REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

DATA MINING OF SMART AGRICULTURAL YIELDING USING CONVOLUTIONAL NEURAL NETWORK (CNN)

Omar Ahmed Razooqi AL-DOORI

Master's Thesis

Supervisor

Asst. Prof. Dr. Ayça Kurnaz TÜRKBEN

Istanbul, 2022

DATA MINING OF SMART AGRICULTURAL YIELDING USING CONVOLUTIONAL NEURAL NETWORK (CNN)

Omar Ahmed Razooqi AL-DOORI

Information Technologies

Master’s Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled OPTIMIZING ROUTING AND SECURITY MECHANISM IN AD HOC NETWORKS prepared by OMAR AHMED RAZOOQI AL-DOORI and submitted on 15/12/2022 has been **accepted unanimously** for the degree of Master of Science in Information Technologies.

____

Asst. Prof. Dr. Ayça Kurnaz TÜRKBEN

the Supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Ayça Kurnaz TÜRKBEN	Department of Software Engineering, Altınbaş University	__________________
Asst. Prof. Dr. Abdullahi Abdu IBRAHIM	Department of Computer Engineering, Altınbaş University	__________________
Asst. Prof. Dr. Serdar KARGIN	Department of Biomedical Engineering, Arel University	__________________

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

Submission data of the thesis to Institute of Graduate Studies: ____/____/____

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Omar Ahmed Razooqi AL-DOORI

Signature

DEDICATION

I devote and pledge this research work to my supervisor who is salient for guiding me through whole research work as well as my family for always assisting me in my hard time.

PREFACE

First and foremost, I would like to thank my supervisor Asst. Prof. Dr. Ayça Kurnaz TÜRKBEN for guiding and helping me along the way in writing this dissertation. Discussing my progress, problems, and ideas with my supervisor Asst. Prof. Dr. Ayça KURNAZ TÜRKBEN a couple of times every week helped me tremendously in understanding the logic behind the research. It made me better realize the technical need for this research work.

ABSTRACT

DATA MINING OF SMART AGRICULTURAL YIELDING USING CONVOLUTIONAL NEURAL NETWORK (CNN)

AL-DOORI, Omar Ahmed Razooqi

M.Sc., Information Technologies, Altınbaş University,

Supervisor: Asst. Prof. Dr. Ayça Kurnaz TÜRKBEN

Date: 12/2022

Pages: 76

This advance research works on the data mining of smart agricultural yielding using Convolutional Neural Network (CNN) and designing an algorithm for the delineation of agricultural parcel objects is non-trivial, both via traditional image segmentation as well as with deep learning techniques. The model needs to correctly dismiss undesired image objects, differentiate between adjacent fields with nearly similar spectral properties or barely visible borders, and correctly delineate field objects with multiple homogenous areas (e.g. caused by varying soil properties within a single field). Overall, the three main challenges for such an algorithm are the heterogeneous and volatile geography of agricultural fields, the specific properties of satellite imagery, inaccuracies in the available ground truth datasets. The results appear very promising in light of the insufficient precedent work on instance segmentation of remote sensing images, the non-optimal training data, and the relatively simple input-output method without extensive post-processing. The correct prediction of small field parcels, miss detections of parcels that are considered environmental areas, and similar physical characteristics between parcels of different crop classes are major obstacles for the current method. The application of CNN based field yield instance segmentation to geospatial and remote sensing data demonstrates a significant future potential in light of the thesis's positive outcomes of 98.92%. The models can use raw data without needing to manually engineer features. The convolutional operation can be described as a moving window that slides across the data. Its size is also called kernel size. Assuming a stride of 1 (i.e. the filter moves across the data in steps of one pixel) and zero-padding of 2 (i.e. the input image boundaries are extended by two rows and two columns of zeros), the moving window is applied to every possible spatial location of the RGB input image in the first network layer. Instead, the S, M and L category area limits were selected by the distribution of field areas in the training data: S (< 162 pixel), M (162 - 322 pixel) and L (> 322 pixel). This roughly corresponds to 43.1% of training field polygons < 0.025 km2, 46.5 % from 0.025 km2 - 0.1 km2 and 10.4 % > 0.1 km2. The processing of satellite images and the training characteristics and resource requirements of deep learning models are made more difficult by the satellite image properties.

Keywords: Image Segmentation, Object Identification, Yield, Agricultural, Geography.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

CLU	:	Common Land Unit
LPIS	:	Land Parcel Identification System
GPUs	:	Graphics Processing Units
IAIS	:	Instance Aware Semantic Segmentation
ANN	:	Artificial Neural Networks
CNN	:	convolutional neural networks
FCN	:	Fully Convolutional Neural Networks

1. INTRODUCTION

1.1. AGRICULTURAL FIELD PARCELS

Geo data of field boundaries and parcels is essential for many agriculture-related applications. Examples include crop type and yield monitoring, subsidy management as well as research and decision support for food security efforts. Initially, such data was primarily utilized by governmental agencies. Today, commercial businesses in the booming agricultural technology sector also require field parcels in various application areas like farm management, yield forecasting and precision farming.

Initiatives for the delineation and mapping of agricultural field parcels include the United States Common Land Unit (CLU) system and the European Union Land Parcel Identification System (LPIS) as given in [1]. Experts in aerial or satellite imagery primarily trace high-quality data on agricultural parcels by hand. Single field objects can be distinguished from the surrounding area in high-resolution imagery by meaningful color, brightness, or texture shifts caused by changes in land use as given in [2]. The task can be supported, if possible, by property or reference parcels, additional farmer-provided geospatial data, information on crop type, field usage status, and other details. Despite the fact that manual crop block segmentation can be very accurate, it takes a long time for larger agricultural areas and is susceptible to intra- and inter-observer variability due to human operators' varying levels of experience, performance, and diligence as given in [3].

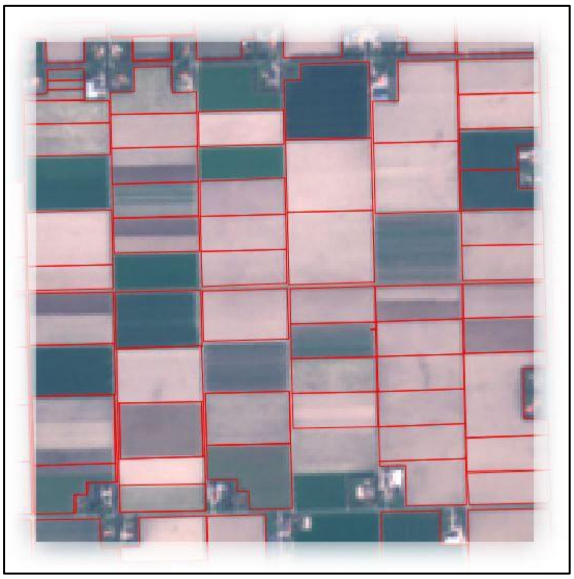


Figure 1.1: Polygons of agricultural field parcels overlaid on satellite imagery [3].

The increasing demand for consistent, regularly updated field parcel data for agricultural regions throughout the world emphasizes the need for reproducible automation of this task. However, most traditional image segmentation models for the automated delineation of agricultural parcels are designed for small-scale applications and tuned to the specific environmental settings of the study area. An ideal automatic method would be able to deliver satisfying speed and accuracy for large areas with variable environmental settings. The thesis investigates whether this goal can be achieved via deep learning instance segmentation, a novel approach for the segmentation and classification of image objects.

1.2. CHALLENGES FOR AUTOMATED DELINEATION METHODS

Agricultural fields in satellite images are relatively simple image objects when compared to most objects in natural photos. E.g., a picture of a human body appears much more complex because it consists of a variety of sub-objects with different textures and shapes (face, hands, clothes, ...). Nevertheless, designing an algorithm for the delineation of agricultural parcel objects is non-trivial, both via traditional image segmentation as well as with deep learning techniques. The model needs to correctly dismiss undesired image objects, differentiate between adjacent fields with nearly similar spectral properties or barely visible borders, and correctly delineate field objects with multiple homogenous areas (e.g. caused by varying soil properties within a single field). Overall, the three main challenges for such an algorithm are:

a. the heterogeneous and volatile geography of agricultural fields, b) the specific properties of satellite imagery, c) inaccuracies in the available ground truth datasets. These challenges will be discussed in more detail below.

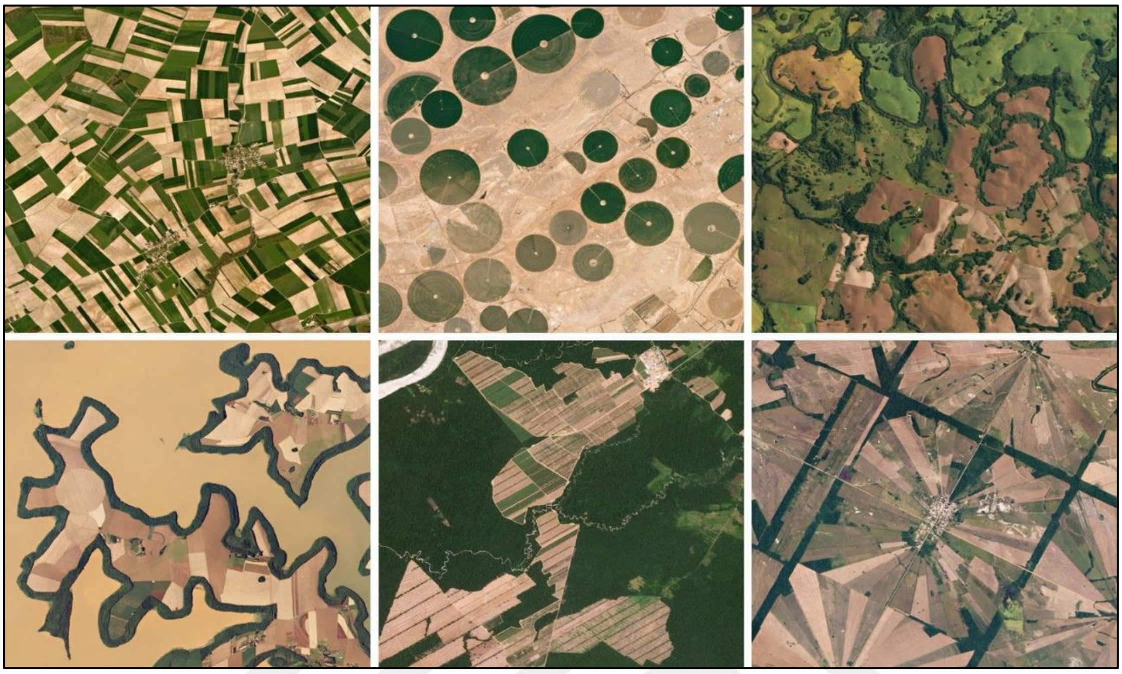


Figure 1.2: Heterogeneous geography of agricultural areas [4].

Field Geography. Shape, size, texture, color, and other physical characteristics of agricultural fields can vary, both from a ground-level perspective and in satellite imagery, depending on numerous physical and human geographical factors. The cultivated species, plant status, topography, soil properties, weather conditions, cultivation methods, and other human influences are all potential sources of variation as mentioned in [5]. It can be difficult to accurately distinguish between multiple field objects due to these local variations. Because of this, it is challenging to develop robust algorithms that can recognize various field instances without overfitting to the particular scene. In addition to variations within a single scene, satellite images may reveal even more pronounced variations within classes. This is true for images taken in various agricultural regions, at various times or during various growing seasons, or under various lighting and atmospheric conditions as mentioned in [6].

Satellite Imagery. In high resolution satellite images, problems can also be caused by local background clutter objects and ground shadows, e. g. from farm machines or utility poles. Locally different atmospheric and lightning conditions (e.g. cloud shadows) can have similar effects. However, unlike for natural photos, image distortions due to viewpoint and distance variations of the imaging sensor are usually insignificant: Satellite image acquisitions are mostly tasked at nadir angle and the satellite height variation is mostly negligible due to the large distance to the imaged earth surface as mentioned in [7].

Ground Truth Data. The precision of the ground truth data sets that are utilized as validation and/or training data for automated algorithms presents yet another obstacle. The majority of these datasets are the result of manual tracing of field boundaries. However, the used imagery and additional data are extremely important for manual tracing of image objects. Additionally, it is a highly subjective task that, depending on the operator's priorities, will unavoidably result in ambiguities and inaccuracies as mentioned in [8]. Existing property parcel data that could be indistinguishable from the imagery alone can be used to supplement the ground truth dataset. Furthermore, due to unreported changes in land use during the growing season, a single field parcel polygon may reveal multiple subfields. Because of these limitations, even a very powerful automated algorithm can only get as good as the ground truth data, but it will never be completely accurate.

1.3. TRADITIONAL AUTOMATED APPROACHES

Several automated and semi-automated methods for the delineation of agricultural parcels from remote sensing imagery have been proposed in the literature. Most of these traditional methods rely on image segmentation techniques applied to high-resolution, multispectral imagery of different band specifications and geographic regions. This includes image segmentation based on edge detection [9], image value gradients [10], deformable “snake” algorithms (adapting to vector contours) [11] and textural properties. Another approach leverages multi-temporal data by combining vegetation indices and edge detection as mentioned in [12]. These studies generally use manually selected features or parameters, which requires a priori knowledge of the scale, physical appearance or distribution of the fields in the scene. No training data is required. Many studies additionally employ region growing as well as post-processing techniques to refine the field detection process. Researcher in [13] use a machine learning approach to iteratively merge selected adjacent super-pixels, with the merge criteria determined by a supervised Random Forest classifier via various spectral indices and texture features. The methodology requires training data, manual feature selection and potentially further post-processing for the completed delineation of independent field parcels.

1.4. DEEP LEARNING FOR COMPUTER VISION AND REMOTE SENSING

Starting in 2012, deep learning, a new area of machine learning, lead to breakthrough advances, especially in computer vision research. Computer vision aims to enable computers to extract and analyses image information in order to gain higher-level understanding of an image. In recent years, deep learning has also enabled groundbreaking results in other fields of machine learning, such as machine translation [14], speech recognition [15], reinforcement learning, and robotics. For a comprehensive review see also in [16]. The term "deep learning" was coined by the use of neural networks (non-linear statistical models) with many layers, so-called "deep" neural networks. By building up a hierarchical structure of more and more abstract representations of the input data, neural networks are able to make sense of complex datasets and use that knowledge to make predictions about similar data. Shallow neural networks have been around since the 1960s and have seen alternating phases of hype and disinterest by the research community. Key factors for the successful training of more powerful, deep neural networks were the increased computing power due to the use of graphics processing units (GPUs), the introduction of several key techniques (e.g. improved network parameter initialization) and a massive increase in the amount of available training data.

