Identifying Urban Canopy Coverage from Satellite Imagery Using Convolutional Neural Networks*

Niamh Donnelly[1] Conor Nugent[2] and Brian Mac Namee[1]

[1] School of Computer Science, University College Dublin, Ireland
[2] Breadboard Labs, Ireland
niamh.donnelly1@ucdconnect.ie

Abstract. The availability of high resolution satellite imagery offers a compelling opportunity for the utilisation of state-of-the-art deep learning techniques in the applications of remote sensing. This research investigates the application of different Convolution Neural Network (CNN) architectures for pixel-level segmentation of canopy coverage in urban areas. The performance of two established *patch-based* CNN architectures (LeNet and a pre-trained VGG16) and two *encoder-decoder* architectures (a simple 4-layer convolutional encoder-decoder and Unet) was compared using two datasets (a large set of images of the German town of Vaihingen and smaller set of the US city of Denver). Results show that the patch-based methods outperform the encoder-decoder methods. It is also shown that pre-training is only effective with the smaller dataset.

Keywords: Convolutional Neural Network · Remote Sensing · Deep Learning · Canopy Coverage · Google Earth Engine.

1 Introduction

Accurate estimation of urban tree canopy coverage is vital to the task of monitoring environmental resources (i.e. soil and air quality, wildlife habitats, levels of CO2 emissions) and for civic planning [17]. Typically, measuring canopy coverage requires human surveyors to manually annotate sample areas of an urban region. The level of coverage in the sample areas is then extrapolated to provide an estimate of the total tree coverage for the full region. This approach, however, is slow, resource intensive, and especially prone to problems of consistency. The recent emergence of affordable, broad coverage remote sensing through improvements in satellite technology and public availability of high resolution aerial imagery has made automated solutions to estimating canopy coverage feasible. In particular, machine learning approaches based on convolution neural networks (CNNs) that have been used in remote sensing tasks offer significant potential.

* This work was partially supported under grant 12/RC/2289 from Science Foundation Ireland. The authors thank Paul Hickey from Breadboard Labs, Tanushree Biswas from The Nature Conservancy, and all data labellers.

This paper describes an experimental study in which the effectiveness of different CNN-based architectures to tree canopy coverage identification are evaluated. The evaluation uses both *patch-based* CNN architectures and *encoder-decoder* CNN architectures, and compares their performance on two datasets. The remainder of this paper proceeds as follows. Section 2 surveys the state of the art in using machine learning approaches on remote sensing data. Section 3 describes the evaluation experiment performed including the architectures used, testing metrics, datasets, and experimental design. The results from the evaluation are presented and discussed in Section 4. Finally, conclusions and directions for future work are presented in Section 6.

2 Related Work

Canopy cover is the percentage cover of tree canopy in a given area, including only trees and shrubs and ignoring any other forms of vegetation [11]. Prior to the accessibility of remote sensing data, measuring canopy was a manual process in which human surveyors identified and measured it from the ground. Manual surveys were typically carried out in small sample areas of an urban region, with the level of canopy coverage measured in the sample area extrapolated to approximate the coverage in an entire region. More recently, surveyors have manually marked areas of canopy coverage in satellite images [1]. Again this manual labelling is usually performed on a sample area with the coverage of the overall region approximated based on this.

Machine learning offers a way to perform the identification and measurement of urban canopy coverage automatically. Not only would this eliminate significant manual effort, but it would also remove the need to extrapolate overall approximations from small sample areas. Stojanova et al. [24] benchmarked different classical machine learning techniques for identifying canopy coverage from satellite imagery and found that random forests models performed well.

CNNs have recently led to step change performance improvements in many image processing tasks, including tasks based on satellite imagery. For example, it is now common to apply CNN based approaches to land use classification. For example, Basu et al [3] describe an approach in which small patches of satellite imagery (sized 28x28 pixels) are classified into six land use classes. In many cases pixel-level land use classification is required, rather than classification of larger patches (see for example Fig. 4). Pixel-level land use segmentation can be achieved using CNNs that classify each pixel in an image individually [16], or using encoder-decoder network architectures that generate a segmentation based on an input image [25]. In the Deep Globe land use segmentation benchmark competition [6] variations of the UNet [21] and SegNet [2] encoder-decoder network architectures lead.

