Optimization of Convolutional Neural Networks for Hedgerow Object Detection in Very High-Resolution Satellite Images

by

Steve Ahlswede

(1388540)

Environmental Science – Remote Sensing

University of Trier

Supervisor: Dr. Achim Roeder

Co-supervisor: Dr. Sarah Asam

Submitted in partial fulfilment of the requirements for the degree of Master of Science in Environmental Science

January 2020

Faculty of Regional- and Environmental Sciences

Universitaet Trier

Erklaerung zur Masterarbeit

Hiermit erklaere ich, dass ich die Masterarbeit selbststaendig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die aus fremden Quellen direkt oder indirekt uebernommen Gedanken als solche kenntlich gemacht habe.

Die Arbeit habe ich bisher keinem anderen Pruefungsamt in gleicher oder vergleichbarer Form vorgelegt. Sie wurde bisher nicht veroeffentlicht.

Datum	Unterschrift

Abstract

Hedgerows are one of the few remaining natural landscape features within agricultural areas. They provide important ecosystem services, living space for numerous species, and mitigate habitat fragmentation by acting as corridors between habitat patches. Environmental agencies in many European countries are thus interesting in monitoring and protecting hedgerows. In order to facilitate the success of such monitoring programs, cost-effective and accurate mapping of hedgerows is required. Here, the use of neural networks is examined in order to determine their feasibility for hedgerow mapping. Two state of the art neural networks were chosen (Mask R-CNN and DeepLab v3+). Both networks were able to successfully detect hedgerows, with DeepLab v3+ outperforming Mask R-CNN. Images from two different seasons (October and May) were tested as inputs. Finding suggest that using all available images, regardless of season, is preferred. Data augmentations were found to greatly increase performance for both networks, with DeepLab v3+ achieving up to 81% recall and 69% precision. However, caution should be taken when applying augmentations which modify spectral values of pixels. A pre-trained network was compared to a network with randomly initialized weights using DeepLab v3+. Here, the pre-trained network was found to produce superior recall and precision. However, the mask boundaries were far more precise using the randomly initialized network. The main limitation to successfully using randomly initialized weights was the amount of data available. Improvements can be made to Mask R-CNN by using rotatable anchors, as the network mainly struggled to detect diagonal hedgerows. Both neural networks were able to perform detections across a large spatial scale, and compared to previous object-based hedgerow mapping techniques, were relatively simple to use. Thus, this study supports the use of neural networks use in regional monitoring strategies for hedgerows.

Acknowledgements

Thank you to DLR for the use of their facilities, as well as for providing me with some financial support during my time spent there. This afforded me the ability to fully invest myself into my thesis while still having time for other aspects of life. Thank you to both Sarah Asam and Achim Roeder for your guidance throughout my time working on the thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial neural networks
ASPP	Atrous spatial pyramid pooling
CNN	Convolutional neural networks
COCO	Common objects in context
DEM	Digital elevation model
DSC	Depthwise separable convolutions
EM	Electromagnetic
ERF	Effective receptive field
FC	Fully connected
FCN	Fully convolutional network
FN	False Negative
FP	False positive
FPN	Feature pyramid network
GIS	Geographic information systems
GPU	Graphical processing units
LfU	Bayerisches Landesamt fuer Umwelt
MODIS	Moderate Resolution Imaging Spectroradiometer
MS	Multispectral
NIR	Near infrared
NMS	Non-maximum suppression
NN	Neural network
OBIA	Object based image analysis
R-CNN	Regional convolutional neural network
ReLU	Rectified linear unit
RF	Receptive field
RF	Random forest
RGB	Red, green, blue
RoI	Region of interest

RPN	Region proposal network
SVM	Support vector machine
TP	True positive

1 Introduction

Given the increasing trend in the human population, growing pressure is being put on natural resources. In particular, the demand for food, fiber, and energy products has led to extensive conversion of natural habitats to agricultural land, as well as an intensified use of existing agricultural land. Thus, agriculture has become one of the major drivers of land cover change and habitat loss globally (Meier et al., 2015; Tilman et al., 2001). The intensification of agriculture commonly results in the homogenization of agricultural fields which, while beneficial from a production standpoint, has led to the loss of biodiversity and ecosystem services in agricultural areas (Ekroos, Heliölä, & Kuussaari, 2010; Power, 2010; Tilman et al., 2001).

Productivity of agricultural lands depends on services typically provided by natural ecosystems, such as pest control, pollination, water management, as well as maintenance of soil structure and fertility. Many agricultural areas today lack the ecological infrastructure required to support such ecosystem services and thus supplement them with fertilizers, pesticides, irrigation, tillage, or importation of pollinators (Richards, 2001). However, such practices have shown to be damaging to the environment, leading to contamination of water supplies, erosion, compaction and salinization of soil, losses of biodiversity, and introduction of invasive species and foreign pathogens (Smith-Ramirez et al., 2018; Solgi, Shiekhzadeh, & Solgi, 2018; Dudley et al., 2017; Graystock et al., 2013). Sustainable management practices, such as organic farming, can reduce some of the negative effects of intensive agricultural land use (Kovacs-Hostyanszki et al., 2017; Schmitz et al., 2017; Hattab-Hambli, Motelica-Heino, & Mench, 2016), but the most relevant factor in improving the biological delivery of ecosystem services is an increase in landscape heterogeneity (Redlich et al., 2018; Power, 2010). Hedgerows have been found as key agricultural landscape features which contribute directly and indirectly to ecosystem services within agricultural systems (Holden et al., 2019; Padoa-Schioppa et al., 2005), arousing an increased concern regarding their protection and re-establishment within agricultural landscapes (Schmitz et al., 2017).

Despite the positive benefits gained from hedgerows, their abundances have declined in Europe (Holden et al., 2019; Lacoeuilhe et al., 2018; Staley et al., 2012). The main causes have been the intensification of agriculture resulting in the removal of hedgerows in order to increase growing space, and the degradation of hedgerows. Hedgerow degradation typically involves the loss of vegetation at the base caused by close proximity ploughing, application of pesticides, and inappropriate pruning schedules (Lacoeuilhe et al., 2018; Dover & Sparks, 2000). In several European countries, Agri-Environment Schemes have been introduced as financial incentives to encourage sustainable farming practices such as hedgerow maintenance (Froidevaux et al., 2019; Staley et al., 2012). Cyprus, Czech Republic, France, Germany, Luxembourg, and the U.K. have explicitly included hedgerow protection into their Good Agricultural and Environmental Conditions, requiring buffer strips established between agricultural lands to stop the drift of pesticide sprays (Lofti et al., 2010; Tansey et al., 2009). Additionally, hedgerows play an important role in the EU 2020 Biodiversity Strategy as well as in the Natura 2000 program, both of which aim to promote habitat connectivity across landscapes in an attempt to mitigate the effects of climate change on species (Lacoeuilhe et al., 2018).

Due to the benefits of hedgerows coupled with the protective measures introduced in many countries, accurate and precise data regarding their location and status appears valuable. Control authorities would benefit from an efficient and robust monitoring approach to help enforce protective legislations and to determine whether current management strategies are effective (Aksoy, Akcay, & Wassenaar, 2010; Tansey et al., 2009). The role of hedgerows in hydrological cycles has also led to some researchers arguing the importance of the inclusion of hedgerow data within hydrological models (Ghazavi et al., 2008; Viaud et al., 2005). In addition, hedgerows act as indicators for many important farmland species, and thus hedgerow maps can be included in species abundance and habitat suitability models (Padoa-Schioppa et al., 2005; Parish et al., 1994; Osborne, 1984).

In the German state of Bavaria, hedgerows have historically been mapped through either field surveys or manual digitization of aerial imagery (S. Boell, pers. comment). In situ mapping leads to very precise results as well as the retrieval of auxiliary characteristics of the hedgerows (e.g. species composition, hedgerow height), but is time consuming and costly (Bock et al., 2005). Aerial imagery has since been utilized to map hedgerows by manually digitizing hedgerow boundaries. However, this too becomes time consuming and labor intensive when performed over large regional areas, leading to less frequent monitoring intervals, reducing the ability to monitor trends (Vannier & Hubert-Moy, 2014; Askoy et al., 2010). Thus, for statewide monitoring of hedgerows, an automated and cost-effective approach would be preferred.

Reliable monitoring schemes for land cover variables require spatially continuous data obtained on a regular basis (Li et al., 2014). Long uninterrupted time series are required in order to understand the links between anthropogenic activities and ecological responses, measure the effectiveness of management strategies, and examine the causes and impacts of phenomenon such as climate change (Nagendra et al., 2012; Wulder et al., 2008). Remote sensing satellite data offers both temporal and spatial data continuity. For remote sensing to be used for monitoring of environmental variables, such as hedgerows, meaningful information must be extracted from remote sensing images through image classification. Here, numerous approaches are available, and the appropriate choice is crucial in determining the quality of the results (Noi & Kappas, 2017).

Image classification is a common technique used with satellite imagery to determine which portions of an image belong to which class(es). Here, two main approaches exist. The first approach uses individual pixels as the primary unit of image classification. Spectral signals across multiple electromagnetic (EM) frequencies are measured by the satellite sensor and recorded for each pixel. Each pixel is thus classified based on its spectral signature, as different land surface types have relatively unique spectral signatures (Jensen, 2006). However, when analyzing images with higher spatial resolutions, a single land cover class will likely have a large variation in spectral signatures across its pixels, leading to misclassifications (Li et al., 2016). The second approach tries to remedy this as it uses groups of pixels with homogenous properties which are spatially connected to form ‘objects’ which are used as the primary unit for image classification (Vannier & Hubert-Moy, 2008; Bock et al., 2005). Object based image analysis (OBIA) approaches thus allow for classification decisions to be made based on spectral features as well as object size, shape, and context, making it a more useful unit of classification in some cases (Blaschke, 2010).

Classification methods can be either parametric or non-parametric. Parametric methods are based upon the statistical assumption that data values (e.g. spectral values) follow a normal Gaussian distribution. Using supervised training methods, the distributions (mean and variance) of each target class are learned, and new data are plotted within feature space and compared to the learned class distributions in order to determine class memberships. Examples of such techniques include maximum likelihood and discriminant analysis. These techniques perform poorly when the data does not fit the assumptions, such as in complex landscapes where classes exhibit high variance and non-linear decision boundaries (Budreski et al., 2007; Hubert-Moy et al., 2001). In such cases, non-parametric methods are required. Here, there are no statistical assumptions with regards to data distribution, and class decision boundaries can be of any geometry. The boundaries are typically learned in an iterative fashion, being slowly adjusted until an optimal solution has been reached (Hubert-Moy, 2001). This learning of decision boundaries is a type of machine learning. Examples of such techniques are the k-Nearest Neighbor, Random Forest (RF) and Support Vector Machine (SVM) algorithms (Noi & Kappas, 2017). The choice of an appropriate classification method has a large affect on the image classification results.

To date, automated hedgerow mapping from aerial or satellite imagery has focused on RF or SVM methods, using objects as the unit of classification (O’Connell, Bradter, & Benton, 2015; Fauvel et al., 2012; Hellesen & Matikainen, 2013). Both methods are robust to overfitting and work well with limited training data (Belgiu & Dragut, 2016; Mountrakis, Im, & Ogole, 2010). However, the use of RF or SVM typically requires the manual design of large amounts of features used to train the classifier (Feng et al., 2015; Meneguzzo et al., 2015; Mishra & Crews, 2014). Such features have often been found to be over-specified, incomplete, and time consuming both in terms of design and validation (Wurm et al., 2019).

Neural networks (NN) are another type of non-parametric classification which are receiving increased amounts of attention in recent years (Voulodimos, Doulamis, Doulamis, & Protopapadakis, 2018). NN are particularly effective in classifying data with differing statistical distributions and complex decision boundaries (Atkinson & Tatnall, 1997). They are capable of learning complex class membership functions without the use of manually crafted feature layers, as NN decide themselves which features are the most relevant to a given classification task (Wurm et al., 2019). For the detection and localization of image objects, two NN approaches can be employed, namely, semantic segmentation and instance segmentation. In the former, all pixels within an image are assigned with a semantic class label (e.g. target class and background), while in the latter, only pixels which correspond to the target class receive a class label (e.g. no background label) and objects are individually segmented (e.g. individual prediction masks for each individual object) (Panboonyuen et al., 2017).

Convolutional Neural Networks (CNN) are a type of NN designed specifically for image classification tasks. While CNNs have been around since the 1980s (Fukushima, 1980; LeCun et al., 1989), they have gained much attention in recent years due to advances in publicly available datasets, graphical processing units (GPU), as well as improved NN architectures (Voulodimos et al., 2018). The structural breakthrough came when Krizhevsky et al. (2012) introduced novel concepts to the CNN, leading to an error rate of almost half of all other approaches in the 2010 Large Scale Visual Recognition Challenge (Liu et al., 2018). Since then, deep learning networks (networks with many layers (e.g. >8)) have experienced a renewed interest and have been applied to a wide range of computer vision tasks such as autonomous driving (Huval et al., 2015), microscope image analysis (Xing et al., 2018), motion tracking (Doulamis, 2018), and remote sensing (Ma et al., 2019). Satellite remote sensing is seen as an area of potential research and development for deep learning networks. This is due to the increasing availability of free high-resolution imagery (e.g. Sentinel-2) which is necessary to provide the level of detail required to detect meaningful spatial features (e.g. contours, edges). However, the majority of remote sensing research to date has focused on the development of deep learning algorithms using stable benchmark datasets; while very few studies have been performed using real data (Ma et al., 2019).

Currently, Mask R-CNN and DeepLab v3+ (henceforth referred to as DeepLab) are the state of the art for the detection and classification of objects within natural images (e.g. photographs taken with a handheld camera). However, these two networks are large, containing over 61 and 41 million trainable parameters for Mask R-CNN and DeepLab, respectively. Thus, these networks are prone to overfitting when training datasets are small, which can often be the case in remote sensing monitoring programs (Ma et al., 2019). Two main techniques are employed to overcome this barrier. The first is the use of pre-trained networks, whereby networks are first trained on a far larger secondary image dataset in order to train the network parameters, followed by a fine-tuning of the network whereby training continues with the target dataset images (Yosinski et al., 2014). This however leads to restrictions on the input data, as pre-trained networks are trained using only three bands as input, and subsequent fine-tuning must match this. Thus, such a strategy means that only three remote sensing bands can be utilized, while satellite sensors typically contain four or more bands. The second method is data augmentation. Here, the target images are modified (e.g. rotated, scaled) in order to increase the size of the dataset (Ma et al., 2019).

At the time of writing, neither of these two NN have been applied to the classification of hedgerows in satellite images, and no studies have applied these NN to vegetation classification using satellite imagery. It is thus currently unclear which of the networks performs better than the other, as this is often dependent on the application and dataset used (Lui et al., 2019; Zhao et al., 2018; Kussul et al., 2017).

Given the protected status of hedgerows in many European countries, improved detection methods would support monitoring efforts, which in turn provides information that aids adaptive landscape management. Hedgerow mapping has been widely studied in the past, with results achieving mixed rates of accuracy (O'Connell et al., 2015; Ducrot et al., 2012; Askoy et al., 2010; Bock, 2005). These works however have all utilized OBIA approaches, and no new work has been carried out with regards to the application of NN image processing methods for hedgerow mapping. As such, a knowledge gap, which this work aims to address, exists as to whether NNs can provide accurate and precise hedgerow detections. This is achieved by investigating the performances between the state of the art Mask R-CNN and DeepLab networks. Networks are trained using both pre-trained weights and randomly initialized weights (see section 6.3.5.2 for more detail). Band combinations are tested to determine the optimal setting for use in the pre-trained network. NN require large amounts of training data to avoid overfitting (Krizhevsky et al., 2012), and as such data augmentation strategies are also explored to determine which augmentations are suitable to the task of hedgerow detection. Optimal hyperparameter settings for detecting hedgerows via Mask R-CNN and DeepLab are also explored. Given that the available satellite scenes for the study area came from differing seasons, differences in results based on seasonal data inputs are explored. While no new methods were developed during this work, the deep learning methods applied here are novel within the domain of hedgerow mapping and are relatively unexplored within the sphere of vegetation mapping.

The hypotheses of this work are as follows. Given that networks employed in Mask R-CNN have been found to be sensitive to training sample size (Lui et al., 2018; Li et al., 2018) it is expected that DeepLab will produce superior results for lower amounts of training data. Furthermore, the inclusion of greater contextual features within the DeepLab network architecture (see section 6.2.1.1) further supports the hypothesis that DeepLab will produce superior results. Data augmentation is expected to increase performances for both networks due to the artificial increase in training data and increased model generalization. However, it is expected that the pre-trained model utilizing only three bands as input will outperform the model using randomly initialized weights with four bands as input as pre-trained models having been trained on far larger datasets.

2 Hedgerow ecology

Hedgerows are linear landscape features comprised of shrubs and trees which were traditionally established along the edges of agricultural fields as a delineation of properties, or as natural fencing systems for agricultural animals (Petit et al., 2003). They are cosmopolitan landscape features which can be found throughout Europe (Baudry, Bunce, and Burel, 2000). Although they account for only a small fraction of agricultural land use, they provide numerous benefits within agricultural landscapes pertaining to biodiversity, hydrology, as well as soil conditions (Baudry et al., 2000; Vannier & Hubert-Moy, 2014; Padoa-Schioppa et al., 2005).

From a biodiversity standpoint, hedgerows provide habitat and food resources for numerous mammal, avian, and invertebrate species (Lacoeuilhe et al., 2018; Staley et al., 2012; Dover and Sparks, 2000; Froidevaux, Broyles, and Jones, 2019). The presence of even a few hedgerows can have large impacts on landscape quality for birds within agricultural areas, as hedgerows are often the only remaining uncropped areas, providing grounds for breeding and shelter, as well as food resources (Padoa-Schioppa et al., 2005; Staley et al., 2012). Farmland birds remove the seeds of harmful weed species, as well as contribute to nutrient cycling (Redlich et al., 2018). Insect species such as butterflies and moths, which perform pollination, also rely on hedgerows as habitat and breeding grounds (Staley et al., 2012). Moths and butterflies constitute the main source of prey for numerous insectivorous birds and bats which in turn provide a biological form of control against agricultural pests (Froidevaux et al., 2019; Power, 2010). Hedgerows further promote biological pest control through higher abundances of arthropod species that prey on crop pests. As such, the promotion of hedgerows reduces the need for external inputs such as pesticides (Froidevaux et al., 2019; Morandin, Long, & Kremen, 2014), the increased use of which has been hypothesized as being responsible for the alarming decline in insect populations within Germany (Hallman et al., 2017), as well as causing adverse effects on human health (Tilman et al., 2001). Hedgerows can reduce the effects of habitat fragmentation between un-cropped land patches, as they act as corridors between the natural habitats which are a part of agricultural mosaics. An increased opportunity for species movement facilitates the flow of genes, leading to greater genetic variation which strengthens a population's resilience to harmful pathogens and the ability to adapt to changes in environmental conditions (Powell, 2018). Furthermore, improved species movement increases resilience to climate change, as altered climates within fragmented habitats may no longer meet the niche requirements of certain species (Davies & Pullin, 2007; Tilman et al., 2001).

Agriculture has been found to account for large amounts of nitrate and phosphate in ground and surface waters due to the transportation (e.g. leaching) of fertilizers, leading to increased risks of eutrophication, which has harmful effects on aquatic species as well as human health if consumed (Firbank et al., 2013). Hedgerows limit agricultural pollution into nearby water bodies by acting as barriers, thereby reducing the loss of topsoil nutrients through the reduced volume and velocity of surface water runoff. Subsurface transportation of phosphorus and nitrogen is also controlled by hedgerows when placed perpendicular to the direction of subsurface flow within a field (Vannier & Hubert-Moy, 2014; Ducrot, Masse, & Ncibi, 2012; Van Vooren et al., 2017). Reductions in surface water velocity also allow more time for water to infiltrate into the soil, providing more water for crops (Alegre & Rao, 1996). Water infiltration and storage is further facilitated by lower degrees of surface soil compaction due to a lack of farm machinery traffic or livestock trampling, increased organic matter, and improved soil porosity as a result of hedgerow root systems (Holden et al., 2019; Oyedele, Awotoye, & Popoola, 2009). Soils nearby to hedges have been found to contain higher abundances of ecosystem engineers such as earthworms and mycorrhizal fungi which, in addition to reducing losses to leaching by fixing nutrients within the soil, generate and stabilize pore spaces within soils, which provide further benefits for water infiltration and reduced runoff (Holden et al., 2019). Hedgerows influence soil fertility by acting as carbon stores within agricultural fields, while hedgerow litter also provides a source of mulch through which soil nutrients can be replenished (Lacoste et al., 2014; Follain et al., 2007; Alegre & Rao, 1996). In a meta-analysis of 60 published studies, Vooren et al. (2017) suggested that soil organic carbon stocks were 22% higher directly underneath hedgerows compared to adjacent fields, and 6% higher in field parcels neighboring hedgerows. Additional soil protection comes from a reduction in wind erosion as hedgerows provide barriers, thus reducing wind velocity as well as intercepting transported soil sediments (Baudry et al., 2000).

3 Remote sensing for hedgerow detection

3.1 Vegetation remote sensing background

Remote sensing is a growing field which can be applied to many domains such as monitoring of change over time or mapping current situations. The principle behind remote sensing is the use of sensors to measure EM energy reflected from different surfaces on Earth. Given differences in surface properties, surfaces exhibit different spectral signatures (Jensen, 2006). Vegetation for example has a distinct pattern of reflectance, as blue and red wavelengths are strongly absorbed by chlorophyll, while green and near infrared wavelengths are strongly reflected. Of course, these properties depend on the health of the vegetation as well as the time of year (Jensen, 2006). However, the general spectral signature of vegetation can be seen in Fig. 1. This figure highlights one common difficulty in vegetation remote sensing; the separation of spectrally similar classes. In the section to come, possible solutions to this issue are presented within the context of hedgerow detection.

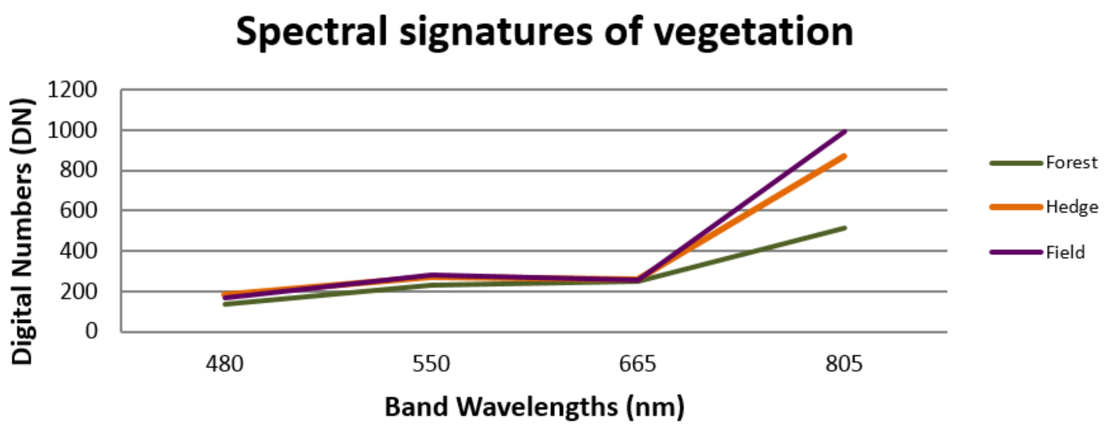


Figure 1 Shows the spectral signatures of three vegetation classes using the IKONOS bands.

3.2 State of the art

3.2.1 Object based image analysis

Land cover classification is one of the most common applications of satellite and aerial imagery. Pixels have been the primary unit of image classification in the past. Such methods have proven effective in the classification of images with coarse to moderate spatial resolutions which cover large spatial extents (e.g. Moderate Resolution Imaging Spectroradiometer (MODIS), Landsat) (Tehrany, Pradhan, & Jebur, 2013; Griffiths et al., 2013; Baker et al., 2012; Friedl et al., 2002). However, as satellite sensor spatial resolutions have increased, issues with pixel-based methods have arisen, as spectral variation between pixels which comprise landscape objects (e.g. crop field) lead to misclassification of pixels within an object, resulting in a ‘salt-and-pepper’ effect in the final classification (Li et al., 2016). Pixel-based approaches, which rely on spectral information alone, also have difficulties in statistical separation of spectrally similar classes (e.g. forests, fields, and hedgerows), typically resulting in increased rates of false positives (Fauvel et al., 2012; Meneguzzo, Liknes, & Nelson, 2013; Vannier & Hubert-Moy, 2008; Tansey et al., 2009; Bock et al., 2005). The separation of various vegetation classes can be solved through multi-temporal data analysis in cases where species exhibit differences in phenological patterns (Zhang et al., 2009). This often requires a sensor with high temporal resolution, especially in areas of persistent cloud cover, which typically implies a reduced spatial resolution (Achard et al., 2007). However, spatial resolution is an important factor in hedgerow detection, as studies have found that resolutions of 30m (e.g. Landsat) are not capable of accurate identification of hedgerows due to their thin widths, leading to pixels with mixed signals which can be difficult to disentangle (Vannier & Hubert-Moy, 2014; Vannier & Hubert-Moy, 2008; Tansey et al., 2009; Padoa-Schioppa et al., 2005). Research has found that to successfully discriminate hedges from other vegetation, spatial, textural, and contextual features offered by OBIA methods are required (Aksoy et al., 2010; Vannier & Hubert-Moy, 2008).

OBIA approaches have been the main focus of research regarding hedge detection (O’Connell et al., 2015; Vannier & Hubert-Moy, 2008; Ducrot et al., 2012; Askoy et al., 2010; Fauvel et al., 2012;

Tansey et al., 2009). Most OBIA approaches regarding hedgerow detection have found the incorporation of edge/line (e.g. Canny's method, Gabor filter) and textural features (e.g. grey level co-occurrence matrix) have improved results. Such features are commonly used within computer vision tasks to extract important object defining characteristics (e.g. object outlines) and typically work as a kernel of a given number of pixels (e.g. 3x3) which are able to extract information from the local region. The increased performance when incorporating such features thus underlines the need for additional features (aside from spectral features) for accurate hedgerow classification (O'Connell et al., 2015; Fauvel et al., 2012; Ducrot et al., 2012; Bock et al., 2005).

Reported results of object-based hedgerow detections range from 35 – 77% for user's accuracy, and 58 - 95% for producer's accuracy (O'Connell et al., 2015; Ducrot et al., 2012; Askoy et al., 2010; Bock, 2005). The 95% producer's accuracy reported by Ducrot et al. (2012) fails to report the total number of false positive misclassifications, but from visual examination of the final classification images and comments made by the authors in the paper, hedgerow detections were overestimated and imprecise, with a large number of forested areas also being classified as hedgerows. False positive overestimation is a common occurrence with regards to hedgerow detections. Fauvel et al. (2012) found that additional inclusion of orientation filters aided in discriminating hedgerows from other woody objects such as forest areas, as hedgerows tend to follow more distinct linear orientations. However, these orientation filters led to the false positive classification of non-woody vegetational linear features within fields (e.g. vegetated ditches) and had difficulties in detecting small hedges which lacked significant orientation.

Some authors have applied an unsupervised approach, using rule-based membership functions to classify segmented objects into various agricultural classes, including hedgerows (Tansey et al., 2009; Vannier & Hubert-Moy, 2008; Bock et al., 2005). Vannier & Hubert-Moy (2008) found that the accuracy of their unsupervised method varied based on hedgerow density within the scene, with higher densities of hedgerows resulting in higher accuracies. While unsupervised methods do not require any ground truth training data, the applicability of membership rules lack transferability across different areas where pixel and object values may differ (Aksoy et al., 2010; Tansey et al., 2009). Reduced accuracies of object-based models across study sites has also been found in supervised approaches, due to the heterogeneity of hedgerow compositions as well as differences in scale and textural characteristics (Vannier & Hubert-Moy, 2014; Ducrot et al., 2012; Aksoy et al., 2010). One further difficulty with regards to OBIA approaches is obtaining a satisfactory object segmentation, as finding segmentation parameters which avoid over or under segmentation of objects across larger spatial scales can be difficult (Sheeren et al., 2012; Taubenboeck et al., 2010).

An alternative to OBIA which has yet to be applied to hedgerow detection is the use of NN. Instead of manually determining which features are most representative of a target object class, NN automatically learn the most important features from the input training data (Zhu et al., 2017; LeCun et al., 1998). Features learned by NN increase in semantic representation as the network becomes deeper due to the combining of multiple low-level features (e.g. edges, corners) or incorporation of contextual features. Higher order semantic labels provide more accurate descriptions for objects within an image. For example, labelling pixels as a swimming pool instead of water, or a residential street rather than urban. High order semantic features extracted by CNNs have been found to be more robust to variances in object scale and position compared to manually engineered features (Lin et al., 2017). This is due to the use of pooling layers, which encode the most prominent features within a local spatial area. NN thus reduce the need for expert knowledge of feature engineering given that the network learns robust and information rich features automatically. This allows for a comparatively simpler extraction of descriptive land cover maps compared to past methods, thus aiding researchers and experts outside of remote sensing fields to derive accurate maps for their own purposes (e.g. hydrological modelling, habitat suitability modelling).

3.2.2 Artificial neural networks

Artificial neural networks (ANN) are analogous to the biological neural system in that they are comprised of nodes (often referred to as neurons) organized into interconnected layers (Voulodimos et al., 2018). Three layer types exist within a NN; the input layer (e.g. input image), output layer (e.g. prediction outputs), and hidden layers (any layers falling between the input and output layers). Neurons represent a simple mathematical model composed of three operations. First, each input (e.g. single image band pixel value) (k_i) is connected to each neuron (h_j) within the proceeding layer, with each connection having its own individual weight (w_{ij}) with which the input value is multiplied, and bias which is added (θ_j). The weighted inputs from all input pixel values across image bands are then summed together as they reach a neuron (Equation 1).

$$h_j = \sum_{i=1}^{n} w_{ij} k_i + \theta_j \tag{1}$$

The value of h_j is then fed through the neuron’s activation function, which will determine the output value of the neuron. Activation functions serve to transform the value of h_j to some range of values (e.g. logistic function). The output value of a single neuron is passed on as input to each of the neurons in the subsequent layer. This feedforward process is continued until the final output layer of the network is reached. Figure 2 shows a hypothetical example of an ANN. Here, the network is classifying the single input pixel into one of two possible output classes represented by each of the output layer neurons. The output neuron with the highest probability value will thus determine the final label for the input pixel.

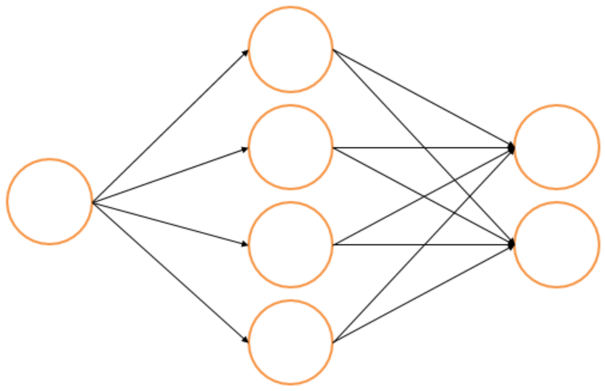


Figure 2 Diagram of a simple artificial neural network containing an input layer, hidden layer, and output layer. Neurons are represented by the circles, and connections are shown by the arrows. Each connection has its own trainable weight and bias parameters.