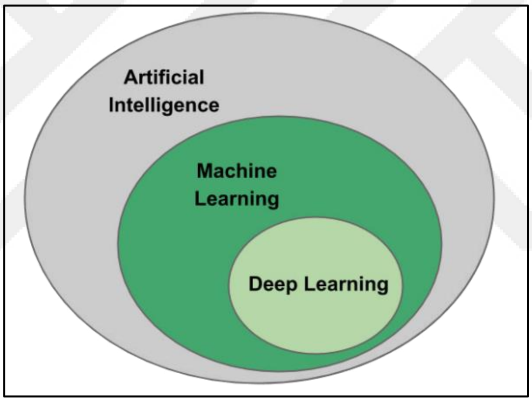


Figure 1.3: The field of deep learning [14].

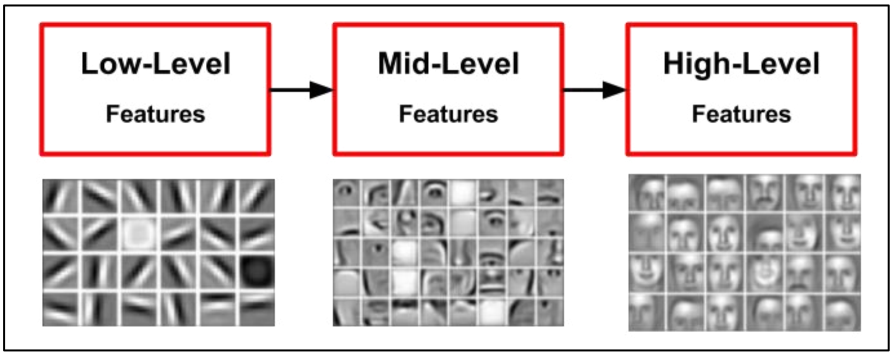


Figure 1.4: Feature hierarchy of image objects [16].

As mentioned above, the deep learning revolution began in 2012 with the introduction of deep convolutional neural networks (CNNs) for the computer vision task of image recognition. CNNs are roughly inspired by the research of Hubel & Wiesel in 1959: After discovering that some neurons in the cat's visual cortex get excited when the cat is shown images of edges of a particular orientation, they proposed a concept of information processing in the visual cortex. In their hypothesis, neurons that are low in the visual hierarchy scan for low-level image features (e.g. oriented edges). These neurons feed to other cells that pick up on more complex features (e.g. the geometric shape of an eye by combining multiple edges), which in turn construct high-level features (partial or full image objects like a face). Convolutional neural networks, which also rely on feature hierarchies and a layered architecture, are a powerful tool for image recognition, the computer vision task of finding a single label for an image from a set of predefined categories. No manual feature engineering is required; the network is able to use raw image pixel data. Supervised training on large, annotated image datasets automatically tunes the CNNs to recognize and process relevant image features. In this way, the CNN maps the input image pixels to abstract feature representations and eventually to class probabilities. The adaption and extension of the CNN architecture has also made deep learning the new standard for more complex computer vision tasks like object detection, semantic segmentation and instance segmentation.

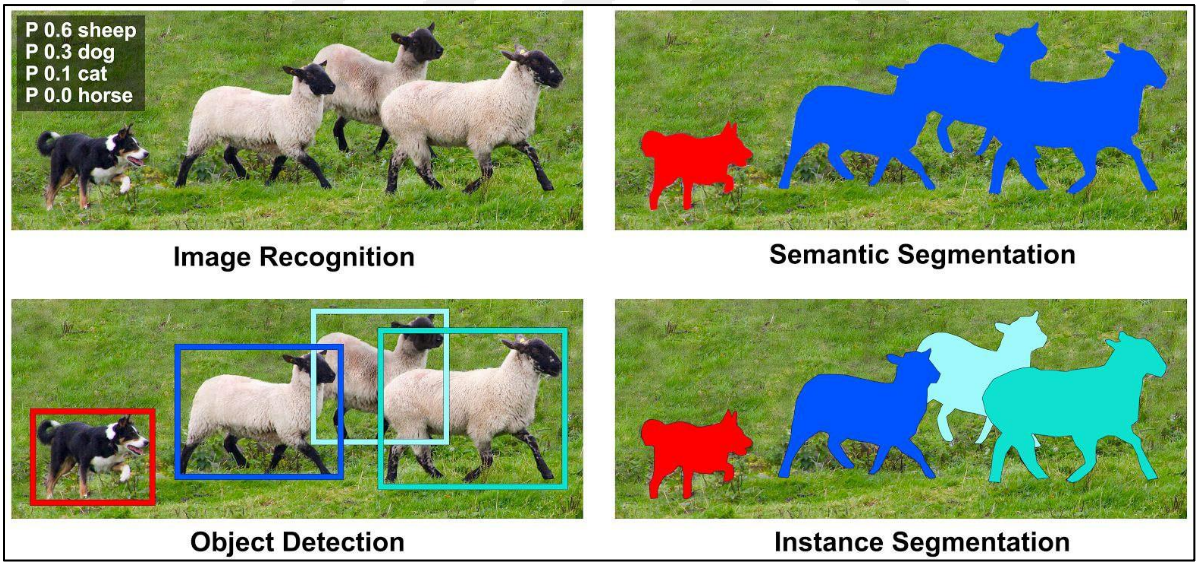


Figure 1.5: Four of the most important computer vision tasks [17].

The appropriate computer vision task for the segmentation and classification of agricultural fields via deep learning is instance aware semantic segmentation (IAIS) (or simply instance segmentation). IAIS delineates and classifies image objects while identifying instances of the same class of objects as mentioned in [18]. Object detection (predicting the bounding box and class of a variable number of image objects but not segmenting them) and semantic segmentation (labeling each image pixel with a semantic category label but not distinguishing between object instances of the same category) are at the intersection of instance segmentation and object detection. If agricultural fields were completely surrounded by areas of various land cover classes, it would be possible to locate individual instances of agricultural fields using semantic segmentation as mentioned in [19]. However, agricultural field objects frequently have touching boundaries and are frequently in close proximity to one another. Semantic segmentation would result in "clumped" polygon objects in this scenario. These connected or overlapping instances of the same class can be distinguished using instance segmentation. In the field of remote sensing, widespread adoption of deep learning has not yet occurred. However, considering the rapidly increasing volumes of earth observation imagery and geospatial data, it promises to be a great fit for many remote sensing applications. Recent studies have explored the potential of deep learning on satellite and aerial imagery for image recognition, semantic segmentation, and object detection as mentioned in [20]. To the author's knowledge, no existing publication has leveraged deep learning instance segmentation to segment and classify objects in remote sensing imagery.

1.5. AIMS AND THESIS STRUCTURE

This thesis aims to show the potential, advantages and challenges of deep learning instance segmentation for the automated delineation and classification of parcels of agricultural land using satellite imagery of medium resolution. Digitized agricultural fields and cloud-free Sentinel-2 imagery (RGB bands) serve as training data for the project's fully convolutional neural network architecture, which was adapted from [21]. In the thesis, two experiments of increasing complexity are presented: First, the model is taught to divide all instances in an agricultural field. The model is trained to simultaneously segment and classify the instances of crops in the field in the second experiment. There are no significant pre- or post-processing steps performed (multi-temporal data or complex vector refinement, which are frequently utilized in other approaches, are not taken into consideration in this thesis), with the exception of some preprocessing to ensure that the data are compatible with the model. Instead, the method emphasizes a straightforward input-to-output mapping. The current objective of establishing a baseline result would be lost, though any additional pre- or post-processing techniques would undoubtedly add value in the future. Also, when considering that the human brain is very successful at delineating field parcels even from simple RGB images, these additional layers of complexity should (in principle) not be necessary to yield promising results.

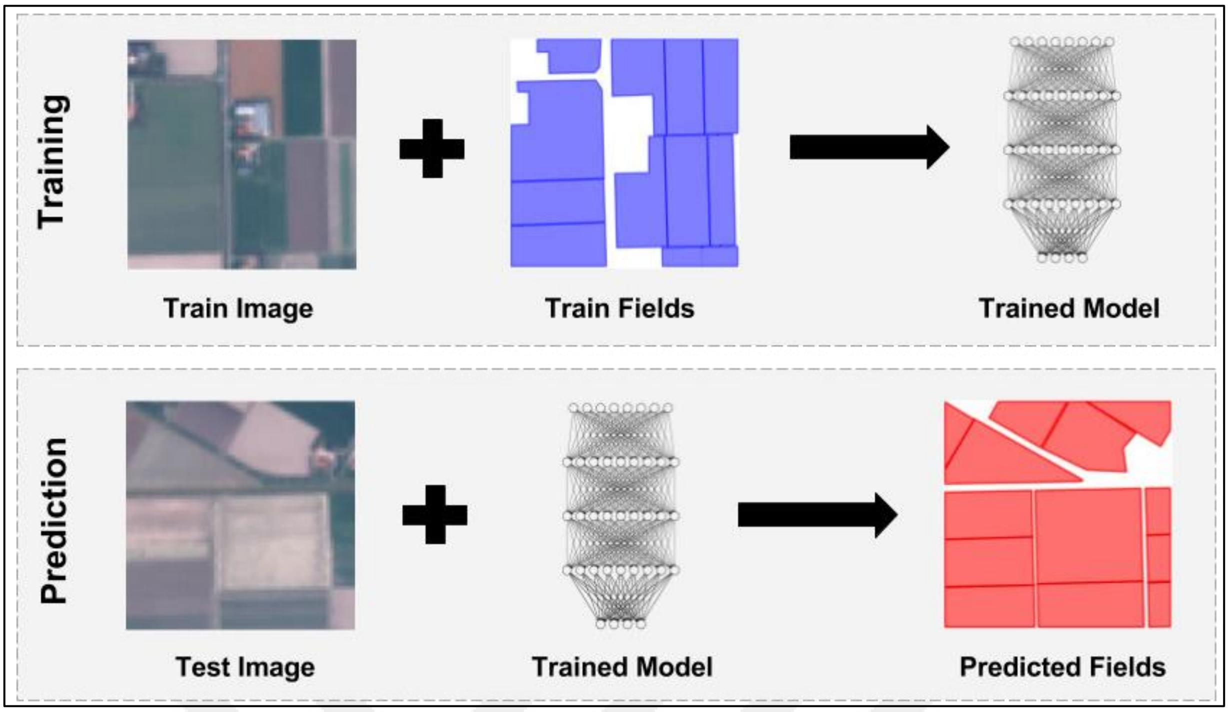


Figure 1.6: High-level overview of Instance segmentation training and prediction.

1.6. THESIS STRUCTURED

The thesis is structured as follows: Chapter 2 introduces the necessary deep learning basics to provide an understanding of the Instance segmentation task. It establishes the concepts of neural networks, image recognition via convolutional neural networks (CNN), semantic segmentation via fully convolutional neural networks (FCN) and object detection. Building on top of these concepts, the chapter then presents the functionality and state of the art of deep learning instance segmentation. Chapter 3 describes the study area, the training data and the methodology of preprocessing the satellite imagery and geospatial data. Also, the implementation and training characteristics of the FCIS model are introduced. Chapter 4 presents and discusses the results. Chapter 5 suggests possible additions and improvements over the current approach. Chapter 6 gives a conclusion.

2. RELATED WORK

2.1. NEURAL NETWORK BASICS

Information processing models known as artificial neural networks (ANNs) are roughly based on the hierarchical and interconnected structure of biological neurons, which are the brain's fundamental signal processing units. Neural networks typically learn by example, which means that the network is set up for a specific task (such as image recognition) through supervised learning on training data. For data clustering applications, neural networks can also be trained through unsupervised learning; however, these methods are not taken into consideration here. ANNs can be trained on raw input data (such as image pixel values) without the need for specialized feature engineering. Because of these properties, neural networks are able to carry out difficult-to-solve, complex prediction tasks that require rule-based methods. The structure, operation, and training of neural networks are discussed in greater detail in the following section.; for a more comprehensive review see [22].

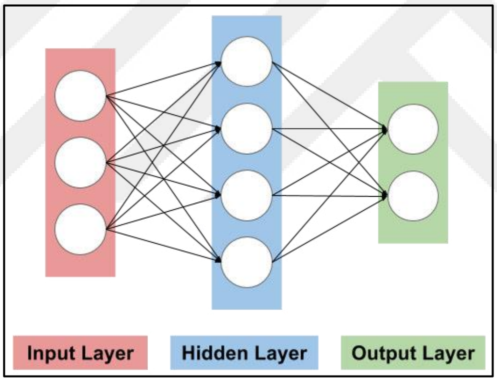


Figure 2.1: Simple neural network with input layer, one hidden layer and output layer, Neurons are depicted by circles, neuron connections by arrows [22].

A neural network is often depicted as stacked layers of neurons. Layers between the input layer (which stores the data input, e.g. image pixel values), and output layer (storing the prediction result) are called hidden layers**.** When the network contains two or more hidden layers, it is usually called a deep neural network. Each neuron in a hidden layer is "fully connected" with all neurons in the adjacent layers via weighted connections. When performing a prediction (forward pass through the network), the network processes the input data through the network by carrying out a series of matrix operations. Each neuron receives the outgoing signals of all neurons in the previous layer, integrates and evaluates them, and passes on a new signal to the neurons in the next layer as mentioned in [23].

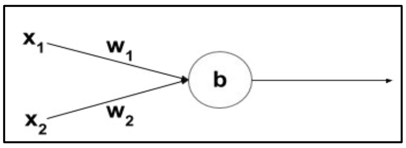


Figure 2.2: A simple perceptron.

The most basic form of a neural network is the perceptron (Fig. 2.2), which has a single neuron. The perceptron's incoming signals x_1 and x_2 are multiplied with the respective connection weights w_1 and w_2. Both products are summed and added to the neuron's bias b. The bias represents a threshold value of the neuron. The perceptron's output signal y is binary: 1 if the evaluated result of the neuron is positive, 0 if it is negative. More generally, the signal processing of a neuron in a multilayer neural network can be described by:

$$y = \sigma(\sum_i x_i w_i + b) \tag{2.1}$$

A nonlinear activation function (such as the sigmoid, tanh, or ReLU function) is applied to the result for each neuron in a hidden layer by adding the product of all incoming signals xi and their respective connection weights w_i to the neuron's bias b. A non-linear activation function is necessary for the network to learn intricate input-output mappings. The neural network can only learn linear mappings by employing a linear activation function, regardless of its number of layers or neurons. The neurons in the subsequent layer receive the output signal y following the application of the activation function. The signals are processed throughout the layers of the network by repeating this calculation for each neuron, resulting in increasingly abstract internal representations of the initial input data. The prediction output, such as a distribution of class probabilities, is created by the final network layer as mentioned in [24].

