Simultaneous regression-based spatial coverage estimation and object detection with deep learning

Andersen, Rasmus Eckholdt; Nalpantidis, Lazaros; Ravn, Ole; Boukas, Evangelos

Published in:
Electronics Letters

Link to article, DOI:
10.1049/ell2.12183

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

Simultaneous regression-based spatial coverage estimation and object detection with deep learning

Rasmus Eckholdt Andersen,✉ Lazaros Nalpantidis, Ole Ravn, and Evangelos Boukas

Section of Automation and Control, Department of Electrical Engineering, Technical University of Denmark, Kongens, Lyngby, Denmark (E-mail: lanalpa@elektro.dtu.dk, or@elektro.dtu.dk, evbou@elektro.dtu.dk)

✉Email: recan@elektro.dtu.dk

Object detection has been in the focus of researchers within varying applications propelled by the recent advances in deep learning and neural networks. Many applications require both detection of class instances as well as a quantification of the spatial coverage of the class instances. While the performance of deep learning approaches for these tasks has been extensively studied there has not been much effort into creating a unified network structure to achieve both goals. The purpose of this paper is to present a regressor to the faster R-CNN architecture that can help quantify the spatial coverage estimation of some detected object. The goal of the regressor is to provide a reproducible result of the spatial coverage. To demonstrate the developed architecture, an example use-case of land cover estimation is used. The experiments conducted in this paper show that the network does not sacrifice object detection accuracy, and indicate that the network is able to estimate the spatial coverage of six different types of land.

Introduction: The ability to quantify areas of interest to get an estimation of how large an area is affected by a given class instance is intrinsic to inspection processes and even occurs in some surveying applications. Using visual sensory information for inspection is a fast way to cover large and hard to access areas, however, it also produces a large amount of information to be processed. If this information has to be processed manually, the results risk being biased towards the humans' subjective assessments, which can vary over time and are not necessarily consistent among humans. Though object detection is capable of analysing large quantities of image data, it does not cover the use cases in which the spatial coverage is of interest rather than the actual object. This missing factor makes object detection a solution to only the localisation of areas of interest, and not the quantification of the areas in the image in relation to the rest of the image.

The advantage of quantification being part of the object detector is to reduce the number of steps required to obtain the desired output. This saves time and complexity when training the object detector. Additionally, by merging the object detection and any quantification network into a single network, the number of parameters that have to be trained get significantly reduced, compared to having separate networks for the two tasks, since a large number of parameters can be shared. Another approach to solve this problem is to perform full segmentation and count the number of pixels in the image for a class instance to obtain a coverage. However, additionally to the computational disadvantage, training such networks requires more extensive labelled data and the segmentation adds potentially unnecessary complexity for an output that can be achieved with a simpler regression. Applications such as space exploration present difficulties both in terms of data availability and computational resources [1, 2].

In this paper, a modification to the faster R-CNN architecture is introduced with the purpose of performing spatial coverage estimation in images. To show that the network does not suffer object detection accuracy degradation after being modified to also support spatial coverage estimation, its detection performance is compared to that of an unmodified faster R-CNN. The accuracy of the spatial coverage estimation is evaluated via an accuracy plot against the ground truth.

In the following section, an overview of some of related work for such an architecture is presented, followed by an introduction to the modifications performed on the faster R-CNN architecture. Finally, as an example of the capabilities of the developed architecture, a dataset of satellite images is used to detect the spatial coverage of six different types of land cover.

Related work: The recent advances in vision-based deep learning has prompted for extensive research in object detection architectures, capable of localising objects within an image. Previously, object detection has been used for corrosion detection for inspection of marine vessels, detecting palm trees, and ice caps.

In marine vessel classification processes, the condition of the vessel is determined by the spread of the corrosion. The popular faster R-CNN [3] architecture has already been used for inspection of corrosion detection [4]. A VGG16 [5] backbone was used to extract features of surfaces on marine vessels containing corrosion and trained a faster R-CNN network to output bounding boxes around corrosion. However, their approach is limited to only localising the corrosion, and not determining the spread. This means the condition of the vessel is still dependent on a human surveyor's input for quantification of the area covered in corrosion. In the presented work, a faster R-CNN is modified to compute the percentage of the image that represents class instance, and thereby alleviating human dependency. Additionally, [6] investigated the applicability of different types of architectures and found that bounding boxes are not suitable for estimating the spread of corrosion.