ANNs can be trained in a supervised manner in which case the network is given both the input as well as the desired output, thus making it possible for the network to assess its own accuracy and make adjustments to neuron weights in order to improve the predictions (O'Shea & Nash, 2015). Supervised learning occurs through the use of a loss function. Loss functions (e.g. mean squared error) measure the error between network predictions and the true label. They can be minimized by estimating the impact which variations in weight values have on the loss function. Loss function gradients with respect to each weight are computed via back propagation in order to determine which weights contributed the most to the error. These weights are thus adjusted in the direction of the steepest slope of descent along the multi-dimensional plane of the loss function until a minimum point is reached (Ruder, 2016). Although the achieved minimum points of the loss function are typically local minima, it has been shown that the majority of local minima are close to being globally optimal for the loss function of deep NN (Nguyen & Hein, 2017; Lu and Kawaguchi, 2017; Kawaguchi, 2016).

Back propagation is performed by calculating the partial derivatives of the cost function with respect to each neuron weight within the network. The error is propagated from the output layer back towards the input layer in a layer-by-layer approach (Li et al., 2012). Take for example a simple three-layer network comprised of an input (k), hidden (h), and output layer (z), using the mean squared error as the cost function. Here, the weight for the connection between neurons i and j of the input and hidden layers, respectively, is denoted as w_{ij}, while the weight for the connection between neurons j and l of the hidden and output layers, respectively, is denoted as v_{jl}. The value for a neuron within the hidden layer has been shown above in Eq. 1, while the value for a neuron in the output layer is calculated using Eq. 2 (Li et al., 2012).

$$z_l = \sum_{j=1}^{n} v_{lj} h_j - \theta_l \tag{2}$$

The cost function (E) is shown in Eq. 3, where t_l denotes the expected value at a given pixel and z_l denotes the predicted value of the network at that position. The change (gradient) in the loss function with respect to a change in the weight v_{jl} is given by Eq. 4. Because v_{jl} is located in the equation for z_l (Eq. 2), which is nested within the functions E (Eq. 3), the chain rule is used (Eq. 4) by multiplying the partial derivative of the function E with respect to z_l (Eq. 5) with the partial derivative of the function z_l with respect to v_{jl} (Eq. 6) (Li et al., 2012).

$$E = \frac{1}{n}\sum_{l=1}^{n}(t_l - z_l)^2 \tag{3}$$

$$\frac{\partial E}{\partial v_{jl}} = \frac{\partial E}{\partial z_l} \cdot \frac{\partial z_l}{\partial v_{jl}} \tag{4}$$

$$\frac{\partial E}{\partial z_l} = -(t_l - z_l) \tag{5}$$

$$\frac{\partial z_l}{\partial v_{jl}} = \max'(0, z_l) * h_j \tag{6}$$

In equation 6, max' is referring to the gradient of the activation function of the neuron z_l. In this case, the rectified linear unit (ReLU) is used. Different activation functions can be applied to NN and the popularity of certain activation functions has changed over time. The sigmoid activation function (also known as the logistic function) was used for many years, but due to increasing sizes of NN the ReLU has come into favor and is used in most state of the art NN (He et al., 2018; Chen et al., 2018; Liu et al., 2019). This is due to problems with the sigmoid function as the function scales input values along an S shaped curve, where gradients become flat as input values tend towards zero and one (Figure 3). The shape of the ReLU function is shown in Fig. 3. From this one can see that the gradient of the ReLU is either zero (when the input value of the function is ≤0) or one (when the input value of the function is >0) (Nwankpa et al., 2018). Since the gradient determines the change in the neuron weight value, weights remain relatively unchanged when along a flat gradient, resulting in the so-called vanishing gradient effect (Glorot & Bengio, 2010). This issue is less pronounced with ReLU activation functions, as input values rather remain unchanged unless they are negative, in which case they become zero (He et al., 2015). Gradients thus only become zero when neural outputs are negative. Given that this will lead to an average of 50% of the neurons being switched to zero, at which point no learning can occur, other researchers have used what are known as Leaky ReLU functions, where small gradients (e.g. 0.001) are assigned to values below zero, allowing for learning to continue across all neurons (He et al., 2015).

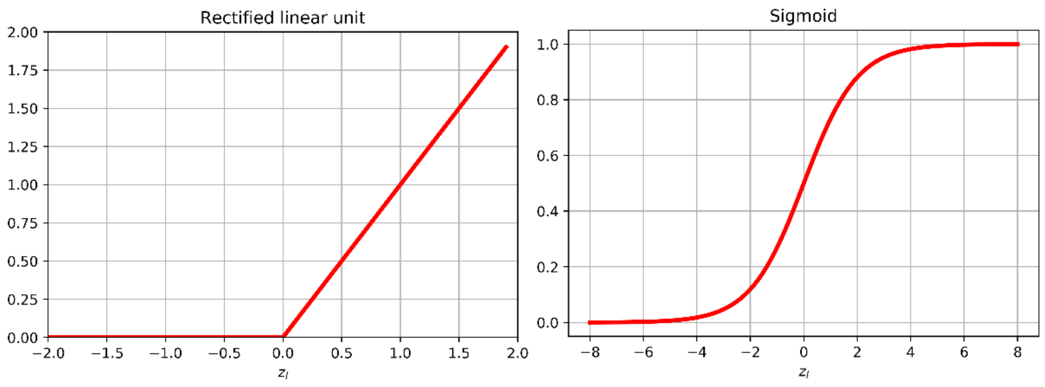


Figure 3 Graph of the rectified linear unit (ReLU) function as well as the sigmoid function.

After each iteration of back propagation, weights are updated in order to bring the predicted output closer to the target output. Updated weight values (u_k) are calculated using Eq. 7.

$$u_k = v_k - r * g_k \tag{7}$$

Here, v_k represents the current weight value, g_k represents the gradient of the current weight with respect to the network error (Eq. 4), and r represents the learning rate hyperparameter (Li et al., 2012). Learning rates thus control the degree of change to weight parameters between each training step. If the learning rate is set too high, weight changes will oscillate at a higher frequency, and the network will struggle to converge on an optimal setting. On the other hand, if the learning rate is set too low, the network may become trapped in a saddle point or sub-optimal local minimum point within the landscape of the cost function (Erb, 1993). Thus, the learning rate is an important hyperparameter to be tuned for NN.

3.2.3 Convolutional neural networks

ANN structures are effective when the input data is low in its dimensionality. However, since each input value is fully connected to each neuron in the subsequent layer, the number of weights required for highly dimensional inputs becomes impractical from a computational standpoint (O'Shea & Nash, 2015). CNNs are a subtype of ANN which provide a solution to this problem. CNNs work with the same feedforward structure and back propagation learning as ANNs. The main feature differentiating CNNs from other ANN structures is that rather than being fully connected to the input layer (e.g. all pixels of an image), neurons connect only to a small gridded region of the input layer (e.g. 3x3, 5x5) known as a convolutional filter (Figure 4). Thus, a neuron needs only to learn the weight parameters corresponding to the filter size. For example, given a 3x3 filter, each neuron learns nine weights. This design allows the network to learn fewer weights, making the processing of images with large numbers of pixels or channels computationally feasible (O'Shea & Nash, 2015).

Each neuron thus has its own gridded filter with a corresponding weight for each grid position, allowing for neurons to act as feature detectors (Figure 4). These filters thus operate in the same fashion as the previously mentioned Canny and Gabor filters traditionally used for OBIA. The important difference between NN filters and manually determined filters are that NNs learn the optimal values of filters based on the performance of the network, rather than the filter values being fixed (Wurm et al., 2019). Filters in the beginning of the network detect features such as color blobs and edges, and as the feature maps are passed deeper into the network, filters combine features from previous layers to form increasingly complex features (Yosinski et al., 2014). Convolutional filters sweep across an input layer, typically shifting by one or more pixels at a time. This movement across the input layer is known as the stride. At each given step, the filter overlaps a range of pixel values from the input layer. The scalar product between the filter weights and pixel values is added to the bias term and then passed through the activation function of the neuron, resulting in the output value for the pixel which lies at the center of the filter window (Figure 4). The output value is recorded to a pixelated feature map comprised of all the neuron output values derived from the movement of the filter across the input layer. Feature maps are essentially a matrix of numerical values, similar to the band of an image. A collection of feature maps from all neurons in a given layer of the network form a feature layer, which is passed on as an input for the subsequent neuron layer (Krenker, Bester, & Kos, 2011). For a feature map to maintain the dimensions of the original image, the filter must use a stride of one. Additionally, zero padding is required around the outside of an image when filters are larger than 1x1, so that pixels on the image edge can be placed at the center of the filter (O'Shea & Nash, 2015) (Figure 4).

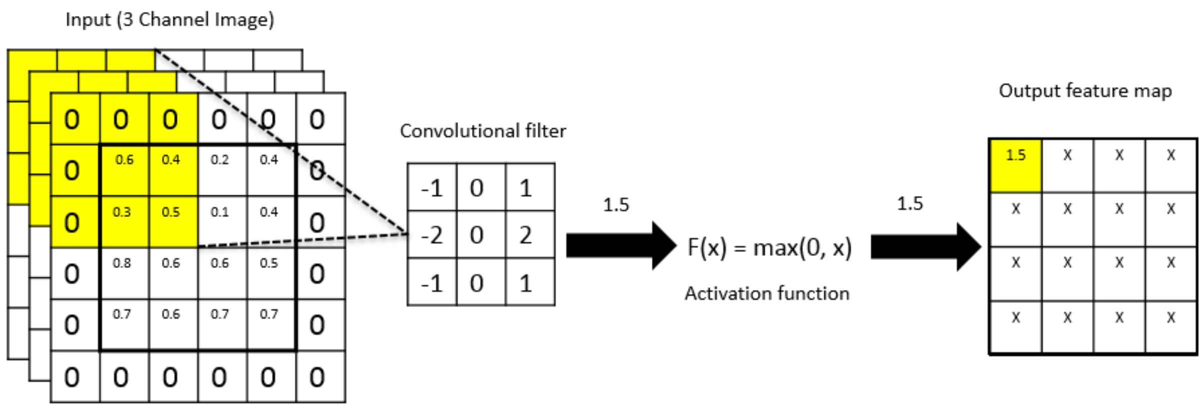


Figure 4 Example of the convolutional process for a single neuron. Sobel edge detection filter is as an example of a convolutional filter which a neuron could possibly learn. To simplify the example, the bias term is considered to be zero. For further information please refer to the above paragraph.

Aggregations of neurons which apply convolutional filters to a given input are known as convolutional layers and serve as one of the three main types of layers involved in the CNN architecture; the other two being pooling and fully connected (FC) layers. Pooling layers are typically 2x2 windows which apply an aggregation function (e.g. maximum function) such that the most prominent feature within a small spatial area (e.g. 2x2) is kept while the others are discarded, resulting in a reduction of the feature map size. The reduction in size makes pooling layers important for reducing computation time, as the number of feature maps within a feature layer tend to increase as an input is passed deeper into the network (e.g. a three-band input image can become 1082 feature maps) (LeCun, 1998). Furthermore, pooling layers provide the capacity to learn translation invariant features. Once a feature has been detected, its exact location is irrelevant as the approximate location in relation to other features is more meaningful for pattern detection. For example, given an image of a single number, knowing where endpoints and corners are located in relation to another, as well as the relative orientations of the lines, would suffice to predict which number is present (LeCun et al., 1998). Windows larger than 2x2 are not recommended due to the destructive nature of pooling layers. Stride values for pooling layers are typically set to two so that no overlap exists as the window slides across the input. This is done to avoid the possible duplication of features. Earlier models performed subsampling where pooling windows recorded the mean of the values rather than the maximum (e.g. LeCun et al., 1998). However, a paper by Scherer et al. (2010) showed the maximum function to be superior in comparison to mean pooling, and as such, the majority of NN today apply the maximum function for pooling layers (Baumhoer et al., 2019; He et al., 2018; Kussul et al., 2017; Girshick et al., 2015).

FC layers are typically the final layer and are connected just as ANN layers described in the previous sub-section. Here each pixel from the input feature layer is connected to each of the neurons of the FC layer. The FC layer provides the network output, integrating information across all locations of the preceding feature layer, and outputting image-wise classification probabilities based on the feature vectors (Hu et al., 2015). For example, if an image is classified as a cat, then all pixels within the image are labeled as belonging to the class cat. For a CNN to produce pixel-wise classifications for images, convolutions must be one dimensional with respect to the spatial coordinates of the image (1x1). Hu et al. (2015) used convolutional filters with 1x1x24 dimensions in order to classify each pixel within a hyperspectral image. Thus, instead of the filter moving across the *x*, *y* coordinates of the input layer, filters moved along the *z* axis across the spectral bands of a hyperspectral image. Given that CNNs typically perform image-wise classifications, they are used as feature extractors and combined with other NNs in order to perform pixel-wise classifications (He et al., 2018; Chen et al., 2018; Ren et al., 2015).

CNNs use filters which aggregate information across a given region (e.g. 3x3) of an input layer. The region which a neuron is directly connected to is known as the receptive field (Figure 5). As the network becomes deeper, larger areas of the original image have an influence with regards to the activation of later neurons. Thus, although neurons are directly connected to only the region equal to the size of the neuron’s filter, neurons are indirectly connected to all receptive fields of the neurons to which they are directly connected. This area is known as the effective receptive field (ERF) of a neuron (Figure 5) (Le and Borji, 2018). A larger ERF leads to inclusions of long-range contextual information in later feature layers, which is important for many computer vision tasks. As such, networks should be deep enough or use large enough convolutional filters in order to capture adequate spatial information (Wei et al., 2016).

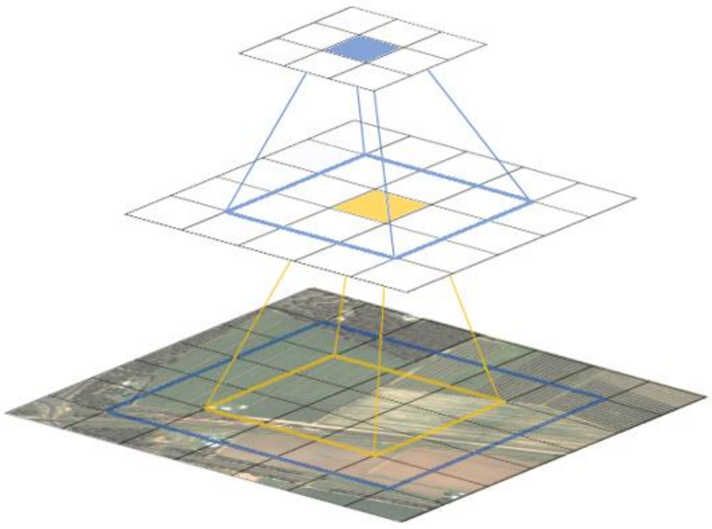


Figure 5 Shows the receptive field (RF) of two highlighted pixels (yellow and blue) as well as the effective receptive field (ERF) of the blue pixel. Pixel RFs are represented on the layers directly below the highlighted pixels connected with vertical lines. The ERF of the blue pixel is shown on the bottom-most layer.

3.2.4 Image analysis using convolutional neural networks

CNNs applied to satellite and aerial imagery have shown improved results over other popular machine learning approaches such as RF and SVM (Li et al., 2019; Liu et al., 2018; Kussul et al., 2017; Längkvist et al., 2016; Ishii at al., 2015). One of the main advantages of CNNs over other machine learning approaches is the ability of the network to learn meaningful features on its own. CNNs learn which features are most significant to the accuracy of a detection task through iterations of supervised learning called epochs. These learned features can be applied to numerous computer vision tasks, including the detection and localization of objects within an image (He et al., 2018). For the detection and localization of image objects, two deep learning approaches can be employed, namely, semantic segmentation and instance segmentation. In the former, all pixels within an image are assigned with a semantic class label, while in the latter, only pixels which correspond to the target class receive a class label, and objects are individually segmented (Panboonyuen et al., 2017).

3.2.4.1 Semantic segmentation

Fully convolutional networks (FCN) are one possible approach to semantic segmentation (Long, Shelhammer & Darrell, 2015). FCN utilize CNN structures to extract features. However, the final classification layer (FC layer) of the CNN is removed and instead a 1x1 convolution is performed, connecting *N* neurons to the final feature layers, where *N* is equal to the number of classes (including a background class). Each of these *N* neurons creates a feature map containing the probability for each pixel belonging to the class represented by the neuron. The class with the highest probability for each pixel across the multiple class neuron layers is then used for the output pixel label. As CNNs perform downsampling through pooling layers, the final output must be up sampled to regain the spatial dimensions of the original image. This is achieved through bi-linear interpolation, where the final layer is up sampled to the input image resolution, resulting in the segmented image (Long et al., 2015). As most deep CNNs down sample images up to 32 times (e.g. Long et al., 2015; He et al., 2018), the final CNN feature map must be aggressively up sampled, resulting in segmentations which often lack spatial detail (Chen et al., 2018; Long et al., 2015). In order to address this, Long et al. (2015) up sample an image in two steps, incorporating information from earlier CNN feature layers into the up sampled layers (known as residual learning). Thus, instead of performing a single up sampling of a magnitude of 32, the final CNN layer is up sampled in two steps of 16, with the result of the first up sampling being added with the CNN feature layer of matching resolution. This layer is then up sampled once more by a magnitude of 16, achieving a classification output at the resolution of the input image. A possible issue with FCN is that it fails to separate individual occurrences of the given object, but rather aggregates all occurrences together, making it potentially difficult to count the number of objects detected (Saindane et al., 2019). This can be overcome however with post processing using geographic information systems (GIS) programs (e.g. ArcMap).

3.2.4.2 Instance segmentation

The second approach to segmentation is instance segmentation. Here, sub-regions of the input image where the network believes a target object to be located are detected and classification is then performed exclusively in these regions. Such approaches are known as Regional CNNs (R-CNN). Early R-CNNs broke images into sub-regions (region proposals) and applied a CNN to each region in order to extract object features. Feature vectors are then passed on to a classifier branch providing probability estimates in order to determine whether the proposed region contains the target class (Girshick et al., 2015). Such R-CNNs separated the region proposal phase from the rest of the processing chain, meaning that a CNN was required for each individual region. This led to redundant calculations as features in overlapping region proposal areas would be recalculated for each individual region proposal. With Faster R-CNN, the region proposal phase was incorporated into the R-CNN processing chain and thus a singular CNN was applied to the entire input image and the generated features were then shared across all region proposal classifications (Ren et al., 2015). As Faster R-CNN only predicted bounding box locations of detected objects, the framework was further extended with Mask R-CNN, incorporating a masking branch which applies an FCN to generate pixel-wise object masks within positively classified regions to precisely localize objects (He et al., 2018).

Mask R-CNN has been found to be state of the art for the Common Objects in COntext (COCO) dataset (Lin et al., 2015); a public benchmark dataset designed specifically for instance segmentation (He et al., 2018). However, as Mask R-CNN performs instance segmentation (object-wise) rather than semantic segmentation (image-wise), it has been seldom applied to remote sensing imagery outside of urban studies targeting individual buildings (Mou & Zhu, 2018; Zhao et al., 2018; Ohleyer, 2018) or ships (Shao et al., 2018; Nie et al., 2018; Zhang et al., 2018). One issue with Mask R-CNN is that the use of an FCN for the binary mask branch has led to reduced precision in object boundary generation (Guo et al., 2018). This has been shown in a study looking at building segmentation, where resulting polygon shapes were irregular and thus required further processing (Zhao et al., 2018).

Recent FCN architecture designs have begun to address the issue of object boundary precision by removing or reducing the amount of down sampling layers by incorporating atrous convolution (Liu et al., 2019; Chen et al., 2018). Atrous convolution allows an expanded field of view from convolutional filters without adding any additional parameters. For example, in a 3x3 atrous convolutional window, nine pixels are sampled by the filter, however the filter values are spread apart, thus skipping pixels in the filter gaps. At an atrous rate of two, there would be a spacing of one pixel between each filter value (Figure 6). Atrous spatial pyramid pooling (ASPP) has extended the use of atrous convolution by applying varying rates of atrous convolution to a single feature map. Outputs from the different atrous convolutions are combined into a single output feature map, allowing for the capture of information at multiple spatial scales (Chen et al., 2018). DeepLab is an FCN architecture which incorporates ASPP layers to perform semantic segmentation which currently produces state of the art performance on the Pattern Analysis, Statistical Modelling and Computational Learning Visual Objects Class 2012 dataset (Chen et al., 2018b); a large public dataset used for benchmarking semantic segmentation algorithms. Liu et al. (2019) applied DeepLab in order to perform cloud segmentation within remote sensing images. The results showed that DeepLab was capable of improved results across all evaluation measures compared to the basic FCN model. The authors further tested their own network, dubbed CloudNet, which outperformed both the basic FCN and DeepLab networks due to more appropriate architectural decisions with regards to remote sensing imagery which removed all pooling and convolutional layers, instead using exclusively ASPP layers.

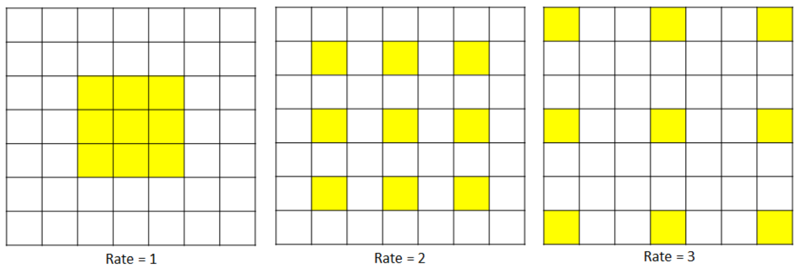


Figure 6 Shows the differences in convolutional filters based on different settings of the rate parameter. A rate of one (left) results in a standard convolutional filter. A rate of two (middle) or three (right) results in the spacing of filter values by one and two pixels, respectively (atrous convolutions).

3.2.5 Deep learning for remote sensing of vegetation

Currently, no deep learning networks have been applied for the purpose of hedgerow detection, although other deep learning studies involving vegetation have been carried out with success. Using Sentinel-2 scenes from northern India, Nijhawan et al. (2017) found that deeper CNNs, comprised of more layers, outperformed more shallow CNN structures when performing a general classification of vegetative versus non-vegetative classes.

Liu et al. (2018) compared the use of a FCN to a CNN known as ResNet (He et al., 2015) for mapping six distinct wetland vegetation land cover classes. The results found that model performances relied on the amount of training samples given to the network, with the FCN outperforming ResNet when less training samples were applied. This suggests that Mask R-CNN performance may also rely on large training samples as most implementations use ResNet as the CNN component. Li et al. (2018) found that for the detection of palm oil tree coordinates, a combination of two CNNs outperformed Faster R-CNN. The poor performance of Faster R-CNN was postulated as being due to either an insufficient number of samples, or the small size of palm tree objects. As Mask R-CNN is an extension of Faster R-CNN, this again brings its performance with small training sample sizes into question. However, neither study stated the use of auxiliary data (e.g. ImageNet) to pre-train their networks, although this approach that has been shown to aid in network training and produce improved results (Girshick et al., 2015).

A study by Zhao et al. (2018b) compared the performance of an FCN known as U-Net (Ronneberger, Fischer, & Brox, 2015) and Mask R-CNN by performing tree canopy segmentation. Their results showed that Mask R-CNN outperformed the FCN on all measures of precision. The downside they found was that Mask R-CNN required longer training time, although this was dealt with by using pre-trained weights from an auxiliary data source.

4 Study area

The study area was located in Freyung-Grafenau, an administrative district located within the Eastern region of the federal state of Bavaria, Germany (Figure 7). The district covers an area of 984 km^2, with a population of about 82,445, making it one of the most sparsely populated regions within Bavaria (Bernhards et al., 2003). This is partly due to a large portion of the district overlapping with the Bavarian Forest National Park which has been reserved for forest conservation following the ideal of "leaving nature alone" set out by the World Nature Protection Organization (Huber, 2005). The largest three cities within the region are Waldkirchen, Grafenau, and Freyung, with populations of 10534, 8256, and 7166, respectively (Bayerisches Landesamt fuer Statistik, 2018). The district is bordered by the Czech Republic to the northeast, and Austria in the southeast.

The region experiences an average rainfall of between 850 – 1,200 mm and average annual temperatures of 6.5°C, although this varies based upon altitude within the region. The region has an average growing period of 190 days per year. The soil is predominately characterized by brown ranker ranging from sandy to silty clay, which are prone to nutrient depletion as well as acidification. The parent material deeper within the soil profile is composed of granite and gneiss (Bernhards et al., 2003).

Agricultural land-use covers a total of 30,263 hectares (ha), comprising 30.8% of the land-use within the district. Of this, only 5,542 ha are used for agricultural plant growth, with the main crops being cereals such as spring barley, oats, as well as corn. The focus of agricultural land use is rather on animal husbandry of cow, cattle, and sheep (Bernhards et al., 2003).

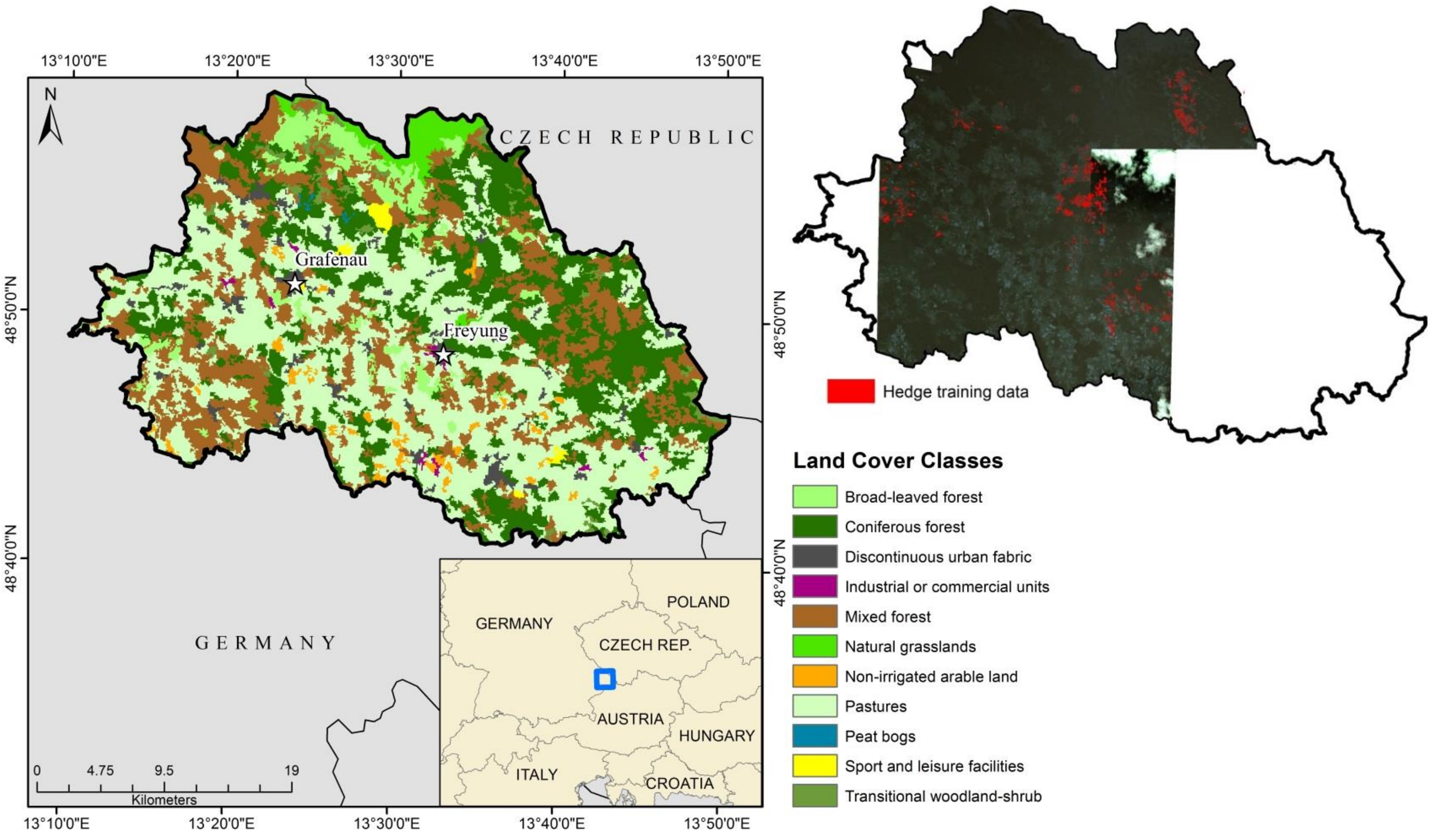


Figure 7 Shows the study area of Freyung-Grafenau as well as the CORINE land cover for the area. Top-right shows the coverage of available training data as well as the IKONOS imagery available to for the study.

5 Data

5.1 Satellite Data

Five scenes from the IKONOS sensor were utilized in order to train the two NNs. Two images were acquired on the 14th of May (2008), while the other three were acquired on the 8th, 10th, and 13th of October (2007). IKONOS acquires panchromatic images with roughly 1m resolution, as well as multispectral (MS) images at roughly 4m resolution. MS images consist of four bands; blue (445 – 516 nm), green (506 – 595 nm), red (632 – 698 nm), and near infrared (NIR) (757 – 853 nm) (Palandro et al., 2003). A pan-sharpening was performed using the Gram-Schmidt algorithm (Label &

Brower, 1998) in ArcMap (version 10.5.1, ESRI). Here, the high-resolution spatial information of the panchromatic band is merged with the lower resolution MS bands, in this case achieving a 1m resolution for all four spectral bands. The algorithm first simulates a low resolution panchromatic band using a linear combination of the MS bands (Eq. 8). In this equation, w_k represents the band-wise weight values. Weight values used were those provided by ArcMap based on the sensor (IKONOS). Thus, weights were 0.85, 0.65, 0.35, and 0.9 for red, green, blue, and NIR bands, respectively. Low resolution bands are then treated as vectors. Taking the simulated panchromatic band as the first vector, all bands are made orthogonal to one another using the Gram-Schmidt vector orthogonalization. Thus, the angle between the vectors of the red and panchromatic bands is calculated, and the red vector is rotated to make it orthogonal to the panchromatic. Then the angle of the green band is calculated and rotated to make the green vector orthogonal to both the red and panchromatic bands. This is repeated until all bands are orthogonal to another (Maurer, 2013). The mean and standard deviation of the original panchromatic band are then adjusted to match those of the simulated panchromatic band. The simulated panchromatic band is then swapped with the original panchromatic band. The inverse of the orthogonal transformation is then performed on the orthogonal MS bands, resulting in an enhanced spatial resolution MS image (Laben & Brower, 1998). Its advantages over Intensity, Hue, and Saturation (HIS) and Principle Component (PC) techniques are that the Gram-Schmidt method can process any number of bands at once, and enhanced preservation of spectral characteristics (Laben & Brower, 1998). A high resolution was required as research has shown that coarser resolutions lead to confusion in classifying thin hedgerows due to mixed pixels being dominated by neighboring land covers (Ducrot et al., 2012; Vannier & Hubert-Moy, 2014).

$$Pan_{sim} = \sum_{k=1}^{n} w_k MS_k \tag{8}$$

5.2 Data pre-processing for DeepLab v3+ and Mask R-CNN

5.2.1 IKONOS imagery

Original IKONOS scenes were clipped to subsets containing hedgerow polygons. These subsets were further split into image tiles of 320x320 pixels. Tiles which contained ground truth polygons (see section 5.2.2) were used as training and validation inputs for the two deep learning networks, while tiles without hedgerow ground truth polygons were discarded. The tiling of images is necessary as larger sized images exhaust GPU memory resources due to the large number of feature maps that are produced by the networks.