There is not extensive research in the literature on canopy coverage identification and measurement using deep learning techniques. Guirado et al. [9] describe a comparative study between the performance of classical machine learning techniques (e.g. SVM, KNN) and a CNN classification approach to determine the

presence of tree shrubs in extracted satellite images (sized 28x28 pixels). Similar patch based approaches are described in [5,19]. Pixel-level segmentation of satellite images into canopy and non-canopy regions, however, remains largely unexplored.

In patch-based approaches to pixel level segmentation a model is trained to label the class of the centre pixel in a small image patch taking into account its surrounding context. This procedure can be efficiently implemented using feed-forward CNN architectures that take a small image patch as input and output the class of the pixel at the centre of the patch. Popular CNN architectures for this type of task include LeNet [15], VGG16 [23], and ResNet [10].

Encoder-decoder techniques produce a complete segmentation of an input image rather than a single class label. The pooling layers of the encoder gradually down-samples the input, before the decoder uses upsample layers to gradually increase the convolutional layer size back to the scale of the original image. Training encoder-decoders requires that the ground truth of a dataset consists of fully segmented image masks. This requires a more involved labelling approach than patch-based techniques. Patch-based methods involve labeling a single pixel and extracting the surrounding patch whereas the encoder-decoder technique requires detailed segmentation of the entire patch by detailing the borders of all objects in an image. Popular encoder-decoder architectures include UNet [21] and FastRCNN [7].

3 Experimental Methods

To explore the use of CNN architectures to automatically identify and measure urban canopy coverage we perform a series of benchmark evaluation experiments. These compare the performance of patch-based (LeNet, VGG16) and encoder-decoder (UNet, and a simple 4-layer network) architectures across two datasets. The remainder of this section describes the datasets, and outlines the experimental procedures for developing and testing the model architectures.[3]

3.1 Remote Sensing Datasets

This study makes use of a pre-existing dataset developed by the International Society for Photogrammetry and Remote Sensing (ISPRS) which contains images of the city of Vaihingen in Germany. In addition to this, a bespoke dataset was generated imagery of the city of Denver, Colorado in the US based on satellite imagery obtained from Google Earth.

3.1.1 The Vaihingen Dataset This dataset contains imagery of Vaihingen, a densely populated town in Germany. The dataset contains 33 tiles of varying sizes, however, most contain approximately 2000 x 2000 pixels at a resolution of

[3] Access to code to run all experiments is provided at `https://github.com/engineevecanopy_coverage_project`

(a) (b) (c) (d) (e)

Fig. 1: Sample of 33x33 pixel patches extracted from the Vaihingen dataset.

9 cm per pixel. For each tile area, a infrared (CIR) image is provided. The CIR images consist of three spectral bands, corresponding to near infrared, red and green color channels (IR-R-G). The IR-R-G channels of the CIR images result in a heightened red hue across images. For each image, a corresponding ground truth image is provided showing segmentation into six land use classes (impervious surfaces, building, low vegetation, tree, car, clutter/background). For the specific task of tree canopy coverage, the Vaihingen dataset was converted to binary format containing the original canopy label and converting all additional classes to non-canopy. Fig. 1 shows an example from the Vaihingen dataset.

The original Vaihingen image tiles were segmented into small 33x33 pixel image patches using a sliding window approach with a stride of 18 pixels. A selection of sample patches are shown in Fig. 1. 25 of the 33 Vaihingen tiles were randomly selected for use as a training dataset, resulting in 372,884 training patches. For the patch based approaches each patch had an associated canopy or non-canopy label based on its central pixel. For the encoder-decoder approaches each patch had a corresponding ground truth image in which pixels were segmented as canopy or non-canopy. The ratio of canopy to non-canopy pixels in the training dataset was approximately 7:2.

