# Bruk av maskinlæring for å gjenkjenne bygninger i flybilder

Mathilde Ørstavik and Terje Midtbø

Vitenskapelig bedømt (refereed) artikkel

*Mathilde Ørstavik and Terje Midtbø: Detecting buildings in aerial images using deep learning methods*

KART OG PLAN, Vol. 78, pp. 221–230, POB 7030, NO-5020 Bergen, ISSN 0047-3278

This paper explores how machine learning can be used to recognize buildings in aerial images (orthophotos). The Norwegian map data: Joint Map Database (*Felles kartbase*) is used as «the true value» for training neural networks to detect buildings in aerial images. The paper introduces two models for machine learning. One basic model ( AirNet) and one model which include more layers in the network (AirNet-extended). The second model is supposed to detect more complex shapes in the images. Several parameters for training the neural network are tested. Complete building maps can still not be made using machine learning, but our results show that these methods can be used, for example, to study of city growth, find unregistered buildings etc.

*Keywords:* Deep learning, Machine learning, Image recognition, Neural Networks, Aerial images

*Mathilde Ørstavik*, Geomatics, Department of Civil and Environmental Engineering, Norwegian University of Science and Technology. E-mail: mathilde.orstavik@gmail.com

*Terje Midtbø*, Geomatics, Department of Civil and Environmental Engineering, Norwegian University of Science and Technology. E-mail: terjem@ntnu.no

## Introduction

When making new maps, and when updating existing maps, the detection of details in the map still requires manual input. This is a labour-intensive process, and over the years, researchers have explored how to automatize this process, or at least parts of the process. In the last decade, there has been a wide-ranging development of relevant technology for the constructions of maps. New and better satellites, laser measurements of earth surface and measurements by the use of drones are some examples.

Researchers have addressed the problem by use of different types of technology. However, most of this research is based on use of aerial images or remotely sensed data from satellites. Based on oblique aerial images, it is possible to generate 3D point clouds that can be used for further segmentation (Xiaofeng et al., 2016; Valinger, 2015). 3D models can also be generated by use of LIDAR, which in the next step may be combined with spectral information from aerial or satellite images (Matikainen et al., 2010; Hermosilla et al., 2011). Singhal and Radhika (2014) used colour information from aerial images for segmentation of roof tops, followed by edge detection, while Sirmacek and Unsalan (2008) combined the colours with shadow information. Other authors are analysing satellite images. Wei Liu and Prinet (2005) segmented high-resolution satellite images into regions, and from these, selected buildings by the use of a probability model, while Zakharov et al. (2015) used spectral clustering involving colours, shapes etc. Recently, there have also been several attempt on using machine learning for automatic detection of buildings (Cohen et al., 2016; Dornaika, et al., 2016; Zhang et al., 2017)

The popularity of machine learning methods has varied over the years since computers became capable to handle neural networks in the 1950's. The continuously grow-

ing capacity of new computers has given the attention for the topic a boost over the last years. In parallel with the increase in computer performance, new methods and techniques for neural networks and machine learning have emerged (Ørstavik and Midtbø, 2017). Visual recognition is one of the fast growing fields within artificial intelligence. It is also one of the fields that are most difficult to handle (Li et al., 2016).

Even if most of the visual recognition research is based on images in general, there are several examples on machine learning in classification of remotely sensed and aerial images (Castelluccio et al., 2015; Long et al., 2017; Marmanis et al., 2016). In Norway, the mapping authorities maintain nation-wide, large-scale map data and high-resolution areal images (orthophotos). These datasets may together form an excellent basis for training neural networks. One of the main objectives of this paper is to study how these Norwegian datasets can be used to meet the challenge of segmentation in aerial images.