Given that spatial context is limited at the edges of images, hedgerows appearing at the edges of image tiles lacked the contextual information that would be available if the object was located within the center of an image tile. In order to reduce this effect, extracted image tiles had 64 pixels of overlap with neighboring image tiles (Liu et al., 2018; Marmanis et al., 2017). This further served to increase the size of the training dataset.

5.2.2 In situ data

Ground truth hedgerow polygons were obtained from the Bayerisches Landesamt fuer Umwelt (LfU). The dataset contained hedgerows which were manually digitized after determining hedgerow coordinates from field surveys. These surveys were carried out over the time period of 1984 – 2019. Given that some hedgerow polygons had been digitized more than 20 years before the IKONOS imagers were taken, some hedgerows had been lost or degraded. Thus, manual inspection was carried out to ensure that hedgerow polygons indeed outlined woody vegetation within the IKONOS scenes. LfU defined hedgerows as linear structures comprised of a mixture of bush and tree species, being a minimum of two years old. As such, not all linear woody vegetation structures in images were labelled as hedgerows.

Annotated images are required when training Mask R-CNN or DeepLab for the network to learn which pixels belong to the target class(es). Image annotations are simply a mask layer which defines the ground truth class label for each pixel in the satellite image. Thus, a binary mask was required for each 320x320 IKONOS image tile, where pixels with a value of 1 belonged to the hedgerow class, and pixels with a value of 0 belonged to the background class. Polygon shapes delineating hedgerows were thus converted to mask images using ENVI (version 5.1, Exelis Visual Information Systems). In the case of DeepLab, single annotation images containing multiple individual hedgerows could be used. However, as Mask R-CNN performs instance segmentation rather than semantic segmentation, annotation images could only delineate a single individual hedgerow within an image tile. Thus, in cases where multiple hedgerows appeared in a single image tile, separate annotation images were required for each hedgerow. As the production of individual annotation images for each hedgerow occurrence would have been time consuming and data intensive, annotations were generated on the fly within Python (version 3.7.1). A link to all Python code used in this paper is available in appendix II. This was accomplished by storing vertex information for each individual polygon within a virtual dictionary which linked individual hedgerows to their respective IKONOS image tiles. Thus, when an image tile was loaded, the file name of the tile was used to query the dictionary and the polygon vertices associated with the image tile were used to generate a temporary annotation image. Annotation images were created used the Python package Numpy (version 1.18.1). Additionally, bounding box information for each individual hedgerow was required for Mask R-CNN as it not only predicts pixel-wise object locations, but also object bounding boxes. Thus, ground truth bounding box coordinates were needed for network training. These were also stored within a dictionary and called upon in the same manner as hedgerow polygons.

Due to the tiling of images, small portions of hedgerows were often cut off along image edges, leading to a few stray hedgerow mask pixels present along tile edges. Small mask regions of less than 200 pixels were thus removed from mask images, as to provide only properly shaped hedgerows to the network for training (Zhao et al., 2018b). Given that some mask images only contained these small hedgerow occurrences, some masks no longer contained hedgerows afterwards. Mask images void of hedgerows were thus removed, leading to a dataset containing a total of 684 tile images. The dataset was then split into training and validation sets with a 75%/25% split, respectively, resulting in 513 training and 171 validation tiles (Figure 8).

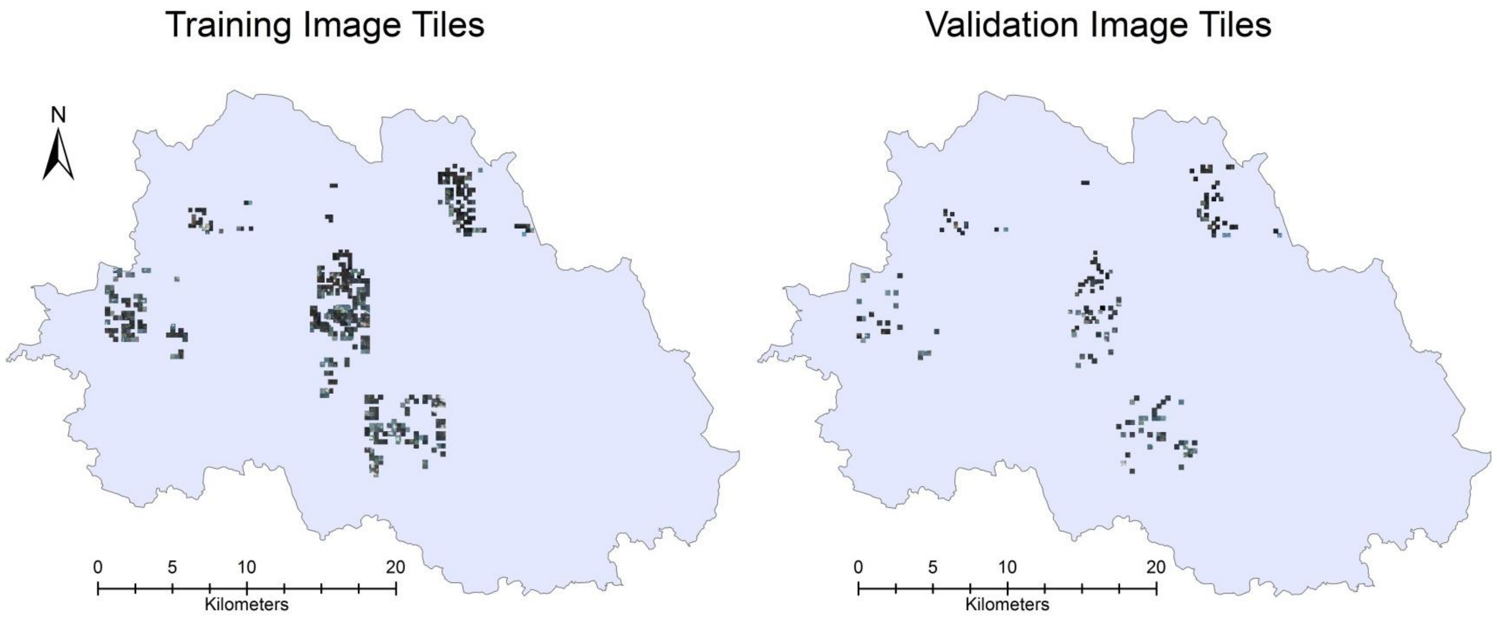


Figure 8 Shows the distribution of both training and validation image tiles across the study area.

6 Methods

The following chapter describes the two NN used in further detail with regards to network architectural designs, loss functions, and hyperparameter settings. After covering these, the chapter provides an outlined description of the experiments carried out using the two NN and describes the metrics used to measure performance in the experiments. The chapter concludes with a short section describing the Python packages and code used to carry out network training and data analysis.

6.1 Mask R-CNN

6.1.1 Network architecture

Mask R-CNN (Figure 9) is comprised of four modules. First is a feature pyramid network (FPN) which creates feature layers with strong semantic features at differing resolutions. Next comes a region proposal network (RPN) which proposes regions of interest (RoI) within an image. Afterwards, the RoIAlign layer standardizes all RoIs to the same dimensions. These are then passed to the final three-pronged module (known as the three ‘heads’ of the network), which classifies and localizes object bounding boxes within RoIs, as well as provides pixel-wise masks of the object. All four modules are supported by a CNN 'backbone', in this case ResNet-101 (He et al., 2015), used for the initial image feature extraction.

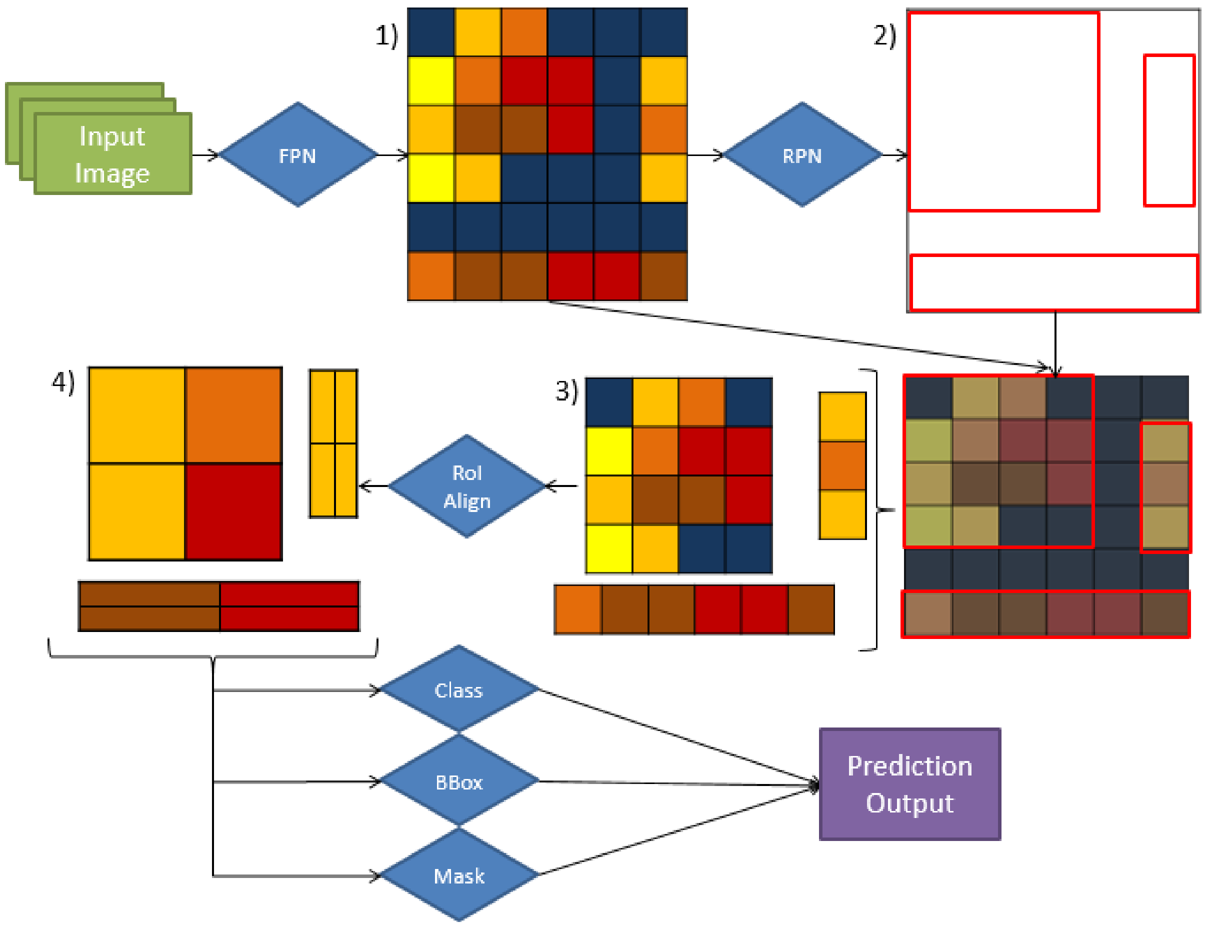


Figure 9 Mask R-CNN architecture. 1) Feature maps generated by the feature pyramid network. 2) Region proposals made by the region proposal network. 3) Region proposals combine with the feature maps to create regions of interest (RoI). 4) RoIs are pooled to equal dimensions (e.g. 2x2) for use in the final layers.

6.1.1.1 Feature extractor network

In Mask R-CNN, a CNN network named ResNet (standing for Residual Network) is used as the feature extracting network. It is referred to as a residual network as it performs additive merging of feature maps from earlier convolutional steps with feature maps generated from deeper in the network (Figure 10). The inclusion of residual information has been shown to be beneficial for the accuracy and training time in object detection tasks (He et al., 2015; Szegedy et al., 2016). The assumption is that, given an unknown function *H(x)* which provides the desired output (*y*) from the input (*x*) (e.g. a feature map), networks can more easily arrive at *H(x)* through the optimization of its residual function (*H(x)* – *x*). Thus, rather than training the network to approximate *H(x)* through some function *F(x)* (*F(x)* ≈ *H(x)*), we instead approximate *H(x)* by solving for *F(x)* ≈ *H(x)* - *x*. We can arrive at *H(x)* by rearranging this equation to *H(x)* ≈ *F(x)* + *x* (He et al., 2015). Figure 10 shows an example of what is known as a residual block, which consists of an input *x* (e.g. feature map) which undergoes three convolutional stages. The resulting feature map from the third convolution is then merged with the information from the input *x*, resulting in the block's output.

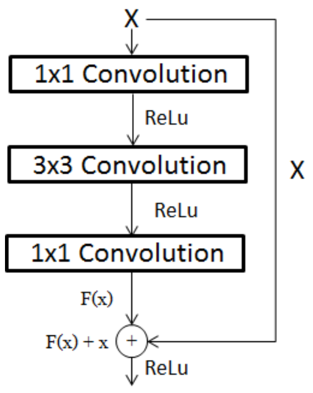


Figure 10 Adapted from He et al. (2015). Shows the residual learning block used by ResNet where *X* is the input layer which is processed through a series of convolutional layers. The input layer (*X*) is added to the output of the last convolutional layer within the block, resulting in the block's output layer.

The ResNet-101 architecture used here contains one 7x7 convolutional layer (non-residual), a max pooling layer, and 32 residual blocks (Figure 11). The original input image is down-sampled by a magnitude of two a total of six times throughout the network using convolutional layers with a stride of two and a single max-pooling layer. The output after each down-sampling step is re-used later within the FPN.

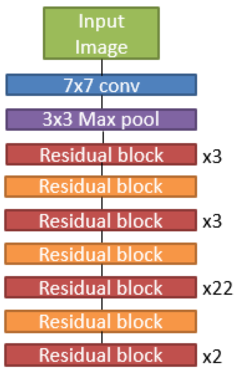


Figure 11 Architecture of the ResNet-101 network. Down-sampling is first performed using a stride of two in both the 7x7 convolution and max pooling layers. Red residual blocks use a stride of one while orange residual blocks use a stride of two in the first convolutional layer of the block, leading to additional downsampling.

6.1.1.2 Feature pyramid network

FPN is used to help detect objects occurring at different scales within an image by generating feature maps of varying spatial resolution and incorporating strong semantic features into each. The FPN thus makes use of feature maps of different resolutions from ResNet, creating a pyramid of feature layers (Figure 12).

The creation of the feature pyramid involves a bottom-up and top-down pathway. Four feature layers are taken from the ResNet backbone to form the bottom-up pathway of the pyramid. The feature layers used here are always the final feature layers before a downsampling step takes place. As the input passes through deeper convolutional layers within ResNet, the semantic features learned become increasingly useful for object detection (Lin et al., 2017). Later layers have undergone some form of downsampling, leading to increasingly lower resolutions, making localization of features less accurate. The strength of the FPN thus comes from the combination of strong semantic features of deeper feature layers with the high-resolution of shallow layers allowing for improved localization (Lin et al., 2017). This occurs within the top-down pathway of the FPN. Here, the lowest resolution feature layer of the bottom-up pathway is sequentially up-sampled by a factor of two (using nearest neighbor upsampling) in order to form a secondary set of feature layers. Lateral connections merge each up sampled feature layer with the feature layer of equal resolution from the bottom-up pathway. Finally, a 3x3 convolution is performed on each of the merged feature maps, resulting in the final output of the FPN which is passed to the RPN.

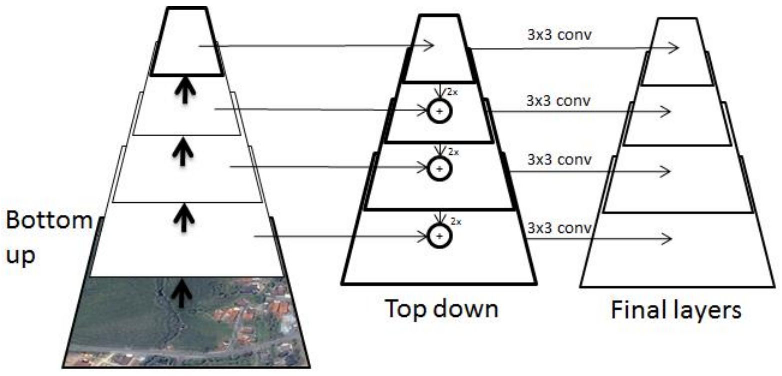


Figure 12 Shows the feature pyramid network (FPN). The bottom-up pathway is taken from the feature extractor network (ResNet). The top-down pathway is created by progressively upsampling by a rate of two. Lateral connections merge up-sampled layers with layers corresponding in size from the bottom-up pathway.

6.1.1.3 Region proposal network

RPN is a sub-network of Mask R-CNN used to propose candidate object bounding boxes. It does this by placing a set of bounding boxes (known as anchors) across each pixel of the input feature layer and performing a classification for both object and non-object classes for each anchor. Anchor sizes are defined by user selected scale and aspect ratios. Mask R-CNN allows for the setting of three aspect ratios and five scales for anchors. Anchor coordinates are later refined by a regression performed by the RPN. Here, the coordinates of the top-left corner of the anchor, as well as its height, and width values are refined in order to more accurately localize possible objects within the FPN feature layers.

The RPN performs both regression and classification tasks using the feature layers generated by the FPN as input. First, a 3x3 convolution is applied to each level of the FPN's output feature layers, resulting in a set of feature layers with resolutions equal to that of the FPN layers. The resulting feature layers undergo a 1x1 convolution, resulting in a feature layer containing 512 feature maps (*RPN conv*) for each FPN scale. *RPN conv* layers are passed on to the 'heads' of the RPN network which perform classification and regression for each anchor. Each pixel of a given *RPN conv* layer receives three anchors of differing aspect ratios. Anchor scales are distributed across the feature pyramid hierarchy, with larger scaled anchors assigned to an *RPN conv* derived from FPN layers at the top of the pyramid. This is due to differences in ERF of the FPN layers. Singular pixels in the upper FPN layers represent a larger field of view than pixels of lower layers. Thus, pixels with smaller ERF would not cover the area spanned by larger anchor scales (Figure 13).

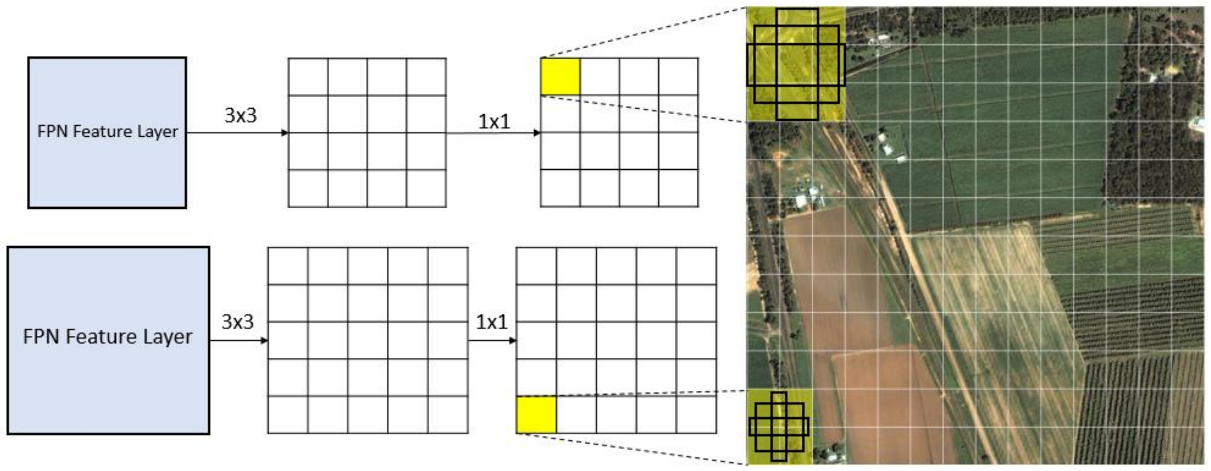


Figure 13 Simplified version of the region proposal network. Layers from the feature pyramid network (FPN) are convolved twice (3x3 and 1x1) to create a set of feature layers. Pixels of FPN feature layers of differing sizes will have different receptive fields with respect to the original image (yellow).

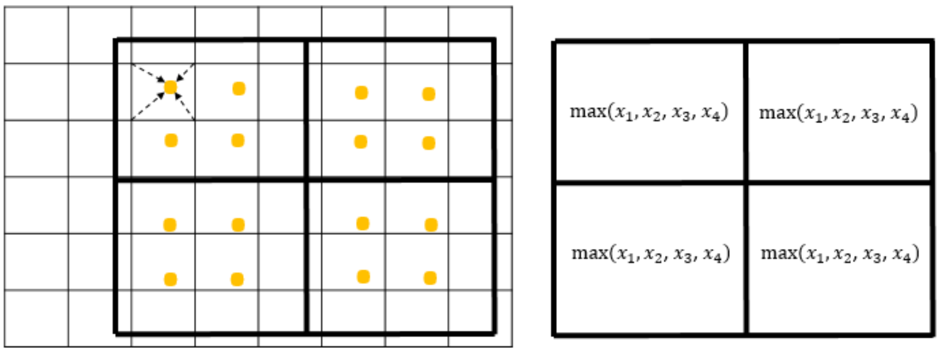
The classification head of the RPN estimates two probabilities (object and non-object) for each anchor, while the regression head estimates four anchor adjustment values for improved object localization. These values provide an adjustment to the x and y coordinates of the top left corner of the anchor, as well as a height and width adjustment for the anchor. The classification and regression heads thus give a total of 2*k* and 4*k* outputs, respectively, where *k* is equal to the total number of anchors generated by the RPN. Some anchors will have high degrees of overlap with another. To remove redundant anchors, a non-maximum suppression (NMS) is applied to anchors. Thus, given a set of anchors which overlap by 70% or more, only the anchor with the highest 'object' classification score will be kept. After NMS, the remaining anchors are again filtered based on classification scores so that only the anchors with the highest 'object' classification scores are kept. Thus, any anchors which did not overlap with other anchors, but had low object scores are removed. This final set of anchors is then passed on as the final RoI region proposals.

Threshold values for NMS and object scores were chosen based on visual inspection of resulting predictions. For NMS, a threshold value of 90% was used. Thus, only anchors with ≥90% overlap were removed. This was found to improve hedgerow detections, as some hedgerows were detected in a piecewise fashion. Thus, keeping detections with higher amounts of overlap allowed for the combination of partial hedgerow detections to form a fuller detection. A threshold of 80% was used for the filtering of anchors based on object score. This ensured that only detections which the network was confident in would be kept, thus reducing false positive detections (e.g. forest edges).

6.1.1.4 RoIAlign

Anchors processed by the RPN network are projected from the original image onto an FPN feature layer to form RoIs (Figure 14). The size of the anchor determines which FPN layer it will be projected to, with anchors RoIs projected to FPN layers of finer resolutions (lower FPN layers) (Lin et al., 2017). Before being sent to the final FC layer of the Mask R-CNN network, RoIs are processed by the 'RoIAlign' layer. Here, RoI dimensions are standardized to a user determined size, which in the case of Mask R-CNN is either 7x7 or 14x14. The 7x7 sized RoIs are used for classification and bounding box regression tasks, while the 14x14 sized RoI is used for the masking task (Huang et al., 2019). This fixed size of RoIs is required by FC layers which only accept fixed-sized inputs (Johnson, Karpathy, & Fei-Fei, 2016).

Since RoIs coordinates are adjusted by the RPN regression head, the coordinates of RoIs are typically misaligned with the FPN feature map pixels (Figure 14). In Faster R-CNN, RoIs were thus shifted to align with feature maps and the FPN feature map pixel value with the highest value within each 7x7 pixel was taken. However, Mask R-CNN aims to produce precise pixelwise object masks, which is negatively affected by shifts in the RoI coordinates (He et al., 2018). In order to avoid this, Mask R-CNN takes the maximum value from four sample points within each 7x7 pixel (Figure 14 shows this as a 2x2 for simplicity). Thus, four regularly spaced points are sampled within each pixel of the standardized RoIs. The values of the four sample points are determined using bi-linear interpolation, whereby the weighted sum of the four nearest neighbor feature map pixels determine the point's value (He et al., 2018; Gribbon & Bailey, 2004).



$\max(x_1, x_2, x_3, x_4)$
$\max(x_1, x_2, x_3, x_4)$
$\max(x_1, x_2, x_3, x_4)$
$\max(x_1, x_2, x_3, x_4)$

Figure 14 Example of the RoIAlign process. The bold area delineates the region of interest (RoI) while the background grid represents a feature map. For simplicity, the RoI is pooled to a size of 2x2. Four points are sampled (yellow dots) in each RoI pixel, whose values are determined using bilinear interpolation (dashed arrows).

6.1.1.5 Mask R-CNN prediction heads

After being transformed to a fixed size, RoIs are passed to the final masking, classification, and bounding box regression heads. Before reaching the classification and regression heads, RoIs are passed through two FC layers which results in a one-dimensional feature vector of length 1024. Classification and regression outputs are obtained by performing a 1x1 convolution on this vector. For the 1x1 convolution performed by the classification head, the number of neurons is set to the number of target classes (two in the case of hedgerow and background) and a softmax probability (Bridle, 1990) is calculated for each neuron, whereby the highest probability across all neurons becomes the classification result (He et al., 2018). Softmax is a transformation commonly used in multi-class classifications which transforms input feature values into a probability distribution which spans across the target classes (Martins & Astudillo, 2016). The regression head estimates four values which are used to create the final object bounding box for the detected object. The values represent x and y coordinates of the top left corner of the object bounding box, and a height and width value of the box.

The masking head is comprised of an FCN (Long et al., 2015) which performs the pixel-wise masking for all classes. The RoI goes through four 3x3 convolutions, followed by a deconvolutional layer which up samples the 14x14 RoI to 28x28. A 1x1 convolution is then applied to obtain pixelwise object masks for each of the target classes (He et al., 2018). By producing masks for all possible classes, the network can perform the masking task independent of the classification branch. He et al. (2018) found this to greatly improve object mask precision as it was easier for the network to train to an optimal solution when producing multiple binomial masks (single class versus background) instead of a single multinomial masks (all classes compete on a per-pixel basis). After all class-wise masks have been predicted, the mask for the class with the highest classification probability for the given RoI is selected. Masks and bounding boxes are projected from the RoI’s FPN coordinates to the original input image coordinates to produce the final outputs (He et al., 2018).

6.1.2 Loss function

The following describes the loss function used during network training. Given that Mask R-CNN contains multiple sub-networks which perform prediction tasks, a multi-task loss is used (Eq. 9). Classification and regression tasks occur twice in the network (RPN and final network heads) and are thus included twice within the loss function. Given that the loss is comprised of multiple tasks it is possible to vary the weight of each loss function within the network in order to emphasize certain tasks when training the network (section 6.3.4 provides information on these settings). The following sub-sections outline the individual loss functions in more detail.

$$L = L_{cls,reg(RPN)} + L_{cls,reg(MRCNN)} + L_{mask} \tag{9}$$

6.1.2.1 Classification and regression loss function

Classification and regression loss functions are calculated based on the resulting accuracies of randomly sampled RoIs. RoIs are first defined as either positive, negative, or neutral. This depends on the intersection over union (IoU) which the predicted RoI has with the ground truth bounding box (Figure 15). To be classified a positive, negative, or neutral detection, the threshold is set to $x \geq 0.7$, $x < 0.3$, and $0.3 < x < 0.7$, respectively. Neutral RoIs are not included in the loss calculations (Ren et al., 2015). From the positive and negative RoIs, a subset is randomly sampled in order to calculate the loss. This is done to avoid biasing the loss calculation towards negative RoIs as these are more abundant within an image than positive ones. Positive and negative RoIs are each randomly sampled at a ratio of 1:3. Given this random sample of RoIs, the classification and regression loss function ($L_{cls,reg}$) is minimized Eq. (10). $L_{cls,reg}$ is comprised of the loss functions for both the classification (L_{cls}) and regression tasks (L_{reg}).

$$L(\{p_i\},\{t_i\})_{cls,reg} = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum p_i^* L_{reg}(t_i, t_i^*) \tag{10}$$

Here, i is the RoI index, p_i is the predicted softmax probability of RoI i being a hedgerow. The ground truth label corresponding to RoI i is denoted as p_i^*, and has the value of 1 if the RoI is a positive detection, and 0 if the RoI is negative. $L_{cls}(p_i, p_i^*)$ is thus equal to the negative log of the probability of the RoI being of class p_i^* ($-\log(p^i_{p_i^*})$). This is then summed up across all positive and negative RoIs and normalized by the number of randomly sampled RoIs (N_{cls}). The second half of this loss function (L_{reg}) examines error in the predicted anchor (if calculating RPN loss) or object bounding box (if calculating the final Mask R-CNN loss) coordinates, where t_i is a vector of length four representing the predicted coordinates, while t_i^* is the coordinate vector of the ground-truth bounding box which matches the positive RoI. $L_{reg}(t_i, t_i^*)$ then equals $R(t_i - t_i^*)$, where R is the smooth L$_1$ loss function shown in Eq. (11) (Girshick, 2015b).

$$Smooth_{L_1}(x) = \begin{cases} 0.5x^2 & if\ |x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \tag{11}$$

As p_i^* is equal to 0 for negative RoIs, the regression loss is only calculated for positive RoIs. The sum of L_{reg} across all positive RoIs is normalized by the number of RoIs produced for a single image. λ is used as a balancing parameter used to give both L_{cls} and L_{reg} terms roughly equal weight (Girshick, 2015b).

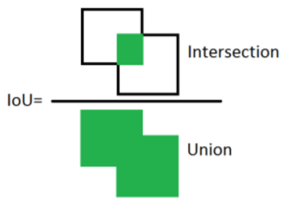


Figure 15 Shows a graphical representation of the intersection of union calculation.

6.1.2.2 Masking head loss function

In addition to the regression and classification losses, Mask R-CNN also incorporates a loss for class masks. Since RoI classification is performed by the classification head, the mask head predicts masks irrespective of class, producing a set of *K* masks for each RoI, where *K* = number of classes. In the case of multi-class tasks, only masks generated for the class corresponding to the true class of the RoI are incorporated into the mask loss. Mask values are output in the form of probabilities, to which a sigmoid function is applied to each of the predicted mask pixels. A sigmoid output value ≥ 0.5 will result in a pixel being classified as the target class. L_{mask} is defined as the binary cross-entropy loss, averaged across all pixels, as shown in Eq. (12) (He et al., 2018).

$$L_{mask} = -\frac{1}{m^2}\sum y_{ij}\log(\hat{y}_{ij}^k) + (1 - y_{ij})\log(1 - \hat{y}_{ij}^k) \qquad (12)$$

Here, y_{ij} represents the value of the ground truth mask for a given pixel (*i*, *j*), while $\hat{y}_{ij}^k$ is the predicted mask value for the same pixel for the class *k*. Thus, when the ground truth label is zero (background pixel), the loss for that specific pixel is given by $-\log(1 - \hat{y}_{ij}^k)$. When the ground truth label is one (hedgerow pixel), the loss for the pixel is $-\log \hat{y}_{ij}^k$. The sum of these losses across all pixels is then divided by the area covered by the ground truth mask (m^2).