All network parameters are typically initialized at random. As a result, the network is unable to make any accurate predictions at the outset. Through supervised learning, the network is trained to perform an initial prediction (forward pass), evaluate the prediction performance (backward pass), and tune the neuron's weights and biases (iteratively) until the network can predict the desired output for a particular input. As outlined, the network needs to evaluate how well the parameters have already been optimized during the training process. After each forward pass, the predicted output and the expected label are compared. This is done by a loss or cost function (e.g. mean squared error or cross entropy), which quantizes the quality of the result. Usually, the loss is averaged over a number of samples in the training data set. The training then aims at minimizing the loss by updating the network parameters during each backward pass, via some form of gradient descent optimization. Recursively going back through the network layers, the gradient of the loss with respect to the network parameters (neuron weights and biases) is calculated using the computationally efficient backpropagation method. The gradient $\partial L/\partial w$ is a vector in the high-dimensional parameter space of the network, pointing in the direction of the maximum increase of the loss. Consequently, in order to decrease the loss, the weights are updated by subtracting the gradient (i.e. moving the parameters into the opposite direction of the gradient in the parameter space):

$$w' = w - \eta \frac{\partial L}{\partial w}$$

Here, η is the learning rate, a hyper-parameter of the network that controls the magnitude of the parameter update. Besides this simple form of (stochastic) gradient descent, many modified optimization methods are used nowadays (e.g. RMS prop or Adam). Due to computational limitations, a forward-backward pass through the network does usually not contain all training samples, but is performed iteratively for mini batches of the dataset. Training for one epoch means that the network has seen all training samples once.

2.2. CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE RECOGNITION

Despite their ability to perform straightforward image recognition on small-scale input images, regular neural networks are inefficient when processing large amounts of image data. One fully connected neuron in the first hidden layer of the network would already have 196608 weights for a medium-sized input image with dimensions of 256x256x3 (height, width, and number of color channels). Training the entire network is very inefficient and prone to overfitting due to the large number of parameters.

A type of neural network that is better suited to image data is called a convolutional neural network (CNN or convnet). Its architecture makes use of the image's spatial structure, making it possible to train the parameters of the network more effectively. Initially, image recognition was the primary application for CNN architectures as mentioned in [25]. CNNs can easily be adapted and repurposed for more complex computer vision tasks like object detection, semantic segmentation, or instance segmentation despite their inherent inability to make spatial predictions.

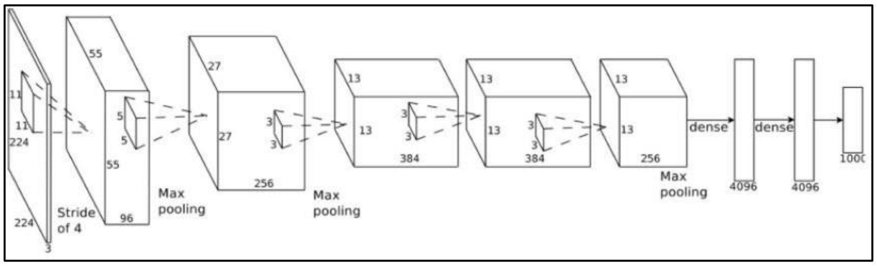


Figure 2.3: Architecture of the Alexnet network [25].

Pioneered by [26], the modern concept of CNNs was introduced and popularized by the Alexnet or Supervision model. Alexnet (fig. 9) appeared as the winning entry of the 2012 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). The challenge aims to predict the labels of 100,000 test images in 1000 categories, with 1.2 million images in the training dataset. Until 2012, the challenge was dominated by approaches leveraging Support Vector Machines. With an accuracy of 85%, Alexnet outperformed all other entries of 2012 and the previous years, beating the 2012 competition's second place by a huge margin of 11%. This prestigious win had an enormous impact on computer vision research and helped spark the deep learning revolution. Since 2012, each year's ILSVRC challenge winner has been a deep learning model.

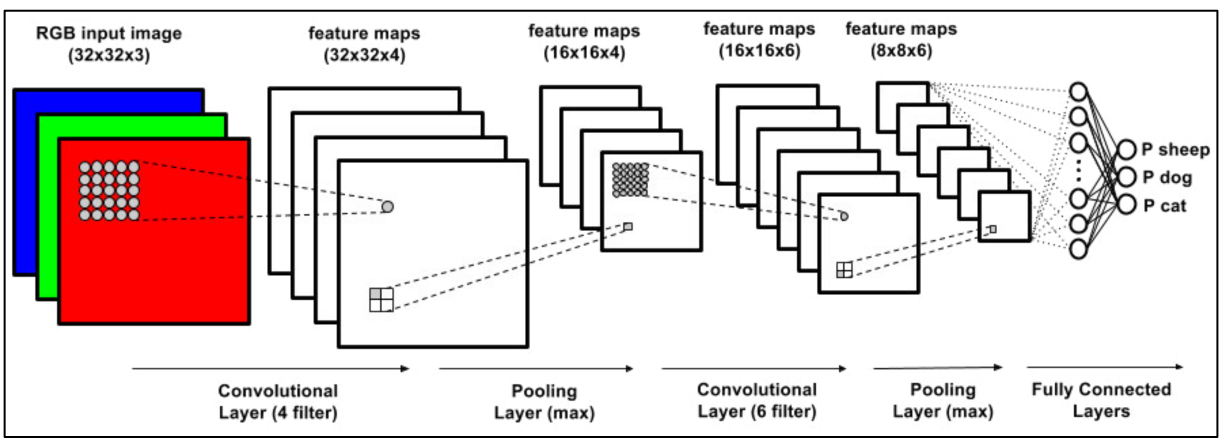


Figure 2.4: Schematic functionality of a convolution neural network [27].

A typical image recognition CNN architecture uses a classifier on the most abstract feature vectors to extract increasingly abstract and computationally efficient image features from an array of image pixels before producing an array of class probabilities. A convolutional layer is at the heart of each feature extractor stage as mentioned in [28]. It applies a number of image filters to the data, each of which is looking for a particular image pattern. A filter activation map, also known as a feature map, is created by looking for these features in the input image by the filters of the first convolutional layer. The produced feature map of the preceding convolutional layer is used as input by the filters of each subsequent convolutional layer. In most cases, the filters of the lower convolutional layers look for straightforward, generic features like color gradients or sharp edges in a particular orientation. Later convolutional layers learn to register increasingly abstract and complex features (polygons, textures) by combining previous layers' knowledge as mentioned in [29]. The network eventually develops filters for entire image objects by incorporating increasingly abstract representations of the initial input data. The patterns of the filters are not fixed; rather, they adjust themselves as the network is trained to recognize relevant image features. In order to speed up the prediction and training process, each feature map undergoes an activation function, also known as non-linearity, and a pooling layer, which reduces the number of parameters. The final stage of the network divides the input image into one of the predetermined categories based on the high-level features that were extracted by the previous convolutional layer. The CNN's various stages are further described below.

Convolution: The convolutional operation can be described as a moving window that slides across the data. Its size is also called kernel size. Assuming a stride of 1 (i.e. the filter moves across the data in steps of one pixel) and zero-padding of 2 (i.e. the input image boundaries are extended by two rows and two columns of zeros), the moving window is applied to every possible spatial location of the RGB input image in the first network layer. Considering the CNN depicted in fig. 10, the image size is 32x32x3. The moving window filter of size [5x5x3] extends over the full depth of the input volume, i.e. it considers the information of all image bands at the same time. For each position in the input image, the moving window computes the filter activations via a dot product between the filter kernel and the raw image pixel values. The resulting 2-D feature map of size 32x32x1 is a representation of the original input image in terms of how strongly the filter was triggered by a particular image region. Closer matches with the filter pattern result in higher activations. A moving window stride bigger than 1, or no zero padding around the input volume yields a compressed feature map. This can result in a faster, but also less accurate CNN, because the feature maps contain less information. The convolutional operation can also be described via artificial neurons: The image filter is then represented by a grid of neurons, each connected to a small, localized pixel region in the input image, called the local receptive field as mentioned in [30]. The filter pattern that each neuron is looking for (the convolutional kernel) is determined by the neuron's weights and biases. Every neuron in the grid shares the same weights and biases. This sharing of parameters enables the detection of a specific feature across the entire image. Each value in the feature map is calculated as the product between a neuron's weight and the RGB-pixel values in the image region that the neuron is connected to, plus the neuron's bias.

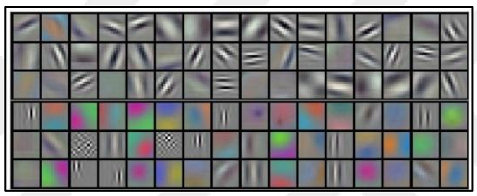


Figure 2.5: Visual representation of the 96 convolutional filter kernels learned by the first convolutional layer of the Alexnet CNN [30].

In practice, each convolutional layer uses multiple feature filters. Similar to the input image, each convolutional layer is arranged in 3 dimensions (width, height, depth). The width and height indicate the moving window dimensions or the arrangements of the neurons in each filter, the depth indicates the number of filters in the convolutional layer. Each filter takes into account the full depth of the 3D-input volume (original input image or stacked feature maps from a previous convolutional layer) as mentioned in [31]. The convolutional operation converts the 3D-input volume to a new 3D volume of activations. As an example, a convolutional layer with 4 filters is able to detect 4 different features across the whole input volume, producing an activation volume of 32x32x4, no matter which depth the input volume has.

Activation and pooling: Before a feature map is processed by the next convolutional layer, it is usually passed through a non-linear activation function and downsampled via a pooling layer. The activation function, in this example a Rectified Linear Unit (ReLU) layer, sets all negative values in the feature map to 0. The dimensions of the feature maps array remain the same as mentioned in [32].

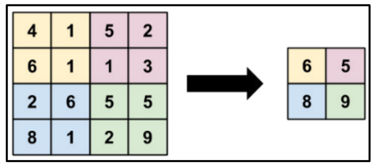


Figure 2.6: Max pooling operation (2x2, stride 2) [32].

The most common operation performed by a pooling layer is max pooling (fig. 12). From each 2x2 square in a feature map, only the maximum value (the strongest activation) is kept. While the array depth remains the same, the array height and width is reduced to half (e.g. in the first feature extractor stage, the pooling layer decreases the array size from 32x32x4 to 16x16x4). The downsampling operation aggregates the activation information, while discarding some detail of the spatial information. This makes the number of parameters and the computations in the network more manageable. After passing through multiple feature extraction stages, the original input image array is reduced into a much smaller volume.

Classification: In the classification stage of the CNN, the 3D-volume of high-level feature maps from the last convolutional layer (respectively last pooling layer) is transformed to a 1D volume of class scores, each representing the probability that the original input image belongs to that class. For that, the feature maps are vectored, i.e. stretched out vertically and concatenated. By performing matrix multiplication of the neuron's weights and the connected filter activations, a number of fully-connected layers and a final softmax layer generate a dedicated score between 0 and 1 for each of these classes. The sum of all class scores is 1.

Training: Training of the CNN works similar as for a regular (feedforward) Neural Network. The CNN is trained via stochastic gradient descent. The aim is to reduce the error between the predicted and expected image classes. While the activation and pooling layers implement fixed functions that do not change during training, the weights and biases in the convolutional layers as well as the parameters in the classification stage are initialized randomly and tuned during the training process. After the initial forward pass through the network, the loss is calculated from the current prediction as mentioned in [33]. Subsequently, in a backwards pass through the network, the gradients are calculated via backpropagation. Then, the network parameters are updated by subtracting the gradients. The parameters are adjusted iteratively, so that over time, the network is able to detect specific image patterns that are common in images of the desired classes. By tuning the weights of the fully connected layers, the network learns which of the extracted high-level features correlate most with which class.

Successfully training a CNN from scratch (with random initialization of all layer parameters) requires datasets of enormous size.

Therefore, Transfer Learning: is often employed to tune existing networks for new use cases. Here, the parameters of a different network, pre-trained on a huge reference dataset within the same application category, are reused. Then, only the classification layers of the network are retrained, so that the network is able to predict the classes of the actual training dataset (equivalent to training a linear classifier on top of the extracted features). Depending on the dataset size and how much it differs from the training data for the initial network, it is also possible or required to fine-tune the network itself: Fine-tuning means to adjust the pre-trained parameters of the last or even all convolutional layers. Transfer learning and fine-tuning can speed up the training process enormously and enable learning on small datasets. They are especially successful in computer vision applications, because the image features recognized by the earlier feature extraction stages (e.g. edges or color gradients) are so generic that they fit almost any dataset.

Following Alexnet [34], several popular, more complex CNN architectures emerged. ZF Net [35], winner of the 2013 ILSVRC challenge, is a modified version of the original Alexnet architecture with tweaked hyper-parameters. GoogLeNet [36] won the 2014 ILSVRC challenge. It introduced average pooling and an Inception module, which enables the combination of pooling and multilevel convolutional operations at different filter sizes, while keeping the number of required model parameters comparably small. VGGNet [37] focused on a simple but powerful architecture (it only performs 3x3 convolution and 2x2 pooling operations throughout the entire network), with 13 convolutional layers and 3 fully connected layers. The model took second place in the 2014 ILSVRC competition. Resnet or Residual Network [38] introduced the concept of deep residual layers via "skip connections". The connections between non-adjacent convolutional layers enable training this very deep network architecture. It won the ILSVRC 2015 challenge and is currently one of the most widely used models for transfer learning.

2.3. FULLY CONVOLUTIONAL NEURAL NETWORKS FOR SEMANTIC SEGMENTATION

Image recognition models yield a single distribution of class probabilities for the entire image. In contrast, semantic segmentation requires a distribution of class probabilities for each image pixel. Conceptually, CNNs can be leveraged for pixel-wise labeling in a patch-based approach via a moving window (CNNs require an input of fixed size). Then, the center pixel of each patch is assigned the predicted class scores for that image region. However, this approach is computationally expensive, as the CNN is applied on each image region independently.

The fully convolutional neural network (FCN) is based on the convolutional neural network architecture, but provides a more efficient solution for the prediction of pixel-wise class labels as mentioned in [39]. CNN architectures can be easily adapted to FCNs, and FCNs can still use transfer-learning with pre-trained CNN parameters. FCN introduces three major changes to the basic CNN architecture: 1) It replaces the CNN's fully connected layers by convolutional layers. This enables the network to predict a class score map for each class for images of arbitrary size. The score maps of each class can then be merged into the desired "all-classes" score map by taking the maximum class probability of each pixel. Compared to the patch-based approach, the FCN method is a lot faster, because computations can be shared between overlapping patch regions. Due to the down-sampling operations in the network architecture, the resulting output maps are coarse and require upsampling to the original image size. 2) Consequently, FCN introduces an in-network upsampling stage, the so-called deconvolutional layer. 3) The detail in the coarse class score map can be improved by adding "skip connections", which incorporate less downsampled features from earlier convolutional layers into the final output. Some concepts of the FCN network are outlined in more detail below.