Two adjoining tiles from the Vaihingen dataset (approximately a 2,500x4,000 pixel area) were used as a test set. In order to obtain a prediction for every pixel in the test tiles, different extraction techniques were used for patch-based and encoder-decoder techniques. For patch-based approaches, a stride of 1 was used when extracting the 33x33 image patches, resulting in a patch for every pixel in the image—9,665,460 test patches. For the encoder-decoder architectures, the test tiles were segmented into 33x33 patches without any overlap, therefore a stride of 32 is used, resulting in 9,438 test images. These images, however, still included over nine million classification opportunities as each pixel was classified.

3.1.2 The Denver Dataset For this study a bespoke dataset was created using the Google Earth Engine (GEE) satellite imagery of the city of Denver, Colorado in the US. An tool was developed in GEE that allowed participants to view imagery of points within the city and apply one of three labels—*tree*, *non-tree*, or *unsure*—to them. A total of 11 people were employed to label approx 500 data points each (one participant labelled an additional 1000 points). After all participants completed the labeling process, a second script extracted

Fig. 2: Sample of images extracted from the Denver dataset.

Fig. 3: Synthetic images for the Denver dataset generated using SMOTE

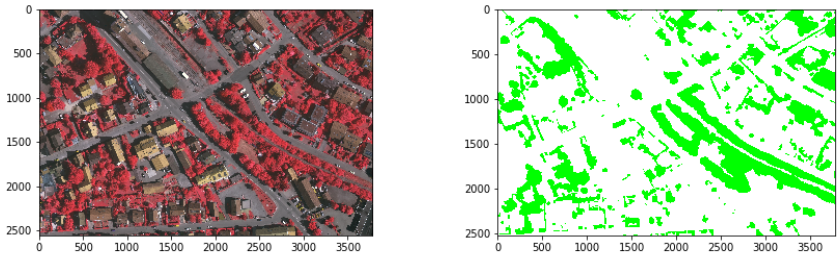
33x33 image patches surrounding each labeled point. A total of 6,895 labelled image patches were collected (patches labelled *unsure* were excluded). Unlike the Vaihingen dataset, the ground truth of the Denver consists of a single label for each patch rather than semantically segmented label encoded images, therefore the Denver dataset is unsuited for encoder-decoder techniques. A sample image patches from the Denver dataset are shown in Fig. 2.

The ratio of canopy to non-canopy pixels in the Denver dataset was approximately 7:1. To counteract effects caused by this imbalance Synthetic Minority Over-sampling Technique (SMOTE) [4] was applied. SMOTE generates synthetic examples of the under-represented class (in this case canopy), resulting in a balanced. Fig. 3 shows examples of synthetic patches generated using SMOTE. the data was randomly split into a training and test set using a ratio of 75:25.

3.2 Experimental Design

Each experiment followed a hold-out test set design (with test sets as described in Section 3.1). Cross-validation was not used due to the significant amount of computation required to train and test models (for example, training a single VGG-16 model on the Vaihingen dataset on a server contianing an Intel(R) Xeon(R) CPU E5-2695 v4 @ 2.10GHz with 72 CPU cores and approximately 500G of RAM took 18 hours). Similarly, due to the excessive computation required, grid searches were not used to identify hyper-parameter values, but rather recommendations from the literature were used.

As all experiments involve binary classification, so precision, recall and macro averaged F1-scores [12] are used to measure performance of all models. The remainder of this section describes the three experiments performed.



(a) Aerial image (b) Ground truth

Fig. 4: Aerial image and ground truth segmentation (green represents canopy) for one of the Vaihingen test area tiles.

3.2.1 Experiment 1 This experiment used the Vaihingen dataset to compare the performance of the LeNet and VGG16 patch-based CNN architectures and two encoder-decoder architectures: a simple convolutional encoder-decoder and the UNet architecture.