6.2 DeepLab v3+

6.2.1 Network architecture

DeepLab is an FCN network which performs semantic segmentation using an encoder-decoder structure (Figure 16). The encoder portion of the network is a CNN network used to generate features used to determine the occurrence of objects. Here, pooling of lower level features (e.g. edges, colors) leads to higher order semantic features in later layers of the encoder. This process results in a down-sampling of an image’s spatial dimensions, and thus a reduced capacity to accurate localize objects within an image. The decoder module helps to restore object locations by performing an up-sampling of the higher order semantic feature layers in order to restore the original spatial dimensions of the input image. Thus, DeepLab is able to assign semantic labels to each pixel within a given image.

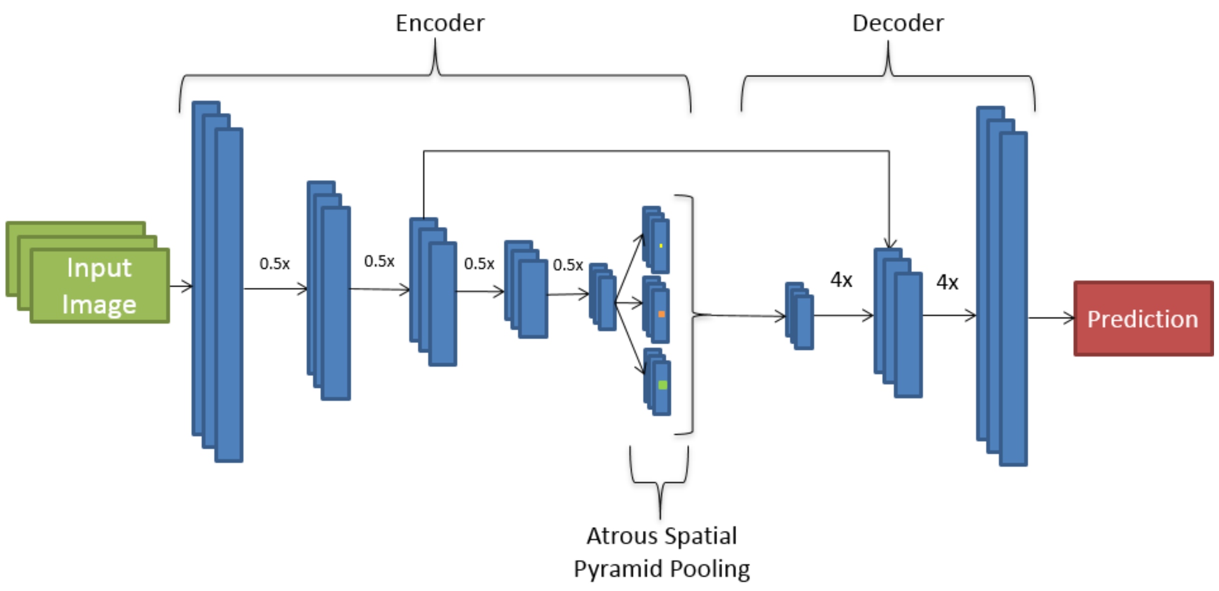


Figure 16 General architecture of the DeepLab v3+ network. The encoder consists of convolutional blocks where downsampled occurs by a rate of 0.5 at each step. Atrous spatial pyramid pooling uses atrous rates of 6, 12, and 18. Upsampling occurs at a rate of four at each step within the decoder.

6.2.1.1 Encoder

The encoder module of the DeepLab network is a CNN known as Xception (Chollet, 2017). Xception uses what are known as depthwise separable convolutions (DSC) instead of normal convolutional filters. Thus, a normal convolution which spans across all input channels is factorized into two steps (Figure 17). First, a depthwise convolution of size *N*x*N* is performed, where a spatial convolution is performed independently for each channel. Each region where the *N*x*N* convolution is applied results in a single value output. A pointwise convolution (1x1) is applied to the output values across all channels, thus aggregating the information across channels. In this way, convolutional filters no longer operate across all three dimensions of the input layers simultaneously, allowing for convolutional filters to learn spatial and cross-channel features separately (Chollet, 2017).

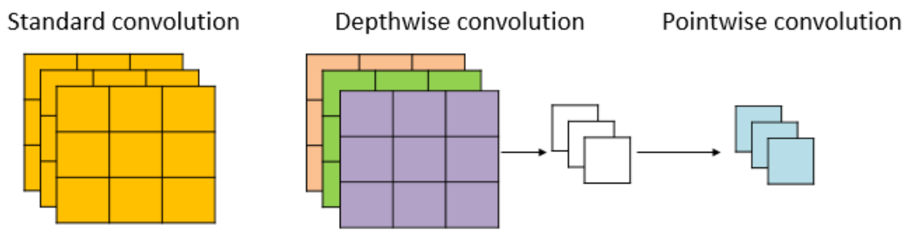


Figure 17 Comparison between standard convolutions and depthwise separable convolutions (DSC). Each color represents a set of learned filter values. DSC is a combination of the depthwise and pointwise convolutions. The white pixels are the intermediate output from the depthwise convolution which is used for the pointwise convolution.

DeepLab modifies version the Xception network by inserting more convolutional layers. The network architecture can be broken down into three main modules (Figure 18). First, the input image is passed through an entry flow, where two standard 3x3 convolutions are applied. The result is then passed through three separate blocks which each consist of a two 3x3 DSC operations with strides of one, followed by a 3x3 DSC operation using a stride of two. The result from the third DSC operation in each of these blocks is merged through a residual connection with the layer produced earlier through a 1x1 convolution employing a stride of two (Chen et al., 2018).

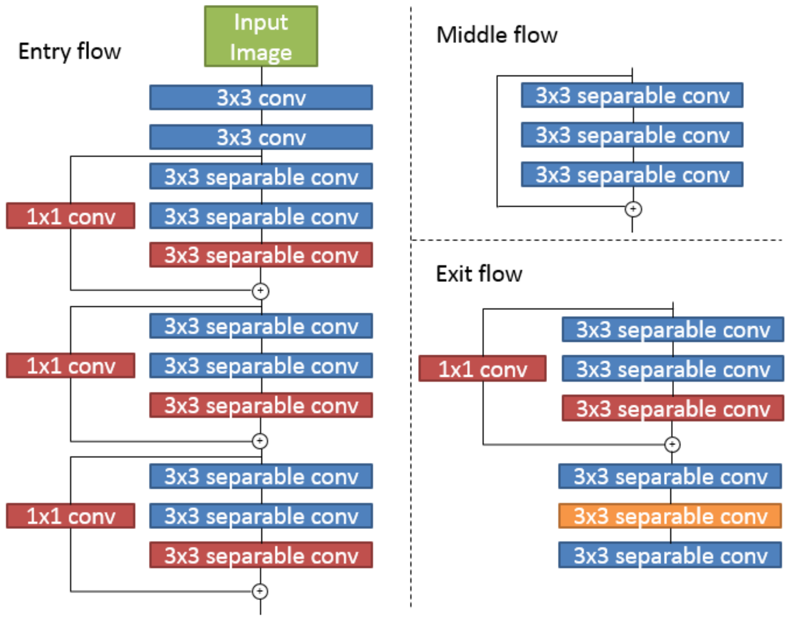


Figure 18 The three portions of the Xception network used as the encoder in DeepLab v3+. Blue convolutional operations use a stride of one, while red convolutions use strides of two. The orange convolution uses a stride of one with an atrous rate of two.

The middle flow consists of one block which DeepLab iterates over 16 times. This block consists of three 3x3 DSC operations with strides of one. The feature layer used as input to a given iteration of the middle flow are then merged with the output of the third DSC to form the block’s output feature layer (Chen et al., 2018).

The exit flow of the network consists of two parts. In the first part, two DSC operations with a stride of one are performed, followed by a DSC operation with a stride of two. The input to the exit flow undergoes a 1x1 convolution with a stride of two, which is merged with the output of the DSC of stride two. The resulting feature layer is passed to the second part of the exit flow, which performs one 3x3 DSC operation, followed by a single 3x3 atrous convolution with a stride of one and rate of two, followed by one final 3x3 DSC with stride of one, resulting in the final output of the exit flow (Chen et al., 2018).

The output from the exit flow portion of the encoder is passed through an ASPP module (Figure 16). ASPP extends the use of atrous convolution by applying varying rates of atrous convolution to a feature layer. Outputs from the different atrous convolutions are merged to form the output feature layer. ASPP thus allows for the capture of objects and contextual features at multiple spatial scales (Chen et al., 2017).

6.2.1.2 Decoder

The encoder portion of the network results in feature maps 16 times smaller than the original image. The decoder module (Figure 16) of DeepLab is meant to restore the feature scores obtained by the encoder module to the original image dimensions. Thus, an up-sampling of the encoder feature layer is required. In order to enhance the spatial precision of mask predictions, this up-sampling is broken into multiple stages and combined with residual information from higher resolution feature layers from the encoder. Thus, the feature maps output by the ASPP layer are first up sampled by a rate of four using bilinear interpolation. The resulting feature maps are then merged with lower level features from the encoder module which match the spatial resolution. This layer then undergoes a 3x3 convolution, followed by an upsampling at a rate of four again using bi-linear interpolation (Chen et al., 2018b). At this point the feature maps have been up sampled to the original dimensions of the input image. Feature scores for each pixel are then used to estimate the probability of a given pixel belonging to each of the target classes using a softmax probability function (Mel, Michieli, & Zanuttign, 2019).

6.2.1.3 Batch normalization

Throughout the network, all convolutional operations use batch normalization layers. The inclusion of batch normalization layers has been found to improve network generalization and significantly improve training performance and are thus incorporated in many state of the art network architectures (Ioffe & Szegendy, 2015; Hoffer, Hubara, & Soundry; 2017; Masters & Luschi, 2018). A batch is a set of randomly chosen images used in a given training step. This is because memory restrictions limit the amount of images which the network can process at once.

The training of networks has been shown to benefit from input data (e.g. IKONOS images) being decorrelated, with unit variance and a mean of zero (LeCun et al., 1998; Wiesler & Ney, 2011). This same effect applies to feature maps which are passed as input to subsequent convolutional layers in a NN. However, continual changes to network weight parameters during network training lead to differences in the distribution of feature layer values between training steps. This complicates the process of training the network as learned parameters must adapt to changing distributions of input values. This problem is known as internal covariance shift and has been addressed through batch normalization layers, which aim to fix the distributions of feature layer values (Ioffer & Szegendy, 2015). It achieves this by normalizing the values of an input feature layer (Eq. 15) using an estimated mean (μ) (Eq. 13) and variance (σ^2) (Eq. 14) for each input value (x_i) across a training batch (m). Thus, for a batch of size eight, the mean and variance of a single pixel position within the matrix of the image (0,0) is estimated using the eight pixel values which occur at this coordinate.

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{13}$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu)^2 \tag{14}$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{15}$$

$$y_i = \gamma * \hat{x}_i + \beta \tag{16}$$

In Eq. (15), ϵ is small constant greater than zero which ensures that division by zero does not occur. One problem with normalizing unit activations is that the expressive power of the values may be reduced. Thus, a shift (β) and scaling (γ) of the normalized values ($\hat{x}_i$) (Eq. 16) is used to maintain the expressive power of the network, leading to batch normalized values (y_i) for each pixel of a feature map. These batch normalized values are then used as the input to the proceeding layer. The shift and scale parameters are learned for each layer where batch normalization is applied (Ioffe & Szegendy, 2015).

6.2.2 Loss function

DeepLab applies a per-pixel loss function as shown in Eq. (17).

$$L = \sum_{k=0}^{N} -\log\,(\hat{y}_{ij}^{k}) \tag{17}$$

Here, $\hat{y}_{ij}^{k}$ is the predicted probability of pixel (i, j) belonging to class k, where k is the ground truth class for the given pixel. This loss is performed across all pixels and averaged for the total loss value (Chen et al., 2016).

6.3 Hyperparameter configurations

6.3.1 Batch size

Both Mask R-CNN and DeepLab were trained in steps using *N* sized batches, where *N* is equal to the number of images which can fit into the memory of the GPU at once. When fine-tuning pre-trained networks, a single Geforce RTX 2080 Ti GPU with 11 GB of memory was used, leading to batch sizes of six. Batch sizes were limited because Mask R-CNN did not allow for the use of multiple GPUs, and thus to provide a fair comparison between Mask R-CNN and DeepLab, batch sizes were kept equal.

Given that the mean and variance parameters of batch normalization layers are calculated within each batch, large batch sizes are required to achieve accurate parameter estimates (Ioffe & Szegendz, 2015).

Given the case of pre-trained networks, batch normalization parameters had already been trained. Thus, these parameters were left unchanged, making the small batch sizes of six acceptable. However, batch normalization parameters did require training when using randomly initialized weights with DeepLab (See section 6.2.1.4). Batch sizes which are too large (e.g. 128 or more) have been found to degrade generalization of networks (Hoffer, Hubara, & Soudry, 2017). However, the use of three Geforce RTX 2080 Ti GPUs only allowed for batch sizes of 18, thus allowing for the training of batch normalization layers.

6.3.2 Class weights

Due to the thin nature of hedgerows, images were dominated by background pixels. Such class imbalances can bias the loss function, leading to poor classification results (Lin et al., 2017b). In such cases, an accuracy measure could achieve high accuracy rates by labelling all pixels as background. In order to avoid this, class specific losses were weighted in favor of the hedgerow class in order to emphasize the correct labelling of this class. For DeepLab, a value of 350 was chosen for hedgerows, while background was left at 1. Values above 350 were found to cause problems with the training, resulting in loss values growing towards infinity (NaN). This was less of a problem for Mask R-CNN given that images are first segmented into smaller regions (RoIs) where the target object is dominant. As such, no class weights were applied for Mask R-CNN.

6.3.3 Anchor sizes

The setting of appropriately sized anchors is important in accurate object detection within the Mask R-CNN network, as these anchors perform the initial region proposals which receive further examination for the target object(s). Thus, if anchors sizes are set inappropriately (e.g. do not approximate the true object bounding box shapes) the network will struggle to accurately locate objects (Yao et al., 2017). The original work by He et al. (2018) used anchor aspect ratios of 0.5, 1, and 2. However, these were found to perform poorly for hedgerows due to their elongated shape. Anchor aspect ratios where length was much greater than the width for both horizontal and vertical cases were found to greatly improve performance. As hedgerows may also run diagonally across an image, a square shaped aspect ratio was also used, resulting in aspect ratios of 0.2, 1.2, and 5. Additionally, the scale parameter for the anchors needs to be adjusted so that the size of anchors loosely resembles that of the ground truth boxes. Anchor scales were set to 30, 60, 100, 150, and 200 to sufficiently cover the most commonly sized hedgerow bounding boxes. Using a scale of 200 resulted in elongated anchors becoming larger than the image itself (Figure 19) but was necessary in order to increase the size of the square shaped anchor which was meant to capture long diagonal hedgerows running across the image.

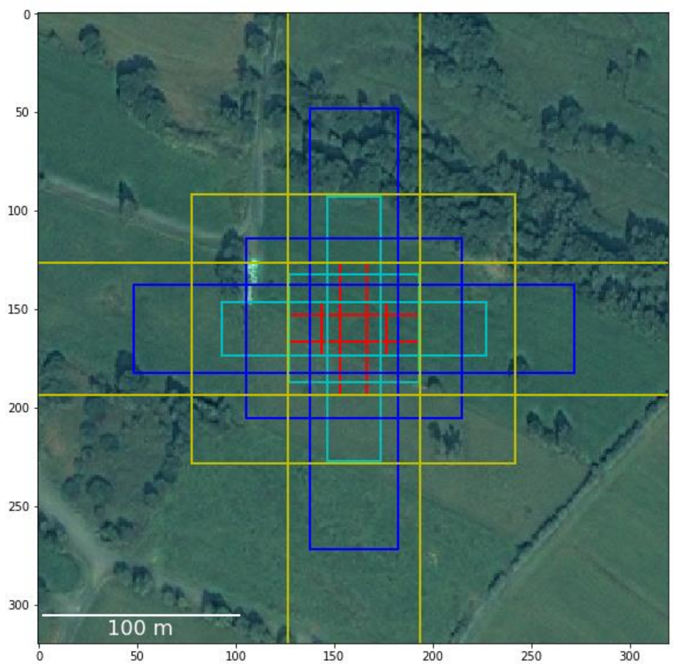


Figure 19 Shows the range of anchors produced by the region proposal network. Each color represents a different scale.

6.3.4 Loss function weights

As Mask R-CNN uses a multi-task loss function, weights can be set for each of the tasks. The classification tasks were given the highest weights as these two classifications are the most important tasks in order to generate hedgerow detections and subsequent masks. As the initial classification of region proposals from the RPN has a cascading effect onto the later tasks, the loss weight associated with this task was set the highest (25). The weight of the later classification at the network head was set lower (6) as the first classification task has already deemed all areas examined by this stage as being possible hedgerows. Masking is an important part of the task and as such the weight for this loss was also increased (2.5). The task of bounding box regression, while useful in refining the coordinates of the initial region proposal, is was not as important a task in this case, as accurate pixel-wise localization of hedgerows was of more importance than their bounding boxes. As such, the weight for both the initial RPN and final network head bounding box estimations were decreased (0.6).

6.3.5 Network weights

6.3.5.1 Pre-trained weights

Pre-trained network weights were used for both DeepLab and Mask R-CNN. The utilization of pre-trained model weights is a practice known as transfer learning which has been shown to greatly improve results across various image analysis tasks (Wurm et al., 2019; Xie et al., 2016; Girshick et al., 2015). It involves the training of a network on a large secondary dataset in order to learn low level features which are universally useful in object detection (e.g. edges, color blobs) (Yosinski et al., 2014), and the subsequent fine-tuning of the learned features using the target dataset. Because low level features are generally useful for most image analysis tasks, these layers are typically left frozen during the fine-tuning so that they cannot change. Thus, only higher levels of the network which combine lower level features are fine-tuned to learn the appropriate feature combinations for the target detection task (Penatti, Nogueiria, & Santos, 2015).

Model weights pre-trained on the public image dataset COCO were utilized as a starting point for both networks. This dataset contains a total of 165,482 training images, 81,208 validation images, and 81,434 test images (Lin et al., 2015). Other popular largescale datasets can also be used for transfer learning (e.g. ImageNet), however such datasets are mostly comprised of images where objects are unobstructed and centered within the frame, leading to difficulties in the detection of small, ambiguous, or partially occluded objects. Such objects within an image require context for proper identification. Given that objects in the COCO dataset are often amid clutter, partially occluded, or residing in the background, networks trained on this dataset place increased importance on learning context based features, leading to increased performance in the detection of objects within natural contexts (Lin et al., 2015). This should thus help with the detection of hedgerows as spectral information alone can lead to misdetections with other vegetation, and contextual information, such as placement within agricultural fields, is important.

The use of transfer learning does however lead to drawbacks when using MS images, as the number of bands used in the pre-training must match the number of bands provided for further training of the network. The COCO images used for pre-training are comprised of red, green and blue (RGB) bands, meaning that only three of the four IKONOS bands could be used as input for pre-trained networks.

6.3.5.2 Training with randomly initialized weights

In addition to using pre-trained networks, DeepLab was also trained from a randomized set of initial weights. The training of a network from ‘scratch’ allows for the incorporation of a greater number of input layers, such as MS images combined with digital elevation model (DEM) layers (Ma et al., 2019). In our case, training from scratch allowed for an increased amount of spectral information to be used by incorporating all four of the IKONOS spectral bands. However, the network received a far smaller amount of training data as only the 513 training images were available. Small training datasets are known to lead to overfitting (Zhu et al., 2017; Larsson, Maire, & Shakhnarovich, 2017; Krizhevsky et al., 2012), which occurs when the number of parameters of a model is too high relative to the number of training samples (Zhong, Fei, and Zhang, 2016). A lack of training data is often overcome using data augmentation, which increases the amount of training data and allows the model to learn the variations of the target object's features. While data augmentation allowed for the increase to a total of 12,655 images, the dataset size was still smaller than the COCO dataset (165,482 images) used to train the pre-trained network.

Weights were initialized following the 'He Initialization' (He et al., 2015b), which has been specifically designed for use with deep NN using ReLU activation functions. Here, weights follow a Gaussian distribution with a mean of zero and a standard deviation of $\sqrt{2/n_l}$, where n_l is the number of input layers which a given neuron is connected to. This is done to standardize the standard deviation of neuron output values between NN layers, as deep NN experience difficulties in training all layers equally due to differences in the standard deviations of layer output values (Ioffe & Szegendy, 2015; Glorot & Bengio, 2010). To explain, each value of a feature map is the dot product between the neuron's weights and the input values (both of which are expected to follow a standard Gaussian distribution). Thus, the distribution of the feature map's values also follows a standard Gaussian distribution, having a standard deviation of one. The sum of all feature maps will give the total standard deviation of a given layer. As such, the total standard deviation becomes $1 * n_l$, and thus we must normalize by n_l to attain a standard deviation of 1 (He et al., 2015b). As the ReLU function sets any values ≤ 0 to zero, roughly half of the activations using ReLU will become 'stuck' during training (they will always return a value of zero regardless of the input). Thus, a factor of two is used ($\sqrt{2/n_l}$) when scaling the range of weight values in order to offset the loss of these neurons so that the standard deviation remains approximately one (He et al., 2015b).

As mentioned here, the input image values should be normalized to a standard Gaussian distribution in order to ensure an even standard deviation of neuron output values across NN layers. As such, image pixel values were normalized on the fly in Python during the loading of images into a batch. Additionally, due to the use of batch normalization layers in the DeepLab network, weight values could have likely been initialized using a standard Gaussian distribution (instead of $\sqrt{2/n_l}$) as batch normalization already ensures that the distributions of layer output values follow a standard Gaussian distribution (Ioffe & Szegendy, 2015).

6.4 Experiments

6.4.1 Band combinations

The use of pre-trained NN is a common practice, as features learned from one dataset often generalize well to new ones (Xie et al., 2016). However, pre-trained networks are trained on three band (RGB) images. Thus, fine-tuning using the target dataset must also be done using three band inputs. In order to determine which three bands are optimal for the fine-tuning of Mask R-CNN and DeepLab, all possible band combinations for IKONOS were tested. Networks were trained for 100 epochs using each of the three-band combinations. Learning rates were set to 0.001 for both NN.

6.4.2 Seasonal inputs

The IKONOS images available for analysis were obtained from two different seasons, with half of the dataset being from October, and the other half from May. A shift in the phenological phase between the two time periods leads to different spectral signals between hedgerow objects as vegetation undergoes senescence in October, leading to higher reflectance in the red wavelength between 550 – 740 nm due to a degradation of chlorophyll in plant leaves (Merzlyak et al., 2002). It was thus investigated whether the use of data from different seasons influences network performance. Three datasets were tested; one containing only October images, one containing only May images, and one containing a mixture of the two. In order to control for the effect of training dataset size, all three datasets were limited to a size of 247 images. Augmentations (See section 6.4.3) were applied to each dataset in order to increase the sizes, leading to total dataset sizes of 5,434 training images for each of the seasons. Each trained network made predictions on three seasonal validation datasets (October, May and mixed). Validation datasets were also standardized between the three seasons, resulting in 87 validation images for each season. Networks were trained for 100 epochs using each of the seasonal datasets. Learning rates were set to 0.001 for both NN.

6.4.3 Data augmentation

As the total dataset comprised of only 513 training images, data augmentation was performed in order to increase the training dataset size. Data augmentation strategies for remote sensing images typically rely on simple geometric augmentations such as rotation, flipping, or translational, with some researchers also applying spectral augmentations (e.g. adding noise) (Stiller et al., 2019; Dodge & Karam, 2016). However, no work has quantitatively examined the effects of these and other possible augmentations regarding the detection of vegetation.

The consensus is that data augmentation leads to improved model results (Ma et al., 2019, Voulodimos et al., 2018; Krizhevsky et al., 2012). Augmentation allows for robust recognition of objects by exposing the network to variations of object features such as size, orientation, position, and other distortions (LeCun et al., 1998). For example, by applying random noise to image pixel values, the network learns to identify objects on features other than their spectral signatures (e.g. shape). This should allow the network to reduce false detections of spectrally similar wooded vegetation such as forest edges, a common problem with hedge object detection (Fauvel et al., 2012; O’Connell et al., 2015). Fawzi et al. (2016) have found that inappropriate augmentations can result in negative effects on network prediction accuracy. For example, a rotated car is still a car, while a rotated “9” can become a “6”. Thus, it is important to investigate which augmentations are appropriate for the given detection task (Dvornik, Mairal, & Schmid, 2019).

Numerous data augmentation strategies were applied, and the resulting model performances were measured in order to quantify the effects of each data augmentation type. While this work focuses only on hedgerows, these data augmentation techniques may be transferable to other vegetation types (e.g. tree canopy). Augmentations which mimic typical variations in the target object as well as in satellite imagery were chosen, such as changes in object orientation and scale, differences in illumination conditions, and image noise (Tong et al., 2019; Mountrakis, Im, & Ogole, 2010). All augmentations applied in this study are summarized in table 1 and Fig. 20 shows an example of how an image changes under the application of each augmentation. Table 1 also provides the shorthand names used in this paper to refer to each augmentation. Augmentations were applied using the Python package imgaug (version 0.3.0).

Given that increases in dataset size lead to positive benefits in model performance, augmentations were examined individually rather than in a stepwise additive fashion to avoid the confounding effect of increasing dataset size. Thus, the augmentation datasets contained 1026 images. In the case of rotated and scaled images, hedgerows at the edges of the image were sometimes cropped out by these augmentations, leading to some images being void of hedgerow objects and thus being discarded. The rotation 45 and scale datasets thus contained 989 and 946 images, respectively. Networks were trained for 100 epochs for each of the individual augmentations with learning rates of 0.001 for both NN.

Name	**Parameter setting**	**Description**
Add	Range: -80 – 80	Image pixel values are uniformly increased or decreased by a value randomly selected within the range parameter.
Add per channel	Range: -80 – 80	Image pixel values are increased or decreased uniformly across each band. For each band a new value is randomly selected from within the range parameter.
Noise	Range: -60 – 60	Image pixel values are increased or decreased on a per-pixel basis. For each pixel a new value is randomly selected from within the range parameter.
Blur	Sigma: 0.75	Gaussian blurring is applied to each pixel with a sigma value of 0.75.
Contrast	Gain: 0.6 – 1.4	Image contrast is increased uniformly by a value randomly selected from within the range of the gain parameter.
Contrast per channel	Gain: 0.6 – 1.4	Image contrast is increased on a per-channel basis. For each channel a new value is randomly selected from within the range of the gain parameter.
Flip	None	Images are either flipped along the horizontal or vertical axis.
Rotate 45	None	Images are rotated by a degree of 45.
Rotate 90	None	Images are rotated by a degree of 90.
Scale	None	Images are either scaled up to 120% of the original image size or down to 80% of the original image size.

Table 1 Gives the shorthand names used for each of the different augmentations used in this study. Parameter settings used for each augmentation as well as a description of how the image changes under each augmentation is provided.

When applying all augmentations together for the final augmented datasets, two different datasets were made. One used only geometric augmentations (flip, rotation, and scaling), applying either one, two or three randomly selected (without replacement) geometric augmentations per image. The other used both geometric and spectral augmentations (blur, noise, contrast, contrast per channel, add, and add per channel), applying one, two, or three randomly selected geometric augmentations and either zero, one, or two randomly selected spectral augmentations per image. Here, a larger range of possible rotations was used, with rotations ranging from 40 - 320° with steps of 40° in between. Final augmented datasets for the geometric dataset and the geometric and spectral augmented dataset contained 12,655 images and 12,499, respectively. The difference in the size of the dataset was caused by the random application of augmentations, leading to differences in the number of images containing hedgerow objects. Networks were both trained for 100 epochs on each of the two final augmentation datasets using a learning rate of 0.001

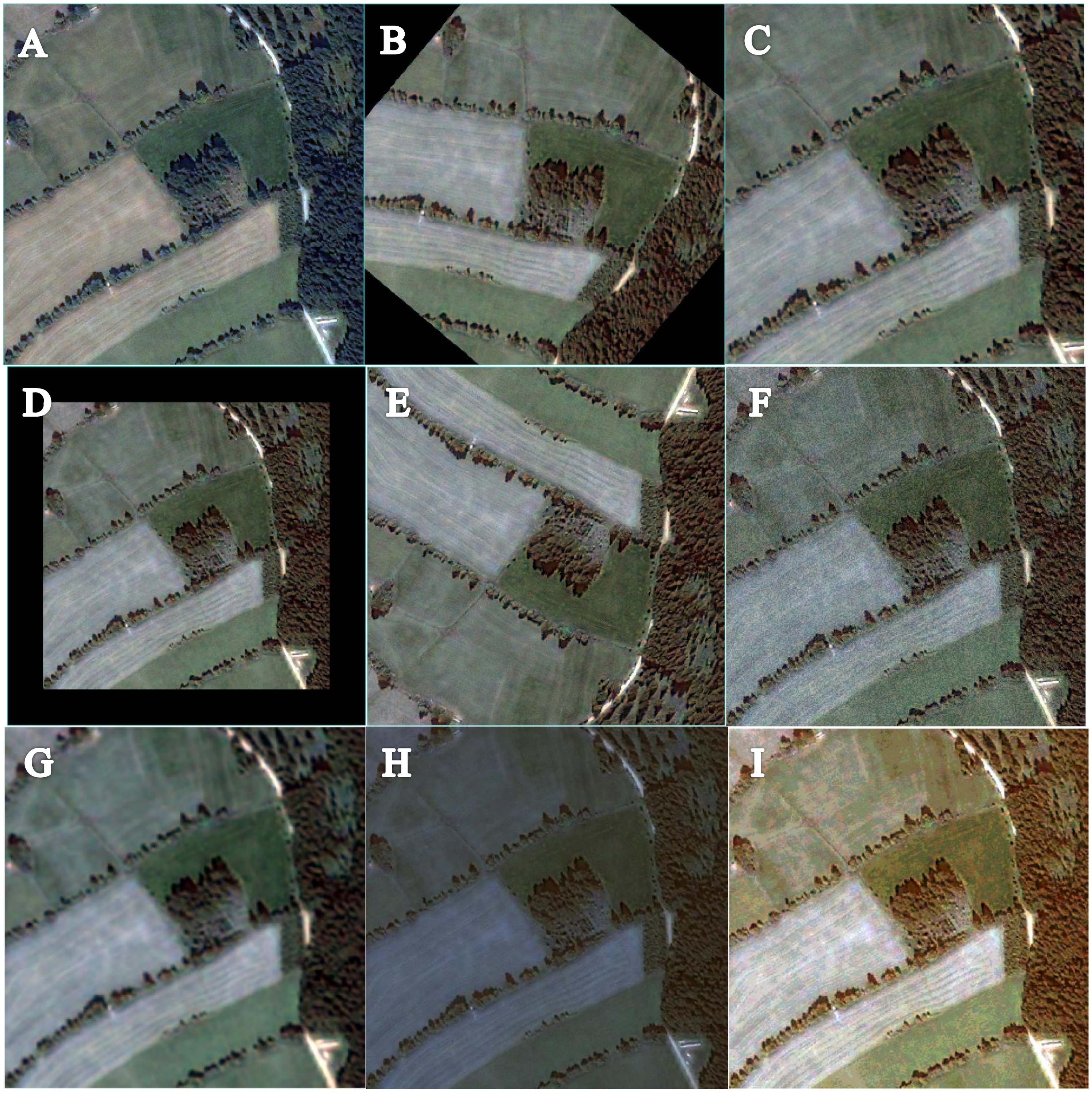


Figure 20 Shows an un-augmented image (a) and the image with augmentations applied. Augmentations involved rotation (b), scaling up (c) and down (d), flipping the image (e), adding noise (f), applying a Gaussian blur (g), uniformly decreasing (or increasing) all image pixels (h), and applying log contrast (image- or band-wise) (i).