In order to explore the use of the Norwegian data as basis for machine learning, a model named AirNet was implemented. The architecture for AirNet is based on SegNet (Badrinarayanan et al., 2015) , which is a relatively new network that is among the top performing networks for semantic segmentation. It is considered to be efficient, both when it comes to memory and to computational time during inference. Consequently, since it is one of the best networks for semantic segmentation, the assumption is that it will work well for building detection. AirNet was implemented in two versions: *AirNet-basic* and a modified version, *AirNet-extended*. The second one of these includes more layers, and can handle neural networks of more complex character. This is often termed as a "deeper model".

In deeper models, the convolution layers are learning a combination of all the previous activation maps. As the model gets deeper, the complexity of the detected shapes increases. This may improve the model's accuracy and its ability to detect complex shapes. On the other hand, when the number of layers increases, more parameters need to be calculated, and the network becomes harder to train. As stated by (He et al., 2015a), a deeper model is not always better.

The paper aims to answer these research questions:

1. Will our proposed model, AirNet, in a satisfactory way, be able to "learn" the difference between buildings and not-buildings by using the Norwegian aerial image database (orthophotos) together with Norwegian map data?
2. How will the proposed method work when introducing deeper architecture?
3. How will the introduction of IR-images influence on the results?

## Methods

AirNet has an encoder-decoder architecture (Long et al., 2015), with several convolutional layers in the encoder and decoder. As well as the convolutional layers, it has max-pooling layers in the encoder, with corresponding transposed convolution layers for up-sampling in the decoder. In front of the encoder, the model has a normalization layer. It normalizes each vector in the input image. This gives performance advantages. In addition, a *softmax classifier* is included after the decoder (*Figure 1*).

Layers two to nine in *Figure 1* is the encoder part of the AirNet-basic network. Convolution layers are added with a filter of 7*7*64. As explained in Badrinarayanan et al., (2015), this «allows for a wide context for smooth labelling». The convolution layers are followed by max-pooling layers which down-sample the images, reduce the number of parameters and thereby increase rotational and positional invariance.

Layers ten to seventeen are the decoder part of the network. As in the encoder, there are convolutional layers. Instead of the pooling layers that exist in the encoder, the images are up-sampled by adding transposed convolution layers. The convolution, that exist in both the encoder and decoder, performs a 2D convolution with a given kernel (filter) size.

An extended version of AirNet was implemented in order to test how a deeper version of the AirNet model would work. It has five

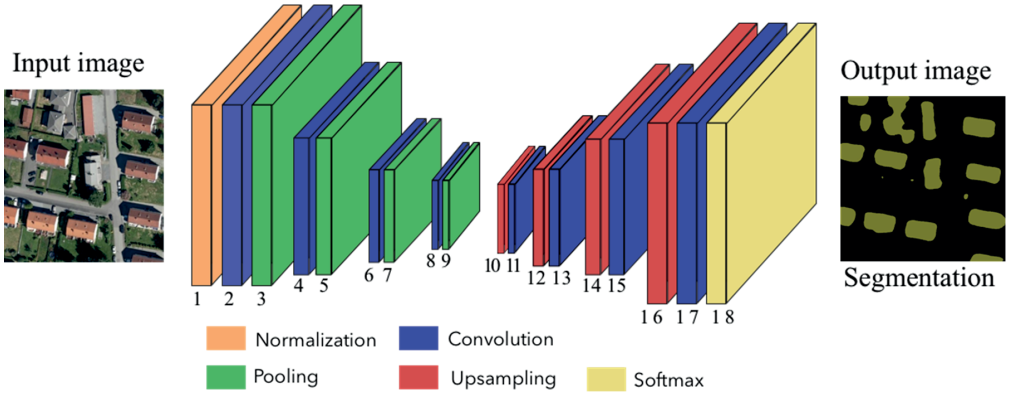


*Figure 1: Architecture for AirNet-basic.*

sets of encoders and decoders, and multiple convolution-layers in each of them (*Figure 2*).

The network is trained by reading the dataset batch by batch and running it through the network. For each batch, the accuracy is calculated, and the weights are updated and saved to *checkpoint files*. For every hundred iterations, a batch of data from the evaluation dataset is passed through the network. The calculated loss for the validation dataset helps to evaluate the training. If the loss starts to increase, the error for the evaluation dataset increases, which means that the model probably will overfit. The model is implemented to support both training from scratch, as well as fine-tuning from *checkpoint files* created in previous training.