The LeNet model replicated the architecture described in [15] with the exception that the size of the input layer was reduced to 33x33 to match the input size of image patches. The LeNet model used binary cross entropy loss and the adam optimiser [14] with a learning rate of 0.000001 and a batch size of 100.

The VGG16 model used model weights that were pre-trained on the Imagenet dataset [22]. The model architecture followed that described in [23]. Due to a size restriction for the pre-built VGG16 Keras model, the training image patches were upsampled to a resolution of 48x48 pixels. The VGG16 model used binary cross entropy loss and the stochastic gradient descent optimiser [13] with a learning rate of 0.0001 and a batch size of 128.

For both of the encoder-decoder models image patches were resized to 32x32 as the up-sampling layers of the encoder-decoder architectures required even numbered input dimensions. The simple encoder-decoder model was implemented using the Keras package and had four convolutional layers. It used mean squared error loss and the Adadelta optimiser [26] using a batch size of 120. The architecture of the simple 4-layer encoder-decoder included two convolution layers for the encoder assembly and two for the decoder. All four convolution layers of the network comprised of 32 filters with a kernel size of 3x3. Convolutions used a stride of 1 with zero padding and a rectified linear unit activation function. The encoder assembly incorporated two max pooling layers (one after each convolution layer) with a kernel size of 2x2, while the decoder assembly included two upsampling layers (one after each convolution layer) similarly with a kernel size of 2x2. The UNet architecture was implemented using the Keras package and followed th specification described in [21]. It used mean squared error loss and the adam optimiser with a learning rate of 0.0002 and a batch size of 112.

3.2.2 Experiment 2 Experiments using the Denver dataset involved only patch-based CNNS—the VGG16 and LeNet architectures—as the ground truth required for encoder-decoder networks was not available. The model architectures used for this experiment were the same as those used in Experiment 1.

4 Results

This section presents results from the two experiments, for each of which performance metrics, confusion matrices and sample segmentations are shown.

In Experiment 1 two tiles that contained substantial dispersion of canopy and a variety of other features were used as the test area. Fig. 4 shows one tile, and its associated ground truth. The performance of each architecture is shown in Table 1, and associated confusion matrices are in Table 2. The areas of canopy identified by each approach are shown in Fig. 5, which can be compared to the ground truth image in Fig. 4.

Only the patch-based CNN models were used in Experiment 2 which used the Denver dataset. The performance of the each of these is shown in Table 3, with associated confusion matrices shown in Table 4. The pre-trained VGG16 architecture produced the highest scores across all metrics.

Table 1: Performance results on the Vaihingen test dataset .

Model	Precision	Recall	F1-Score
LeNet	0.9100	0.9017	0.9057
Pre-trained VGG16	0.8844	0.8989	0.8913
4-layer encoder-decoder	0.8716	0.6398	0.6631
UNet	0.8714	0.6007	0.6078

Table 2: Confusion Matrices for Vaihingen dataset (X-axis: Predicted, Y-axis: True)