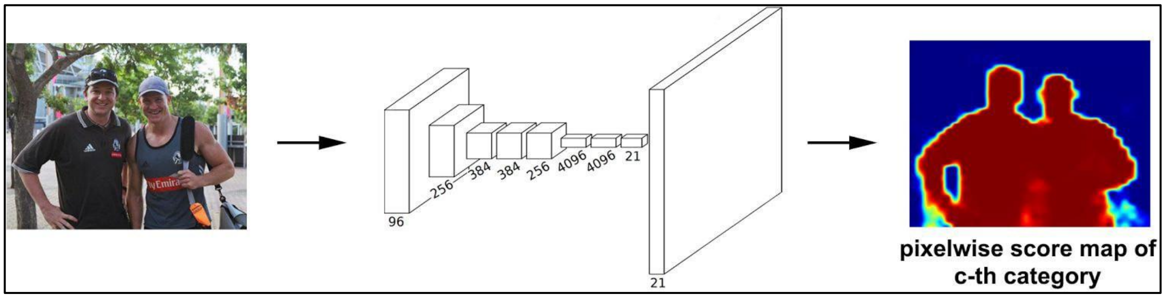


Figure 2.7: Class score map output and network architecture of FCN for semantic segmentation [39].

Convolutionalization: Due to the fixed amount of neurons, a fully-connected layer requires an input of fixed size and outputs a 1D-array. Instead, the filter kernels of a convolutional layer can be applied to an input volume of arbitrary size, and produce a 3D-array of spatial maps. Ultimately, both a fully-connected and a convolutional layer compute weighted sums of their input values. This means that every fully connected layer can be replaced by a convolutional layer with a specific setup: The number of filters in the convolutional layer has to equal the number of neurons in the fully connected layer, and the receptive field has to be of the same size as the height and width dimensions of the input volume to the fully connected layer. E.g., take a fully connected layer with 4096 neurons, that processes feature maps of size 8x8x256 from the previous convolutional layer (i.e. 32 times downsampled from an input image of size 256x256). This fully-connected layer can be replaced by a convolutional layer with 4096 filters (stride 1, padding 0) and a receptive field of size 8x8x256. As the feature maps and filters have the same size, each filter will only be applied once, computing a single weighted sum. The 4096 resulting feature maps of size 1x1x4096 have the same number of network parameters and computational requirements as the fully connected layer. For n classes, a final 1x1 convolutional layer (trained via a pixel-wise loss function) reduces the feature maps to n class scores.

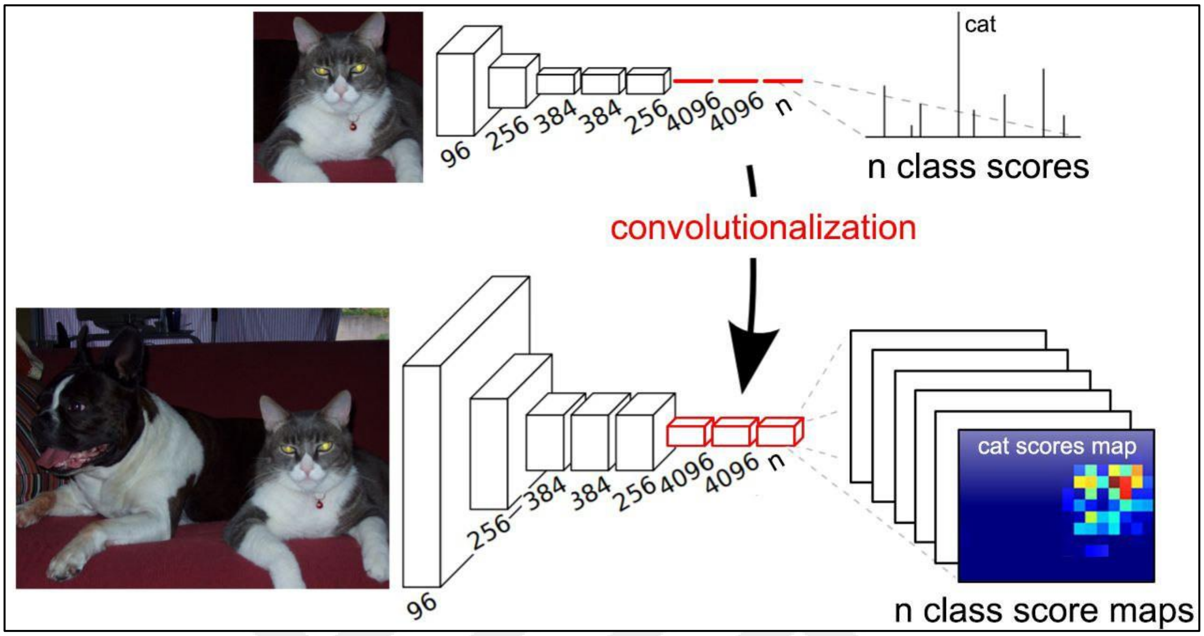


Figure 2.8: Convolutionalization of fully-connected to convolutional layer. CNN outputting class scores (top) and CNN with convolutionalized layers outputting coarse spatial class score maps (bottom) [40].

By convolutionaliazing the final, fully-connected layers in an image recognition CNN, the entire network can be treated as a series of filters. It can be applied to images of arbitrary size. This is the first main idea behind the FCN. When using the FCN on an image, the resulting output is not just a 1D-array containing one class score per class, but a 3D-array that contains, for each class, a spatial "heatmap" or 2D-array of pixel-wise class scores (fig. 14). E.g. if the filter kernels of size 8x8x256 of the last convolutionalized layer look at feature maps of 12x12x256 (i.e. 32 times downsampled from an input image of size 384x384), the filter kernels fit the feature maps five times in length and width, resulting in n spatial class score maps of size 5x5. All in all, the FCN maps the 3D input image to a 3D output of class score maps. This configuration is a lot faster than applying a CNN on every possible image location in a patch-based approach, as the parameters of overlapping patch regions are shared in the network.

Deconvolution: Due to the downsampling operations in the network architecture, the class score maps are smaller than the input image size and exhibit limited spatial detail. They require upsampling to the original image resolution. Instead of fixed bilinear interpolation (each upsampled pixel is the weighted average of the four diagonally neighboring pixels in the input), the FCN network uses deconvolutional layers (also called transposed convolution, backward strided convolution, upconvolution or half strided convolution) for learnable, fast, in-network upsampling as mentioned in [41]. The deconvolutional layer at the end of the network performs the upsampling of the class score maps by carrying out the inverse operation of a convolutional layer. An upsampling of factor f by the deconvolutional layer with stride f corresponds to a convolutional operation with stride 1/f.

Skip connections: A more detailed or denser segmentation can be achieved by incorporating the higher resolution feature maps of earlier convolutional layers via "skip connections" as mentioned in [42]. In addition to the feature maps of the last convolutional layer, the feature maps of the two previous convolutional layers (downsampled by factors of 8, 16 and 32 respectively) are directly forwarded to the final 1x1 convolution classification stage. For each class, this yields three class score maps of different size. The score maps resulting from the feature maps of the earlier convolutional layers are of higher spatial resolution, but were generated from less abstract features, and thus contain "worse" semantic information. The class score maps are then upsampled (by their respective downsampling factor) to the original image resolution via deconvolutional layers. For each class, summing the respective three score maps yields a final output class score map with good semantic information and detail. While most subsequent semantic segmentation models adopted the paradigm of fully convolutional networks, they introduced even more sophisticated approaches to improve the spatial detail in the class score maps as given in [43].

2.4. OBJECT DETECTION VIA REGIONAL CNN

The goal of object detection is to locate the classes and bounding box locations of image objects. The total number of items is unknown. By suggesting a number of rectangular box regions in the input image to see if any of them correspond to actual image objects, object detection is conceptually enabled. This could be accomplished by first suggesting boxes at every possible scale and location, and then applying a CNN to the image content of each box separately. However, this method would require a lot of performance as mentioned in [44].

With convolutional neural networks, R-CNN (Regional CNN) makes it possible to detect bounding boxes with greater accuracy. In addition, Fast-RCNN and Faster-RCNN derivatives 2016) significantly speed up the training and testing of models. At the time of this writing, Faster-RCNN is the most popular object detection framework and the foundation of numerous instance segmentation models as mentioned in [45].

There are two important metrics for evaluating object detection and instance segmentation models: The proportion of instances of a bounding box or segment that are correctly delineated is known as the average precision (AP). The Intersection-over-Union (IoU) ratio is the criterion for accurate delineation. This is the area of union divided by the area of intersection between the predicted and ground truth bounding boxes. For a more in-depth explanation of these evaluation metrics.

R-CNN with Selective Search: R-CNN first searches for a manageable number of class agnostic region proposals, i.e. a set of approximate image regions which are most likely to contain image objects. To find these regions, R-CNN uses the Selective Search algorithm [46], but is also compatible with other proposal methods. Selective Search greedily merges adjacent super-pixels that share textures, colors, or intensities at multiple scales. R-CNN considers only the bounding boxes of these proposals.

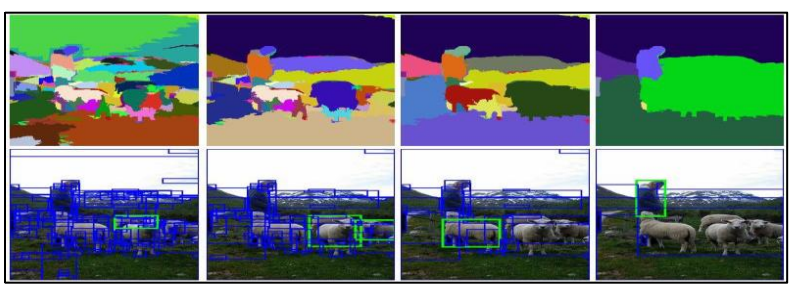


Figure 2.9: Superpixels (top) and corresponding bounding boxes (bottom) of Selective Search applied at different scales. The green bounding boxes corresponding to entire image objects show the necessity of multiple scales [46].

The image content of each region proposal bounding box is then warped to a standard square size to fit the CNN specifications. The CNN is separately applied to each of these image snippets to extract their convolutional features. For each class (plus a dedicated background class), a linear SVM that is specifically trained on convolutional features of objects of that class is applied to the convolutional feature vector of each proposal. For each region proposal or region of interest (RoI), this yields a class score for each of the c+1 predefined classes. The highest class score determines the proposal's class label and is also used as a detection score, indicating the confidence in the detection as mentioned in [47].

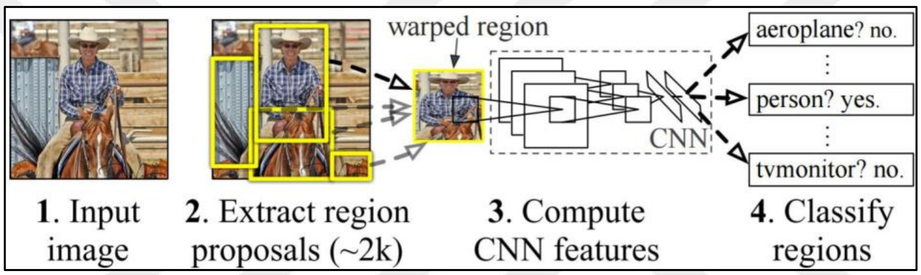


Figure 2.10: R-CNN stages from input to region extraction [47].

Redundant proposals, for which an overlapping, higher scoring proposal of the same class exists are then eliminated via non-maximum suppression (NMS). This method loops through every region proposal and compares it with all remaining region proposals of the same class. If region proposal A and another proposal B have a significant IoU overlap and the detection score of A is lower than that of B, region proposal A is rejected and removed from the loop. The remaining proposals are considered as the actual predicted image objects. Finally, to reduce localization errors and ensure tighter fitting bounding boxes, a class-specific, simple linear regression is applied on the remaining, classified region proposal. This step mildly corrects their bounding box coordinates as mentioned in [48].

Fast R-CNN: In the multi-stage pipeline of R-CNN, the CNN is applied separately on every region proposal. Because many region proposals are partially overlapping, R-CNN performs a lot of redundant calculations. The main achievement of Fast-RCNN is the introduction of Region of Interest Pooling (RoI-Pool) for more efficient feature extraction. The classification stage of the

CNN is removed, and the network is applied only once to the full image. Then, the feature vectors of the region proposals are extracted via RoI-Pooling directly at the corresponding locations on the output feature maps of the last convolutional layer as mentioned in [49]. Subsequently, the network performs classification and bounding box regression in parallel. RoI-Pool is much quicker than applying the CNN multiple times and on partially overlapping regions. Instead of training and applying three different models separately (CNN feature extraction, SVM classification, bounding box regression), Fast R-CNN combines these tasks into one joint network with shared parameters, making the network much more efficient. The training uses a multi-task loss and updates all network layers jointly as mentioned in [50].

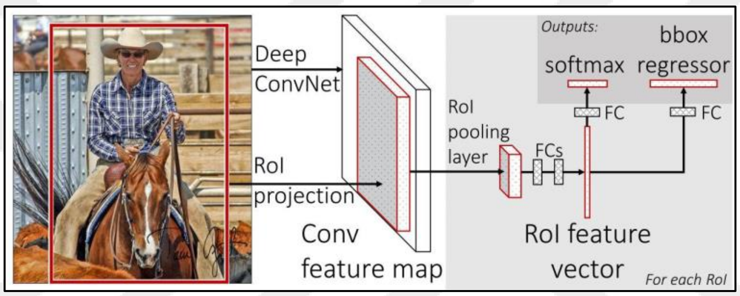


Figure 2.11: Fast-RCNN architecture from input image to feature extraction [50].

Faster R-CNN: The final iteration of R-CNN, Faster R-CNN focuses on accelerating the region proposal step, introducing the fully convolutional Region Proposal Network (RPN) in replacement of the fairly slow, external Selective Search algorithm. The class agnostic region proposals are instead generated directly on the convolutional feature maps, which is much more efficient and enables end-to-end training of the whole object detection model as mentioned in [51]. While moving a sliding window over the feature maps of the whole image, each window position is checked with predefined anchor boxes (reference bounding boxes of common sizes and aspect ratios, by default 3 anchor boxes at 3 different scales). For each anchor box, the RPN then regresses the proposal box coordinates (reg layer) and an “objectness” score via two parallel, convolutionalized fully-connected layers. The objectness score indicates how likely the box contains an image object or belongs to the background category. The region proposals are ranked by their objectness scores, and non-maximum suppression is applied with an IoU score of 0.7. From the remaining proposals, the top n ranked region proposals are used as the candidates for the object classification and bounding-box regression branches of Fast-RCNN. By using anchor boxes, fewer candidates need to be evaluated. Also, because the convolutional feature maps are shared between the RPN and Fast R-CNN, the proposals can be created at minimal computational cost as mentioned in [52].