6.4.4 Randomly initialized weights

Many researchers using NN for remote sensing applications use pre-trained network weights followed by fine-tuning (Wurm et al., 2019; Sa et al., 2016). However, these methods restrict the number of band inputs to three. In addition, research implies that the use of features extracted from natural images may not be well suited to remote sensing images due to the differences between natural and satellite images (Chu et al., 2016; Yosinski et al., 2014). Thus, DeepLab was trained using randomly initialized weights in order to compare with results from pre-trained network. Here, the two final augmented datasets described in the preceding section were each used to train the network for a total of 100 epochs. Here, the optimal learning rate was found to be 0.01.

6.5 Performance assessment metrics

Object detection tasks often utilize precision and recall measures (Wen et al., 2019; Zhao et al., 2018; Zhang et al., 2018b). Contrary to typical remote sensing performance measurements which are performed on a pixel basis, precision and recall are calculated based on objects. Thus, the overlap between a ground truth mask and a predicted mask are used to determine a positive detection. Thresholds are required in order to determine what amount of overlap between the predicted object mask and the ground truth mask constitutes a true positive. Here, three thresholds were examined; >70%, >50%, and >30% overlap.

Precision (Eq. 18) and recall (Eq. 19) are calculated using true positive (TP), false positive (FP) and false negative (FN) values (Zhao et al., 2018). TP is thus deemed as an area where ground truth and predicted mask overlaps are higher than the given threshold value, FP occurs when a predicted mask does not meet the requirements of a TP, while FN occurs when a ground truth mask does not adequately overlap with any predicted masks. The >70% threshold is considered the most important indicator of network performance as hedgerow masks should cover the majority of the ground truth objects. However, the inclusion of lower thresholds offered an improved quantitative representation of the network performances.

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

Per-pixel accuracy was also used to measure performance. However, as background pixels were not of interest and would lead to inflated rates of accuracy due to large class imbalances, per-pixel accuracy was calculated by looking at only areas where either the ground truth or predicted masks occurred, thus leaving true negatives from the calculation (Eq. 20).

$$\frac{TP_{pixel}}{TP_{pixel} + FP_{pixel} + FN_{pixel}} \tag{20}$$

Here, TP_{pixel} is deemed as a pixel where ground truth mask and predicted mask both produce a one, FP_{pixel} is deemed as a pixel where the ground truth produces a zero and the predicted produces a one, while FN_{pixel} is deemed as a pixel where the ground truth produces a one and the predicted mask produces a zero.

6.8 Python implementation

All data pre-processing tasks, network training, and analysis of results were carried out using Python (version 3.7.1), and the scripts used can be found in an online repository (Appendix II). DeepLab was written with the Python package Tensorflow, while Mask R-CNN was written using the package Keras, which is an extension of the Tensorflow package designed for easier implementation of NN. The code for DeepLab was written by Chen et al., (2018) and can be located in an online repository provided by Tensorflow (https://github.com/tensorflow/models/tree/master/research/deeplab). In this study, the Matterport Inc. implementation of Mask R-CNN which was released under a Massachusetts Institute of Technology license was utilized (https://github.com/matterport/Mask_RCNN). Given that the version dependencies of the two networks differed, two separate Docker (version 1.8.0) containers were created to allow for different work environments containing the appropriate package versions (See the README.md files in each network's online code repository).

Given that these two NN were designed for use with 8-bit RGB images, the source code was edited to accept 16-bit image values. Additionally, the code for DeepLab was changed in order to allow for four input bands. These can also be found within the repository linked in appendix II. As Tensorflow currently does not support the reading of .tif image files, all image tiles were converted to .png format.

7 Results

7.1 Band combinations

NN networks are typically pre-trained on large secondary datasets of RGB images (e.g. COCO). Pre-trained networks are thus constrained to three bands as input. As IKONOS images contain four bands (RGB and NIR), different band combinations were tested in order to find the optimal three band combination. Table 1 shows the results from each combination for both Mask R-CNN and DeepLab.

For Mask R-CNN, per-pixel accuracy for hedgerows was highest using GRNIR images (32.6%), followed by RGB (31.9%), BGNIR (31.7%), and BRNIR (30.6%). To get a better quantitative representation of the networks, three different thresholds (>70%, >50%, and >30%) were used to define a true positive for precision and recall. Differences in precision given a >70% overlap were minimal between the different band combinations. RGB and GRNIR both achieved the highest precision (46.5%) while BGNIR had the lowest (45.7%). Recall rates at >70% were highest in the

GRNIR dataset (48.6%), while BGNIR was again the lowest (43.5%). At the >50% threshold, BGNIR had the highest precision (58.7%) with BRNIR (56.2%) having the lowest. Here, the highest recall came from GRNIR (59.7%), with BRNIR (53.9%) again with the lowest. At the >30% threshold, RGB had both the highest precision and recall (65.2% and 66.3%, respectively), while GRNIR (61.6%) had the lowest precision, and BGNIR (59.9%) the lowest recall.

For DeepLab, hedgerow per-pixel accuracy was highest using RGB bands (30.5%), while BRNIR was the lowest (29.4%). At the >70% threshold, GRNIR had the highest precision score (60.7%), while BGNIR (57.6%) had the lowest. Here, the highest recall was BGNIR (63.9%), with the lowest being RGB (56.7%). At the >50% threshold, precision was highest for GRNIR (71.5%) and lowest for BGNIR (64.2%). Here, recall was the highest for BGNIR (71.2%) and lowest for RGB (65.6%). At the >30% threshold, precision was highest for BRNIR (79.9%) and lowest for BGNIR (68.6%), while recall was highest for BRNIR (77.3%) and lowest for RGB (73.5%).

GRNIR was deemed as the optimal band combination given high precision, recall, and per-pixel accuracies at the >70% threshold between both networks. Thus, all subsequent data analyses were performed using only the GRNIR bands.

Band combination and network	Per-pixel accuracy (%)	Precision @70% (%)	Recall @70% (%)	Precision @50% (%)	Recall @50% (%)	Precision @30% (%)	Recall @30% (%)
RGB (M)	31.9	**46.5**	47.3	58.3	59.2	**65.2**	**66.3**
GR+NIR (M)	**32.6**	**46.5**	**48.6**	57.1	**59.7**	61.6	64.4
BR+NIR (M)	30.6	46.4	44.5	56.2	53.9	62.4	61.0
BG+NIR (M)	31.7	45.7	43.4	**58.7**	55.8	64.1	59.9
RGB(D)	**30.5**	60.1	56.7	69.6	65.6	78.0	73.5
GR+NIR(D)	30.3	**60.7**	59.5	**71.5**	70.1	76.3	75.1
BR+NIR (D)	29.4	60.5	58.6	70.4	68.2	**79.9**	**77.3**
BG+NIR(D)	30.2	57.6	**63.9**	64.2	**71.2**	68.6	76.1

Table 2 Results of four possible band combinations for IKONOS. Bands include red (R), green (G), blue (B), and near infrared (NIR). Results for the Mask R-CNN network are denoted with (M), and DeepLab v3+ with (D). Precision and recall are presented at three levels (70, 50, and 30%). The highest result from a column for each method is bolded.

7.2 Seasonal inputs

The IKONOS images available for analysis were obtained from two different seasons, with half of the dataset being from October, and the other half from May. It was thus investigated whether the use of data from different seasons influences network performance.

Results showed that the highest per-pixel accuracies, as well as precision and recall (at >70% threshold), were achieved when using October images for both training and prediction. Here, Mask R-CNN and DeepLab attained 41.7% and 42.8% per-pixel accuracies, 49.6% and 50.0% precisions, and 60.5% and 67.6% recalls, respectively. Per-pixel accuracy, precision, and recall all decreased when using the dataset containing a mixture of both October and May images (henceforth referred to as the mixed dataset) at the prediction stage after training with images from October. Using images from May at the prediction stage after using October for training resulted in the lowest scores across all performance metrics (Figures 21, 22, 23).

Networks trained with the mixed dataset achieved the highest per-pixel accuracies, recall, and precision when using images from October in the prediction stage. Here, Mask R-CNN and DeepLab attained per-pixel accuracies of 40.5% and 39.1%, precisions of 48.9% and 42.2%, and recalls of 56.0% and 57.2%, respectively. Using the mixed dataset in the prediction stage lowered per-pixel accuracies as well as precision and recall, while using May images for predictions resulted in the lowest performance across all three metrics (Figures 21, 22, 23).

For networks trained with only images from May, results were highest using May images during the prediction stage, with per-pixel accuracies of 36.2% and 38.6%, precisions of 46.7% and 37.8%, and recalls of 56.7% and 54.5%, for Mask RCNN and DeepLab, respectively. Using the mixed seasons images for predictions led to decreases in performance metrics scores, while using October images for prediction resulted in the lowest performance metrics scores (Figures 21, 22, 23).

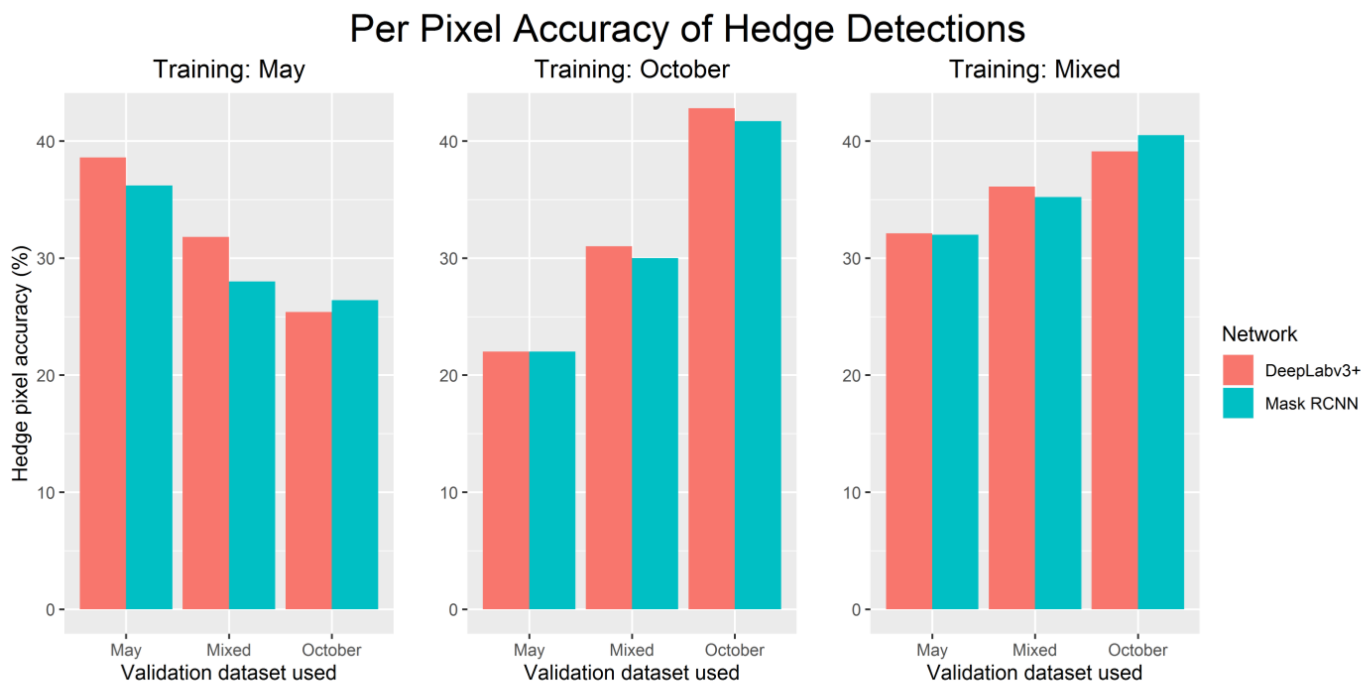


Figure 21 Per-pixel accuracy scores for both DeepLab and Mask R-CNN using three different seasonal datasets (May, October, and a mixture of the two) split into training and validation sets. Networks were trained on one of the three datasets and subsequently assessed using each of the three validation datasets.

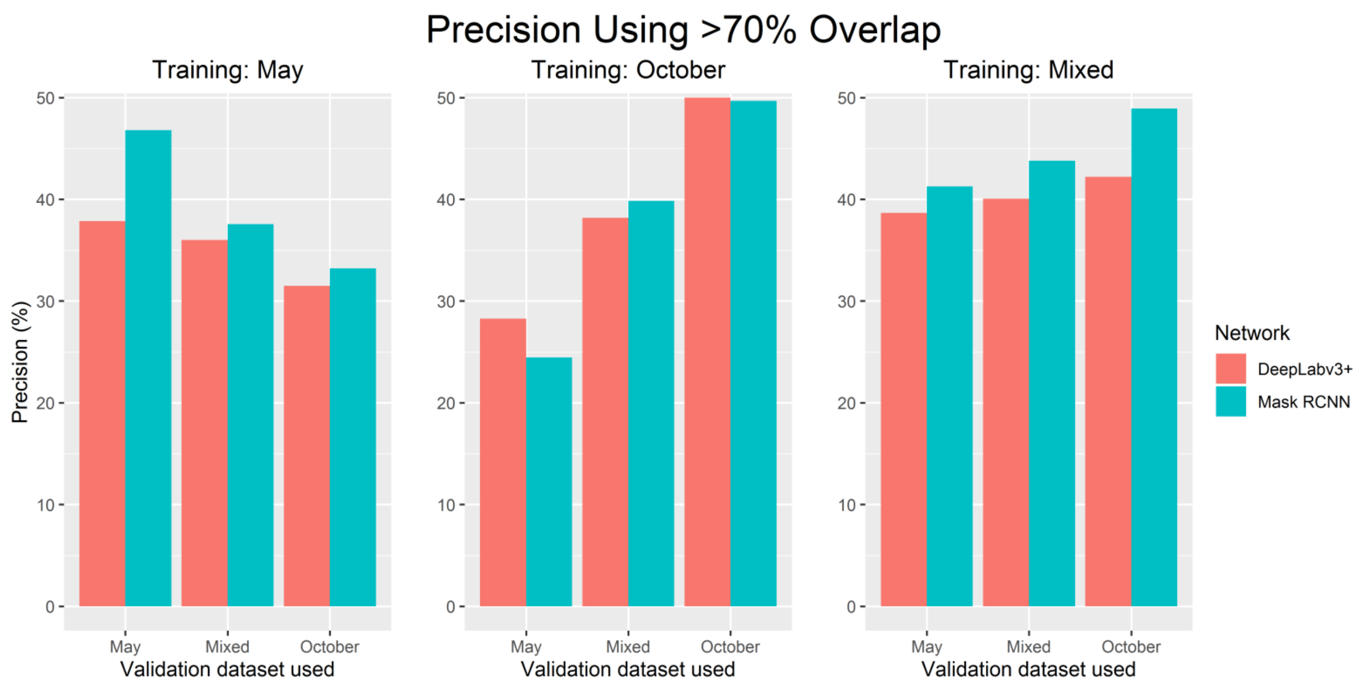


Figure 22 Precision scores for both DeepLab and Mask R-CNN using three different seasonal datasets (May, October, and a mixture of the two) split into training and validation sets. Networks were trained on one of the three datasets and subsequently assessed using each of the three validation datasets.

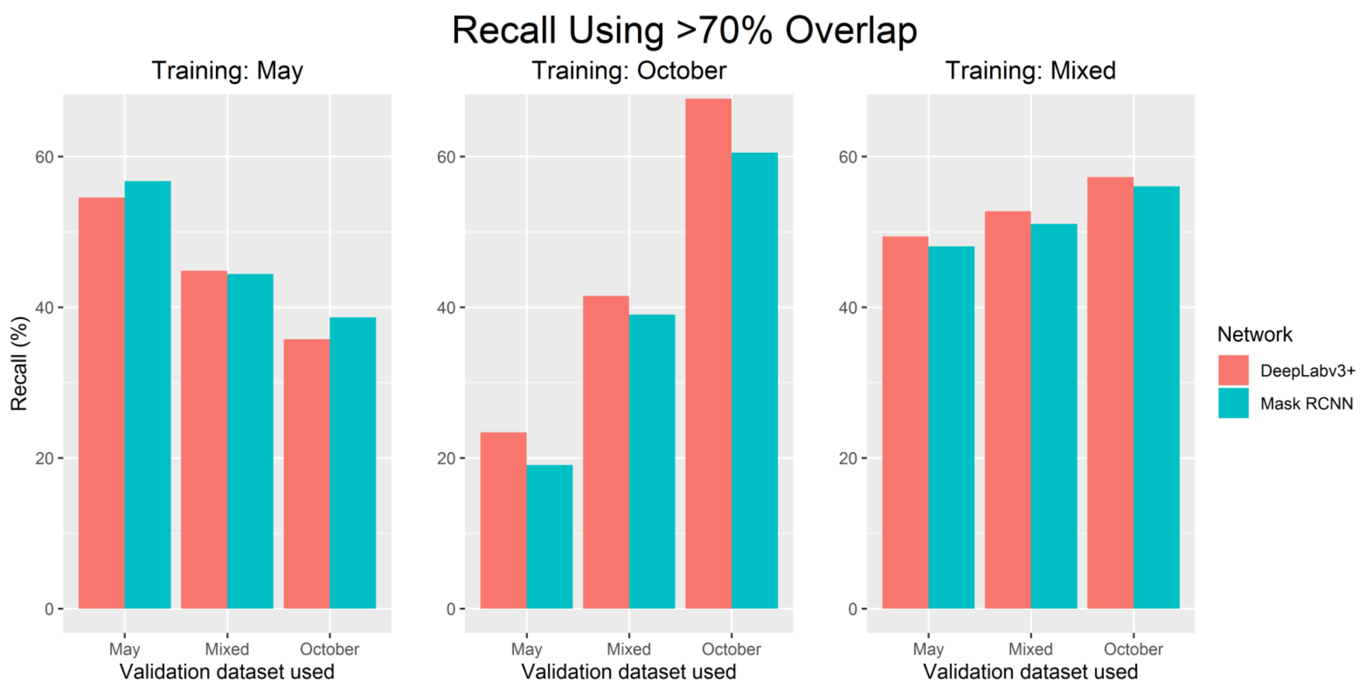


Figure 23 Recall scores for both DeepLab and Mask R-CNN using three different seasonal datasets (May, October, and a mixture of the two) split into training and validation sets. Networks were trained on one of the three datasets and subsequently assessed using each of the three validation datasets.

7.3 Data augmentation

7.3.1 Individual augmentations

Ten different data augmentations were systematically tested to determine which techniques can be applied to hedgerow detection. Test results from all augmentations, for both Mask R-CNN and DeepLab are shown in Fig. 24. Here, the non-augmented dataset (‘none’) is included as well for comparison.

With regards to Mask R-CNN, spectral augmentations (add, noise, blur, add per channel, contrast, and contrast per channel) all achieved improvements over ‘none’ in the per-pixel accuracy of hedgerow predictions, with the addition of noise achieving the highest accuracy (35.8%). With respect to ‘none’, per-pixel accuracies decreased in three of the four geometric augmentations (flip, rotate 45, and scale), with only rotate 90 leading to improvement (32.5%). Precision and recall rates were only examined at the >70% threshold. Precision improved with respect to ‘none’ in three of the spectral augmentations (contrast (48.8%), contrast per channel (48.0%), and noise (48.0%)), decreased in two (add (46.0%) and blur (44.6%)), and remained unchanged using add per channel (46.5%). Precision rates from geometric augmentations increased with respect to ‘none’ in both flip and rotate 45 (49.1% in both) and decreased in scale (45.1%) and rotate 90 (42.3%). Recall improved with respect to ‘none’ in all spectral augmentation datasets aside from add which had the same recall rate as ‘none’ (48.6%). The highest recall from spectral augmentations was achieved by noise (54.4%), followed by add per channel (52.0%) and blur (51.6%), Recall rates decreased in all geometric augmentation datasets when compared to 'none', with the lowest recall coming from rotate 90 (40.3%).

With regards to the DeepLab network, per-pixel accuracies all improved compared to 'none' aside from flip (27.6%). Rotate 90 and contrast per channel both had the highest per-pixel accuracies (33.3%), followed by blur (33.1%) and add (32.4%). Precision rates increased with respect to 'none' in two of the six spectral augmentations (blur (62.6%) and add per channel (61.6%)), as well as in two of the four geometric augmentations (rotate 45 (69.0%) and rotate 90 (67.6%)). The lowest precision from all tested augmentations came from flip (35.5%), contrast (53.6%), and contrast per channel (56.8). Recall rates decreased with respect to 'none' in all but three of the tested augmentations; rotate 45 (63.1%), rotate 90 (62.7%), and scale (59.9%). The largest decreases in recall were seen in flip (49.3%), contrast (50.7%) and contrast per channel (51.4%).

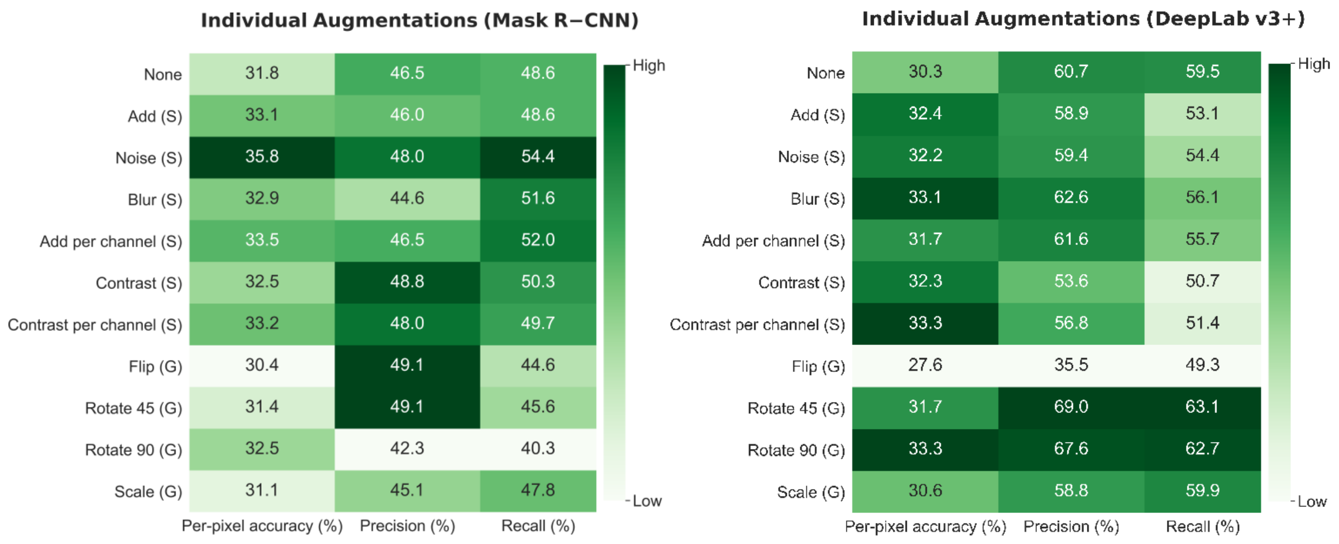


Figure 24 Shows per-pixel accuracy, precision, and recall for all augmented datasets. Precision and recall are calculated using a >70% threshold to determine object detections. Darker colors represent high values within a column, while brighter colors represent lower values within a column. Spectral augmentations are labeled with (S), while geometrical augmentations are labeled with (G). None refers to the original dataset without augmentation.

7.3.2 Grouped augmentations

7.3.2.1 Quantitative results

Results from the fully augmented datasets are shown in table 2. For Mask R-CNN, per-pixel accuracy was higher using only geometric augmentations. At the >70% detection threshold, recall and precision were both higher using both geometric and spectral augmentations (53.6% and 64.6%, respectively). At the >50% threshold, precision was higher using only geometric augmentations (60.3%), but recall was higher using both geometric and spectral augmentations (72.4%). At the >30% threshold this pattern continued, with the highest precision using geometric augmentations (64.8%) but higher recall using both geometric and spectral augmentations (76.1%). Looking at DeepLab, all performance metrics were higher when using only geometric augmentations.

Augmentation	Per-pixel accuracy (%)	Precision (70%)	Recall (70%)	Precision (50%)	Recall (50%)	Precision (30%)	Recall (30%)
Geometric (M)	**40.6**	51.3	59.9	**60.3**	70.3	**64.8**	75.6
Geometric + Spectral (M)	39.5	**53.6**	**64.6**	60.1	**72.4**	63.1	**76.1**
Geometric (D)	**36.9**	**69.4**	**80.8**	**73.9**	**86.1**	**77.6**	**90.4**
Geometric + Spectral (D)	36.4	68.1	79.9	70.9	83.1	73.8	86.5

Table 3 Results of using the fully augmented datasets. Mask R-CNN is denoted by (M) while DeepLab v3+ is denoted with (D). Bolded values represent the highest value of the column for a given neural network.

7.3.2.2 Qualitative results

Figure 25 and 26 show images of some of the detections made by both Mask R-CNN and DeepLab, respectively. In both figures, the top row highlights how augmentations improved mask predictions compared to when augmentations were not applied, the middle row shows cases where the use of geometric and spectral augmentations led to improvements over using only geometric augmentations, whereas the bottom row shows cases where the use of geometric and spectral augmentations was problematic compared to using only geometric augmentations.

With respect to Mask R-CNN, the application of either augmented dataset improved the quality of hedgerow masks. Hedgerow masks in areas of dense hedgerow occurrences were sometimes poorly masked without the added training gained from using augmented images (Figure 25, A). Additionally, more hedgerows were detected using the augmented datasets (Figure 25, B, C), and the network was able to fully detect longer hedgerows after augmentation had been used (Figure 25, C). In the middle row of Fig. 25 is shown how the use of spectral augmentations led to increased detections of hedgerows which were missed when using only the geometric augmentations. However, the generalization that came from using spectral augmentations led to numerous false positive detections, as seen in the bottom row of Fig. 25. In particular, forest edges which neighbored agricultural fields were misclassified as hedgerows (Figure 25, H & J). Non-linear clusters of woody vegetation within agricultural fields were occasionally misclassified as hedgerows when spectral augmentations were added to the dataset (Figure 25, I).

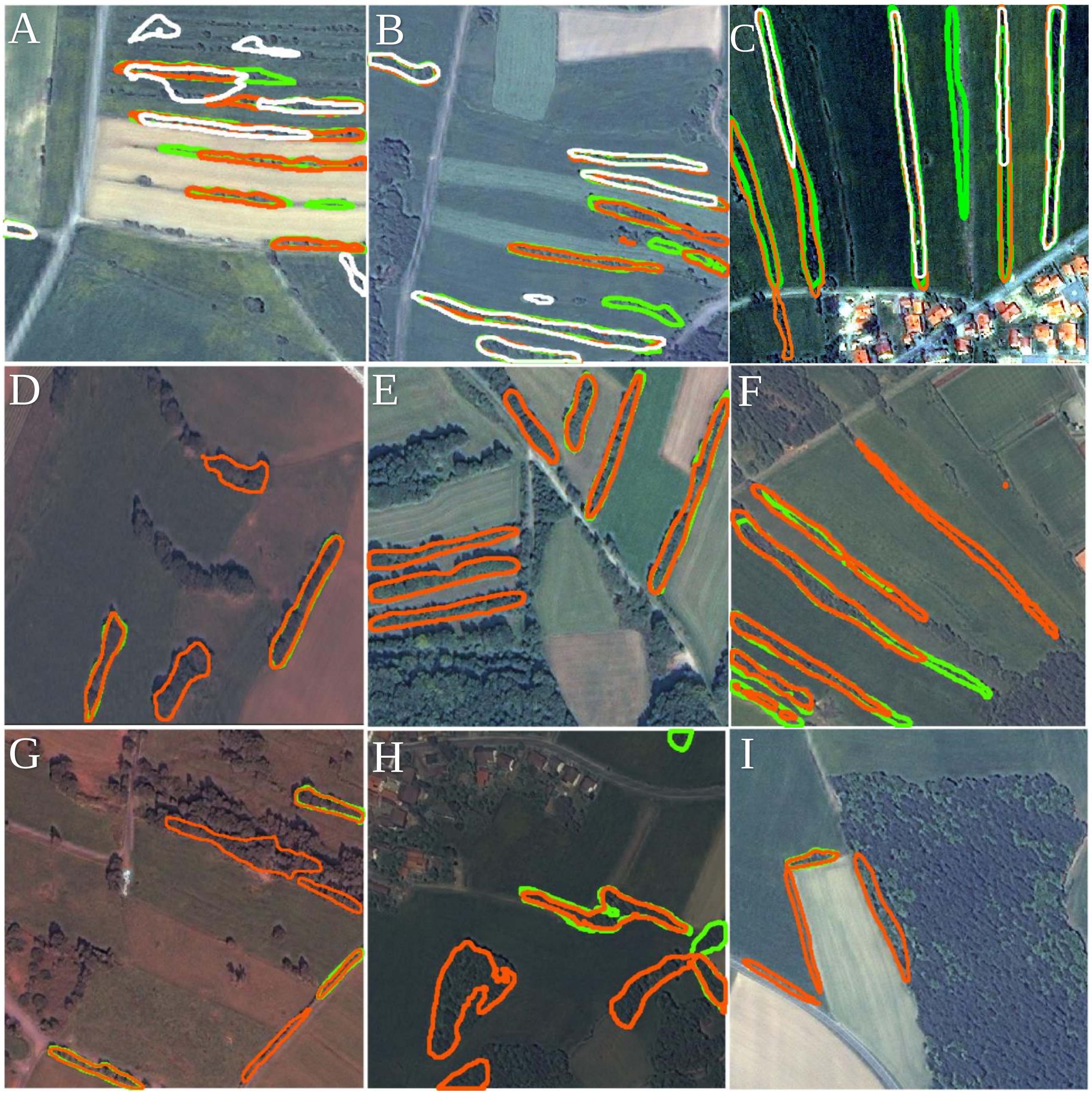


Figure 25 Shows hedgerow predictions generated from Mask R-CNN using different augmentation strategies. Network predictions using the un-augmented dataset are shown in white, while predictions using geometric augmentations and geometric plus spectral augmentations are shown in green and orange, respectively. Tiles are each 320m^2 in size.