After the training process is completed, the model can be run in test mode by passing the *checkpoint file*, which is saved during training. Each image is passed through the network, using the calculated weights (which were saved to the checkpoint files). It is then possible to calculate the accuracy of each image. The results from running the test dataset are used for evaluation of the model.

When working with deep learning the models are computationally expensive, and to be able to train them in a reasonable time you need adequate hardware. Since the nodes in a layer are calculated independently, they can also be calculated simultaneously. The architecture of the GPU makes it pos-

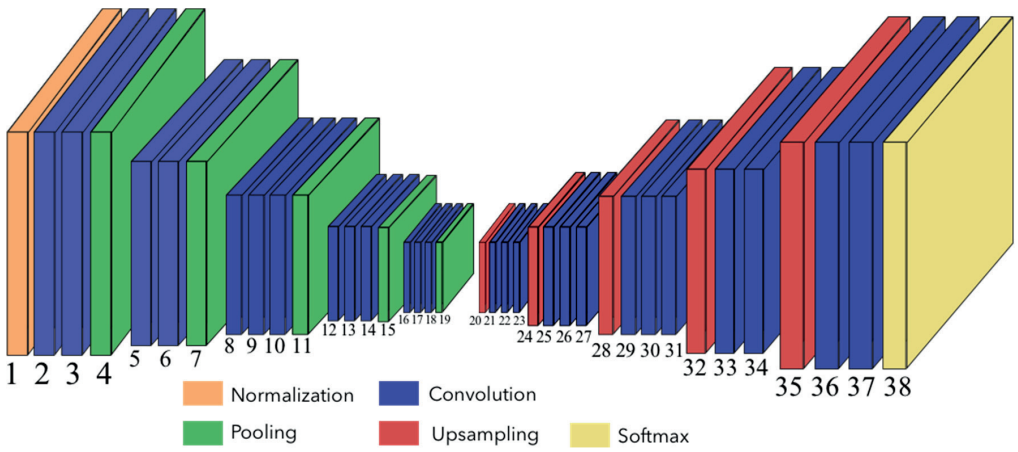


*Figure 2: Architecture of AirNet-extended*

sible to run these processes in parallel and perform mathematical operations that the CPU is not designed to handle. Consequently, the most important hardware requirement is the GPU, which should be at least 8GB RAM for training a network within reasonable time. *Table 1* shows the specification for the computer used in our research.

*Table 1: Computer specification*

| | |
|---|---|
| CPU | Intel®Core™i7-7700K CPU @ 4.20GHz x 8 |
| GPU | GeForce GTX 1080/PCIe/SSE2 (8GB RAM) |
| Operating system | Ubuntu 16.04 LTS |

**Training data**

The training dataset is a crucial component for neural networks. A segmentation dataset needs two sets of images: the input image and the label image. The network tries to segment the input image correctly, while the label image represents the desired segmentation output (*Figure 3*).

As input images, this project used aerial images from the Norwegian mapping authorities. When selecting images for the training data, three criteria were set:

1. Equal resolution in the whole area, ideally between 0.1 and 0.5 meter.
2. Available infrared counterpart of the images
3. "New" images, not more than six years

Table 2 shows the five areas that met these criteria.

*Table 2: The selected aerial images*

| Area | Year | Resolution | Size |
|---|---|---|---|
| Lørenskog | 2013 | 0.1 m | 70.5 $km^2$ |
| Skedsmo | 2013 | 0.1 m | 77.0 $km^2$ |
| Rælingen | 2013 | 0.1 m | 71.7 $km^2$ |
| Nittedal | 2013 | 0.1 m | 186.2 $km^2$ |
| Farsund | 2014 | 0.1 m | 267.6 $km^2$ |
| **Total size:** | | | **672.4 $km^2$** |

Earlier research indicate that using infrared images in deep learning can improve feature detection (Marmanis et al., 2016; Bollinger, 2017; Nogueira et al., 2017). Since the images in *Table 2* did have an infrared counterpart, it was possible to evaluate if infrared images improve the segmentation of the images.