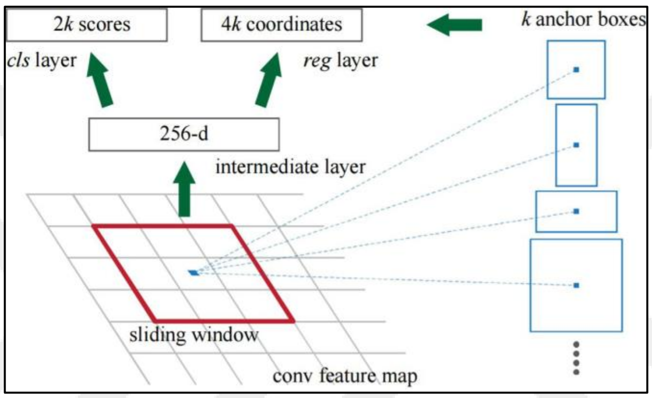


Figure 2.12: Region Proposal Network (RPN) of Faster-RCNN [52].

2.5. INSTANCE SEGMENTATION

2.5.1. Overview of Task and Models

The process of categorizing and segmenting distinct object instances is referred to as instance-aware semantic segmentation, or simply instance segmentation. Object detection (predicting bounding boxes and classes of objects without segmenting them) and semantic segmentation (labeling each image pixel with a label for a semantic category without distinguishing between objects in the same category) are two aspects of this technique that are combined as mentioned in [53].

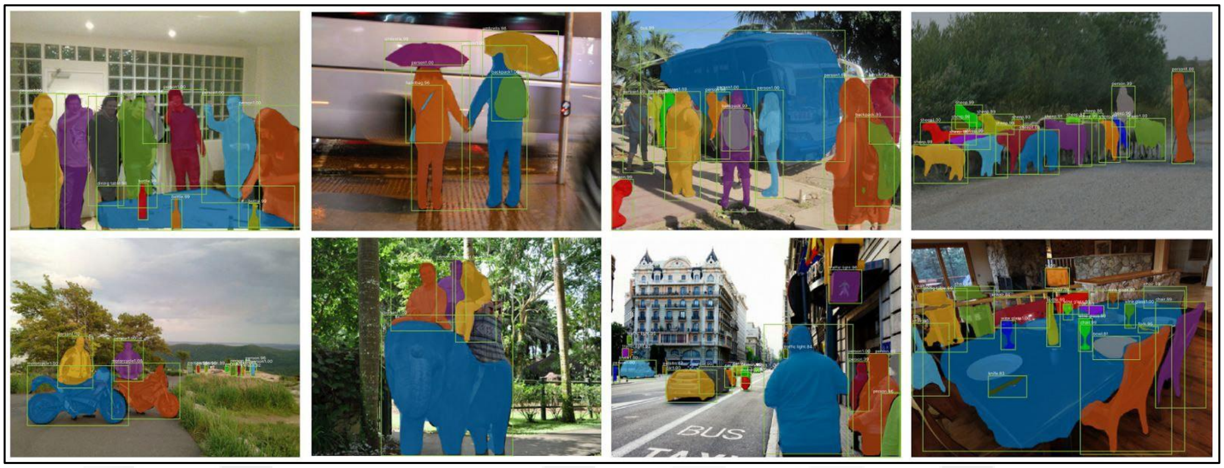


Figure 2.13: Mask-RCNN results on the COCO test dataset. Displayed are object instances, bounding boxes and object class labels [53].

Microsoft Common Objects in Context (COCO), with its 80 classes and 300 k images, and Pascal Visual Object Classes (VOC), with its 20 classes and 5 k images, are two of the most widely used reference datasets for instance segmentation [54].The Mask-RCNN model, which is currently the most effective for deep learning segmentation, on the COCO reference dataset, achieves 37.1 mean average precision (mAP) at [.5,.95] IoU (the average AP across all classes, averaged over IoU thresholds from 0.5 to 0.95 in 0.05 steps).19).FCIS comes in second with 29.2 mAP@ [.5,.95] IoU. In the practical portion of this thesis, the FCIS model is utilized.

Because the focused image region or object instance can assign distinct semantic categories to the same image pixel, instance segmentation must be performed at the region level. A pixel may, for instance, be part of an image object's object mask (mask foreground), but it may also be outside of another image object's mask (mask background) as mentioned in [55]. This situation cannot be distinguished by regular CNNs and FCNs when applied to the whole image, because convolution is translation invariant: A pixel will receive the same foreground/background scores, regardless of the relative position of the pixel in the focused context. Inspired by the success of region proposal techniques for object detection, most instance segmentation models use per-RoI application via segment proposal as mentioned in [56].

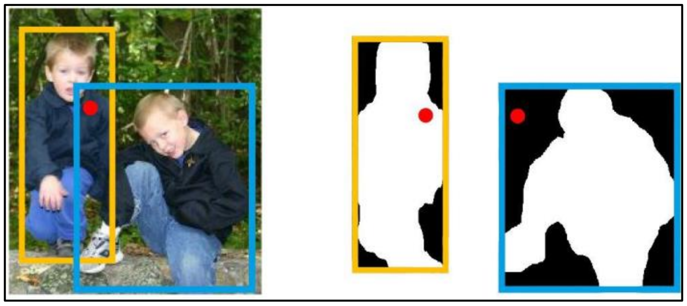


Figure 2.14: Different semantic categories for the same pixel when focusing different instances. The same pixel (red point) belongs to the mask foreground (white) in the yellow instance, but the mask background (black) in the blue instance [55].

Earlier instance segmentation models relied on existing bottom-up segment proposal algorithms that extract and merge super-pixels, e.g. Selective Search or Multiscale Combinatorial Grouping. The category-agnostic segment proposals are then classified using R-CNN or Fast R-CNN variants, much like an object detection workflow. Later on, neural networks that learn to predict more accurate segment proposals, such as Deep-mask or the fully-convolutional approach Instance FCN took the place of the simple proposal techniques.2016. Parameter sharing is not possible in these segment proposals networks, which necessitate a downstream classification network. For more precise object localization, Multipath-net makes use of proposals from Deep-mask and a modified version of Fast-RCNN. The first end-to-end, trainable instance segmentation solution is Multi-Task Network Cascades (MNC) [57]. It first predicts proposals for bounding boxes, which are then used to regress segment masks. Following that, Faster-RCNN is used to classify using both segment proposals and bounding boxes. Despite the fact that MNC's subtasks share convolutional features, each task still relies on the output of the previous subnetwork, necessitating a complex network architecture and training. Recent models attempt to combine segmentation and classification into parallel network architectures rather than performing them sequentially. FCIS2017 is the first fully trainable, end-to-end convolutional segmentation network architecture. For fully convolutional segment proposal prediction, it builds on the concepts of position-sensitive class score maps and object detection. The system as a whole is extremely quick because FCIS efficiently and simultaneously performs object segmentation and classification. RCNN-Mask, the current state of the art for instance segmentation, adds a parallel, fully convolutional network branch for the prediction of RoI segmentation masks to the object detection model Faster-RCNN.The following chapters provide a more in-depth description of some of the most significant models.

2.5.2. Deepmask and Multipathnet

Deepmask introduced the first method of using convolutional neural networks to propose segments. The VGG network, a well-known CNN for image recognition, serves as the basis for the architecture. The input image's convolutional features are extracted after the fully-connected layers of VGG are removed. After that, the Deepmask architecture splits into two network branches. These branches are used in a moving window method to directly apply to the convolutional feature maps of the final convolutional layer at multiple scales. A binary, class-independent RoI segmentation mask is predicted by the first network branch. The likelihood that the RoI is centered on a full-image object is shown by the second branch, which regresses a class-agnostic objectless score. Through stochastic gradient descent, both tasks are trained simultaneously. The logistic regression losses of the two network branches are added up to create the loss function. Deepmask bi-linearly resamples the proposals to the original input image because the segmentation mask is delineated from the downsampled convolutional feature maps. The suggestions that come out of it are usually pretty rough and don't always align exactly with the boundaries of the actual object. The purpose of the optional Sharpmask iteration of the Deepmask model is to enhance the precision and overall fit of the Deepmask proposals for coarse segments. By incorporating information from earlier convolutional layers or less downsampled feature maps, it improves the alignment of the proposal with the image object. Deepmask and Sharpmask's segment proposals are identified and categorized by Multi-PathNet. Several modifications have been made to the successful object detection model Fast R-CNN with RoI-Pooling, which serves as the foundation for the architecture: By cropping the corresponding feature map location at multiple scales, the network creates four "foveal regions" while extracting the proposals' convolutional features using RoI-Pooling. The network can use the object context at multiple resolutions because of this. Additionally, it uses skip connections to incorporate the more in-depth information from earlier convolutional layers for each foveal region. After that, the network uses bounding box regression to better localize the object instances and classifies the segment suggestions. To eliminate predictions of the same object that overlap, non-maximum suppression is applied to the proposals using the objectless scores that Deepmask provides. Fast R-CNN training is similar to that of Multipathnet, but Multipathnet uses an "integral loss" function that applies the loss function at multiple IoU threshold values.

2.5.3. MNC

Multi-task Network Cascades (MNC), which won the COCO-2015 segmentation challenge, is a multi-stage structure of three consecutive VGG networks. The model is end-to-end trainable and shares parameters between its subtasks, but each task is still dependent on the output of the previous subnetwork.

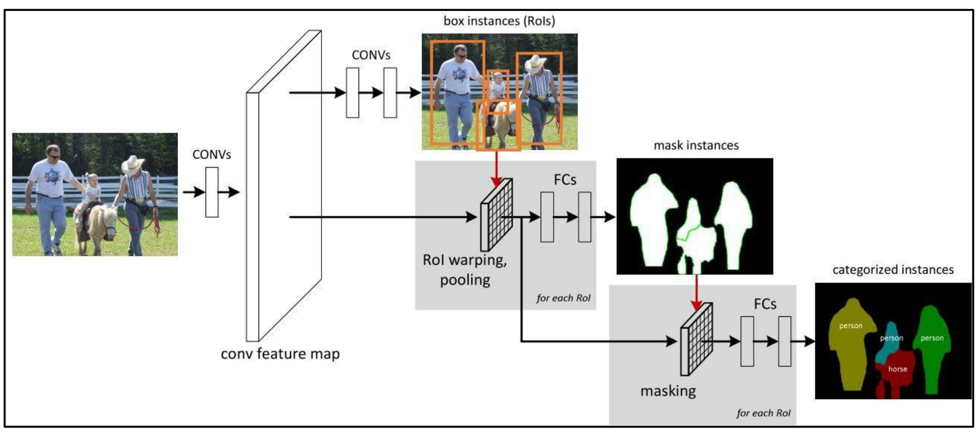


Figure 2.15: Structure of the Multi-Task Network Cascades [57].

Firstly, the region proposal network of Faster-RCNN predicts class agnostic bounding boxes and their respective objectness scores. Redundant proposals are filtered out via non-maximum suppression. The second stage takes the bounding box proposals and shared convolutional feature maps and extracts the feature vectors for the respective image regions via RoI-Pooling. For each bounding box, it predicts a class agnostic, pixel-level mask via binary logistic regression. Compared to Deepmask, which applies the segment proposal prediction at every image position, the preselection of RoIs via the bounding box prediction is computationally more efficient. The final classification stage takes the feature maps of the bounding box proposals and masks out the pixels inside the bounding box that do not belong to the respective segment proposal (mask background; these values are set to zero). The resulting feature maps are limited to the foreground of the segment proposals, which are then used to predict the class scores. The entire workflow can be trained end-to-end via stochastic gradient descent, but the causal relation of the various outputs and subnetworks make the training via backpropagation non-trivial. As the loss of some network stages depends on other stages, the network is trained using a unified loss function.

2.5.4. FCIS

As described above, MNC shares convolutional features among its subnetworks. However, it does not share the per-RoI computation during the final classification step. Also, resizing the RoIs to a fixed size representation (which is required by the fully connected layers) can impair the spatial detail and segmentation accuracy. In contrast, fully convolutional neural networks can be applied to images of arbitrary size and computations can be shared between overlapping RoIs. Therefore, they are much more suitable for spatial prediction. However, like CNNs, regular FCNs are translation-invariant. A possible solution to this problem in instance segmentation is presented by the fully convolutional instance-aware semantic segmentation (FCIS) model. It incorporates partially translation-variant, position-sensitive inside/outside class score maps. The model performs simultaneous detection and segmentation and shares parameters between sub-tasks as well as RoIs, which makes it a fast solution for instance segmentation. FCIS won the 2016 COCO segmentation challenge. It relies and extends upon the main ideas of Instance-FCN for the fully convolutional prediction of segment proposals and R-FCN for fully convolutional object detection. The FCIS model works in the following way: Firstly, the convolutional feature maps for the full image are extracted using the 100 convolutional layers of the ResNet-101 model (the final pooling and fully connected layers are removed). As the downsampling factor of 32 in the last convolutional layer of ResNet produces too coarse feature maps for instance segmentation, the effective filter stride is reduced to 16 pixels. For each of the c+1 classes (the additional class is dedicated to background), two sets of k^2 position-sensitive inside/outside score maps are generated fully convolutionally. The $2*k^2*(c+1)$ score maps represent the pixel-wise probabilities that a pixel is at a specific relative position inside or outside an object instance of the particular category. High pixel values in the top-middle inside score map indicate that these pixels have a high probability to be near the top-middle border area of image objects. A parallel RPN that shares the convolutional features with FCIS proposes RoIs. The RoI area is split by a grid of k x k, regularly distributed bins. For each RoI, one inside and one outside score map is assembled from the respective bins on the sets of inside/outside score maps of the full image. This makes the RoI score maps partially translation-variant: A pixel can receive different score values for different RoIs if the pixel's relative position inside the RoIs is different.

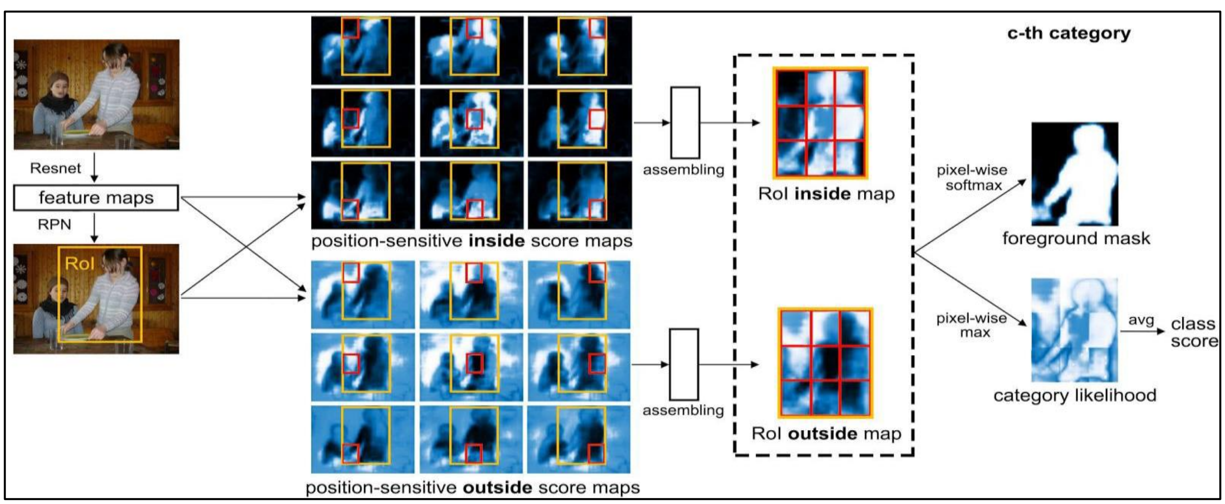


Figure 2.16: FCIS scheme [56].