	Non-canopy	Canopy
Non-canopy	6 689 190	295 981
Canopy	368 562	2 020 403

(a) LeNet

	Non-canopy	Canopy
Non-canopy	6 416 847	459 193
Canopy	319 203	2 039 213

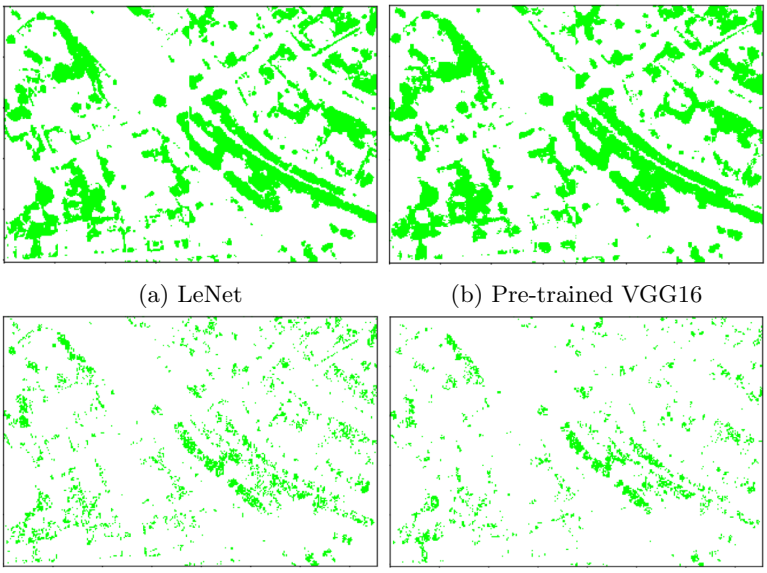
(b) Pre-trained VGG16

	Non-canopy	Canopy
Non-canopy	696 639	1 741 136
Canopy	43 743	7 064 210

(c) Simple encoder-decoder

	Non-canopy	Canopy
Non-canopy	420 858	1 939 099
Canopy	22 082	7 085 871

(d) UNet



(a) LeNet (b) Pre-trained VGG16

(c) 4-layer encoder-decoder (d) UNet

Fig. 5: Segmentation results of models applied to the test area of the Vaihingen dataset.

Table 3: Test results of patch-based models on the Denver dataset.

Model	Precision	Recall	F1-Score
LeNet	0.5823	0.5031	0.4763
Pre-trained VGG16	0.7840	0.7940	0.7889

Table 4: Confusion Matrices for Denver dataset (X-axis: Predicted, Y-axis: True)

	Non-canopy	Canopy
Non-canopy	1 535	6
Canopy	180	3

(a) LeNet

	Non-canopy	Canopy
Non-canopy	1 421	86
Canopy	77	140

(b) Pre-trained VGG16

5 Discussion

It is clear from Table 1 that for the Vaihingen dataset CNNs can accurately detect urban canopy coverage. The maximum F1-score 0f 0.91 achieved by the LeNet model on the Vaihingen dataset, exceeds that reported in the literature for classical machine learning methods applied to this task, which are typically in the range 0.85 to 0.9 [9]. Results demonstrate that CNNs can be used to accurately perform pixel-level segmentation of urban canopy coverage.

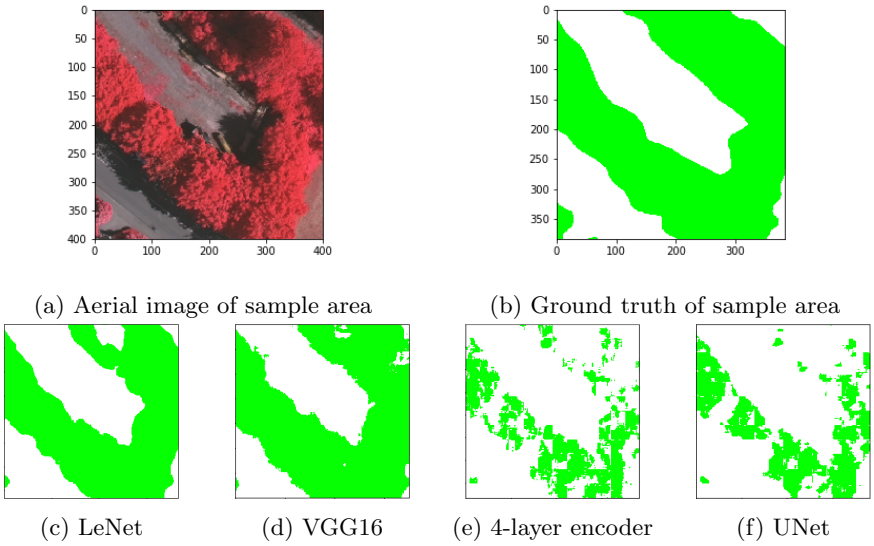
(a) (b) (c) (d) (e)

Fig. 6: Examples from Denver test dataset misclassified as non-canopy by LeNet and VGG16 models