Norway is also covered by large-scale primary map data, Felles KartBase (FKB). These maps have detailed building information, and footprints of the building roofs where used as label image.

For the evaluation of our deep learning methods, the dataset needed to be split into three parts, one for training, one for validation and one for testing. The training set (80%) was used to feed the network with examples to learn from, while the validation dataset (10%) was used during training to evaluate the process and to indicate when to terminate the training. Finally, the test dataset (10%) was used to evaluate the model after the training.

All together 672 $km^2$ of land were covered by the chosen images. However, Norway is a country of large wilderness areas, and when we introduced the requirement for at least one house on images used in the input dataset, only 12.2 $km^2$ fulfilled the criteria. This area was covered by 4645 images, each on 512x512 pixels.

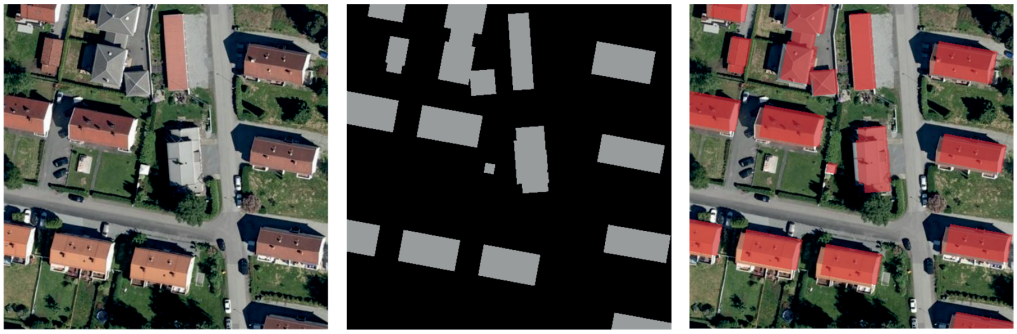

*Figure 3: Input image, label image and a combination of those two*

**Training parameters**
A central issue within deep leaning is to test the models in order to find the optimal training parameters. There are no definitive rules for when to use some of the parameters (e.g. optimizer and initializer). Consequently, various combinations have to be tested. In other words, trial and error is still a big part of implementing a network. This paper uses the mean intersection over union (mIoU) in to evaluate the different models. This measure is commonly used (Badrinarayanan et al., 2015; Long et al., 2015; Noh et al., 2015; Dai et al., 2016; Li et al., 2017) and is considered to be a better measurement than only looking for the ratio of correctly predicted pixels. The mIoU for each class is:

$$\frac{truePositive}{truePositive + falsePositive + falseNegative}$$

The models are tested thoroughly to find optimal training parameters. We tested the following training parameters for AirNet-basic and AirNet-extended:

**Batch size** (number of images in each batch) depends on the hardware configuration of the computer. Based on the complexity of the model, AirNet-basic could handle eight images, while AirNet-extended could handle four. The larger batch size performs better, as expected.
**Class weights** in this particular case depend on the ratio between "building" and "not building". If the buildings area is small, this should have stronger weights. These weights were first calculated based on the actual ratio between areas of "buildings" and "no buildings". However, since these weights are uncertain, we tested several weight combinations in order to find the optimal values (*Table 3*).

*Table 3: Weights between «buildings» and «no buildings»*

| | Weight for «no building» | Weight for «building» | mIoU |
|---|---|---|---|
| Based on ratio | 0.55 | 6.01 | 71.03 |
| Optimal weights | 0.8 | 1.1 | 78.93 |

**Initializers and optimizers** are important for the result when training the neural network. To find the optimal solution, all combinations of the parameters were tested. For AirNet we tested the optimizers *SGD* (Badrinarayanan et al., 2015; Long et al., 2015; Szegedy et al., 2015), *Adam* (Kingma & Ba, 2014) and *AdaGrad* (Mineault, 2014). *Xavier* (Glorot & Bengio, 2010) and *Variance scale* (He et al., 2015b) represented the initializers.