For each class independently, the assembled RoI-specific inside and outside score map is then combined in two different ways: A pixel-wise softmax operation yields the class-specific foreground probability mask (the pixel has a high probability to belong to an object mask if it has a high RoI-inside and low RoI-outside score). A pixelwise maximum operation yields a class-specific category likelihood score map (high pixelwise probability to belong to the bounding box of an actual image object in case of combined low RoI-inside with high RoI-outside scores or low RoI-inside with low RoI-outside scores). The category likelihood map is then reduced to a single class score via average pooling. A softmax operation on all extracted class scores of the RoI results in a vector of c+1 class probabilities. The highest class probability value indicates the class label of the RoI and acts as the confidence detection score. When a RoI bounding box only partially overlaps an image object, some of the position sensitive bins that are assembled for the RoI score maps are not activated, which leads to a low detection score. To filter out overlapping segment proposals, the RoIs are ranked by their detection score, and non-maximum suppression with an IoU threshold of 0.3 is applied. For the remaining RoIs, the foreground probability mask is selected by the class label and the final segment mask is determined by mask voting: For each remaining RoI, the foreground masks of all RoIs with an IoU overlap of 0.5 or higher (including the ones removed during NMS) are selected. The masks are averaged pixel-wise (weighted by the RoI's detection scores) and then binarized. The introduction of partial translation-variance by the position sensitive score maps improved the AP@0.5 by 13.2 % (tested on the PASCAL VOC 2012 data) over using a translation-invariant baseline FCIS configuration with a single score map with 52.5 AP@0.5. FCIS is trained via stochastic gradient descent with Online Hard Example Mining (OEHM) bootstrapping. OEHM addresses the usual imbalance of many easy, but few hard examples (also due to the imbalance of positive and background RoIs). For each iteration, instead of randomly subsampling RoIs, the SGD modification automatically selects only high-loss examples, which significantly boosts the model accuracy with fewer iterations. The Resnet-101 convolutional layers are initialized with parameters pre-trained on ImageNet. All other learnable layers are initialized randomly. The shared score maps are generated fully convolutionally for the whole input image and correct aspect ratio. Each RoI is evaluated by the detection loss over c+1 categories, the segmentation loss (sum of the pixelwise comparison of RoI foreground probabilities with the ground truth mask, normalized by RoI area) and the bounding box regression loss. All layers after the RoI and score map generation implement simple, fixed functions, enabling very efficient per-RoI computations.

2.5.5. Mask R-CNN

Mask R-CNN is (at the time of writing) the state of the art instance segmentation model and outperforms all other solutions. It takes the object detection architecture of Faster-RCNN (including the region proposal network) and adds a fully convolutional network branch for the pixel-wise prediction of class-agnostic, binary segmentation masks (fig. 23). The segmentation branch is parallel to the bounding box regression and classification branch of Faster R-CNN. All branches share the convolutional features of the input image and are applied to each RoI directly on the convolutional feature maps, which makes Mask-RCNN very efficient. The (multi-task) loss is the sum of the loss of each of the three network branches. For each RoI, Mask-RCNN predicts a binary segmentation mask for each class fully convolutionally. The authors found that with conventional RoI-Pooling of Faster R-CNN, the regions of interest in the convolutional feature maps do not completely align with the exact location of the respective regions in the original image. For the classification branch of Mask R-CNN (and also for bounding box object detection in general), such small pixel misalignments are mostly irrelevant. However, instance segmentation requires pixel-level accuracy to delineate the exact object instances. Even small translations can have a big impact on the evaluation result. They are caused by rounding issues due to the different scales or pixel dimensions of the feature maps and input image. Therefore, Mask-RCNN replaces the RoI-Pool of Faster-RCNN with RoI-Align, which uses bilinear interpolation to exactly align the RoIs with the corresponding regions in the original input image. RoI-Align greatly improves the mask accuracy by 10 % to 50 % depending on the evaluation metric's overlap threshold. To further improve the network accuracy, the convolutional feature extraction stage of Mask R-CNN uses the Feature Pyramid Network proposed by [58], which enables efficient, in-network processing and exploitation of convolutional feature maps at multiple scales. Compared to FCIS, Mask-RCNN also increases the number of RPN anchors and evaluates the advanced Resnext-101 instead of only Resnet-101.

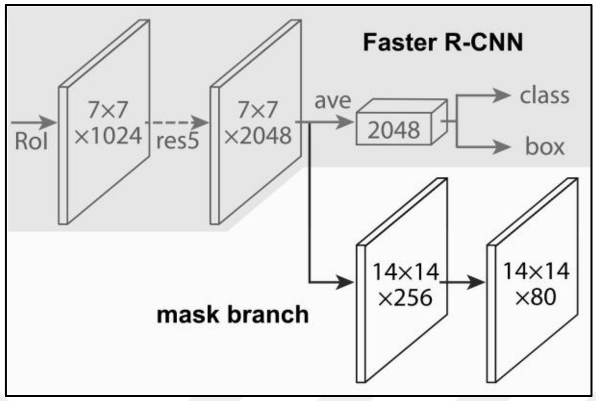


Figure 2.17: Mask-RCNN model with Faster-RCNN architecture (top) and parallel instance mask branch (bottom).

3. METHODOLOGY

The following chapter presents the training data, preprocessing, model training and evaluation methodology that was used for the automated delineation of field parcels. In short, medium resolution satellite images (Sentinel-2, RGB) and the corresponding, georeferenced field boundaries were preprocessed and cut into small image chips to fit the model requirements and available computational resources. Then, a fully convolutional neural network architecture, adapted was trained and tested in two configurations: 1) Segmentation of all agricultural field instances, 2) Segmentation of field instances and simultaneous classification of instance crop classes.

3.1. STUDY AREA AND DATASETS

It is defined by the extent of the Sentinel-2 satellite image mosaic and corresponding digitized agricultural parcels that were used as training and validation data. With an area of 20900 km2 (190 km x 110 km), the region covers the eastern part of the mainland, adjoining parts of the Baltic Sea and the second biggest Danish island Funen. The area is characterized by dominant agricultural land use, temperate climate and mostly flat terrain. The mosaic is made up of two cloud-free Top of Atmosphere (TOA) Sentinel-2 images of 10 m spatial resolution, captured at the beginning of the growing period in May 2016. The image tiles originate from the same Sentinel-2 image acquisition and overlap partially in north-south direction. The agricultural field polygons are from the 2016 ‘Marker’ dataset (MFLF 2017), which is part of the European Union Land Parcel Identification System (LPIS) initiative. The dataset contains nearly 600 k field parcel polygons, each attributed with a unique identification number, one of 293 crop type classes and the field area. 249298 parcels are contained in the study area, from which 159042 with a mean area of 52451 m2 were used in this work (some classes are removed during preprocessing).

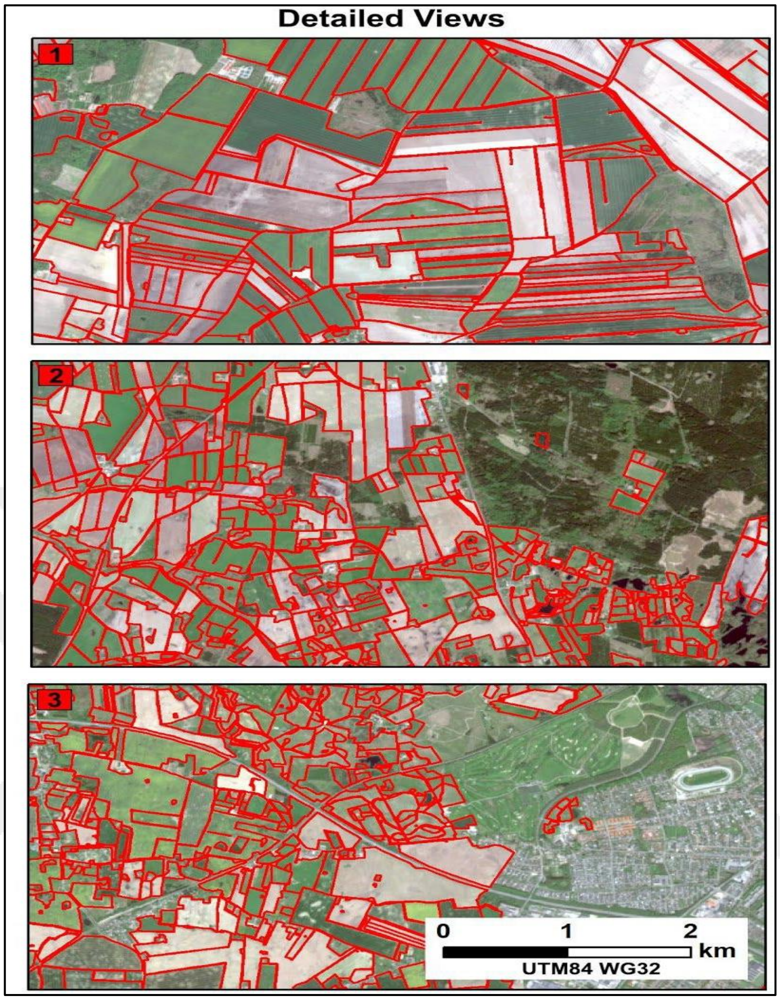


Figure 3.1: Study area and detailed views showing the variety of agricultural parcel shape and geographical settings: (1) Large, regular field parcels in flat terrain, (2) Irregular shapes near forested areas (3), Urban setting with recreational areas.

3.2. PREPROCESSING

The Sentinel-2 images were retrieved and mosaiced via Google Earth Engine (GEE), a cloud-based platform that offers access to geospatial datasets with analysis capabilities. All mosaic image bands except the red, green and blue channel (10 m spatial resolution) were discarded. The image values were clamped to their 0.25 to 97.75 percentile to preserve the dynamic range against outliers. Further preprocessing of image and polygon data was carried out with Python 2.7.8, mostly relying on the Python libraries rasterio, geo-pandas and shapely.

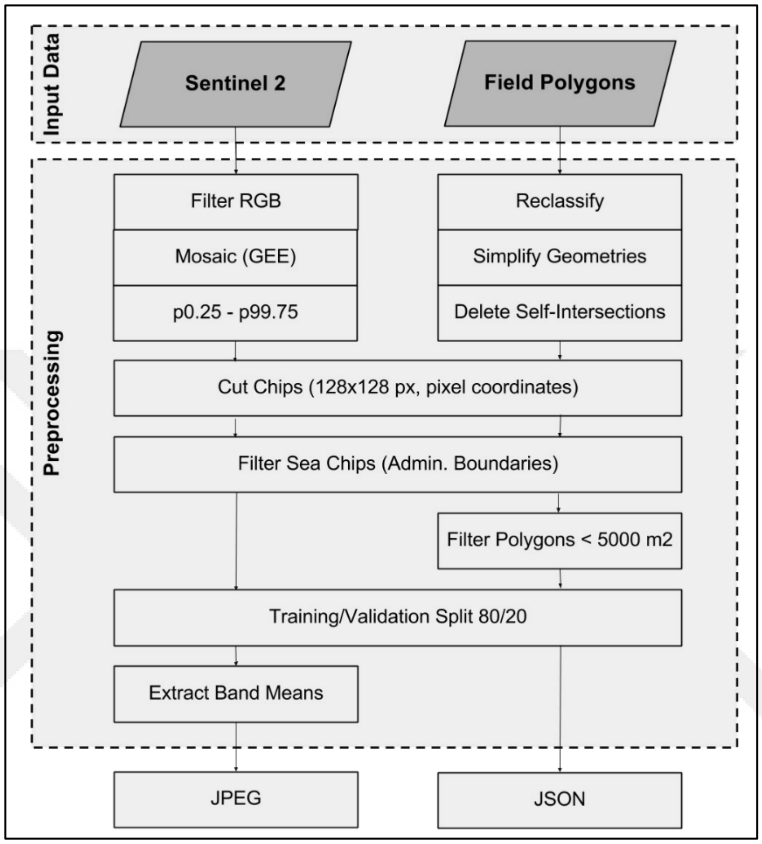


Figure 3.2: Preprocessing workflow**.**

The parcel classes provided in the original dataset were reclassified to the five classes spring cereal, winter cereal, maize, grass-fields and other fields. Parcels of classes not related to agricultural usage, e.g. natural or permanent environment, forest, recreational areas, or wasteland, were removed. The distribution of (reclassified) crop classes and parcel areas inside the study area is displayed in Figure 3.3.

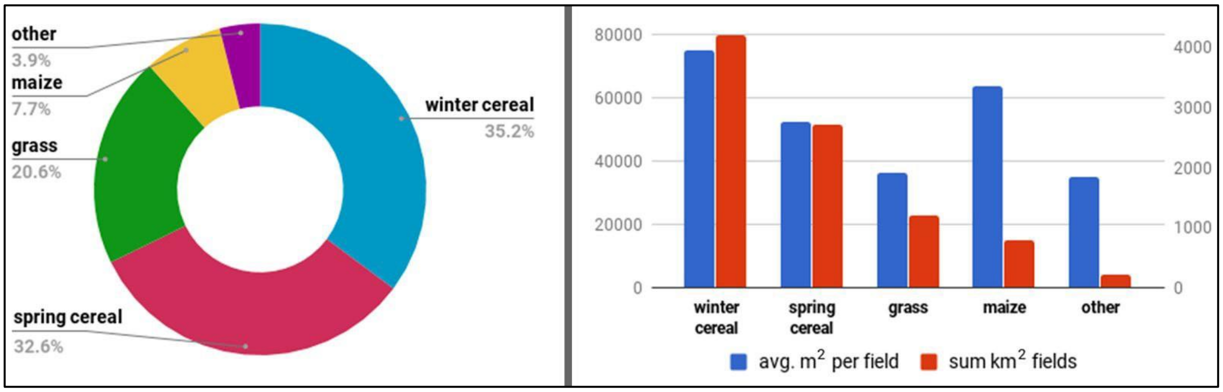


Figure 3.3: Distribution of (reclassified) parcel classes (left) and average area per class and sum of area per class (right).

After reclassification, the polygons were projected to the coordinate reference system of the image mosaic and intersected with the mosaic's bounding box. The remaining polygon geometries were simplified within a small distance tolerance of each point of the original geometry. The original topology was preserved. The original dataset contained polygons with geometry self-intersections, which were resolved by zero-distance buffering. The polygons were then transformed to pixel coordinates of the image mosaic.