Deep learning has also been used in an aerial observation. One application has been to detect and localise palm trees [7]. In this work, images collected with a drone are used to localise palm trees using a faster R-CNN architecture. While the palm trees are successfully detected, and subsequently counted, the network does not provide a quantification of the density of the trees. By measuring the area the palm trees cover, an estimation of not only the physical size of the forest or plantation can be obtained, but it would also allow for tracking the growth of the area. Tracking the ice caps has been of great interest in the recent years as the ice melts away. A typical way of tracking the ice employs synthetic aperture radar (SAR) measurements from satellite imagery. The ice cover can then be tracked from the acquired images as shown in [8] in which the ice is segmented using a U-net. The need for segmenting the whole image can be more computationally intensive and not necessarily required if the only interest is a general estimation of the ice cap decay or growth over time.

Architecture: The object detector used in this paper is the faster R-CNN due to its separation of feature extraction (i.e. backbone) and detection. The advantage of the separation is that the features can easily be re-used without modifying any other branch of the network. The backbone used for this faster R-CNN is a feature pyramid that utilises not only the deeper layers, but also the more shallow layers. This means the network can learn to utilise both lower level abstract features from the image, but also higher level features directly, resulting in more robust scale invariance. The combination of the aforementioned features makes the network capable of both detecting class occurrence and producing bounding boxes. In order to perform spatial coverage estimation, a third head is added to the faster R-CNN architecture. Independent of the object detector component, this head is connected to the backbone, and adds directly to the loss function, such that the backbone is also optimised towards minimising the spatial coverage estimation error. In Figure 1, an overview of the modified architecture can be seen. The faster R-CNN architecture supports a wide variety of backbones and generally the ResNet [9] backbone family performs well. Specifically, ResNet50 will be used in this work.

The feature maps passed on to regression and classification heads have the shape batch size $\times$ channels $\times$ height $\times$ width where the batch size is user defined. The number of feature channels can also be adjusted, however, for simplicity, it has been left to the default 256 as in the original paper [10]. Since ResNet does not contain any linear layers that are dimensionally constrained in the width and height channels of the image, the output of the backbone also varies. Since the classification and regression heads rely on fully connected layers, the features have to be both downsampled to a fixed size and to a manageable number of parameters —to be learned— before any spatial coverage estimation can be performed.

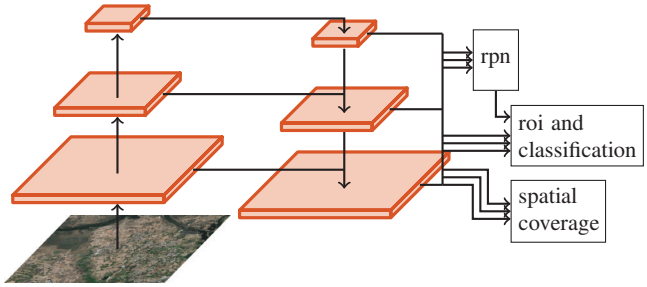

Fig. 1 *A faster R-CNN architecture with the original region proposal network (RPN) head and region of interest (ROI) pooling and classification head. Additionally, the added spatial coverage head is shown*

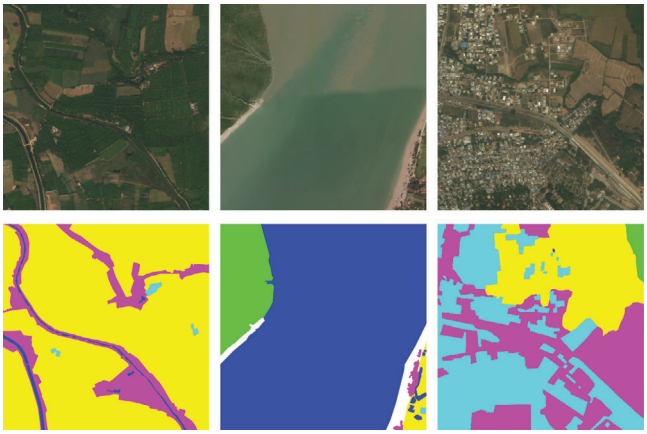

Fig. 2 *First row: original images used for training and validation. Second row: the corresponding mask. See Table 1 for colour encoding*