**Result**
*Figure 4* shows how the learning effect was influenced by the parameters during the first 5000 steps of training. As we can see from *Figure 4*, there are small differences in

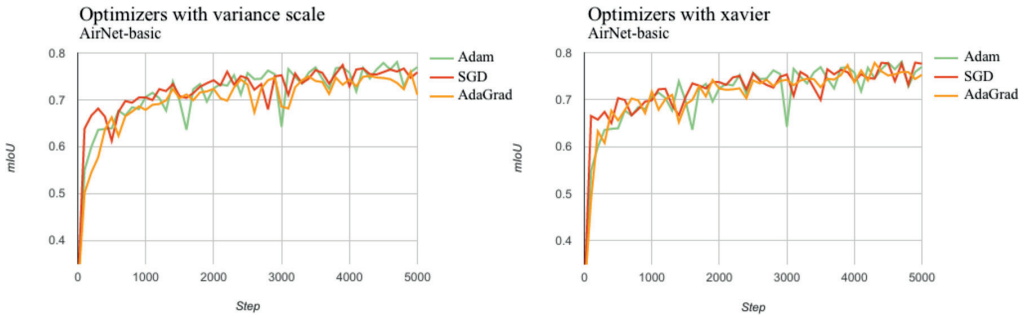


*Figure 4: Testing different optimizers and initializers on AirNet-basic*

the results from different parameters. However, the most stable training is achieved by using AdaGrad together with Xavier.

The different parameters caused a larger variation when testing the AirNet-extended model. This supports the claim that it can be more challenging to train deeper models. Adam performs worse than the other optimizers do, independent of the type of initializer. Variance scale initialization makes the training more stable than Xavier initialization. The best combination seems to be AdaGrad with variance scale initialization, as it leads to the fastest convergence and is the most stable (*Figure 5*).

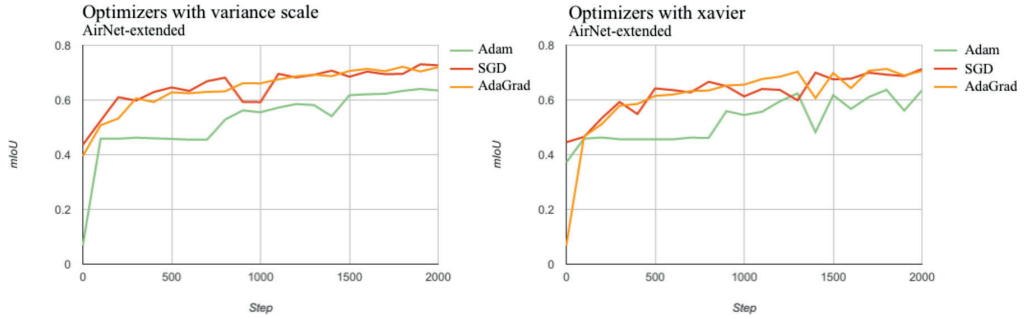


*Figure 5: Testing different initializers and optimizers for AirNet-extended*

Next, AirNet-basic and AirNet-extended were tested using the optimal initializers and optimizers. Each of the models were trained until they *overfitted*, and then tested with the weights calculated before the overfitting occurred. *Table 4* compares the results from the two models.