With respect to DeepLab, augmentations again improved the quality of hedgerow masks when compared to the predictions made from the network trained on the un-augmented dataset (Figure 26, row 1). Hedgerow mask boundaries were generally more precise when training with the augmented datasets. Without augmentation, the separation of individual hedgerows was problematic, as predictions often included pixels from the field in between hedgerows, especially in areas of dense hedgerow occurrences (Figure 26, B and C). However, in certain areas of dense occurrences, the separation of individual hedgerows still remains an issue even after using augmented images (Figure 26, C). The use of spectral and geometric augmentations compared to only using geometric augmentations had some positive effects with regards to the accuracy of hedgerow prediction boundaries. As can be seen in row two of Fig. 26 (D, F), there were cases where the boundaries of hedgerow masks are more precise when using spectral and geometric augmentations compared to those made from the geometric dataset. However, this was not the case in all image predictions, as can be seen in the third row of Fig. 26. Here can be seen that the use of spectral augmentations can lead to problematic false positive detections within forested areas, in which hedgerows which were detected using only the geometric augmentations are completely missed.

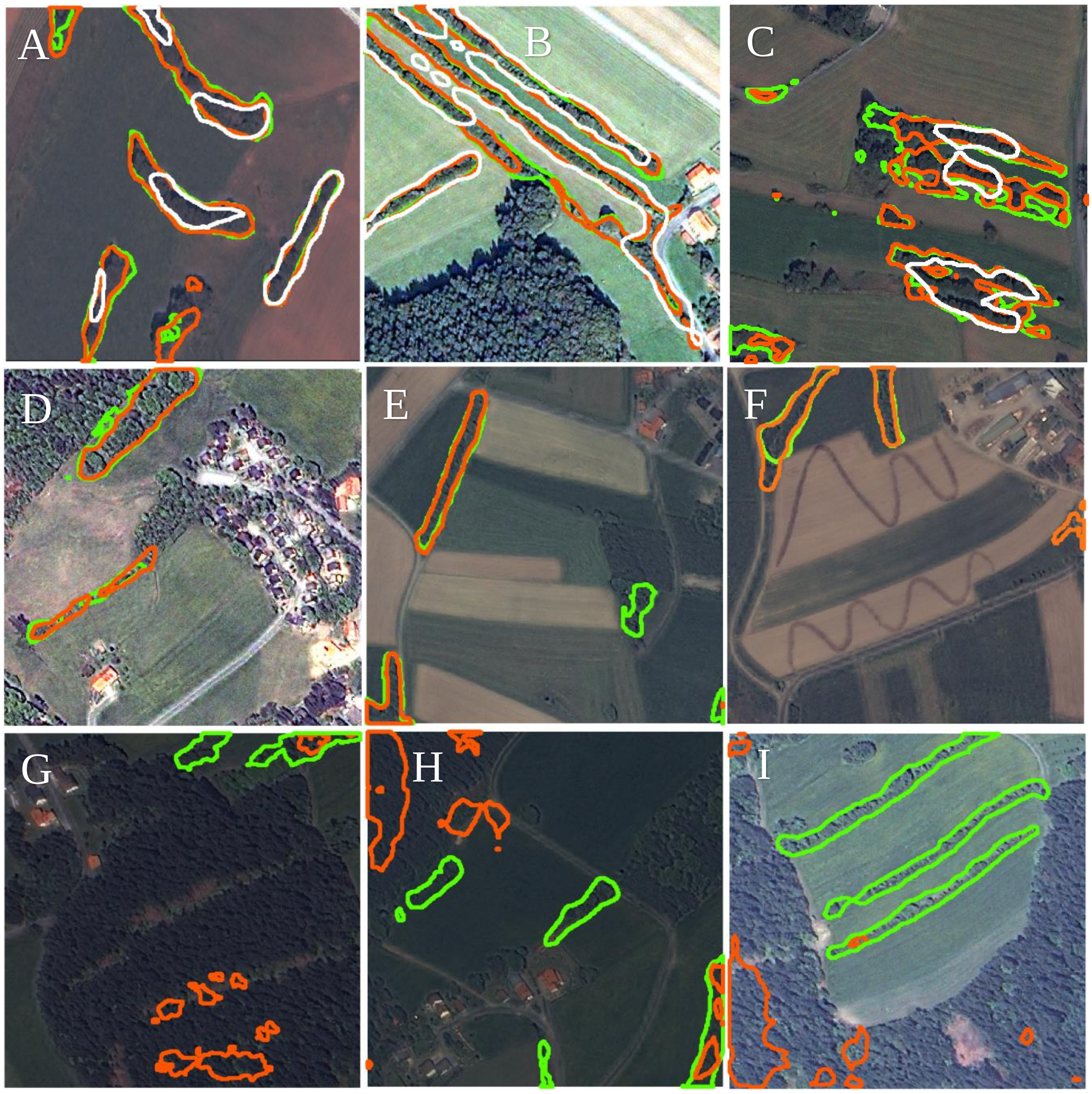


Figure 26 Shows hedgerow predictions generated from DeepLab v3+ using different augmentation strategies. Network predictions using the un-augmented dataset are shown in white, while predictions using geometric augmentations and geometric plus spectral augmentations are shown in green and orange, respectively. Tiles are each 320m^2 in size.

7.4 Comparison between Mask R-CNN and DeepLab v3+

As previous analysis focused on the comparison of different data inputs, an analysis between the two deep learning networks with regards to their predictions is presented here. Based on the outputs of the final augmented datasets, a prediction map was produced from each of the fine-tuned models. For Mask R-CNN, the network trained on geometrically and spectrally augmented images was used, while for DeepLab the network trained on the geometrically augmentation images was used for the prediction map.

From a quantitative standpoint, Mask R-CNN outperformed DeepLab on per-pixel accuracy, while DeepLab had higher precision and recall scores across all detection thresholds (Table 2). Qualitatively, the predictions from Mask R-CNN performed better with regards to the precision of mask boundaries, as the boundaries of mask predictions made by DeepLab were imprecise in most cases (Figure 27). In areas where only small gaps separated hedgerows, DeepLab failed to individually mask each hedgerow, instead adjacent hedgerows were predicted by one singular mask (Figure 27, B). Mask R-CNN struggled the most with the identification of longer diagonal hedgerows, leading of only partial predictions, or complete omission of some hedgerows (Figure 27, A). Predictions made by DeepLab were unaffected by angular orientation.

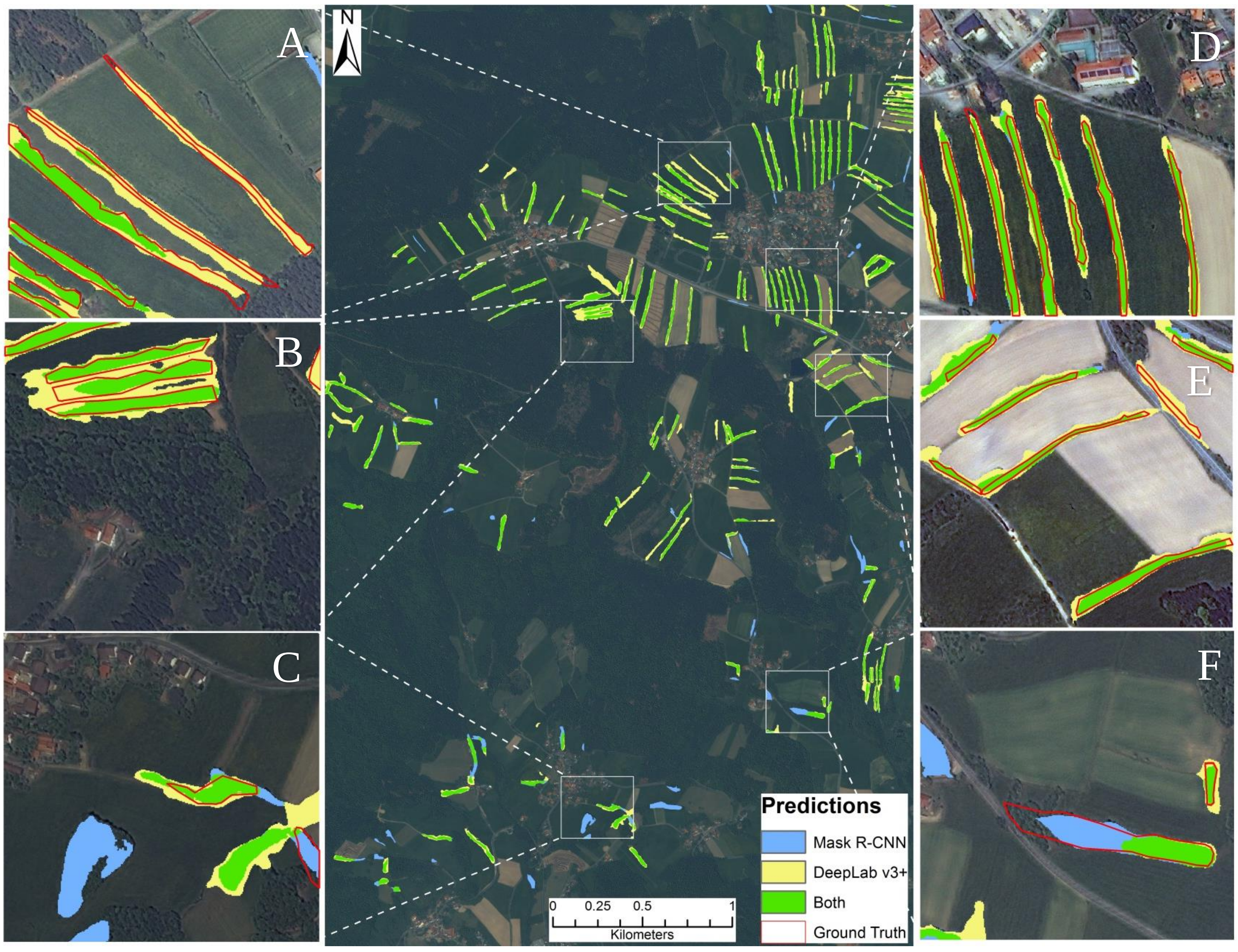


Figure 27 Map showing hedgerow detections at a landscape scale within the study area. Zoomed in areas are of 320m^2 areas.

7.5 Randomly initialized weights

7.5.1 Comparison between augmentation strategies

Given that the use of pre-trained models limits the user to three input bands, DeepLab v3+ was trained from scratch to utilize all four bands of spectral information. The network was trained using each of the augmentation datasets (geometric and geometric plus spectral). Training with data geometrically augmented images led to a per-pixel accuracy of 31.5%, with precision and recall (>70% threshold) of 46.2% and 58.4%, respectively (Table 3). Training with data geometrically and spectrally augmented led to a per-pixel accuracy of 20.9%, while precision and recall were 26.2% and 66.1%, respectively.

Training data Augmentation	Per-pixel Accuracy (%)	Precision >70% (%)	Recall >70% (%)
Geometric	**31.5**	**46.2**	58.4
Geometric + Spectral	20.9	26.2	**66.1**

Table 4 Results of the DeepLab v3+ network trained using randomly initialized weights. Object detections for precision and recall metrics are considered as a true positive when the predicted and ground truth masks overlap by at least 70%. Bold values represent the highest value within a column.

Figure 28 shows images of the predictions made by DeepLab after training from scratch using only geometric augmentations, as well as geometric and spectral augmentations. When the network was trained using only geometric augmentations, predictions were relatively accurate, with good separation of individual hedgerows. The inclusion of spectral augmentations led to increased amounts of predictions, but these predictions also included more non-hedgerow pixels, such as adjacent fields, or urban pixels (Figure 28, row 2). Both networks performed well when predicting scenes containing few woody vegetation objects and when hedgerows were bordered by agricultural fields (Figure 28, row 1). However, the use of spectral data augmentations resulted in misclassifications within urban scenes, as urban vegetation (e.g. garden bushes) was often misclassified a hedgerow. Urban scenes also resulted in some roads being predicted as hedgerows, as well as the inclusion of some rooftops within hedgerow predictions. Images with large amounts of mixed forests also resulted in misclassifications when adding spectral augmentations into the training dataset.

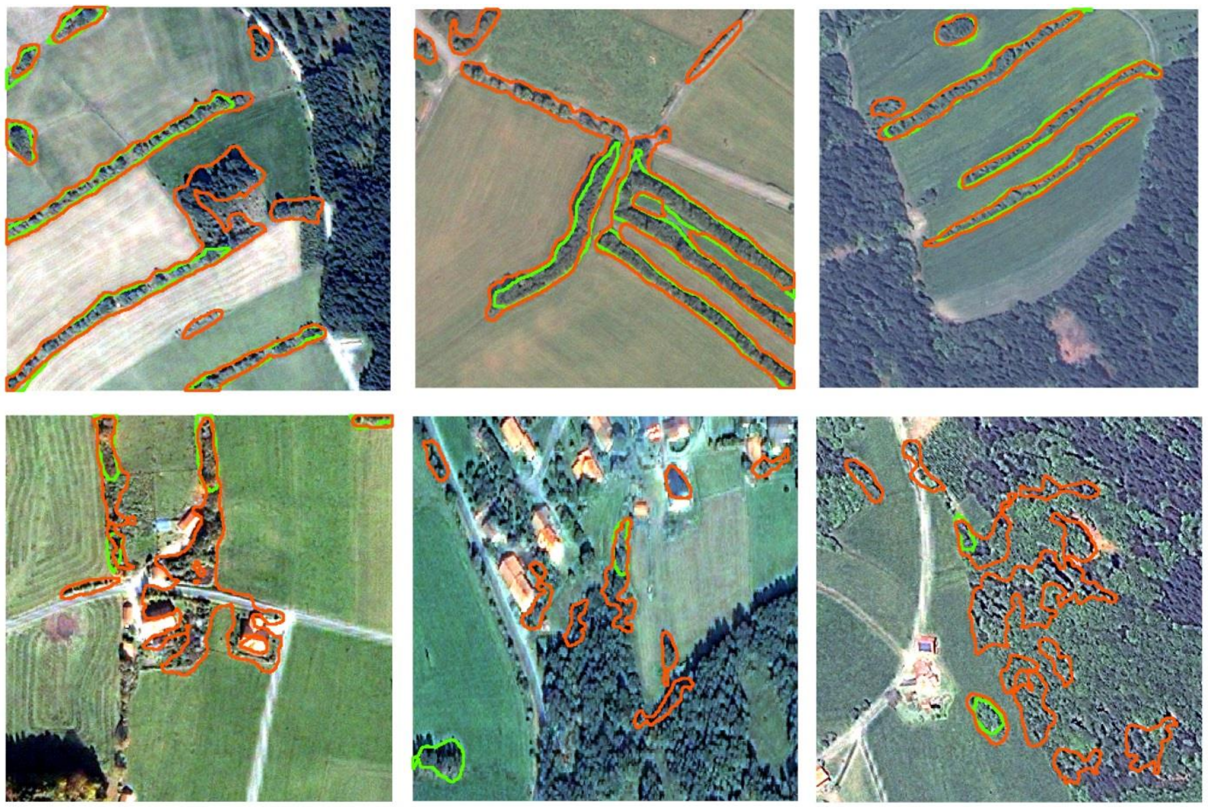


Figure 28 Predictions from the DeepLab v3+ network trained using randomly initialized network weights. Predictions made by the network trained using only geometric augmentations are in green, while predictions made from the network trained using geometric and spectral augmentations are shown in orange. Tiles are each 320m^2 in size.

7.5.2 Comparison with fine-tuned networks

As the predictions from the geometrically augmented dataset obtained the better results when training from scratch, the predictions from this model were compared to those made by the top pre-trained networks from Mask R-CNN and DeepLab (Figure 29). The top row of Fig. 29 shows predictions made by the pre-trained Mask R-CNN which was fine-tuned using geometric and spectral augmentations, while the middle and bottom rows show predictions made by the pre-trained DeepLab fine-tuned using the geometrically augmented dataset, and DeepLab trained from scratch using the geometrically augmented dataset, respectively. By training from scratch, the network was able to predict hedgerows of varying angular orientations which Mask R-CNN missed (Figure 29, first column). The network trained from scratch was also able to perform object masking with more precise mask boundaries than the fine-tuned DeepLab. However, some areas of dense hedgerow occurrences were difficult for the network trained from scratch to mask correctly, although the mask boundaries were still typically an improvement in these cases over the other two fine-tuned networks (Figure 29, third column). In some areas though, the network trained from scratch had difficulties with fully masking detected hedgerows and also missed more hedgerows than the two pre-trained networks (Figure 29, fourth column).

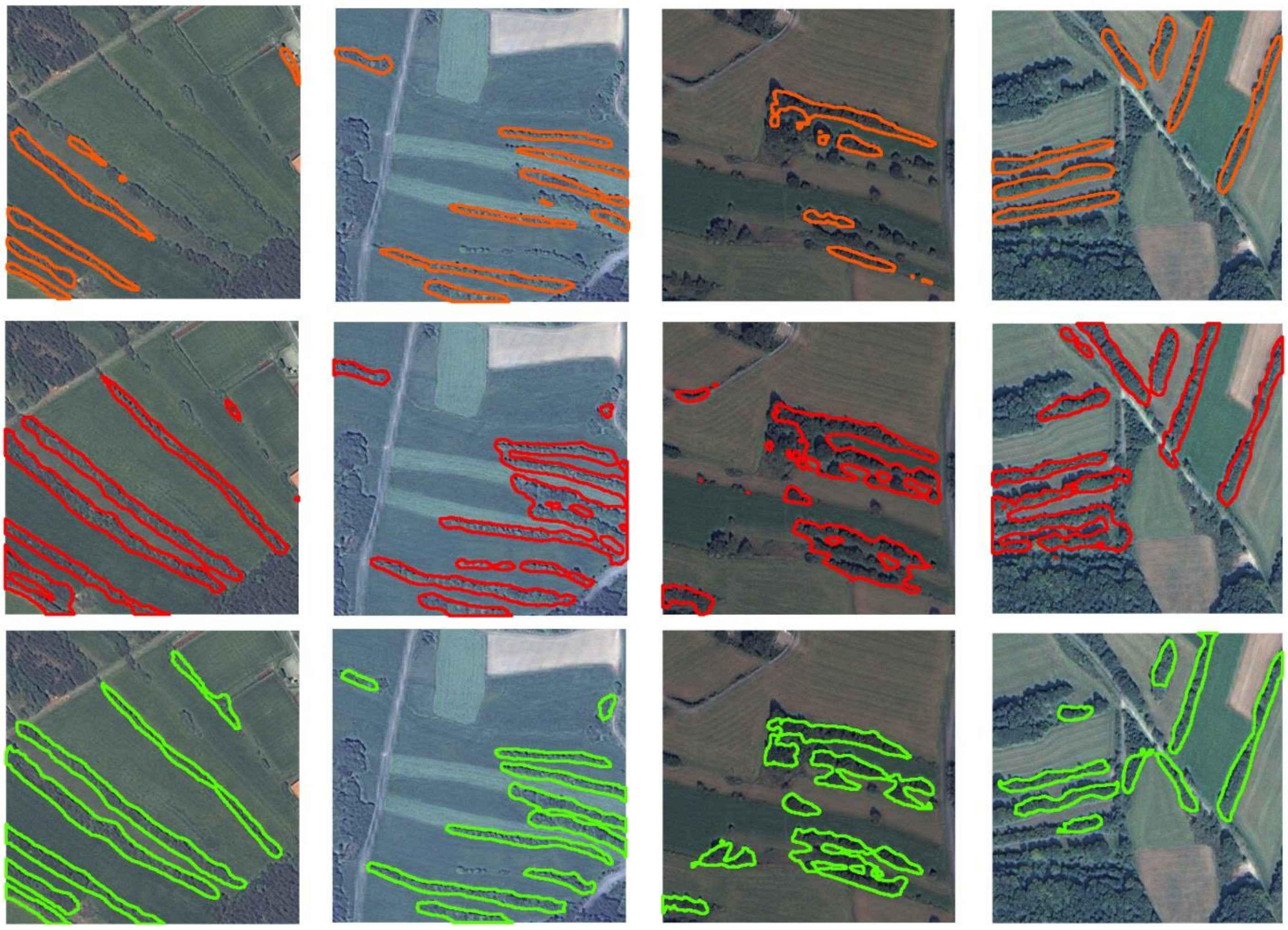


Figure 29 Shows the predictions from the best performing Mask R-CNN (top row, orange) and DeepLab (middle row, red) networks fine-tuned from pre-trained weights, and the DeepLab network trained from scratch (bottom row, green).

7.6 Final predictions map

Given that the pre-trained DeepLab network fine-tuned using geometric augmentations produced the highest performance metric scores, this network was used to make predictions across the entire area of satellite scenes which were available for Freyung-Grafenau (Figure 30). The network was able to make hedgerow predictions across a large spatial scale, with predictions being made for image scenes which had not been included in the training or validation dataset due to the restricted coverage of the available ground truth data. Predictions across the 562km^2 sized area were obtained within 30 minutes, as an entire IKONOS scene was first loaded, and 320m^2 tiles were extracted on the fly and processed within roughly one second per tile.

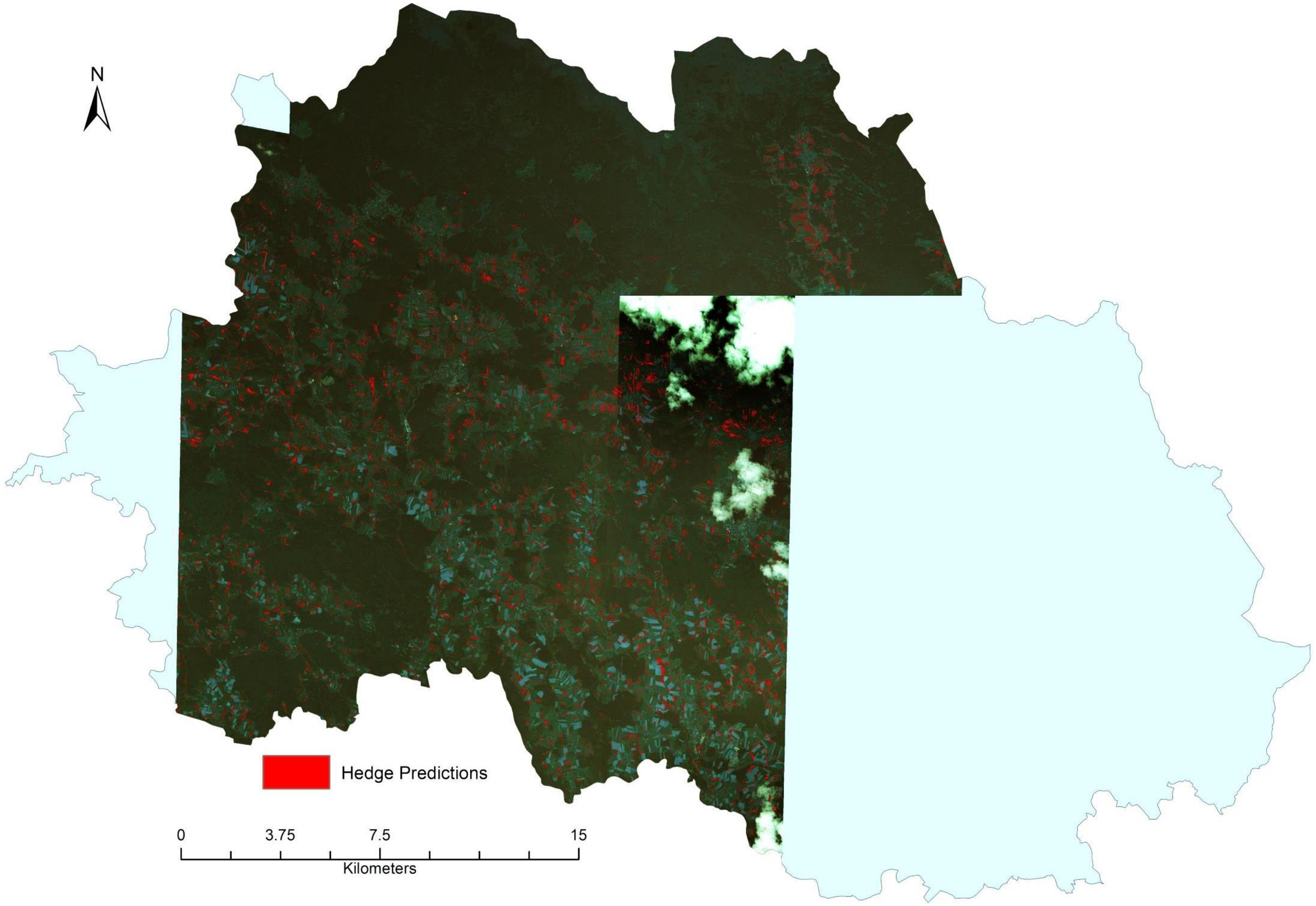


Figure 30 Map showing the distribution of hedgerow predictions across the entire study area.

8 Discussion

The main restriction of NN seems to be the amount of training data available. Given that data annotation can be expensive, deep learning can help to provide larger datasets through iterative training (Tong et al., 2019). Here, the network is trained first on the set of annotated images. Afterwards, the network predicts the target class within images without annotations. The predictions must go through some manual quality assurance to check which images have been correctly annotated, and these images are then added to the training pool. The network can then be trained further on the increased dataset, leading to an iterative process where annotated images are generated. Such as approach could eliminate the work of manually annotating images. With Mask R-CNN, confidence levels can also be set, so that only the prediction which the network is highly confident in (e.g. 95%) are kept, which would help ensure annotation quality.

8.1 Band combinations

The GRNIR band combination was found to be the optimal three band combination for detecting hedgerows. Other researchers have also shown that green, red and NIR bands are the most correlated to vegetation parameters (Baloloy et al., 2018; Price, Guo, & Stiles, 2002), and these are also the three bands used for making false color composite images which are popular for quick human identification of vegetation within a scene (Baeck et al., 2016). However, GRNIR did not perform the best across all performance metrics or across all levels of detection. Using DeepLab, GRNIR had lower recall rates across different detection thresholds compared with BGNIR and BRNIR, while using Mask R-CNN, precision rates at 50% and 30% thresholds were higher using BGNIR and RGB as input, respectively. Here the emphasis was placed on the detection of hedgerows with a minimum of 70% of the hedgerow correctly masked, as this gives a more accurate map in terms of hedgerow coverage. If complete delineation of hedgerows is of less concern and partial detection is sufficient, then other band combinations may be more desirable. Thus, the decision on band combination should be made depending on the desired outcomes of the project.

The fact that BGNIR had the lowest precision in 70% threshold for Mask R-CNN but highest precision rates at lower thresholds suggests that the band combination does offer good precision but has trouble fully masking hedgerows. Further experiments using augmentations showed that spectral augmentations can help detections occur at higher thresholds. Thus, further experimentation with three band combinations after having augmentations applied could lead to different results with respect to the optimal band combination. Here only the original dataset without augmentation was used to test the performance of band combinations due to faster training times of smaller datasets.

The good performance of RGB implies that images taken using RGB cameras could also be applied for hedgerow monitoring. RGB cameras have a long history of use in aerial photography as they produce images which are easily interpretable at a relatively cheap cost (Zheng et al., 2018; Pepe, Fregonese, & Scaioni, 2018). Given that some organizations use RGB imagery for the manual digitization of hedgerows (Dover, 2019), these flight campaigns could still be useful for hedgerow monitoring. Aerial imagery tends to be of higher spatial resolution (Ruwaimana et al., 2018; Lehmann et al., 2017) which may be of greater benefit than the addition of the NIR band, as research has shown that increased spatial resolution leads to improved results in CNNs (Jozdani, Johnson, & Chen, 2019; Farooq, Hu, & Jia, 2018). The results obtained in this work also support this notion, as the pre-trained networks using three bands as input outperformed the network trained from scratch using four bands, thus showing the addition of a fourth band can lead to decreased performance. Mask R-CNN would likely be the better candidate for the use of RGB aerial imagery as performance metrics between the different band combinations showed less variation compared to using DeepLab, suggesting Mask R-CNN to be more robust concerning band combination.

8.2 Seasonal inputs

Given that the dataset of IKONOS imagery contained scenes from both October and May, hedgerows were characterized by different spectral features due to differences in phenological phases between May and October. Thus, the effect of seasonal input was investigated in order to determine which time frame is best to obtain hedgerow images, as well as to investigate whether the use of mixed seasonal input images would hinder network performance.

Images from October provided the best performance for hedgerow detection. This is likely due to the unique spectral characteristics captured during this time of year, as hedgerows are typically a combination of multiple bush and tree species (Forman & Baudry, 1984). Hedgerows should thus contain more heterogeneous spectral features due to greater variation in the phenological patterns between different species compared to forests and fields which are typically more homogeneous, being dominated by one or a few species (Ghiyamat & Shafri, 2010, Hycza, Sterenczak, & Balazy, 2018).

While images from October obtained the highest ratings for performance metrics; using the mixed seasons dataset during training and October images for evaluation performed only a few percentages worse than using the October dataset in both stages. This supports the use of mixed images for network training, as predictions were not significantly harmed. Given that the comparison made here controlled for the size of datasets, the increased size garnered from using imagery from multiple seasons instead of a smaller single season dataset should lead to performance increases which outweigh the decreased performance of using mixed rather than October images.

The inclusion of imagery from varying phenological time periods can be conceptualized as a type of data augmentation, in that the network should have improved generalization by learning of a variety of hedgerow spectral characteristics caused by both phenological cycles as well as differences in reflection intensities due to differences in sunlight incidence angles across the seasons (Liu et al., 2018b). Thus, multi-season datasets should lead to more robust network predictions across new prediction data, as well as across gradients of elevation where phenological stages can differ over spatial scales (Vitasse et al., 2009; Ziello et al., 2009).

The final observation from these experiments are the differences in the performances between the two NN networks based on seasonal inputs. The performance of Mask R-CNN varied less when using either October or mixed seasonal inputs, while DeepLab showed larger losses in performance when using the mixed seasons data compared to the October dataset. This suggests that Mask R-CNN is more robust across seasonal data inputs, and thus may perform better across different test sites where spectral features of hedgerows may differ.

8.3 Data augmentation

8.3.1 Effects of individual augmentations

The effects of numerous geometric and spectral augmentations were individually examined and compared to another. When looking at geometric augmentations, both networks performed poorly when images were augmented through random flips in the image (vertical or horizontal). This could be because the flipping of hedgerows does not add as much new information to the dataset since hedgerow positions are simply mirrored. This may lead to overfitting towards hedgerows of a certain orientation and spectral features. Thus, for the detection of relatively symmetrical objects which appear similar when mirrored, the addition of flipped images may not be as beneficial as other augmentations. The results of the remaining three geometric augmentations (rotation of 45°, 90°, and scale) were mostly dependent on the NN, as Mask R-CNN showed lower rates of recall and per-pixel accuracies when applying these three, while DeepLab showed improvements in all performance metrics. The use of rotations produces a greater variety of hedgerow orientations, providing the network more opportunity to learn a variety of orientation features. However, the negative effects seen in Mask R-CNN may stem from the network continually learning the same spectral features, leading to overfitting to certain spectral features. This seems to be the case, as all individually tested spectral augmentations led to increases in at least two of three performance metrics for Mask R-CNN.