Despite performing well on the Vaihingen dataset, the LeNet model did not perform well on the Denver dataset. This is most likely due to the much smaller size of the second dataset (just 9,088 training examples after SMOTE upsampling compared to 372,884 for the Vaihingen dataset). Improved performance on the Denver dataset was observed using the pre-trained VGG16 model, a common finding for smaller datasets [18]. It is interesting, however, that the pre-trained VGG16 network did not outperform the much simpler LeNet architecture on the Vaihingen dataset. Although the LeNet had a higher success rate in terms of performance metrics examining the reconstructions of the sample patch in Fig. 7 (described below), the VGG16 is the only model to identify an area of tree covered by shadow. This indicates it might be possible, with further network tuning and the application of some post-processing techniques for the VGG16 model to obtain better overall performance.

Development of the Denver dataset required human annotators to apply a label to a point from a Google Earth Engine electronic map. Although an option of 'Unsure' was provided to participants, it is possible some labels suffered from response bias, whereby ambiguous labels were recorded as either canopy or non-canopy. Fig. 6 shows a sample of some ambiguous points that were incorrectly classified as non-canopy by the VGG16 model although participants labeled them as canopy. This analysis revealed common sources of error to be points lying on the border of canopy and non-canopy (see Figs. 6a and 6d); within an area of shadow (see Figs. 6c and 6e); or over areas of other vegetation (see Fig. 6b) which were all incorrectly labelled as canopy. It is likely that further cleaning of this dataset to correct these types of errors could further improve performance.

Both patch-based approaches out-performed both of the encoder-decoder approaches. This was largely due to the poor recall of the encoder-decoder models (see Table 1). Further insights into the differences between these methods can be gained by examining Fig. 7 which show a small 400x400 pixel area of the test tile from Fig. 4 with associated segmentations from the four model types. While the outputs of the patch-based approaches closely mirror the ground truth the encoder-decoder models produce a more speckled, less defined representation. Further investigation of this disparity indicates that the reduced performance is a result of models miss-classifying areas of low vegetation for the canopy class, a common problem in tree identifying classifiers [20].



(a) Aerial image of sample area (b) Ground truth of sample area

(c) LeNet (d) VGG16 (e) 4-layer encoder (f) UNet

Fig. 7: Image and ground truth for a sample patch and the segmentations generated by different model architectures.

It is worth noting, however, that there are large differences in the computational cost when models using the different architectures are used to segment images. The encoder-decoder models predict the segmentation class of all pixels in a 32x32 patch in one pass through the network. So, to segment the 400x400 sample patch in Fig. 7 requires 144 passes through the network (allowing for padding). The patch based models, however, require one pass through the network to predict the segmentation class of every pixel. So, a total of 147,456 passes through the network are required to segment the same 400x400 pixel patch (again allowing for padding). This is an important difference would be applied across large areas and so significant reduction in computational effort to segment a region could be made by using an accurate encoder-decoder network.

Training encoder-decoder networks, however, requires significantly more labelling effort than training patch base models. A ground truth image that provides the segmentation class for every pixel is required. In contrast, patch-based methods simply require the label of the centre pixel to represent the class of a patch. The latter type of labelling is much faster (for the Denver dataset, participants produced approximately 500 labeled points per hour).

6 Conclusion

This research addresses the need for accurate classification of canopy coverage in urban areas, which has important applications in large-scale urban planing and environmental monitoring. The research pioneered the use of two distinct

CNN approaches: patch-based architectures and encoder-decoder architectures for pixel level segmentation. Experiments compared the performance of two variants of each approach on the problem of canopy coverage for a dataset containing aerial images of the German town of Vaihingen. It was found that the patch-based approaches outperformed the encoder-decoder approaches by a significant margin, with a LeNet model demonstrating the best performance. A follow on study using a smaller dataset collected using the Google Earth Engine platform exclusively for this study, focused on the town of Denver, Colorado. In this experiment (which used only patch based approaches) a pre-trained VGG16 model produced substantially higher performance levels than a fully trained LeNet architecture.