*Table 4: Comparison of performance and training parameters*

| **Parameter** | **AirNet-basic** | **AirNet-extended** |
|---|---|---|
| Optimizer | AdaGrad | AdaGrad |
| Initializer | Xavier | Variance scale |
| Batch size | 8 | 4 |
| Class weights | 0.8&1.1 | 0.8&1.1 |
| Iterations | 24000 | 26000 |
| Total training time | 11h 56m | 14h 47m |
| Number of inputs per sec | 5.6 | 2.5 |
| Inference time per input | 0.07s | 0.14s |
| **mIoU accuracy** | **81.05** | **80.32** |

AirNet-basic shows the best mIoU (81.05). When using the IR images in the same model, with the same parameters as in *Table 4*, there is a small improvement in performance (mIoU = **82.43)**.

A larger dataset is likely to give better results, and to compensate for our limited dataset we next tried to pre-train the model. Datasets for pre-training do not depend on the same strict requirements as the constructed dataset, and may consist of any type of images. However, the closer the images are to the dataset used for fine-tuning, the better. Consequently, a mixed aerial dataset was created. The dataset consisted of 8300 instances of RGB images in multiple resolutions. To fine-tune the network, the original IR dataset was used because it gave better performance than the RGB dataset. It is expected that comprehensive training improve deeper models. The deeper a model is, the more advanced shapes the filters in the convolutional layers can recognize. In addition, you avoid «wasting» valuable information in the dataset on recognizing basic shapes when the data is pre-trained (typically in the first layers of the network). The results in *Table 5* confirm this presumption.

*Table 5: Results for pre-trained models*

| | **AirNet-basic** | **AirNet-extended** |
|---|---|---|
| mIoU when fine-tuned with IR | 82.58 | **83.20** |
| Iterations fine-tuned | 21000 | 12000 |
| Training time for fine-tuning | 10h 42m | 6h 40m |

**Discussion**

The mIoU is a good measure for evaluation of a model and for the study on how parameters affect the results. However, it does not give any information on how the results actually look like, what type of buildings the model segments well, and in which areas it has trouble. *Figure 6* gives us a visual impression on how the pixel-based segmentation works.

In most of the areas in the test dataset, the segmentation fits the buildings well. Even though the edges of the buildings are usually not square, it still reflects the shape. It manages to segment buildings of different size and colour, and rarely misclassifies other areas as buildings. It manages to detect a large part of the pixels in almost all buildings. Often some of the pixels surrounding

*Figure 6: Examples on segmentation. Green pixels are classified as building by the computer.*

the buildings are included in the buildings class. This reduces the quality of the extracted shape.

The segmentation showed in *Figure 6* illustrates that deep learning segmentation is per-pixel-classification based on the surrounding pixels. The network classifies pixels as buildings. However, which building each pixel depicts is unknown. The training process for the neural network depends on datasets where there are accordance between the images and the labels (*Figure 3*). Unfortunately, there are some inconsistence between the aerial images and the FKB-data from the Norwegian Mapping Authorities. The dates of the two datasets are not identical. Consequently, some new buildings from FKB may be missing in the aerial image. This gives the neural network contradictory information and weakens the algorithm.

An orthographic image also contains geometric distortion. Rectification of the oblique aerial images may result in tiny dislocation of rooftops. Next, when placing the label above the image, some of these will appear shifted in relation to where the roof is visible in the image. Based on this, it makes sense that the edges of the buildings are the most difficult pixels to detect. To create an optimal dataset, the dataset for the training data could be adjusted manually. Some of the polygons in the labels might be adjusted to fit the buildings in the images in a better way, and the buildings not present in the images could be deleted.

Deep models have a large set of parameters, e.g. learning rate, regularization, optimizers, filter size and activation functions. Particular combinations of these parameters will work best for given problems, and to find the best combination researchers have to search through a vast space of possibilities. The testing of optimizers and initializers in this paper differ from previous research. Kingma and Ba (2014) claim that *Adam* optimizer works better than *SGD*. This was however never the case for the AirNet model. Further, He *et al.* (2015b) state that *Variance scale* initializer makes deeper models converge, while *Xavier* initialization cannot. Based on this one could expect that *Xavier* would not make AirNet-extended converge. However, our experiment showed the opposite. Still, this demonstrates the importance of testing each model. Testing combinations of parameters are time demanding, as it takes several hours, or even days, to train the network sufficiently.