Overall, it seems that the spectral augmentations helped to increase the number of true positives while simultaneously reducing the number of false negatives in predictions made by Mask R-CNN. The addition of image noise led to the greatest improvements. Previous research has found the addition of noise to have negative effects on network performances (Dodge & Karam, 2016), finding higher rates of misclassification. Here however, it seems that the opposite is the case, as the addition of noise likely allows the network to generalize across a greater variance of spectral features possessed by hedgerows, or to detect hedgerows using non-spectral features (e.g. orientation, contextual). Dodge & Karam (2016) found that blurring images has negative effects on network classification accuracy as well. This was partially the case here as well, as although the recall did increase using blur, the precision of predictions decreased. This trade-off between recall and precision implies that the inclusion of blurred images depends on the desired output of the user. Other spectral augmentations (add, contrast per channel) showed relatively unchanged recall and precision, yet an increase in per-pixel accuracy. This suggests that spectral augmentations lead to more precise mask boundaries by forcing the network to learn contextual and morphological features, rather than relying on spectral features which are often similar to surrounding fields and forests.

While Mask R-CNN obtained improved results when applying spectral augmentations, DeepLab performed poorly when trained with spectrally augmented images. Recall decreased across all tested spectral augmentations, while precision decreased in all but two spectral augmentations (blur and add per channel). The largest decreases in both precision and recall came from the contrast and contrast per channel augmentations. Given that these two augmentations caused changes to pixel values by the largest scale (scaled by two) suggests that training images with pixel values which deviate largely from the true values can lead to poor predictions from DeepLab.

In cases where hedgerow predictions were TP, both networks produced more precise mask boundaries using spectrally augmented training images. This effect of increased boundary precision when using spectral augmentations has been found by Stiller et al. (2019), where the application of a hue transformation increased the precision of urban building masks. However, the use of spectrally augmented images did introduce more FP detections, especially when using DeepLab. Ma et al. (2019b) found the opposite effect, as DeepLab performed more accurate object detections when augmenting the spectral values of RGB images for goat detection. This conflicting result suggests that the application of spectral augmentations may be better suited to detection tasks where spectral characteristics are less of a defining feature of the target objects (e.g. buildings, animals).

8.3.2 Effects of applying multiple augmentations

After testing augmentations individually, the effect of applying all augmentations was investigated by training the networks on two different datasets, one containing images augmented using only the geometric augmentations (*G*), and one containing images augmented using both geometric and spectral augmentations (*GS*). Overall, the increased size of these two datasets compared to the original un-augmented dataset (UA) led to large increases in performances for both Mask R-CNN and DeepLab.

Using Mask R-CNN, precision rates at the >70% threshold were highest using *GS*, while precisions at both lower thresholds were higher using *G*. In addition to this, while *GS* had higher recall rates, the difference between the recall rates of the two augmentation strategies decreased as the threshold for detection decreased. These results suggest that the addition of spectral augmentations helped make the network more generalizable, leading to the acceptance of more hedgerow pixels into mask predictions and masking hedgerows more completely (thus the increased rates of precision and recall at the 70% threshold). However, given that recall rates were more similar at lower detection thresholds implies that the use of spectral augmentations did not lead to a large increase in overall hedgerow detections, but rather that hedgerows detected only partially by the network trained with *G* were masked more fully using *GS*. The higher precision from *GS* decreased at the lower thresholds, leading to more precise predictions using *G*. This is because while more hedgerows were detected using *GS*, false detections also increased. Given the trade-off between recall and precision between the two augmentation strategies, the decision to incorporate spectral augmentations into the training data depend on the desired outcome of predictions.

With respect to DeepLab, the experiments incorporating multiple augmentation strategies into the training dataset reinforced the conclusions from the individual augmentation experiments. Here, the inclusion of spectral augmentations from *GS* led to decreases in model performance compared to training with *G*. Although the performance metrics show only a few percentages of difference between the two augmentation strategies, visual inspection of the predictions showed the problem of applying spectral augmentations. Here it was found that some large forested areas were misclassified as hedgerows. Objects typically detected as false negatives from other trained networks shared some contextual or morphological characteristics with hedgerows (e.g. neighbouring agricultural fields, linear shape). However, the addition of spectral augmentations led to the inclusion of false detections which possessed few hedgerow characteristics. These predictions were not neighboring any open fields and were devoid of any linear shape, appearing seemingly random in shape. This goes against the notion that spectral augmentations lead to improved learning of morphological features (Stiller et al., 2019; Ma et al., 2019b). However, in the case of TP detections, prediction masks were often improved with respect to mask boundary precision when using *GS* compared to *G*. Thus, the results on the use of spectral augmentations with DeepLab were mixed, prompting the need for further investigation into whether the application of more conservative spectral augmentations could result in improved mask predictions.

One possible counter to the negative effects of spectral augmentations could be training with *GS* at first, followed by a fine tuning of a few epochs with *G*. This should lead to a learning of morphological and textural features from the spectrally augmented images in the first stage, followed by a learning of the appropriate spectral features there-after. Additionally, since some of the individually tested spectral augmentations did perform better than others, it is possible that applying only the best spectral augmentations could have been more beneficial. This effect was found in Ma et al. (2019b), where although all augmentations resulted in improvements over their un-augmented dataset, the application of only the highest-ranking augmentations led to higher performance than when all augmentations were applied. However, due to time constraints further experimentation with combined spectral and geometric augmentations was not possible.

8.4 Randomly initialized weights

Using a pre-trained network and fine-tuning it with satellite imagery was compared to training a network from scratch. When performing network training from scratch, large batch sizes are important to properly train the batch normalization layers (Masters & Luschi, 2018). As the Mask R-CNN source code did not allow for multiple GPU processing, batch sizes were restricted by the memory of a single GPU and thus only DeepLab could be trained in this manner.

Although the pre-trained network (*PT*) outperformed the network trained from scratch (*TS*) with regards to the performance metrics, *TS* did show promising results. For example, *TS* performed far better with regards to mask boundary precision than *PT*. With the *PT*, clusters of hedgerows were often masked with one continuous polygon. With *TS*, DeepLab was able to successfully separate hedgerow instances in many cases. In areas where *TS* masked hedgerow clusters with a single polygon the mask boundaries were still more precise than the masks generated for the same area from *PT*. This suggests that the features learned by a network pre-trained on natural images may not be optimal for remote sensing images. This is likely due to the difference between remote sensing images and natural images, as it has been found that features from pre-trained networks are less transferable given larger differences between the source (e.g. natural images) and target (e.g. remote sensing) images (Chu et al., 2016; Yosinski et al., 2014).

Reasons why pre-trained weights outperformed weights learned from scratch are most likely related to differences between the two training datasets. First, the hedgerow dataset may have been too small to properly train the network from scratch. This has been found in other research where datasets ranging from 8,000 – 81,000 were found to be too small for training from scratch (Tajbakhsh et al., 2017; Chu et al., 2016; Cadene, 2016). Additionally, not just the number of images, but also the number of target objects within each image may influence training. The pre-training dataset COCO contains a total of 2.5 million target objects across all images (Lin et al., 2014), whereas the hedgerow training dataset contained only 1,383 hedgerow objects. Furthermore, the quality of the dataset annotations could have also led to poor performance, as hedgerows were not always outlined with great precision. On the other hand, the COCO dataset incorporated numerous quality assurance measures to ensure high quality annotations of object masks (Lin et al., 2014). One recent study looking at the issue of annotation quality found that while pixel precise polygon masks performed best, the use of rough masks, such as object bounding boxes, still managed to predict accurate object masks. The main issue they found with the use of object bounding box annotations was that it led to increased amounts of false positive pixels (Mullen, Tanner & Sallee, 2019). It may often be the case that remote sensing data is only roughly annotated as the cost of producing precisely digitized object polygons can be high (Mullen et al., 2019), thus potentially limiting the performance of NN in such cases. Future work could further study this potential effect by comparing networks trained using annotations with high and low precisions. Lastly, the COCO dataset may be better suited for network training as it includes occluded objects, forcing the network to learn improved contextual features (Lin et al., 2014). The inclusion of contextual information has been shown to improve network performance, as Liu et al. (2018) found that including class annotations for all pixels rather than only labelling target classes improved semantic segmentation results. The authors used very precise class annotations (e.g. labelling shadow pixels), which across large spatial scales would be infeasible due to the work required to manually label with such detail. However, freely available raster layers (e.g. CORINE land cover) could potentially be used as labels for the background classes. While such layers may often be more general with regards to class labels, it would be interesting to see if such data could improve results.

Here, only spectral images were used as inputs. However, given the possibility of training from scratch, more input layers could improve the accuracy of the method. Layers such as point cloud data (e.g. light detection and ranging (LiDAR) systems) should help to eliminate any false positive detections which occurred from linear features within fields (e.g. ditches), as well as potentially reducing forest detections, as structural features typically differ between forests and hedgerows (Schlinkert et al., 2016; Wehling & Diekmann, 2009). However, the coverage of such data is often limited and may not be available across larger spatial scales (O'Connell et al., 2015)

8.5 Overall performances of Mask R-CNN and DeepLab v3+

False positives were typically made on objects that resemble hedgerows, or vegetated areas directly neighboring fields (Appendix I). Research has found that species compositions tend to be similar between hedgerows and forest edge areas (Herlin & Fry, 2000; Forman & Baudry, 1984), which may lead to higher rates of misclassifications of forest edges as hedgerows. The per-pixel accuracy metric was partially affected by hedgerow annotations not always fully covering the hedgerow pixels (Appendix I), leading to correct pixel masks being penalized and introducing a negative bias in this performance metric. LfU defined hedgerows as linear structures comprised of a mixture of bush and tree species, being a minimum of two years old. Thus, numerous linear vegetation structures which appear similar to hedgerows from a satellite (e.g. tree lines, young hedgerows) were often misclassified as hedgerows, leading to reduced precision rates. However, such features would likely be just as difficult for a human observer to correctly label given the same image, and thus the only way to improve such misclassifications would be through field surveys.

This study found that the FCN network DeepLab outperformed Mask R-CNN in both precision and recall measures. To date only one other study has been carried out performing instance/semantic segmentation for vegetation objects (Zhao et al., 2018). In this study, Zhao et al. (2018) compared Mask R-CNN to an FCN network known as U-Net (Ronneberger, Fischer, & Brox, 2015) for detection of pomegranate trees. Their results achieved much higher rates of precision and recall for Mask R-CNN than achieved in this paper. However, this was likely due to the simplified detection task of Zhao et al. (2018), as training and validation images were comprised of only the target trees against a bare soil background. On the other hand, images used for detecting hedgerows often contained instances of wooded vegetation which the network could potentially confuse as hedgerows (e.g. bushes, forests, tree lines). Additionally, hedgerow objects were typically located within spectrally similar areas (e.g. pasture fields), making the separation of hedgerows from the background more difficult. Zhao et al. (2018) found that Mask R-CNN outperformed U-Net on both precision and recall, whereas we found DeepLab performed better than Mask R-CNN. This is likely due to U-Net being an older FCN architecture, which lacked the use of atrous convolutions. This suggests the importance of the use of atrous convolutions within NN architectures used for remote sensing image analysis, as they provide increased ERF without downsampling the spatial resolution of the input (Chen et al., 2017).

Zhao et al. (2018) also found the predicted masks of Mask R-CNN to be more precise, as the FCN mask predictions would span over multiple individual objects in areas where objects were densely clustered. This same result was also found here in the case of hedgerows, as DeepLab also had troubles with separating object masks when individual objects were close together, leading to a single mask spanning several hedgerows. This effect can be attributed to the large scale upsampling performed by FCN networks such as DeepLab and U-Net (Marmanis et al., 2017; Chen et al., 2016). While DeepLab performs one less stage of downsampling than Mask R-CNN due to the use of atrous convolutions, Mask R-CNN employs an FPN which allows for fine-scale objects to be masked using feature maps with spatial dimensions closer to the original input. Thus, fine-scale objects such as hedgerows are masked using lower FPN feature layers, thus mitigating the effects of downsampling without the loss of higher-level semantic features typically only available in low resolution layers. One additional advantage of the FPN is that it performs upsampling at a slower rate (rate of 2) compared to DeepLab (rate of 4). Slower up-sampling rates have been found to lead to better mask outputs (Long et al., 2015) and allow for more residual connections during upsampling steps.

An inability to separate hedgerows with individual masks can have negative effects on monitoring programs, as the number of hedgerows would be underestimated. Additionally, the loss of hedgerows in areas of dense occurrences may not be detected, as a mask containing multiple hedgerows may only decrease in size rather than a decrease in the total number of hedgerow masks. Given that hedgerow structural features, such as width, influence mammal and avifaunal species abundance and richness (Schlinkert et al., 2016; Padoa-Schioppa et al., 2005), the accurate delineation of hedgerow masks would be important if used for certain environmental models.

One way to increase the precision of mask boundaries in FCN networks would be to reduce or eliminate downsampling operations from the network architecture. Given that atrous convolutions are used for the purpose of reducing downsampling, the increased use of such layers is a likely solution. One concern stated by Chen et al. (2018) regarding designs employing large amounts of atrous convolutions was the increased computational requirements, due to the increasing number of feature layers generated as the network becomes deeper. However, Liu et al. (2019) proposed a network which successfully applied only atrous convolutions by constraining the number of feature layers at multiple points within the network architecture. This was done using 1x1 convolutional layers as bottlenecks which summarize the information across many feature maps (e.g 256) into fewer feature maps (four) after each convolutional block in the network. They thus exploited atrous convolutions to provide spatial and contextual features without the need for any down-sampling, leading to results which outperformed DeepLab v3+ for cloud detection in remote sensing images.

Mask R-CNN struggled to detect and fully mask long diagonal hedgerows. Here, there appear to be two main issues. The first is caused by the interaction between the RPN and FPN of Mask R-CNN. Diagonal hedgerows require larger anchor bounding boxes to encapsulate them (Figure 31), leading to large RoIs. These RoIs are then assigned to the top layers of the FPN characterized by coarse spatial resolution. However, hedgerows are fine-scaled objects, comprising a small area within these large bounding boxes. Research within object-based image analysis which incorporates multi-scale feature pyramids have found that fine-scale objects are best classified using fine-scale feature layers, while coarse feature layers are best for classifying large objects (e.g. forests), (Zhou et al., 2019; Blaschke, 2010). Thus, hedgerows require fine-scale spatial resolution feature layers to be accurately classified and masked, such as those at the lower levels of the FPN. The second issue with detecting long diagonal hedgerows is that large class imbalances occur within RoIs of diagonal hedgerows as much of the area is dominated by background pixels.

A solution to both issues would be the use of rotational anchors, allowing for anchor bounding boxes with dimensions which share a relationship with the physical size of the object (Wen et al., 2019; Li et al., 2018; Azimi et al., 2018). Rotated anchors would thus reduce RoI scales, in turn mapping objects to more appropriate layers within the FPN, as well as reducing the amount of background pixels within the bounding box. Rotational anchors could also improve hedgerow detections within areas of dense hedgerow occurrences, as Mask R-CNN also struggled in such areas. This is because Mask R-CNN assumes an anchor contains a singular instance of a target object (Liu, Pan & Lei, 2017). However, in cases of densely occurring objects, anchors are unable to separate individual objects. This is most often the case when dealing with angular hedgerows, as their large bounding boxes often contain neighboring objects (Figure 31).

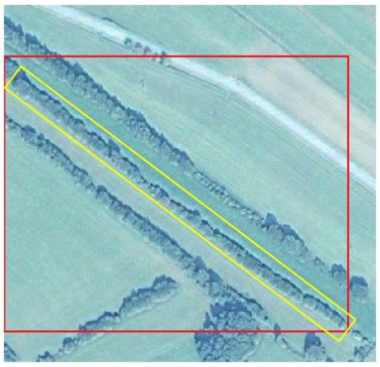


Figure 31 Shows an anchor bounding box without (red) and with (yellow) rotation for a single hedgerow.

8.6 Comparison with past object-based studies

Past studies typically used pixelwise measures of producer's and user's accuracy while here object-based measures of precision and accuracy were used. However, precision ($\frac{TP}{TP+FP}$) and recall ($\frac{TP}{TP+FN}$) are measured in the same sense as user's and producer's accuracy, respectively. Thus, a comparison between results of this study and past studies is possible.

This study managed to achieve precision of 69% and recall of 81%. Bock et al. (2005) achieved 77% producer's accuracy and 70% user's accuracy. Thus, our method outperformed that of Bock et al. (2010) by 4% with respect to producer's accuracy and underperformed by 1% on user's accuracy. Bock et al. (2010) used higher resolution imagery of Quickbird (0.7m) as well as a DSM model for the classification. The sample size for the study of Bock et al. (2010) was however very small, covering a 9km^2 area having only nine ground truth hedgerows. Their classification was based on multiple rules, which may not transfer well across study sites (Aksoy et al., 2010). The study by O'Connell et al. (2015) achieved 77% for both producer's and user's accuracies. Thus, they performed 6% higher than our study with respect to producer's accuracy, and 3% lower than our study with respect to user's accuracy. Here they used a RF classifier and 0.5m resolution RGB+NIR images. Thus, the higher resolution of the imagery likely improved the results of their method. The classification method of O'Connell et al. (2015) required the production of auxiliary layers (NDVI, EVI, Canny edge), and the assessment of 60 additional feature variables (e.g. Grey level co-occurrence matrix, hue, saturation, lightness (HSL) transformation) in order to find the optimal classification setting. Additionally, their study was limited to a small landscape area of 12km^2. Ducrot et al. (2012) reported a 95% rate of correctly detected hedgerows within their study area (5km^2). However, the authors fail to report the amount of overestimation which occurred within the study and admit that hedgerow detections were "extended and over represented" (Ducrot et al., 2015, pp. 6349). From visual inspection of the classification map provided in their study one can also see that many large non-linear (likely forested) areas were classified as hedgerows. However, their method utilized lower resolution imagery (2.5m) than other studies. Askoy et al. (2010) used the same precision and recall measures as we used in this study, achieving lower rates than ours of 35% and 59%, respectively. Here, they used Quickbird-2 imagery pansharpened to 0.6m resolution. Multiple feature layers were also generated (Gabor, opening and closing granulometry) which required testing across multiple kernel scales in order to determine the optimal feature settings.

Thus, our results were either relatively similar or superior to those of past OBIA approaches. Our approach also seems much more straightforward compared to the OBIA approaches, as they required the testing of multiple feature variables and the generation of additional layers. Given that past studies have come to different conclusions regarding the optimal features for hedgerow classification, new studies would also require testing of different feature variables. This makes these methods less practical for those with limited or no experience in OBIA. OBIA studies also utilized higher resolution imagery (0.5-0.7m) than this current study (1m). Given that higher resolution has been shown to improve NN performance (Jozdani et al., 2019; Wurm et al., 2019; Farooq et al., 2018), it is likely that our results would have outperformed OBIA by a greater margin given higher image resolutions. The OBIA studies were also performed on much smaller areas (5 – 12km^2) whereas this study produced results across a 562km^2 area.

8.7 Environmental implications

Results showed differences in hedgerow densities across the study area, with some areas having very few hedgerows within the agricultural mosaic. Specialist species living within such areas may become prone to local extinctions due to changes in habitat conditions caused by climate change (Schlinkert et al., 2016). The establishment of hedgerows which act as corridors between habitat patches could thus allow for greater mobility of species between habitat patches and improved resilience against climate change (Davies & Pullin, 2007; Tilman et al., 2001).

The soils most commonly present in the Freyung-Grafenau area are prone to nutrient depletion and acidification (Bernhards et al., 2003). This can have negative effects on crop growth as acidic soils may inhibit or reverse plant uptake of important cations such as calcium, magnesium, and potassium (Uexkuell & Mutert, 1995). While acidic soil can be balanced through the applications of lime or phosphate fertilizers, hedgerows can offer an alternative management strategy, as hedgerows have been found to have higher pH levels than in surrounding nearby forest areas. Soils underneath hedgerows also contain higher contents of nutrients important for plant growth (Holden et al., 2019; Wehling & Diekmann, 2009). However, the range of such positive effects is limited, and depends on physical characteristics of the hedgerows, such as width, length, and height (Vooren et al., 2017).

The establishment and protection of hedgerows thus lead to a trade-off between the positive and negative aspects of incorporating hedgerows with agricultural landscapes (e.g. loss of crop area). However, given that most agricultural areas in the district are used for animal husbandry (Bernhards et al., 2003), the establishment of hedgerows should have minimal effects on productivity factors.

9 Conclusion

Overall, both DeepLab and Mask R-CNN performed well with the detection of hedgerows, resulting in accurate and precise hedgerow mapping across a large spatial scale. Both networks were able to learn how to detect hedgerows from minimal input, as only the ground truth annotations and matching satellite images were used to train both networks. Performance was mainly limited by the availability of training data, as only a few IKONOS scenes were available, several of which partially overlapped the same areas. Although augmentation was performed in order to increase the dataset size, no additional feature layers were required, making this a more straightforward approach compared to previous work on hedgerow detection which incorporated manually derived feature layers. While no direct comparison with past approaches to hedgerow detection (e.g. OBIA) were made here, the precision and accuracy rates found in this work were either similar or higher than those reported in many of the previous published works on hedgerow detection (O'Connell et al., 2015; Ducrot et al., 2012; Askoy et al., 2010; Bock, 2005). Further work should be done in order to validate the notion that the NN approaches tested here truly do offer improvements over past OBIA approaches.

Mask R-CNN was able to produce mask predictions with greater precision with respect to object mask boundaries than DeepLab did. However, DeepLab greatly outperformed Mask R-CNN by 15% for both precision and accuracy. This was contrary to past work comparing Mask R-CNN to an older FCN architecture (Zhao et al., 2018). Given that DeepLab has advanced the FCN architecture through the application of atrous convolutions, it seems that atrous convolutions are important architectural features for NN in remote sensing imagery, a result which has been found another recent study (Liu et al., 2019). Training time of 100 epochs was also performed much faster when using DeepLab, further supporting its use over Mask R-CNN. This was likely due to the .tfrecs (Tensorflow records) file format which was used by DeepLab, as it allows for files to be read into memory at faster speeds.

The main limitation with Mask R-CNN was the poor detection of long diagonally oriented hedgerows. This seems to be due to the currently anchor implementation, as anchors for diagonal hedgerows were far larger than the hedgerow object itself. Thus, any future work applying Mask R-CNN to detect hedgerows or other objects which are characterized by uneven aspect ratios should incorporate anchors with rotation parameters which can be learned by the network (Wen et al., 2019; Li et al., 2018; Azimi et al., 2018). On the other hand, DeepLab struggled to produce object masks with precise mask boundaries matching that of the ground truth, as has been found in previous work looking at vegetation object detection (Zhao et al., 2018). This problem was improved upon by using randomly initialized weights for DeepLab. This suggests that the features learned during pre-training are not optimal for remote sensing images, a notion which has been implied in the past (Chu et al., 2016; Yosinski et al., 2014). However, the performance with respect to precision and accuracy was reduced when using randomly initialized weights compared to pre-trained weights. Issues with the training dataset, such as the size of the dataset and the quality of the object annotation masks, are suspected to be the cause for the inferior performance of the randomly initialized network. Thus, in cases where insufficient training data is available, using pre-trained weights would be recommended. Future work should also examine strategies for enhancing the quality of remote sensing annotations masks (Liu et al., 2018) (e.g. incorporating CORINE land cover annotations) in hopes this may enhance the performance of networks trained using randomly initialized weights.

Geometric augmentations were found to improve the results for both NN, while spectral augmentations gave mixed results, as Mask R-CNN performed slightly better when incorporating spectral augmentations and DeepLab performed worse. The decreased performance of DeepLab when using spectral augmentations contradicts past findings where spectral augmentations (e.g. hue transformation) led to improvements in network performance (Stiller et al., 2019; Ma et al., 2019b). These studies aimed at detecting buildings and animals, both of which are defined more by their morphological characteristics than specific spectral properties (Herold et al., 2002; Small, 2001), whereas vegetation typically emits a more consistent spectral signature. Thus, the optimal augmentation strategies may often differ between applications as well as NN architectures. Future work looking at applying only the optimal spectral augmentations could lead to further improvements in the performance of the Mask R-CNN and DeepLab.

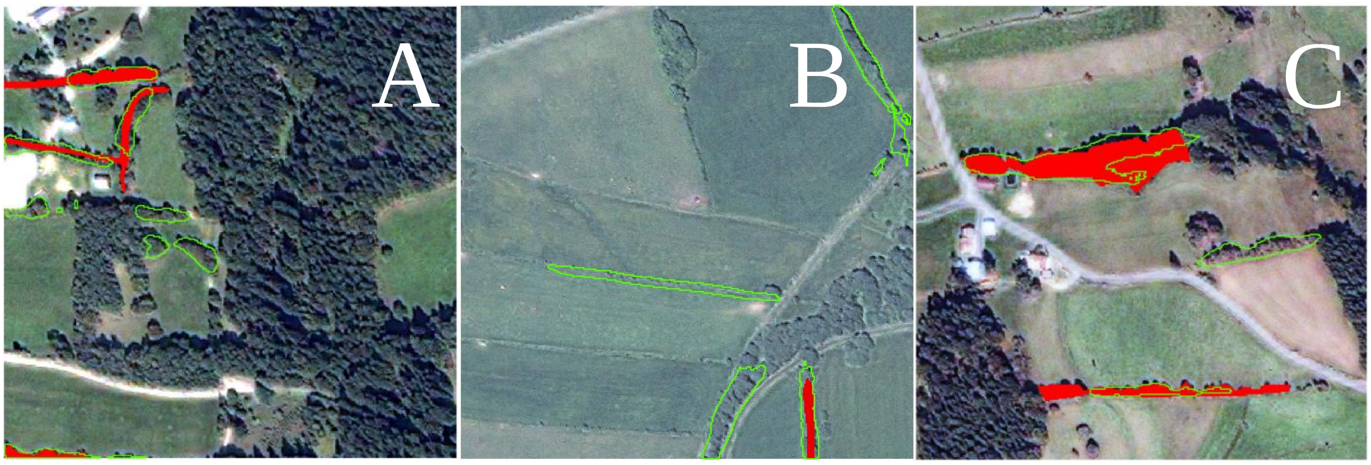
DeepLab was more sensitive across different data inputs (band combinations and seasons), with larger decreases in performance between different input datasets (band combinations, seasons), while Mask R-CNN was more robust to different input datasets. Thus, when applying a network to new study areas outside of the training area, Mask R-CNN is likely to be preferred. The optimal three band combination was found to be GRNIR. However, other band combinations performed similarly well, suggesting that basic sensors (e.g. RGB) could also be successfully applied for hedgerow detection. The inclusion of images from varying seasons (e.g. spring/summer and autumn) was found not to hamper the results of the NN predictions. Thus, datasets using images from mixed seasons should be preferred over a single season dataset as the increase in the dataset size should lead to an increase in performance (Liu et al., 2018).

While in the UK hedgerow data is publicly available (Countryside Survey Data), other countries, such as Germany, lack openly available hedgerow data. Since object-based classification methods rely on expert knowledge regarding feature engineering (Zhong, Fei, & Zhang, 2016; Bock et al., 2005) they may not be easily employed by professionals in both conservational fields and within government agencies. The methods utilized here offer a very cost-effective and relatively accurate (81%) and precise (69%) method for mapping hedgerow occurrences, making NN a valuable tool for hedgerow detection and monitoring. The main limitations of NN are the data and hardware requirements. However, these restrictions can be largely mitigated using pre-trained networks and fine-tuning to the target dataset. Support for the use of pre-trained networks is provided by the results showing our pre-trained models to outperform those trained from randomly initialized weights. Currently, network weights from training on large image datasets such as COCO are readily available, but there are few open source network weights from networks trained using large amounts of remote sensing data. Open access to such weights would be beneficial to the conservation community and increase the capability for interested parties to perform certain image analysis tasks using NN.

The results of this work should thus act as a guide for those wishing to perform hedgerow detection with NN and highlights some of the issues which can arise during network training. Numerous findings of this work can also be transferred to the more general problem of vegetation mapping, such as the optimal combination of bands, and data augmentation strategies. As some parameters should hold for hedgerow detections in other regions, NN should allow for a relatively simple approach to hedgerow mapping given one follows the methodology of this paper.

Appendix

Appendix I. Shows examples where ground truth and hedgerow detections were mismatched. Green outlined areas are areas where hedgerow predictions were made by one of the networks while red areas are the labelled ground truth pixels.



Appendix II. Link to the online repository containing all Python scripts used in this thesis.

https://github.com/SevenFoldDeep/DeepHedge

References

Aksoy, S. (2010). Automatic mapping of linear woody vegetation features in agricultural landscapes using very high resolution imagery. *IEEE Transactions on Geoscience and Remote Sensing, 48*(1), 511-522.

Alegre, J. C., & Rao, M. R. (1996). Soil and water conservation by contour hedging in the humid tropics of Peru. *Agriculture, Ecosystems and Environment*, *57*(1), 17–25.

Atkinson, P. M., & Tatnall, A. R. L. (1997). Introduction neural networks in remote sensing. *International Journal of Remote Sensing*, *18*(4), 699–709.

Azimi, S. M., Vig, E., Bahmanyar, R., Körner, M., & Reinartz, P. (2018). Towards multi-class object detection in unconstrained remote sensing imagery. *arXiv*:1807.02700.

Baatz, M., & Schäpe, A. (2000). Multiresolution Segmentation: An optimization approach for high quality multi-scale image segmentation. *Angewandte Geographische Informations verarbeitung XII*, 12–23.

Baeck, P., Blommaert, J., Delalieux, S., Delauré, B., Livens, S., Nuyts, D. Sima, A., Jacquemin, G. L., & Goffart, J. (2016). High resoltion vegeation mapping with a novel compact hyperspectral camera system. *Proceedings of the 13th Internation Conference on Precision Agriculture*, 1-12.

Baker, B. A., Warner, T. A., Conley, J. F., & McNeil, B. E. (2013). Does spatial resolution matter? A multi-scale comparison of object-based and pixel-based methods for detecting change associated with gas well drilling operations. *International Journal of Remote Sensing*, *34*(5), 1633–1651.

Balidoy Baloloy, A., Conferido Blanco, A., Gumbao Candido, C., Jay Labadisos Argamosa, R., Bart Lovern Caboboy Dumalag, J., Lee Carandang DImapilis, Lady, & Camero Paringit, E. (2018). Estimation of mangrove forest aboveground biomass using multispectral bands, vegetation indices, and biophysical variables deerived from optical satellite imageries: Rapideye, Planetscope and Sentinel-2. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*(3), 29–36.

Baudry, J., Bunce, R. G. H., & Burel, F. (2000). Hedgerows: An international perspective on their origin, function and management. *Journal of Environmental Management*, *60*(1), 7–22.