The task of fixing the size of the feature output is done using an adaptive pooling layer where the stride is dynamic such that the height and width of the features becomes constant. It was chosen that each of the outputs of the pyramid will be pooled into the shapes 64×64, 32×32, 16×16, 8×8 from shallow to deeper layers respectively. The number of channels is reduced by applying a single convolutional layer on each output level of the feature pyramid which will reduce the number of channels from 256 to 16, i.e. a reduction of 94%. With a fixed feature size, the spatial coverage head can be used with linear layers to perform a regression estimation of the spatial coverage. The final operation is applying a regular linear operation on the output. Since the output is a percentage it would be natural to apply a sigmoid activation on the output, however, if the dataset does not contain evenly distributed target values from 0% to 100% and instead is concentrated around the extremities, the network suffers from a vanishing gradient due to the properties of the sigmoid. The use of a linear output does not produce values outside the bounds of the interval [0,1] for the dataset considered in this paper anyway, as will be demonstrated in one of the following sections.

Data preparation: To demonstrate the versatility of the architecture, a dataset on land cover [11] is used. The dataset originates from a segmentation challenge where the goal was to segment types of land cover from satellite imagery. The six classes in the dataset are all annotated with a corresponding mask with the bounding box defined as the minimum enclosing rectangle for each segmentation. Some examples images from the dataset can be seen in Figure 2.

All images in the dataset are orthophotos of size 2448×2448 pixels where each pixel corresponds to 0.5 meters, thus each image roughly covers 1.5 square kilometres. Since the datasets are relatively small, three data augmentations were randomly applied to the images: vertical and horizontal flip is applied with a probability of 50%, and the hue of the image is shifted with a uniformly random sampled factor in the interval $[-0.5, 0.5]$. The reasoning for applying the colour "jitter" is to force the network to focus on learning shapes independently of the colour, since the colour of the landscapes tends to vary with the seasons.

Table 1. The number of images for each class along with the total number of annotations in the landcover dataset used in this paper

		Training		Validation		
	Class	# images	# annotations	# images	# annotations	
1	Urban	436	3359	211	1776	
2	Agriculture	451	1426	230	688	
3	Rangeland	342	2631	177	1461	
4	Forest	126	297	61	159	
5	Water	312	1436	162	721	
6	Barren	280	859	141	491	

The ground truth spatial coverage is obtained by counting and summing each pixel for each class and dividing with the total amount of pixels in the image, i.e. a percentage in the interval [0,1] that corresponds to that objects spatial coverage in the image.

Training procedure: For each of the classes listed in Table 1 both a regular faster R-CNN network and a modified network was trained. The loss function for the spatial coverage estimation is an L1 loss, i.e. a mean absolute error. The loss function was chosen for the behaviour in the interval [0,1], which is where the target output lies. The loss functions for the faster R-CNN heads have not been changed. Thus the final loss function is as shown in Equation (1).

$$L = L_{\text{cls}} + L_{\text{reg}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N} |y_i - x_i|}_{L_{\text{spatial}}} \tag{1}$$

L_{cls} and L_{reg} denotes the unmodified loss functions from [3], and L_{spatial} denotes the loss from the added spatial coverage estimation. The reason to not just freeze the faster R-CNN feature extraction network and train only the spatial coverage head was that an end-to-end approach allows all the heads attached to backbone to equally influence the learned features. Additionally, the network can be trained in an end-to-end fashion with no intermediate steps required.