The difference between using RGB images and IR images was somewhat disappointing. For future work, several spectral bands should be tested for possible increase in accuracy. The presence of IR images was the limiting factor when selecting test dataset. If we excluded this premise, the test dataset could have covered a larger area. Moreover, a larger dataset might have given a higher performance gain than the IR images gave.

The methods, as presented in this paper, are still immature when it comes to making precise maps of buildings. However, to get an overview over a given area the methods are applicable. Deep Learning methods could for example be employed in analysing the percentage of an area that is covered by buildings. This again can be used to evaluate city growth. The method can also help keeping maps (vector data) updated when we have new images over an area.

When organizations have to respond to disasters or political crises, a critical resource is up-to-date maps. Humanitarian OpenStreetMap is working on this challenge. Software, like AirNet, could help them in a rapid analyse of where there are buildings in the affected area - and thereby humans that might need help.

## Conclusions

Using Norwegian map data together with orthographic aerial images of Norway worked fairly well as training data. The basic model resulted in a mIoU accuracy of 81.05. Visual inspections did however show occurrences of small offsets between the buildings in the map data and the images. This reduces the computer's ability to learn how to detect edges of the buildings.

Out of our two models, AirNet-basic performs best when trained from scratch on the constructed RGB dataset. It gives the highest accuracy and it trains faster. However, which of the two AirNet models that should

be used for a particular problem, is a question that depends on several factors: The size of the dataset, the available time for training the network, if the model should be as fast as possible during inference, or if a higher accuracy is more important. If the accuracy of the results is the most important factor, a pretrained AirNet-extended model is recommended. If fast inference time is the most important factor, the AirNet-basic model is preferred.

The introduction of IR-images in the network did improve the results. The improvement could be seen mainly in occluded areas in the images. However, the improvement was less than expected. Since IR-images are rare, it might be worth considering larger data sets with RGB images only.

Using deep learning methods for building detection has still a considerable potential for improvement. However, by using methods as the proposed AirNet, it is possible to solve several general tasks. This includes tasks as the study of city growth, finding unregistered buildings and immediate disaster evaluation. This, together with the fact that the methods within neural networks and deep learning is constantly evolving, indicates a promising future for the technology.

## References


Badrinarayanan, V., Handa, A., & Cipolla, R. (2015). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Robust Semantic Pixel-Wise Labelling. *ArXiv:1505.07293 [Cs]*.

Bollinger, D. (2017). Open Source Machine Learning – Development Seed. Retrieved May 3, 2018, from https://developmentseed.org/blog/2017/01/30/machine-learning-learnings/

Castelluccio, M., Poggi, G., Sansone, C., & Verdoliva, L. (2015). Land Use Classification in Remote Sensing Images by Convolutional Neural Networks. Retrieved from http://arxiv.org/abs/1508.00092

Cohen, J. P., Ding, W., Kuhlman, C., Chen, A., & Di, L. (2016). Rapid building detection using machine learning. *Applied Intelligence*, *45*(2), 443–457. https://doi.org/10.1007/s10489-016-0762-6

Dai, J., He, K., & Sun, J. (2016). Instance-Aware Semantic Segmentation via Multi-task Network Cascades. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3150–3158). IEEE. https://doi.org/10.1109/CVPR.2016.343

Dornaika, F., Moujahid, A., El Merabet, Y., & Ruichek, Y. (2016). Building detection from orthophotos using a machine learning approach: An empirical study on image segmentation and descriptors. *Expert Systems with Applications*, *58*, 130–142. https://doi.org/10.1016/J.ESWA.2016.03.024

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *PMLR* (pp. 249–256).