The land area of the Sentinel-2 image mosaic was cut to 11320 chips of 128x128 pixels, each representing an area of 1.6384 km2. Chips outside of the high resolution Global Administrative Areas (GADM) boundaries (GADM 2016) were filtered out. The agricultural fields dataset was intersected with the same grid and transformed to pixel coordinates of the 128x128 pixel chips.

Field polygons that overlap chip boundaries of the grid are split in at least two smaller polygons. As this can result in very small field instances, all polygons with an area below 5000 m2 (corresponding to 50 image pixels) were filtered out. This resulted in 196704 field polygons with a mean area of 45583 m2. All image chips that contain at least one ground truth polygon were then randomly split into 80 % training data (8439 chips) and 20% validation data (2110 chips). A secondary validation dataset contains the same validation chips in addition to all 771 "empty" image chips that do not contain any ground truth polygons. To normalize the image chips to zero mean during training, the mean value of each image channel was calculated based on the training data. As all pixel values have the same range from 0 to 255 and similar relative scales, the standard deviation was not normalized. The image chips were then saved in RGB JPEG format without any compression. The polygon data was saved in the COCO JSON annotation format.

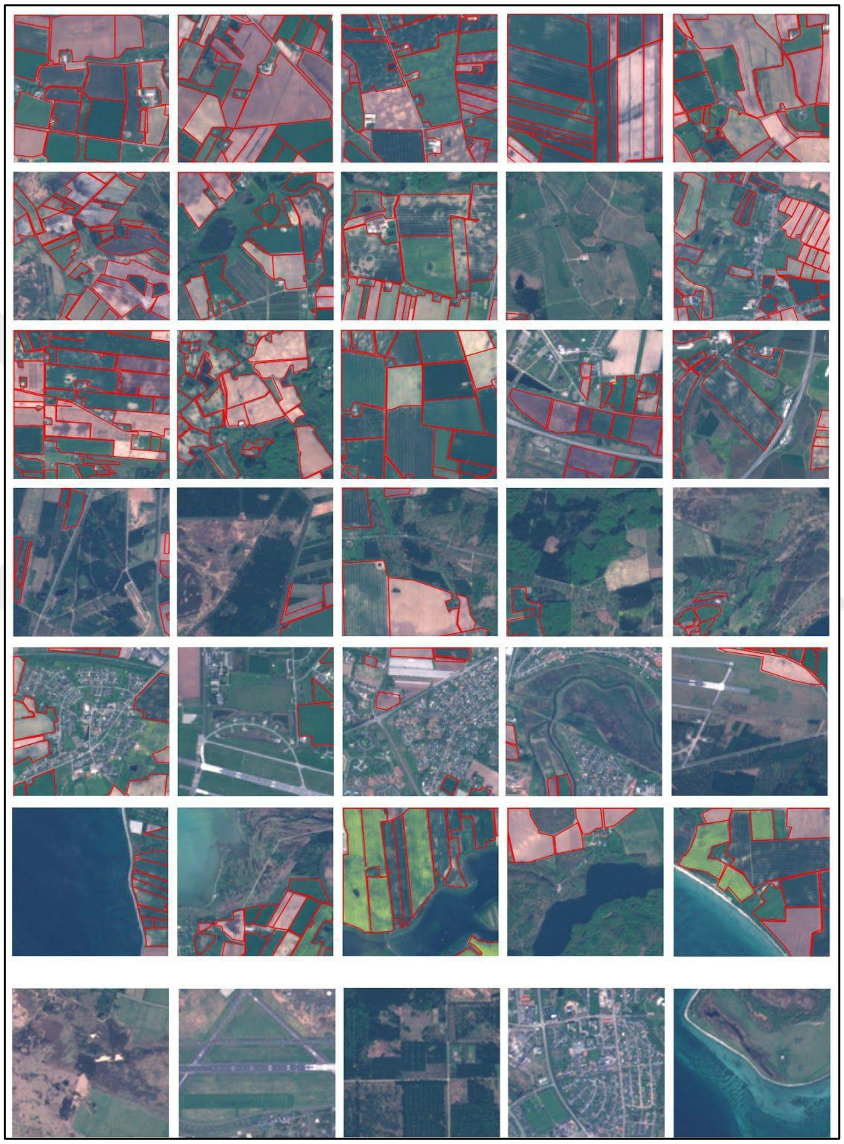


Figure 3.4: Example 128x128 image chips with ground truth agricultural field polygons after preprocessing. Chips in the last row do not contain ground truth polygons.

3.3. MODEL IMPLEMENTATION DETAILS

The instance segmentation task is carried out using the FCIS model. The architecture and functionality of FCIS. Although Mask R-CNN is the current state of the art instance segmentation model, at the time of writing the authors have not yet released their implementation. The baseline FCIS model in combination with horizontal flip data augmentation was trained and evaluated on the training respectively validation satellite image chips (RGB channels, 128x128 pixel) and corresponding Geo data of agricultural fields. Two different configurations were applied: 1) All fields treated as a single class (referred to as “single-class”), 2) each field annotated with one class label from the five reclassified crop classes (“multi-class”). The successfully trained algorithm takes in the validation satellite images and outputs instances of agricultural field parcels. The model implementation is based on MX-Net, a lightweight, efficient deep learning framework. Training and evaluation were performed on a Tesla k80 graphics card of an Amazon Web Services (AWS) EC2 p2.xlarge instance. Following the considerations and recommendations for best practice deep learning instance segmentation in the literature, extensive testing of the internal FCIS-network parameters and training data properties (e.g. image chip dimensions, RGB vs. Near Infrared-R-G channels) was carried out. Eventually, most of the chosen model hyper-parameters follow the default settings of the original FCIS model implementation as they provide a good balance of prediction accuracy, training and inference time.

During model training, FCIS only considers image chips with existing ground truth instances, i.e. empty image chips are filtered out. The image chip values (range 0-255) are mean-centered by subtracting the per-channel mean values, which were extracted during the preprocessing. The order of the training data chips is randomly shuffled before each epoch, as a regular order could potentially bias the neuron parameters. Single-scale training is carried out, the shorter side of each image chip is resized to 600 pixels. Each image chip and the corresponding Geo data is horizontally flipped during the training, the RoI specific results are averaged. This data augmentation helps making the model more accurate and robust to overfitting.

The model was trained end-to-end for 8 epochs, via stochastic gradient descent with OEHM bootstrapping (around 3 hours per epoch on the Tesla k80). Due to OHEM bootstrapping, each mini-batch consists of a single image chip. The number of iterations is therefore equal to the number of training image chips times two (due to horizontal flipping). The SGD used a learning rate of 0.0005 and momentum of 0.9 (i.e. the parameter update takes into account previous update steps to avoid getting stuck in a local minimum). The learning rate was reduced to 0.00005 during the fifth epoch. Using transfer learning, the ResNet layers for the extraction of the convolutional features were initialized with the parameters of the ImageNet pre-trained ResNet-v1-101 model. Other learnable layers in the FCIS network were initialized with random values uniformly sampled between -0.1 and 0.1. During the forward pass, two sets of seven position-sensitive inside and outside score maps per category are generated. For shared convolutional features between FCIS and the RPN, a joint training scheme similar to Faster-RCNN is used: The training alternates between tuning the region proposal and the detection/segmentation task. The FCIS loss is defined by the mask loss, detection loss and bounding box loss. The RPN uses twelve anchors (four scales at three aspect ratios). A RoI is considered a positive example when it has an IoU score greater 0.5 with the nearest ground truth object.

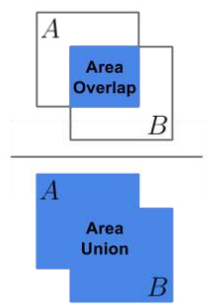
Inference time is about 0.25 s per image chip. Non-maximum suppression with an IoU threshold of 0.3 is applied to the RoIs. For each of the remaining RoIs, mask voting over all instances with an IoU overlap of 0.5 or higher (including the RoIs removed during NMS) is applied. The pixel-wise averaged masks (weighted by the RoIs' detection scores) are binarized at a threshold of 0.5. Each final instance mask is encoded as a binary array in the dimensions of the input image chip. The binary masks are converted to polygon format via Python opencv. The original topology is preserved. The resulting instance polygons and classes are exported to JSON format.

3.4. EVALUATION METRICS

The predicted field instances are evaluated by comparing them with the ground truth polygons in the validation dataset. The reported metrics include the average precision at different IoU thresholds (0.5, 0.75, [0.5:0.95]) and for three different object scales (S, M, L) as well as the corresponding recall values. These metrics are described in more detail below. The evaluation is carried out using the Python MS COCO API.

For object detection and instance segmentation models, both the training and the evaluation stage require the metrics Intersection-over-Union (IoU) and average precision (AP): When comparing the position and appearance of a predicted object bounding box (object detection) or object segment mask (instance segmentation) with the respective ground truth instance, one needs to differentiate between a correctly predicted object (true positive or TP) and a falsely predicted object that does not exist in reality (false positive or FP). This differentiation is based on the IoU score of the two polygons. Commonly, an IoU score of 0.5 or above is considered a true positive. IoU, also called Jaccard Index, evaluates the similarity and dissimilarity of two polygons A and B. IoU is calculated by dividing the area of overlap between both objects by their area of union. For an IoU score of 1 the regions are exact matches, for a score of 0 they do not overlap at all. IoU is scale invariant.

$$IoU(A,B) = \frac{|A \cap B|}{|A \cup B|} = \tag{3.1}$$



The instance segmentation method predicts a number of instances, each with a confidence value, the detection score. For each class independently, the instances are then ranked by their detection score. Each instance is compared with each ground truth object. Based on an IoU threshold (e.g. 0.5), each predicted instance is labeled as a TP or FP. For each ground-truth object, only the predicted instance with the highest detection score is labeled as the true positive: All other predicted instances that fulfill the IoU criterion but have lower detection scores are labeled as false detections. If a true positive is found, both the ground truth and predicted object are removed from the loop. If a ground truth object is not matched by any predicted instance, it can be considered a false negative (FN). The number of false negatives results from the number of ground truth objects minus the number of true positives. From the TP/FP assignments, the precision and recall for each instance in the ranked list can be calculated. Precision (P) is the percentage of all predicted instances that match a ground truth object. Recall (R) is the percentage of ground truth objects that were correctly predicted. Each calculation takes into account the number of TP/FP/FN up to this point in the list, thus also includes the assignments from all prediction with higher detection scores. Per image chip, a maximum of 100 instances with the highest detection scores are considered. An example calculation of precision and recall for multiple predictions and ground truth objects.

$$P_{class\ c} = \frac{\#TP(c)}{\#TP(c) + \#FP(c)} \tag{3.2}$$

$$R_{class\ c} = \frac{\#TP(c)}{\#TP(c) + \#FN(c)} \tag{3.2}$$

The precision values of all instances are averaged within 101 bins of 0.01 recall. When the resulting precision and respective recall values are plotted, the area under the resulting precision-recall curve gives the average precision (AP) for class c. To reach high AP values respectively have a high area under the curve, the method needs to have high recall and high precision. AP and AR are calculated for different IoU thresholds, e.g. 0.5 and 0.75. AP@[0.5:0.95], which is the standard COCO segmentation challenge metric, averages the AP at IoU thresholds from 0.5 to 0.95 in 0.05 steps. For the assessment of multi-class problems, the AP over all classes or mean average precision (mAP) is used.

In addition to the overall AP and AR, the "per chip" AP and AR metrics are calculated over the predicted field instances of each chip independently. This enables a reasonable placement of the instance segmentation success for a particular image chip. Only instances with a detection score above the selected detection score threshold are taken into account. The per chip metrics cannot be directly compared to the overall AP and AR metrics (as each chip contains a different amount of instances, the average of all per chip APs slightly deviates from the overall AP). Analogous to the COCO dataset specifications and to make the results more comparable, the evaluation and analysis focuses on chips with existing ground truth objects. For completeness, the results on a secondary validation dataset that includes all empty land area chips are briefly described as well. Given the spatial resolution of the Sentinel-2 images and the size of the agricultural field polygons, the area-specific COCO evaluation categories do not correspond to the field delineation task. Instead, the S, M and L category area limits were selected by the distribution of field areas in the training data: S (< 162 pixel), M (162 - 322 pixel) and L (> 322 pixel). This roughly corresponds to 43.1% of training field polygons < 0.025 km2, 46.5 % from 0.025 km2 - 0.1 km2 and 10.4 % > 0.1 km2.

4. RESULTS

4.1. EXPERIMENT 1: SINGLE-CLASS

First, the results for instance segmentation in the single-class setting are reported. In this experiment, which is the main focus of this work, all agricultural fields were set to the same class index. Therefore, the sole task was to predict the polygons of all field instances in a given image patch, but not to classify them based on their crop type.

4.1.1. Metrics

Overall, the validation dataset contains 2110 image chips. Running the model on all of these chips yields 100004 field proposals. On average, the model predicts 47.4 proposals per chip. This number is approximately normally distributed, with a minimum of 10 and a maximum of 76 polygons per chip. The confidence detection scores for these predictions range from 0.001 to 0.989. Visualizes the detection scores in more detail. The selected cutoff detection score of 0.8 (red line in Figure. 4.1) leaves 42640 predicted field instances after filtering. In comparison, the validation dataset contains a similar number of 39160 ground truth polygons (an average of 18.6 polygons per chip).

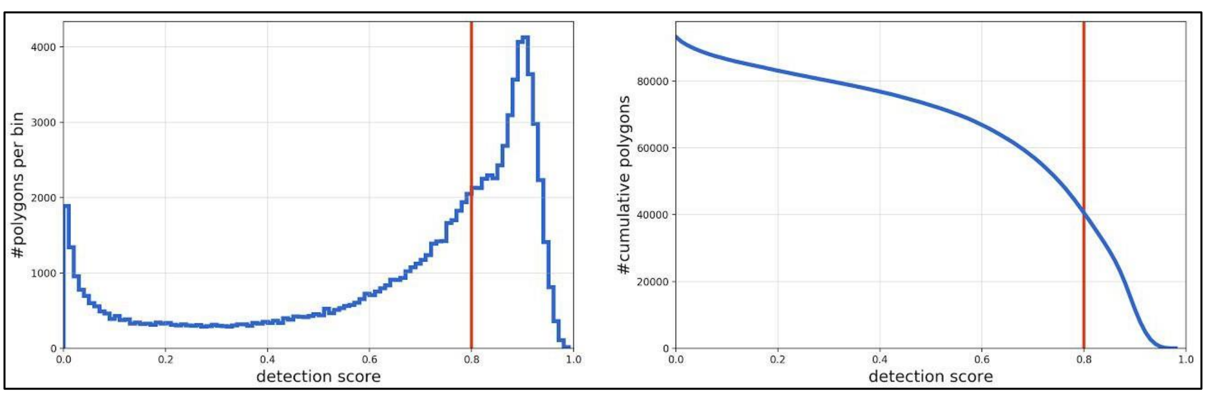


Figure 4.1: Histogram of detection scores vs. number of polygons (left) and number of polygons remaining at a specific detection score threshold (right). The red line is the detection cutoff score.