Baumhoer, C. A., Dietz, A. J., Kneisel, C., & Kuenzer, C. (2019). Automated extraction of antarctic glacier and ice shelf fronts from Sentinel-1 imagery using deep learning. *Remote Sensing*, *11*(21), 1-22.

Bayerisches Landesamt fuer Statistik (2018). *Einwohnerzahlen stand: 31. Dezember 2018*. Retrieved from https://www.statistik.bayern.de/mam/produkte/veroffentlichungen/statistische_berichte/a1210c_201800.pdf.

Belgiu, M., & Drăgu, L. (2016, April 1). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing, 114*, 24–31.

Bernhards, U., Klockenbring, C., Plankl, R., & Rudow, K. (2003). Pilotfallstudie zur Bewertung der Ausgleichszulage in benachteiligten Gebieten im Landkreis Freyung-Grafenau. *Arbeitsbericht // Bundesforschungsanstalt für Landwirtschaft (FAL), Institut für Betriebswirtschaft, Agrarstruktur und Ländliche Räume, No. 04/2003.* Retrieved from https://www.econstor.eu/bitstream/10419/39422/1/385724411.pdf.

Bock, M., Xofis, P., Mitchley, J., Rossner, G., & Wissen, M. (2005). Object-oriented methods for habitat mapping at multiple scales - Case studies from Northern Germany and Wye Downs, UK. *Journal for Nature Conservation, 13*(2–3), 75–89.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. *Neurocomputing*, 227–236.

Buckland, M., & Gey, F. (1994). The relationship between recall and precision. *Journal of the American Society for Information Science, 45*(1), 12–19.

Budreski, K. A., Wynne, R. H., Browder, J. O., & Campbell, J. B. (2007). Comparison of segment and pixel-based non-parametric land cover classification in the Brazilian Amazon using multitemporal Landsat TM/ETM+ imagery. *Photogrammetric Engineering and Remote Sensing, 73*(7), 813–827.

Cadène, R., Thome, N., & Cord, M. (2016). Master’s thesis : Deep learning for visual recognition. *arXiv*:1610.05567.

Canny, J. (1987). A Computational Approach to Edge Detection. *Readings in Computer Vision*, 184–203.

Chen, L., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *arXiv*: 1706.05587.

Chen, L., Zhu, Y., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L. (2018). DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4), 834-848.

Chen, L., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018b). Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv:* 1802.02611.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 1800–1807.

Chu, B., Madhavan, V., Beijbom, O., Hoffman, J., & Darrell, T. (2016). Best practices for fine-tuning visual classifiers to new domains. *ECCV Workshops*, 435-442.

Davies, Z. G. & Pullin, A. S. (2007). Are hedgerows effective corridors between fragments of woodland habitat? An evidence-based approach. *Landscape Ecology, 22*(3), 333-351.

Dodge, S., & Karam, L. (2016). Understanding how image quality affects deep neural networks. *arXiv*:1604.04004.

Doulamis, N. (2018). Adaptable deep learning structures for object labeling/tracking under dynamic visual environments. *Multimedia Tools and Applications*, *77*(8), 9651–9689.

Dover, J. W. (2019). *The ecology of hedgerows and field margins.* New York, NY: Routledge.

Dover, J., & Sparks, T. (2000). A review of the ecology of butterflies in British hedgerows. *Journal of Environmental Management*, *60*(1), 51–63.

Ducrot, D., Masse, A., & Ncibi, A. (2012). Hedgerow detection in HRS and VHRS images from different source (optical, radar). *International Geoscience and Remote Sensing Symposium (IGARSS)*, 6348–6351.

Dudley, N., Attwood, S. J., Goulson, D., Jarvis, D., Bharucha, Z. P., & Pretty, J. (2017, May 1). How should conservationists respond to pesticides as a driver of biodiversity loss in agroecosystems? *Biological Conservation*, *209*, 449–453.

Dvornik, N., Mairal, J., & Schmid, C. (2018). On the importance of visual context for data augmentation in scene understanding. *arXiv*:1809.02492.

Ekroos, J., Heliölä, J., & Kuussaari, M. (2010). Homogenization of lepidopteran communities in intensively cultivated agricultural landscapes. *Journal of Applied Ecology*, *47*(2), 459–467.

Erb, R. J. (1993). Introduction to Backpropagation Neural Network Computation. *Pharmaceutical Research: An Official Journal of the American Association of Pharmaceutical Scientists*, *10*, 165–170.

Farooq, A., Hu, J., & Jia, X. (2019). Analysis of spectral bands and spatial resolutions for weed classification via deep convolutional neural network. *IEEE Geoscience and Remote Sensing Letters*, *16*(2), 183–187.

Fauvel, M., Sheeren, D., Chanussot, J., & Benediktsson, J. A. (2012). Hedges detection using local directional features and support vector data description. *International Geoscience and Remote*

Sensing Symposium (IGARSS), 2320–2323.

Feng, Q., Liu, J., & Gong, J. (2015). UAV remote sensing for urban vegetation mapping using random forest and texture analysis. *Remote Sensing*, *7*(1), 1074–1094.

Firbank, L., Bradbury, R. B., McCracken, D. I., & Stoate, C. (2013). Delivering multiple ecosystem services from Enclosed Farmland in the UK. *Agriculture, Ecosystems and Environment*, *166*, 65–75.

Follain, S., Walter, C., Legout, A., Lemercier, B., & Dutin, G. (2007). Induced effects of hedgerow networks on soil organic carbon storage within an agricultural landscape. *Geoderma*, *142*(1–2), 80–95.

Forman, R. T. T., & Baudry, J. (1984). Hedgerows and hedgerow networks in landscape ecology. *Environmental Management*, *8*(6), 495–510.

Friedl, M. A., McIver, D. K., Hodges, J. C. F., Zhang, X. Y., Muchoney, D., Strahler, A. H., Woodcock, C. E., Gopal, S., Schneider, A., Cooper, A., Baccini, A., Goa, F., & Schaaf, C. (2002). Global land cover mapping from MODIS: Algorithms and early results. *Remote Sensing of Environment*, *83*(2), 287–302.

Froidevaux, J. S. P., Broyles, M., & Jones, G. (2019). Moth responses to sympathetic hedgerow management in temperate farmland. *Agriculture, Ecosystems and Environment*, *270*, 55–64.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*(4), 193–202.

Gao, Y., Zhang, Y., Li, X., Li, L., Gao, A., & Yu, P. (2016). A review of guided filter based pansharpening methods. *Advances in Computer Science Research*, 64, 44-49.

Ghazavi, G., Thomas, Z., Hamon, Y., Marie, J. C., Corson, M., & Merot, P. (2008). Hedgerow impacts on soil-water transfer due to rainfall interception and root-water uptake. *Hydrological Processes*, *22*(24), 4723–4735.

Ghiyamat, A., & Shafri, H. Z. M. (2010). A review on hyperspectral remote sensing for homogeneous and heterogeneous forest biodiversity assessment. *International Journal of Remote Sensing*, *31*, 1837–1856.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2015). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 142-158.

Girshick, R. (2015b). Fast R-CNN. *arXiv:* 1504.08083.

Glorot, X. & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 249-256.

Graystock, P., Yates, K., Evison, S. E. F., Darvill, B., Goulson, D., & Hughes, W. O. H. (2013). The Trojan hives: pollinator pathogens, imported and distributed in bumblebee colonies. *Journal of Applied Ecology*, *50*(5), 1207-1215.

Griffiths, P., Van Der Linden, S., Kuemmerle, T., & Hostert, P. (2013). Erratum: A pixel-based landsat compositing algorithm for large area land cover mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *6*, 2088–2101.

Guo, Y., Liu, Y., Georgiou, & Lew, M. S. (2018). A review of semantic segmentation using deep neural networks. *International Journal of Mulitmedia Information Retrieval*, *7*(2), 87-93.

Hallmann, C. A., Sorg, M., Jongejans, E., Siepel, H., Hofland, N., Schwan, H., Stenmans, W., Mueller, A., Sumser, H., Hoerren, T., Goulson, D., & de Kroon, H. (2017). More than 75 percent decline over 27 years in total flying insect biomass in protected areas. *PLOS ONE*, *12*(10), e0185809.

Hattab-Hambli, N., Motelica-Heino, M., & Mench, M. (2016). Aided phytoextraction of Cu, Pb, Zn, and As in copper-contaminated soils with tobacco and sunflower in crop rotation: Mobility and phytoavailability assessment. *Chemosphere*, 145, 543–550.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv:* 1512.03385

He, K., Zhang, X., Ren, S., & Sun, J. (2015b). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. *arXiv*:1502:01852.

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2018). Mask R-CNN. *arXiv*: 1703.06870.

Hellesen, T., & Matikainen, L. (2013). An object-based approach for mapping shrub and tree cover on grassland habitats by use of LiDAR and CIR orthoimages. *Remote Sensing*, *5*(2), 558–583.

Herold, M., Gardner, M., Hadley, B., & Roberts, D. (2002). The spectral dimension in urban land cover mapping from high-resolution optical remote sensing data. *Proceeding of the 3rd Symposium on Remote Sensing of Urban Areas*, 1-8.

Hoffer, E., Hubara, I., & Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *arXiv*:1705.08741

Holden, J., Grayson, R. P., Berdeni, D., Bird, S., Chapman, P. J., Edmondson, J. L., Firbank, L. G., Helgason, T., Hodson, M. E., Hunt, S. F. P., Jones, D. T., Lappage, M. G., Marshall-Harries, E., Nelson, M., Miller, M., Shaw, H., Wade, R. N., & Leake, J. R. (2019). The role of hedgerows in soil functioning within agricultural landscapes. *Agriculture, Ecosystems and Environment*, *273*, 1–12.

Hu, W., Huang, Y., Wei, L., Zhang, F., & Li, H. (2015). Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, *2015*, 1-12.

Huang, Z., Huang, L., Gong, Y., Huang, C., & Wang, X. (2019). Mask scoring R-CNN. *arXiv*:1903.00241.

Huber, C. (2005). Long lasting nitrate leaching after bark beetle attack in the highlands of the Bavarian Forest National Park. *Journal of Environmental Quality*, *34*(5), 1772–1779.

Hubert-Moy, L., Cotonnec, A., Le Du, L., Chardin, A., & Perez, P. (2001). A comparison of parametric classification procedures of remotely sensed data applied on different landscape units. *Remote Sensing of Environment*, *75*(2), 174–187.

Hurd, J. D., Civco, D. L., Gilmore, M. S., & Wilson, E. H. (2006). Tial wetland classifcation from Landsat imagery using an integrated pixel-based and object-based classification approach. *ASPRS Annual Conference*, 1-11.

Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., Andriluka, M., Rakpurkar, P., Migimatsu, T., Cheng-Yue, R., Mujica, F., Coates, A., & Ng, A. Y. (2015). An empirical evaluation of deep learning on highway driving. *arXiv:* 1504.01716.

Hycza, T., Stereńczak, K., & Bałazy, R. (2018). Potential use of hyperspectral data to classify forest tree species. *New Zealand Journal of Forestry Science*, *48*(1).

Ishii, T., Nakamura, R., Nakada, H., Mochizuki, Y., & Ishikawa, H. (2015). Surface object recognition with CNN and SVM in Landsat 8 images. *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, 341–344.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *32nd International Conference on Machine Learning*, 448–456.

Jensen, J. R. (2006). *Remote sensing of the environment: An earth resource perspective* (2nd ed.). London, UK: Pearson.

Johnson, J., Karpathy, A., & Fei-Fei, L. (2015). DenseCap: Fully convolutional localization networks for dense captioning. *arXiv*:1511.07571

Juel, A., Groom, G. B., Svenning, J. C., & Ejrnæs, R. (2015). Spatial application of Random Forest models for fine-scale coastal vegetation classification using object based analysis of aerial orthophoto and DEM data. *International Journal of Applied Earth Observation and Geoinformation*, *42*, 106–114.

Kawaguchi, K. (2016). Deep learning without poor local minima. *arXiv*:1605.07110.

Kovács-Hostyánszki, A., Espíndola, A., Vanbergen, A. J., Settele, J., Kremen, C., & Dicks, L. V. (2017). Ecological intensification to mitigate impacts of conventional intensive land use on pollinators and pollination. *Ecology Letters*, *20*(5), 673–689.

Krenker, A., Bester, J., & Kos, A. (2011). Introduction to the artificial neural networks. In K. Suzuki (Ed.), *Artificial Neural Networks - Methodological Advances and Biomedical Applications* (pp. 1-18). London, UK: IntechOpen.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25(2), 1-9.

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, S. (2017). Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778-782.

Label, C. A. & Brower, B. V. (1998). Process for enhancing the spatial resolution of multispectral imagery using pan-sharpening. *United States Patent No. US6011875A*. Retrieved from https://patents.google.com/patent/US6011875A/en.

Lacoeuilhe, A., Machon, N., Julien, J.-F., & Kerbiriou, C. (2018). The relative effects of local and landscape characteristics of hedgerows on bats. *Diversity*, *10*(3), 1-16.

Lacoste, M., Minasny, B., McBratney, A., Michot, D., Viaud, V., & Walter, C. (2014). High resolution 3D mapping of soil organic carbon in a heterogeneous agricultural landscape. *Geoderma*, *213*, 296–311.

Längkvist, M., Kiselev, A., Alirezaie, M., & Loutfi, A., (2016). Classification and segmentation of satellite orthoimagery using convolutional neural networks. *Remote Sensing*, *8*(4).

Larsson, G., Maire, M., & Shakhnarovich, G. (2016). FractalNet: Ultra-deep neural networks without residuals. *arXiv*:1605.07648

Laura, V. V., Bert, R., Steven, B., Pieter, D. F., Victoria, N., Paul, P., & Kris, V. (2017). Ecosystem service delivery of agri-environment measures: A synthesis for hedgerows and grass strips on arable land. *Agriculture, Ecosystems and Environment*, *244*, 32–51.

Le, H., & Borji, A. (2017). What are the receptive, effective receptive, and projective fields of neurons in convolutional neural networks?. *arXiv*:1705.07049.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324.

Li, W., Dong, R., Fu, H., & Yu, L. (2018). Large-scale oil palm tree detection from high-resolution satellite images using two-stage convolutional neural networks. *Remote Sensing, 11*(1), 1-20.

Li, M., Zang, S., Zhang, B., Li, S., & Wu, C. (2014). A review of remote sensing image classification techniques: The role of Spatio-contextual information. *European Journal of Remote Sensing, 47*(1), 389–411.

Lin, T., Dollar, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *arXiv*: 1612.03144.

Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017b). Focal loss for dense object detection. *arXiv*:1708.02002.

Lin, T. Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Ramanan, D., Zitnick, C. L., Dollar, P. (2014). Microsoft COCO: Common objects in context. *arXiv*:1405:0312.

Liu, C., Zhang, Y., Chen, P., Lai, C., Chen, Y., Cheng, J., & Ko, M. (2019). Clouds classification from Sentinel-2 imagery with deep residual learning and semantic image segmentation. *Remote sensing, 11*(2), 1-16.

Liu, T., Abd-Elrahman, A., Jon, M., & Wilhelm, V. L. (2018). Comparing fully convolutional networks, random forest, support vector machine, and patch-based deep convolutional neural networks for object-based wetland mapping using images from small unmanned aircraft system. *GIScience & Remote Sensing*, *55*(2), 243-264.

Liu, S., Ding, W., Liu, C., Liu, Y., Wang, Y., & Li, H. (2018b). ERN: Edge loss reinforced semantic segmentation network for remote sensing images. *Remote sensing*, 10, 1-23.

Long, J., Shelhammer, E., & Darrel, T. (2015). Fully convolutional networks for semantic segmentation. *arXiv*:1411.4038.

Lotfi, A., Javelle, A., Baudry, J., & Burel, F. (2010). Interdisciplinary analysis of hedgerow network landscapes' sustainability. *Landscape Research*, *35*(4), 415–426.

Lu, H., & Kawaguchi, K. (2017). Depth creates no bad local minima. *arXiv*:1702.08580.

Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS Journal of Photogrammetry and Remote*

Sensing, *152*, 166–177.

Ma, R., Tao, P., & Tang, H. (2019b). Optimizing data augmentation for semantic segmentation on small-scale dataset. *ACM International Conference Proceeding Series*, 77–81.

Marmanis, D., Schindler, K., Wegner, J. D., Galliani, S., Datcu, M., & Stilla, U. (2018). Classification with an edge: Improving semantic image segmentation with boundary detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, *135*, 158–172.

Martins, A. F. T., & Astudillo, R. F. (2016). From softmax to sparsemax: A sparse model of attention and multi-label classification. *arXiv*:1602.02068

Masters, D., & Luschi, C. (2018). Revisiting small batch training for deep neural networks. *arXiv*:1804.07612

Meier, M. S., Stoessel, F., Jungbluth, N., Juraske, R., Schader, C., & Stolze, M. (2015). Environmental impacts of organic and conventional agricultural products - Are the differences captured by life cycle assessment? *Journal of Environmental Management*, *149*, 193–208.

Mel, M., Michieli, U., & Zanuttigh, P. (2019). Incremental and multi-task learning strategies for coarse-to-fine semantic segmentation. *Technologies*, *8*(1).

Meneguzzo, D. M., Liknes, G. C., & Nelson, M. D. (2013). Mapping trees outside forests using high-resolution aerial imagery: A comparison of pixel- and object-based classification approaches. *Environmental Monitoring and Assessment*, *185*(8), 6261–6275.

Mercioni, M. A., Tiron, A., & Holban, S. (2019). Dynamic modification of activation function using the backpropagation algorithm in the artificial neural networks. *International Journal of Advanced Computer Science and Applications*, *10*(4).

Merzlyak, M. N., Gitelson, A. A., Chivkunova, O. B., & Rakitin, V. Y. (1999). Non-destructive optical detection of pigment changes during leaf senescence and fruit ripening. *Physiologia Plantarum*, *106*(1), 135–141.

Mishra, N. B., & Crews, K. A. (2014). Mapping vegetation morphology types in a dry savanna ecosystem: Integrating hierarchical object-based image analysis with Random Forest. *International Journal of Remote Sensing*, *35*(3), 1175–1198.

Morandin, L. A., Long, R. F., & Kremen, C. (2014). Hedgerows enhance beneficial insects on adjacent tomato fields in an intensive agricultural landscape. *Agriculture, Ecosystems and Environment*, *189*, 164–170.

Mou, L. & Zhu, X. X. (2018). RiFCN: recurrent network in fully convolutional network for semantic segmentation of high resolution remote sensing images. *arXiv:* 1805.0209.

Mountrakis, G., Im, J., & Ogole, C. (2011, May). Support vector machines in remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, *66*, 247–259.

Mullen, J. F., Tanner, F. R., & Sallee, P. A. (2019). Comparing the effects of annotation type on machine learning detection performance. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 1-7.

Nagendra, H., Lucas, R., Honrado, J. P., Jongman, R. H. G., Tarantino, C., Adamo, M., & Mairota, P. (2013). Remote sensing for conservation monitoring: Assessing protected areas, habitat extent, habitat condition, species diversity, and threats. *Ecological Indicators*, *33*, 45–59.

Nguyen, Q., & Hein, M. (2017). The loss surface of deep and wide neural networks. *arXiv*:1704.08045.

Nie, S., Jiand, Z., Zhang, H., Cai, B., & Yao, Y. (2018). Inshore ship detection based on Mask R-CNN. *IEEE International Geoscience and Remote Sensing Symposium*, 1-4.

Nijhawan, R., Sharma, H., Sahni, H., & Batra, A. (2017). A deep learning hybrid CNN framework approach for vegetation cover mapping using deep features. *2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, 192–196.

Noi, P. T., & Kappas, M. (2018). Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using sentinel-2 imagery. *Sensors*, *18*(18), 1-20.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *arXiv*:1811.03378.

O'Connell, J., Bradter, U., & Benton, T. G. (2015). Wide-area mapping of small-scale features in agricultural landscapes using airborne remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, *109*, 165–177.

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *arXiv:* 1511.08458.

Ohleyer, S. (2018). Building segmentation on satellite images. Retrieved from https://project.inria.fr/aerialimagelabeling/files/2018/01/fp_ohleyer_compressed.pdf.

Osborne, P. (1984). Bird numbers and habitat characteristics in farmland hedgerows. *The Journal of Applied Ecology*, *21*(1).

Padoa-Schioppa, E., Baietto, M., Massa, R., & Bottoni, L. (2006). Bird communities as bioindicators:

The focal species concept in agricultural landscapes. *Ecological Indicators*, *6*(1), 83–93.

Palandro, D., Andréfouët, S., Dustan, P., & Muller-Karger, F. E. (2003). Change detection in coral reef communities using Ikonos satellite sensor imagery and historic aerial photographs. *International Journal of Remote Sensing*, *24*(4), 873–878.

Panboonyuen, T., Jitkajornwanich, K., Lawawirojwong, S., Srestasathiern, P., & Vateekul, P. (2017). Road segmentation of remotely-sensed images using deep convolutional neural networks with landscape metrics and conditional random fields. *Remote Sensing*, *9*(7), 1-19.

Parish, T., Lakhani, K. H., & Sparks, T. H. (1994). Modelling the relationship between bird population variables and hedgerow and other field margin attributes. I. Species richness of winter, summer and breeding birds. *The Journal of Applied Ecology*, *31*(4), 764-775.

Penatti, O. A. B., Nogueira, K., & Dos Santos, J. A. (2015). Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 44–51.

Pepe, M., Fregonese, L., & Scaioni, M. (2018). Planning airborne photogrammetry and remote-sensing missions with modern platforms and sensors. *European Journal of Remote Sensing*, *51*(1), 412–435.

Petit, S., Stuart, R. C., Gillspie, M. K., & Barr, C. J. (2003). Field boundaries in Great Britain: stock and change between 1984, 1990 and 1998. *Journal of Environemtal Management*, *67*(3), 229-238.

Powell, J. R. (2018). Genetic variation in insect vectors: Death of typology? *Insects*, *9*(4), 1-15.

Power, A. G. (2010). Ecosystem services and agriculture: Tradeoffs and synergies. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *365*, 2959–2971.

Price, K. P., Guo, X., & Stiles, J. M. (2002). Optimal landsat TM band combinations and vegetation indices for discrimination of six grassland types in eastern Kansas. *International Journal of Remote Sensing*, *23*(23), 5031–5042.

Redlich, S., Martin, E. A., Wende, B., & Steffan-Dewenter, I. (2018). Landscape heterogeneity rather than crop diversity mediates bird diversity in agricultural landscapes. *PLOS ONE, 13*(8), e0200438.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. *arXiv:* 1506.01497.

Richards, A. J. (2001). Does low biodiversity resulting from modern agricultural practice affect crop pollination and yield?. *Annals of Botany, 88*(2), 165-172.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv:* 1609.04747.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, *16*(8), 1-23.

Saindane, P., Ganapathy, G., Prabhavalkar, N., Bhatia, N., & Vaidya, A. (2019). Comparison and analysis of algorithms used for detecting slums in satellite images. *International Journal of New Technology and Research, 5*(2), 48-52.

Sarlöv Herlin, I. L., & Fry, G. L. A. (2000). Dispersal of woody plants in forest edges and hedgerows in a Southern Swedish agricultural area: The role of site and landscape structure. *Landscape Ecology*, *15*(3), 229–242.

Scherer, D., Müller, A., & Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. *Artificial Neural Networks - ICANN 2010 - 20th Internation Conference,* 1-10.

Schieber, B. (2014). Effect of altitude on phenology of selected forest plant species in Slovakia (Western Carpathians). *Folia Oecologica*, *41*(1), 75-81.

Schmitz, M. F., Herrero-Jáuregui, C., Arnaiz-Schmitz, C., Sánchez, I. A., Rescia, A. J., & Pineda, F. D. (2017). Evaluating the role of a protected area on hedgerow conservation: The case of a spanish cultural landscape. *Land Degradation & Development*, *28*(3), 833–842.

Shao, Z., Wu, W., Wang, Z., Du, W., & Li, C. (2018). SeasShips: a large-scale precisely annotated dataset for ship detection. *IEEE Transactions on Multimedia, 20*(10), 2593-2604.

Small, C. (2001). Multiresolution analysis of urban reflectance. *IEEE/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas, DFUA 2001*, 15–19.

Staley, J. T., Sparks, T. H., Croxton, P. J., Baldock, K. C. R., Heard, M. S., Hulmes, S., Peyton, J., Amy, S. R., & Pywell, R. F. (2012). Long-term effects of hedgerow management policies on resource provision for wildlife. *Biological Conservation*, *145*(1), 24–29.

Sun, Y., Tian, Y., Xu, Y. (2018). Problems of encoder-decoder frameworks for high-resolution remote sensing image segmentation: structural stereotype and insufficient learning. *Neurocomputing, 330*, 297-304.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception architecture for computer vision. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2818–2826.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning?. *IEEE Transactions on Medical Imaging*, *35*(5), 1299–1312.

Tansey, K., Chambers, I., Anstee, A., Denniss, A., & Lamb, A. (2009). Object-oriented classification of very high resolution airborne imagery for the extraction of hedgerows and field margin cover in agricultural areas. *Applied Geography*, *29*(2), 145–157.

Taubenböck, H., Esch, T., Wurm, M., Roth, A., & Dech, S. (2010). Object-based feature extraction using high spatial resolution satellite data of urban areas. *Journal of Spatial Science*, *55*(1), 117–132.

Tehrany, M. S., Pradhan, B., & Jebuv, M. N. (2014). A comparative assessment between object and pixel-based classification approaches for land use/land cover mapping using SPOT 5 imagery. *Geocarto International, 29*(4), 351–369.

Tilman, D., Fargione, J., Wolff, B., D'Antonio, C., Dobson, A., Howarth, R., Schindler, D., Schlesinger, W.H., Simberloff, D., & Swackhamer, D. (2001). Forecasting agriculturally driven global environmental change. *Science, 292*, 281–284.

Tong, X.-Y., Xia, G.-S., Lu, Q., Shen, H., Li, S., You, S., & Zhang, L. (2018). Land-cover classification with high-resolution remote sensing images using transferable deep models. *arXiv*:1807.05713

UN-DESA/DSD (2014). *United Nations Sustainable Development Goal 2: end Hunger, Achieve Food Security and Improved Nutrition and Promote Sustainable Agriculture*. Available online at: https://sustainabledevelopment.un.org/?page=view&nr=164&type=230.

Vannier, C., & Hubert-Moy, L. (2008). Detection of wooded hedgerows in high resolution satellite images using an object-oriented method. *International Geoscience and Remote Sensing Symposium (IGARSS), 4*(1).

Vannier, C., & Hubert-Moy, L. (2014). Multiscale comparison of remote-sensing data for linear woody vegetation mapping. *International Journal of Remote Sensing*, *35*(21), 7376–7399.

Viaud, V., Durand, P., Merot, P., Sauboua, E., & Saadi, Z. (2005). Modeling the impact of the spatial structure of a hedge network on the hydrology of a small catchment in a temperate climate. *Agricultural Water Management*, *74*(2), 135–163.

Vitasse, Y., Delzon, S., Bresson, C. C., Michalet, R., & Kremer, A. (2009). Altitudinal differentiation in growth and phenology among populations of temperate-zone tree species growing in a common garden. *Canadian Journal of Forest Research*, *39*(7), 1259–1269.

von Uexküll, H. R., & Mutert, E. (1995). Global extent, development and economic impact of acid soils. *Plant and Soil*, *171*(1), 1–15.

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, *2018*, 1–13.

Walker, M. P., Dover, J. W., Sparks, T. H., & Hinsley, S. A. (2006). Hedges and green lanes: Vegetation composition and structure. *Biodiversity and Conservation*, *15*(8), 2595–2610.

Wehling, S., & Diekmann, M. (2009). Hedgerows as an environment for forest plants: A comparative case study of five species. *Plant Ecology*, *204*(1), 11–20.

Wen, Q., Jiang, K., Wang, W., Liu, Q., Guo, Q., Li, L., & Wang, P. (2019). Automatic building extraction from google earth images under complex backgrounds based on deep instance segmentation network. *Sensors*, *19*(2).

Wulder, M. A., White, J. C., Hay, G. J., & Castilla, G. (2008). Pixels to objects to information: Spatial context to aid in forest characterization with remote sensing. *Lecture Notes in Geoinformation and Cartography*, *30*, 345–363.

Wurm M., Stark, T., Zhu, X., Weigand, M., & Traubenböck, H. (2019). Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, *150*, 59-69.

Xie, M., Jean, N., Burke, M., Lobell, D., & Ermon, S. (2016). Transfer learning from deep features for remote sensing and poverty mapping. *30th AAAI Conference on Artificial Intelligence*, 3929–3935.

Xing, F., Xie, Y., Su, H., Liu, F., & Yang, L. (2018). Deep learning in microscopy image analysis: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(10), 4550–4568.

Yao, Y., Jiang, Z., Zhang, H., Zhao, D., & Cia, B. (2017). Ship detection in optical remote sensing images based on deep convolutional neural networks. *Journal of Applied Remote Sensing*, *11*(4).

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks*?. arXiv*:1411.1792.

Zhang, Y., You, Y., Wang, R., Liu, F., & Liu, J. (2018). Nearshore vessel detection based on scene-mask R-CNN in remote sensing image. *International Conference on Network Infrastructure and Digital Content.*

Zhang, Y., Zhang, Y., Li, S. X., Zhang, J. H. (2018b). Accurate detection of berthing ship target based on Mask R-CNN. *2018 International Conference on Image and Video Processing, and Artificial*

Intelligence.

Zhang, X., Friedl, M. A., & Schaaf, C. B. (2009). Sensitivity of vegetation phenology detection to the temporal resolution of satellite data. *International Journal of Remote Sensing*, *30*(8), 2061–2074.

Zheng, H., Cheng, T., Li, D., Zhou, X., Yao, X., Tian, Y., Cao, W., & Zhu, Y. (2018). Evaluation of RGB, color-infrared and multispectral images acquired from unmanned aerial systems for the estimation of nitrogen accumulation in rice. *Remote Sensing*, *10*(6), 824.

Zhao, K., Kang, J., Jung, J., & Sohn, G. (2018). Building extraction from satellite images using Mask R-CNN with building boundary regularization. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).*

Zhao, T., Yang, Y., Niu, H., Wang, D., & Chen, Y. (2018b) Comparing U-Net convolutional network with Mask R-CNN in the performances of pomegranate tree canopy segmentation. *Conference: Multispectral, Hyperspectral, and Ultraspectral Remote Sensing Technology, Techniques and Applications VII.*

Zhong, Y., Fei, F., & Zhang, L. (2016). Large patch convolutional neural networks for the scene classification of high spatial resolution imagery. *Journal of Applied Remote Sensing*, *10*(2), 1-20.

Zhou, K., Chen, Y., Smal, I., & Lindenbergh, R. (2019). Building segmentation from airborne vhr images using mask r-cnn. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, 42*(2), 155–161.

Zhu, X. X., Tuia, D., Mou, L., Xia, G. S., Zhang, L., Xu, F., & Fraundorfer, F. (2017). Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine, 5,* 8–36.

Ziello, C., Estralla, N., Kostova, M., Koch, E., & Menzel, A. (2009). Influence of altitude on phenology of selected plant species in the Alpine region (1971 - 2000). *Climate Research, 39*(3), 227-234.