The network uses a stochastic gradient descent optimiser with a learning rate of $5 \cdot 10^{-4}$ for the first 20 epochs, after which the learning rate is decreased to $5 \cdot 10^{-5}$ for the remaining epochs. Additionally, a $L2$ regularisation of $5 \cdot 10^{-4}$ and a momentum of 0.9 was used with the optimiser. Since training the feature extracting backbone of the network is a time consuming process that requires a large amount of labelled data, all weights in the network were initialized using pretrained weights from a faster R-CNN network trained on the COCO train2017 dataset [12]. The remaining weights were initialized using a Kaiming uniform distribution.

Trials/results: To ensure that the addition to the original faster R-CNN does not sacrifice accuracy when the modifications have been done, a vanilla faster R-CNN is trained and used as a baseline. These results can be seen in Table 2 which lists the intersection over union (IoU). The results indicate that adding a spatial coverage head to the faster R-CNN architecture does not heavily impact the accuracy of the produced bounding boxes. Specifically, for the case of urban and agriculture land quantification, the modified network performs approximately on par with the baseline with a negligible improvement in both metrics. For the class rangeland, the network produces a better average precision (AP) with the vanilla network, however, the modified network produces a better average precision for the IoU$=0.5$ case which indicates that the accuracy overall is better at IoU$=0.5$ than it is for higher thresholds, i.e. a small sacrifice in accuracy at high thresholds has gained a higher accuracy at IoU$=0.5$. For the class forest, a drop in accuracy of 14.7% when comparing the vanilla network to the modified network indicates that the modified network was severely impacted. One reason for this may be the small number of examples in the dataset. With only 127 images

Table 2. Results from six faster R-CNN networks used as a baseline. 0.5 : 0.95 *denotes the average of thresholds in the interval* [0.5, 0.95] *at 0.05 increments*

	Class	Vanilla faster R-CNN		Modified faster R-CNN	
		AP@IoU 0.5:0.95	AP@IoU 0.5	AP@IoU 0.5:0.95	AP@IoU 0.5
1	Urban	0.141	0.292	**0.144**	**0.301**
2	Agriculture	0.273	0.396	**0.285**	**0.409**
3	Rangeland	**0.126**	0.254	0.123	**0.256**
4	Forest	**0.259**	**0.351**	0.221	0.305
5	Water	**0.180**	**0.327**	0.176	0.325
6	Barren	**0.086**	0.163	**0.086**	**0.165**

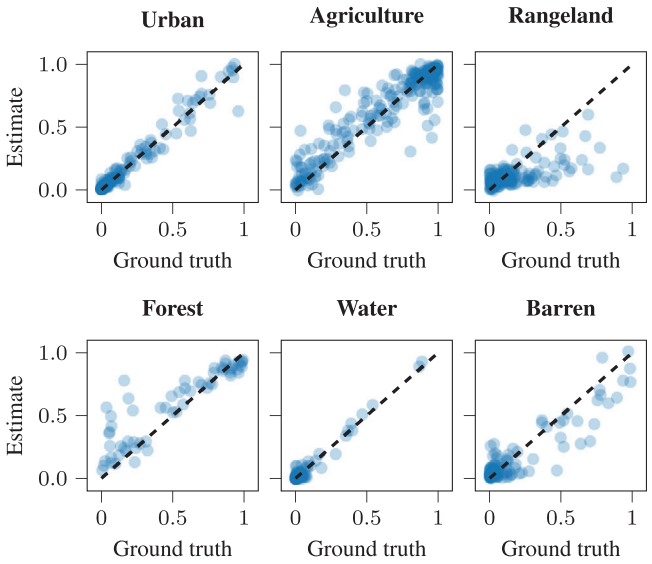

Fig. 3 *Accuracy of spatial coverage of different land types*

including 297 annotations, the class is significantly under-represented when compared to the other classes. This may be because of the network being more sensitive to the low number of training examples when there is a larger number of additional parameters to be learned. The last two classes water and barren also provided similar accuracy, however, the accuracy for the class barren is noticeably lower than any other class even though the number of images is more than twice that of the forest class.

In Figure 3, an accuracy plot of the spatial coverage estimation for the six classes is shown. For each point for each class, there exists a ground truth, which corresponds to the actual coverage of the given class in an image. The estimate is the network's output, meaning in an ideal scenario, all points should follow a straight line with slope 1, as indicated by the dotted black line.