These findings provide strong evidence of the high performance potential of CNN architectures for canopy coverage identification and measurement, although they also suggest the need for further research. Due to time and computational constraints, it was not feasible to perform grid searches on any of the models. Model performance could potentially improve if optimal model parameters were identified through these searches. Another avenue for improvements could be obtained through post processing techniques. Dilation and erosion techniques [8] are commonly use to add definition to object boundaries by filling in holes/gaps in images and separating objects that overlap. These techniques often increase model accuracies of final predictions. Lastly, it is common known that performance of CNNs is sensitive to the size of the dataset. Potentially increasing the number of examples in the dataset could also add further improvement.

References

1. Measuring the tree canopy cover in london: An analysis using aerial imagery. Tech. Rep. MSU-CSE-06-2, London SE1 2AA (September 2015)
2. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(12), 2481–2495 (2017)
3. Basu, S., Ganguly, S., Mukhopadhyay, S., DiBiano, R., Karki, M., Nemani, R.: Deepsat: A learning framework for satellite imagery. In: Proc of SIGSPATIAL '15. pp. 37:1–37:10 (2015)
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
5. Chen, X., Xiang, S., Liu, C.L., Pan, C.H.: Vehicle detection in satellite images by hybrid deep convolutional neural networks. IEEE Geoscience and Remote Sensing Letters **11**(10), 1797–1801 (2014)
6. Demir, I., Koperski, K., Lindenbaum, D., Pang, G., Huang, J., Basu, S., Hughes, F., Tuia, D., Raskar, R.: Deepglobe 2018: A challenge to parse the earth through satellite images. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2018)
7. Girshick, R.: Fast r-cnn. In: Proc of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
8. Gonzalez, W., Woods, R.E.: Digital Image Processing. Prentice Hall (2004)

9. Guirado, E., Tabik, S., Alcaraz-Segura, D., Cabello, J., Herrera, F.: Deep-learning versus obia for scattered shrub detection with google earth imagery: Ziziphus lotus as case study. Remote Sensing **9**(12), 1220 (2017)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: In Proc CVPR 2016. pp. 770–778 (2016)
11. Jennings, S., Brown, N., Sheil, D.: Assessing forest canopies and understorey illumination: canopy closure, canopy cover and other measures. Forestry: An International Journal of Forest Research **72**(1), 59–74 (1999)
12. Kelleher, J.D., Mac Namee, B., D'Arcy, A.: Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies. The MIT Press (2015)
13. Kiefer, J., Wolfowitz, J., et al.: Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics **23**(3), 462–466 (1952)
14. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
15. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc of the IEEE **86**(11), 2278–2324 (1998)
16. Mnih, V., Hinton, G.E.: Learning to label aerial images from noisy data. In: Proc of ICML-12. pp. 567–574 (2012)
17. Moskal, L.M., Styers, D.M., Halabisky, M.: Monitoring urban tree cover using object-based image analysis and public domain remotely sensed data. Remote Sensing **3**(10), 2243–2262 (2011)
18. Nogueira, K., Penatti, O.A., dos Santos, J.A.: Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recognition **61**, 539 – 556 (2017)
19. Okafor, E., Pawara, P., Karaaba, F., Surinta, O., Codreanu, V., Schomaker, L., Wiering, M.: Comparative study between deep learning and bag of visual words for wild-animal recognition. In: IEEE SSCI 2016. pp. 1–8 (2016)
20. Paisitkriangkrai, S., Sherrah, J., Janney, P., Van-Den Hengel, A.: Effective semantic pixel labelling with convolutional networks and conditional random fields. In: Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on. pp. 36–43. IEEE (2015)
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. CoRR **abs/1505.04597** (2015)
22. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015)
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
24. Stojanova, D., Panov, P., Gjorgjioski, V., Kobler, A., Deroski, S.: Estimating vegetation height and canopy cover from remotely sensed data with machine learning. Ecological Informatics **5**(4), 256 – 266 (2010)
25. Volpi, M., Tuia, D.: Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. IEEE ,, **55**(2), 881–893 (2017)
26. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)