recommended future work on this network would be to train on a larger dataset and add awareness to orientation.

6.3 SCRDet-R^{2}CNN

SCRDet-R^{2}CNN is a large network with orientation awareness. Currently, it has only been trained for 21 000 steps due to time limitations. Yang Xue et al have trained their functional network for 600 000, which gives better results, however, their use case includes other unnecessary objects for this project such as detecting football fields, water reservoirs, etc. which in the case of this project, results in unnecessary detections, hence why the use of their pre-trained network is not encouraged.

References

[1] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, Liangpei Zhang (May 21, 2019) DOTA: A Large-scale Dataset for Object Detection in Aerial Images. Retrieved from https://arxiv.org/pdf/1711.10398.pdf

[2] Audebert, N.; Le Saux, B.; Lefèvre, S (13 April 2017) Segment-before-Detect: Vehicle Detection and Classification through Semantic Segmentation of Aerial Images. Remote Sens. Retrieved from https://www.mdpi.com/2072-4292/9/4/368

[3] Ammour, N.; Alhichri, H.; Bazi, Y.; Benjdira, B.; Alajlan, N.; Zuair, M. (27 March 2017) Deep Learning Approach for Car Detection in UAV Imagery. Remote Sens. Retrieved from https://www.mdpi.com/2072-4292/9/4/312

[4] Joseph R., Ali F. YOLOv3: An incremental improvement. Retrieved from https://pjreddie.com/media/files/papers/YOLOv3.pdf

[5] Joseph R., Ali F., Santosh Divvala., Ross G., Yolo: Unified, Real-time object detection. Retrieved from https://pjreddie.com/media/files/papers/yolo.pdf

[6] Kaiming He, Georgia G., Piotr D., Ross G.. (2018) Mask-RCNN. Retrieved from https://arxiv.org/pdf/1703.06870.pdf

[7] Matterport (2019) Mask-RCNN. Retrieved from https://github.com/matterport/Mask_RCNN

[8] Xiang Zhang (Apr 22, 2018) Simple Understanding of Mask-RCNN. Retrieved from https://medium.com/@alittlepain833/simple-understanding-of-mask-rcnn-134b5b330e95

[9] Jason Brownlee (Apr 22, 2019) A Gentle Introduction to Pooling Layers for Convolutional Neural Networks. Retrieved from https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/

[10] BVLC (2018).Caffe. Retrieved from https://github.com/BVLC/caffe

[11] Lei Liu, Zongxu Pan, Bin Lei (2017). Learning a Rotation Invariant Detector with Rotatable Bounding Box. Retrieved from https://arxiv.org/abs/1711.09405

[12] Xue YangJirui Yang, Junchi Yan, Yue Zhang, Tengfei Zhang, Zhi Guo, Xian Sun, Kun Fu (2018) SCRDet: Towards More Robust Detection for Small, Cluttered and Rotated Objects. Retrieved from https://arxiv.org/pdf/1811.07126.pdf

[13] Sean White (n.d) Measuring Tourism Locally. Retrieved from https://www.visitbritain.org/sites/default/files/vb-corporate/Documents-Library/documents/England-documents/2_eco_impact.pdf