He, K., Zhang, X., Ren, S., & Sun, J. (2015a). Deep Residual Learning for Image Recognition. *ArXiv:1512.03385 [Cs]*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015b). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1026–1034). Washington, DC, USA: IEEE Computer Society. https://doi.org/10.1109/ICCV.2015.123

Hermosilla, T., Ruiz, L. A., Recio, J. A., & Estornell, J. (2011). Evaluation of automatic building detection approaches combining high resolution images and LiDAR data. *Remote Sensing*, *3*(6), 1188–1210. https://doi.org/10.3390/rs3061188

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*.

Li, F.-F., Karpathy, A., & Johnson, J. (2016). *Lecture 7: Convolutional Neural Networks*.

Li, Y., Qi, H., Dai, J., Ji, X., & Wei, Y. (2017). Fully Convolutional Instance-Aware Semantic Segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4438–4446). IEEE. https://doi.org/10.1109/CVPR.2017.472

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3431–3440). https://doi.org/10.1109/CVPR.2015.7298965

Long, Y., Gong, Y., Xiao, Z., & Liu, Q. (2017). Accurate Object Localization in Remote Sensing Images Based on Convolutional Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing*, *PP*(99), 1–13. https://doi.org/10.1109/TGRS.2016.2645610

Marmanis, D., Wegner, J. D., Galliani, S., Schindler, K., Datcu, M., & Stilla, U. (2016). Semantic

segmentation of aerial images with an ensemble of cnns. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *3*, 473.

Matikainen, L., Hyyppä, J., Ahokas, E., Markelin, L., & Kaartinen, H. (2010). Automatic Detection of Buildings and Changes in Buildings for Updating of Maps. *Remote Sensing*, *2*(5), 1217–1248. https://doi.org/10.3390/rs2051217

Mineault, P. (2014). Adagrad – eliminating learning rates in stochastic gradient descent.

Nogueira, K., Penatti, O. A. B., & dos Santos, J. A. (2017). Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, *61*, 539–556. https://doi.org/10.1016/j.patcog.2016.07.001

Noh, H., Hong, S., & Han, B. (2015). Learning Deconvolution Network for Semantic Segmentation. *ArXiv:1505.04366 [Cs]*.

Singhal, S., & Radhika, S. (2014). Automatic Detection of Buildings from Aerial Images Using Color Invariant Features and Canny Edge Detection. *International Journal of Engineering Trends and Technology*, *11*(8), 393–396. Retrieved from http://www.ijettjournal.org

Sirmacek, B., & Unsalan, C. (2008). Building detection from aerial images using invariant color features and shadow information. In *2008 23rd International Symposium on Computer and Information Sciences* (pp. 1–5). IEEE. https://doi.org/10.1109/ISCIS.2008.4717854

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9). https://doi.org/10.1109/CVPR.2015.7298594

Valinger, J. (2015). *Automatic rooftop segment extraction using point clouds generated from aerial high resolution photography*. Umeå universitet.

Wei Liu, & Prinet, V. (2005). Building detection from high-resolution satellite image using probability model. In *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05.* (Vol. 6, pp. 3888–3891). IEEE. https://doi.org/10.1109/IGARSS.2005.1525759

Xiaofeng Sun, Shen, S., & Hu, Z. (2016). Automatic building extraction from oblique aerial images. In *2016 23rd International Conference on Pattern Recognition (ICPR)* (pp. 663–668). IEEE. https://doi.org/10.1109/ICPR.2016.7899710

Zakharov, A., Tuzhilkin, A., & Zhiznyakov, A. (2015). Automatic building detection from satellite images using spectral graph theory. In *2015 International Conference on Mechanical Engineering, Automation and Control Systems (MEACS)* (pp. 1–5). IEEE. https://doi.org/10.1109/MEACS.2015.7414937

Zhang, A., Liu, X., Gros, A., & Tiecke, T. (2017). Building Detection from Satellite Images on a Global Scale. In *30th Conference on Neural Information Processing Systems (NIPS 2016)*. Retrieved from http://arxiv.org/abs/1707.08952

Ørstavik, M., & Midtbø, T. (2017). A New Era for Feature Extraction in Remotely Sensed Images by The Use of Machine Learning, *77*(2), 93–107.