As mentioned earlier, the output of the spatial coverage network is a single number with no activation function applied to it. The reason for this was to avoid vanishing gradients near the extremes 0% and 100%. From Figure 3, the heavy imbalance in points is clearly visible. There are either a lot of samples near 0% as shown by the classes urban, rangeland, water, and barren, whereas the classes agriculture and forest have more points near 100%. If an activation function like the sigmoid was to be used, the network would be forced to learn from a gradient that is near horizontal, i.e. approaching 0 which gives no information to back propagate in the network. Even though a linear output layer is used, the network does not produce negative samples that would not be rounded to zero when using a reasonable number of significant decimals. This justifies the choice of activation function and provides precedence for such a decision design to be used in other fields when the training data representation is non-homogeneous.

As shown in Figure 3, the network is able to learn an indication of the trends within the different classes. Specifically, the urban class shows a good match between the ground truth and the estimated spatial coverage. While the agriculture class also exhibits a linear trend, there is significantly more noise on the estimations. From these results, the network has a better chance of quantifying the area with water in an image than any other class. From Figure 3, some outliers can also be seen for rangeland, forest and barren land, indicating the network has difficulties quantifying these types of land.

Conclusion: There are several use-cases where performing spatial coverage is of interest to track progress or for inspection processes. In this paper, a modified faster R-CNN architecture with a feature pyramid backbone was presented capable of estimating spatial coverage. The modifications consisted of adding a third head that, after performing down sampling, forwards the features to a regression head trained using a L_1 loss. As an example, the network was trained on a dataset consisting of land cover from satellite images. The results show that the modification to the faster R-CNN architecture did not suffer significant loss of accuracy, and in half of the test cases even performed the same or better than an unmodified network. Thus, this indicates that the network is capable of solving cases in wide range of applications where the spatial coverage is of interest.

Acknowledgement: This work has been supported by the **Inspectrone** (Autonomous and high-level commanded system for remote inspection of marine vessels to support classification and commercial operations) project, under contract number 8090-00080B.

Received: *1 March 2021* Accepted: *24 March 2021*
doi: 10.1049/ell2.12183

References

1 Bampis, L., Gasteratos, A., Boukas, E.: Cnn-based novelty detection for terrestrial and extra-terrestrial autonomous exploration. IET Cyber-Systems and Robotics (2021). https://doi.org/10.1049/csy2.12013
2 Boukas, E., Gasteratos, A.: Modeling regions of interest on orbital and rover imagery for planetary exploration missions. *Cybernetics and Systems* **47**(3), 180–205 (2016)
3 Ren, S., et al.: Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(6), 91–99 (2015)
4 Liu, L.,,et al.: CNN-based automatic coating inspection system. *Advances in Science, Technology and Engineering Systems* **3**(6), 469–478 (2018)
5 Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556, (2014)
6 Andersen, R., et al.: Investigating deep learning architectures towards autonomous inspection for marine classification. In: 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 197–204.IEEE, Piscataway, NJ (2020)
7 Liu, X., et al.: Oil Palm Tree Detection and Counting in Aerial Images Based on Faster R-CNN. In: *Lecture Notes in Electrical Engineering vol. 632*, pp. 475–482.Springer, Singapore (2020)
8 Wang, Y.-R., Li, X.-M.: Arctic sea ice cover data from spaceborne SAR by deep learning. *Earth System Science Data Discussions*, 1–30 (2020). https://doi.org/10.5194/essd-2020-332
9 He, K., et al.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, IEEE, Piscataway, NJ (2016)
10 Lin, T.-Y., et al.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 936–944, IEEE, Piscataway, NJ (2017)
11 Demir, I., et al.: Deepglobe 2018: A challenge to parse the earth through satellite images. In: The IEEE Conference on Computer Vision and Pattern Recognition, pp. 172–17209, IEEE, Piscataway, NJ (2018)
12 Lin, T.-Y., et al.: Microsoft COCO: Common objects in context. In: European Conference on Computer Vision, pp. 740–755.Springer, Berlin (2014)