7 Appendices

A

Data Collection Process Diagram

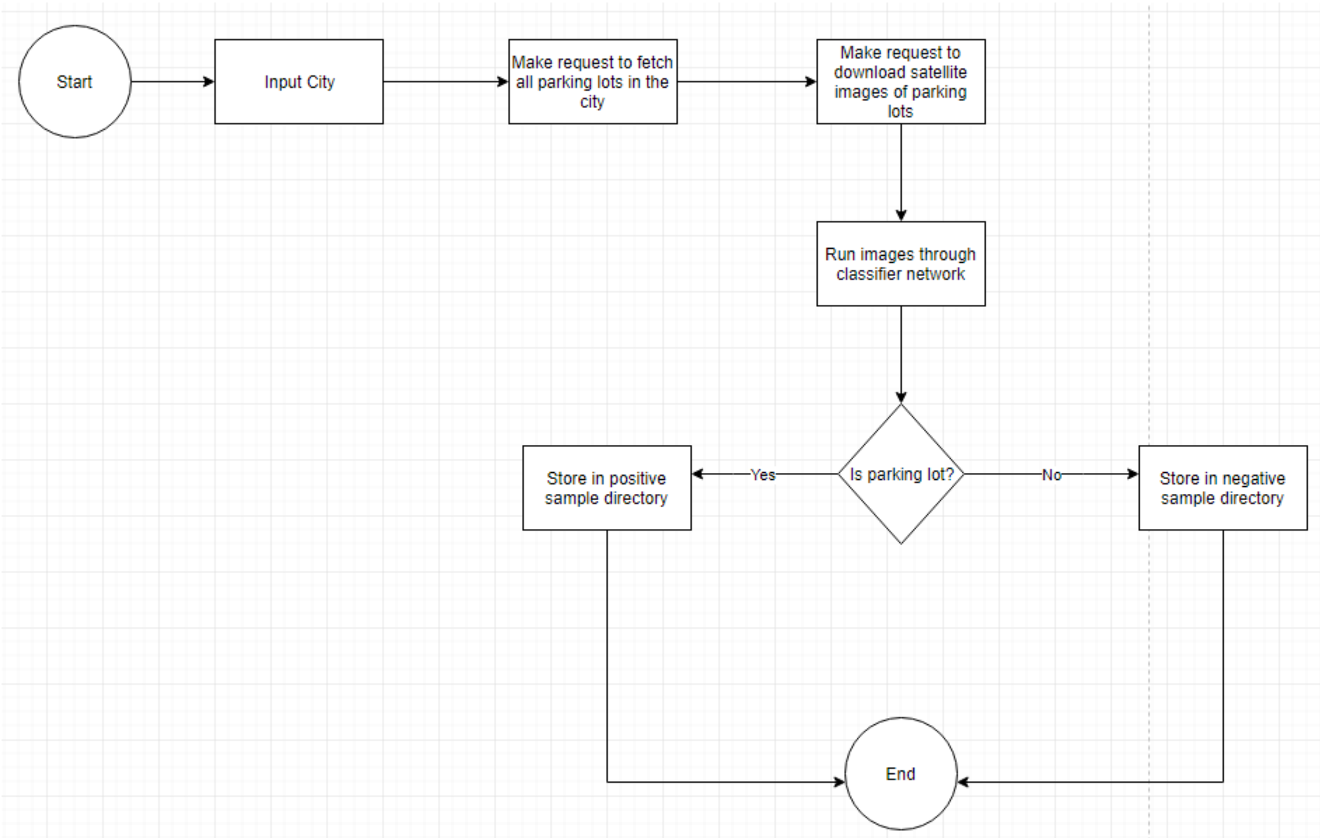

Fig. 16. Data collection process diagram through the web interface

B

Classifier Architecture

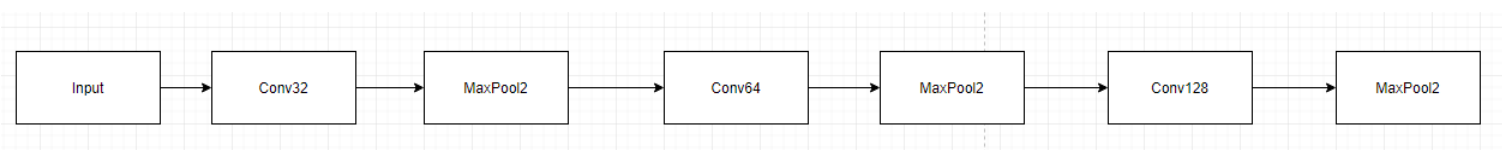

Fig. 17. Parking Lot Classifier architecture

C
SCRDet-R^{2}CNN structure

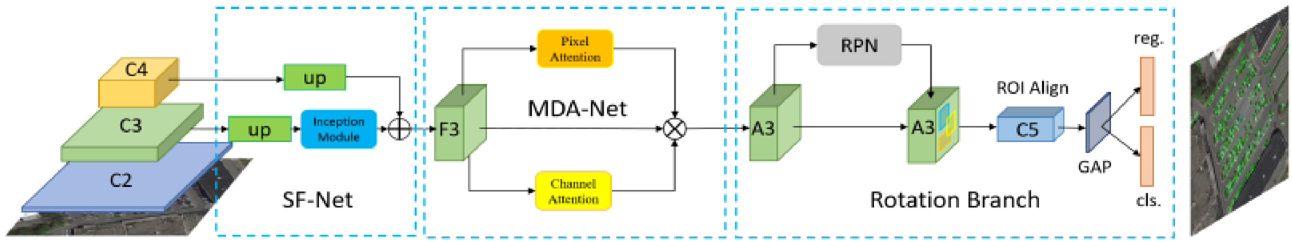

Fig. 18. SCRDet-R^{2}CNN Neural Network structure

D
Requirements for SCRDet-R^{2}CNN

- Python 2.7
- opencv 2
- CUDA >= 8.0
- Tensorflow >= 1.2.0
- Pre-trained models Resnet50 and Resnet101, Mobilenetv2 and a custom-trained R2CNN model