Table 4.1 summarizes the average precision (AP) and average recall (AR) at different IoU thresholds on the validation set. This dataset was not shown to the network during training. The metrics. As can be seen in the table, applying a low IoU threshold of 0.5 yields modest results: 71.4 % of the predicted field instances are correct (i.e. they coincide with actual field instances; precision), while 83.4 % of the ground truth instances are successfully delineated (i.e. the model predicted a field instance at their location; recall). As expected, stricter IoU thresholds lead to disproportionately less successful delineations, because they require closer matches with the ground truth objects: Interestingly, AP and AR are very similar for IoU thresholds of 0.75 and [0.5:0.95], but substantially lower than for a threshold 0.5 (in relative numbers, AP and AR are reduced by around 40%).

Table 4.1: Average precision (AP) and average recall (AR) at different IoU thresholds for the single-class experiment (validation dataset). Metrics are calculated in two ways: By averaging over all field instances in all chips together (overall) and by calculating AP and AR for each image chip individually, then averaging these values over all image chips (per chip).

	AP@IoU			**AR@IoU**		
	0.5	**0.75**	**0.5:0.95**	**0.5**	**0.75**	**0.5:0.95**
overall	71.4	41.1	39.1	83.4	56.4	51.2
per chip (mean)	74.8	47.0	43.5	85.0	58.0	52.5
per chip (standard deviation)	16.3	20.9	13.8	12.5	19.1	12.1

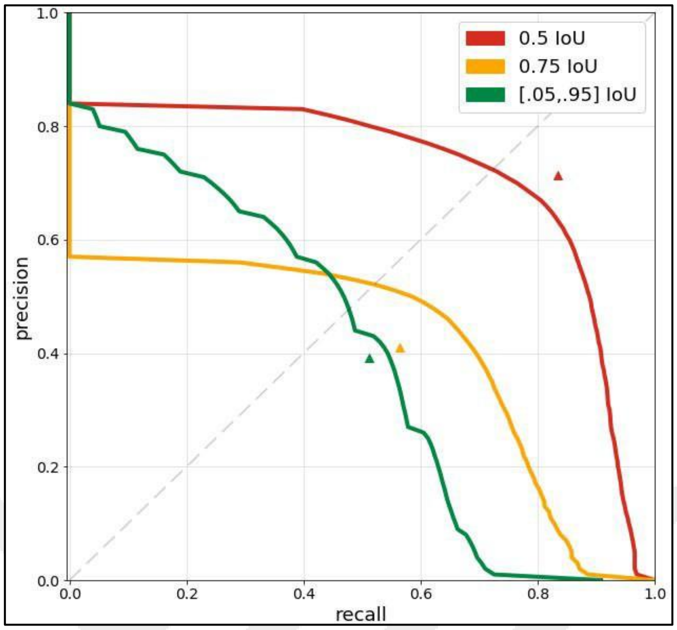


Figure 4.2: Precision-recall curves at different IoU thresholds for the single-class experiment (validation dataset). Triangle symbols indicate the overall AP and AR values.

A general sense of the results can be gained by comparing them to the corresponding metrics on the COCO segmentation reference dataset: The AP of 39.1 at [0.5:0.95] IoU achieved in the single-class setting is in line with the performance on the COCO segmentation dataset evaluation by FCIS (29.2@[0.5:0.95]) and Mask-RCNN (37.1@[0.5:0.95]). However, note that the COCO dataset is much more complex and diverse than the field delineation task investigated here. To gain a deeper understanding beyond the average metrics, the entire precision-recall curves (for the different IoU thresholds). The area under the curve corresponds to the AP of the predictions. To reach high AP values (i.e. high area under the curve), the method needs to have both high precision and high recall. The plot shows that the model requires a significant initial drop in precision to increase recall (for all IoU thresholds). In contrast, a perfect model would retain full precision until all ground truth objects are successfully delineated. After the initial drop off, the curves for AP at 0.5 and 0.75 IoU show a similar, balanced decrease of precision and recall. Although the APs at 0.75 and [0.5:0.95] have similar values, their precision-recall curves are significantly different: The averaging effects over the multiple IoU thresholds of AP at [0.5:0.95] IoU lead to a rippled curve pattern with a much stronger slope. Higher initial precision is balanced by lower recall.

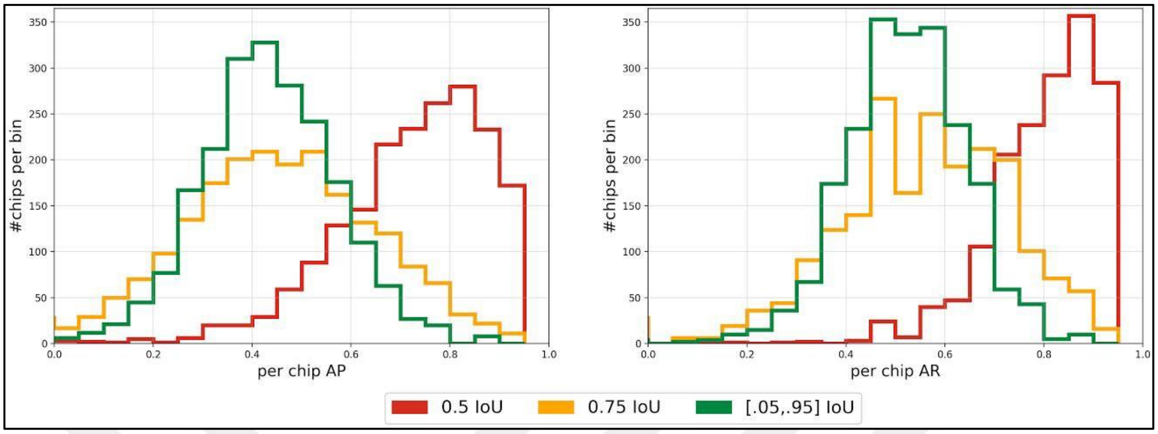


Figure 4.3: Histograms of the number of chips with a specific per chip AP and per chip AR at different IoU thresholds (validation dataset).

The results presented so far were averaged over all field instances and image chips at the same time. In addition, the metrics (AP and AR) were calculated for each chip individually, in order to investigate how the model performs on the level of a single image chip. The mean and standard deviation of these per chip-values are also shown in table 1, they are consistently higher than the respective overall results. It should be noted that these values are only calculated over the predictions of the specific image chip and cannot be directly compared to the overall AP and AR metrics. Fig. 30 shows the distributions of AP and AR values on the chip level (i.e. how many image chips have a specific AP/AR value). The difference between low and high IoU thresholds can be observed clearly: For a low IoU threshold of 0.5, the distribution is right-skewed and limited to a certain range of relatively high per chip AP and AR values. In contrast, the distributions for IoU thresholds of 0.75 and [0.5:0.95] IoU approximately follow a normal distribution.

4.1.2. Classification of Yielding

Figure 4.4 shows example image chips from the validation set. The ground truth field instances and the predictions by the model (with a detection score > 0.8) are indicated. From top to bottom, the examples have decreasing per chip AP (all at IoU [0.5:0.95]). For the examples with high per chip

AP, the ground truth and the prediction results are in good accordance with each other in terms of the amount of field instances, their boundaries and the resulting overlap. Therefore, the consensus area (right column) closely resembles the actual field instances.Complex shapes and borders are to some extent modelled by the predictions, even though small indentations and sharp edges often appear slightly rounded in the predictions. Also, straight boundaries of ground truth objects appear more irregular in the predictions. These effects hold for examples with high as well as low per chip AP. Heterogeneous texture inside single field parcels seem to be handled quite well. Sometimes, the predictions show some overlap with adjacent instances, i.e. the separation is not as strict as in the ground truth polygons.

Image chips with low per chip AP often show three main characteristics, which are discussed later on: 1) Predicted instances that cover an area which is actually covered by multiple small instances in the ground truth data, 2) full overlap of a smaller prediction by a bigger prediction, 3) false positives due to assumed field instances that are not existing in the ground truth data.

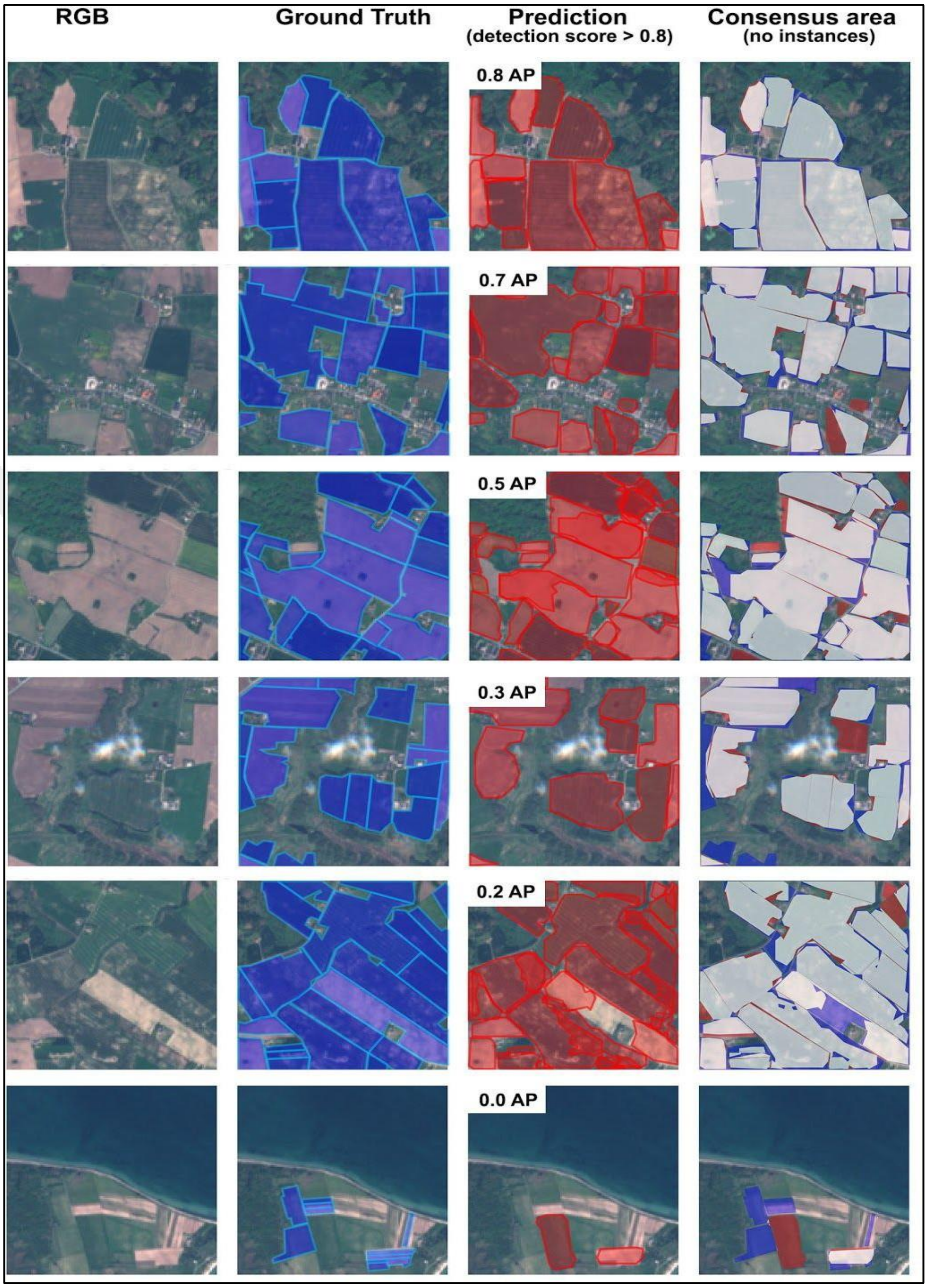


Figure 4.4: Example image chips from the validation dataset and predictions for the single-class setting (ranked by per chip AP@[0.5:0.95]). Only field instances with detection scores over 0.8 are displayed and considered for the per chip AP calculation. The consensus area (white) shows the overlap of ground truth and predictions.

4.1.3. Training Process

Figure below displays the development of the AP metrics and loss functions during the training process. Due to transfer learning initialization, OHEM bootstrapping and the single image mini-batch, the training losses (running average per epoch) quickly decrease within the first epoch. Afterwards, they slowly decrease further, until they plateau around epoch six to eight. The small peaks of the loss function at the beginning of each epoch are artifacts of the training process and insignificant for the performance of the model.

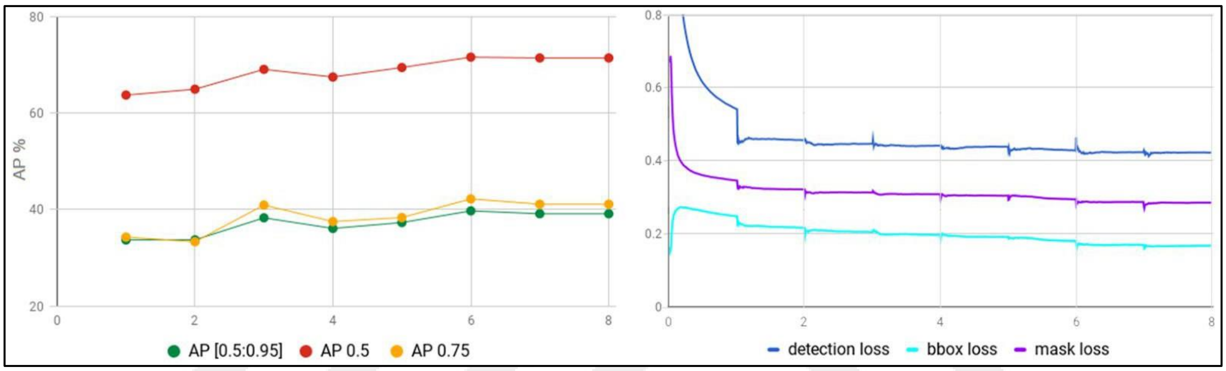


Figure 4.5: Development of the model over the training epochs: AP at different IoU thresholds (left) and loss functions (right).

4.1.4. Field Size

In manual segmentation, it is arguably harder to recognize and segment small agricultural fields. To investigate the effect of field size on the performance of the model, all field instances were assigned to one of three categories based on the size of their predicted polygon: S for small fields (< 0.025 km2), M for medium-sized fields (0.025 km2 - 0.1 km2) and L for large fields (> 0.1 km2). Table 2 shows the AP and AR values for each category, as well as the distribution of fields across the categories. As expected, the performance of the model increases with field size. Especially small field instances (category S) have a rather low AP and AR, while medium-sized and large fields show more similar results. Also, with an amount of 55.9 % of the predicted field instances, un-proportionally many small field objects are predicted - compared to only 43.1% of small objects in the ground truth training data. A possible reason for these bad results on small field instances, in the example with 0.0 per chip AP: Apparently, the FCIS algorithm has problems to delineate field instances in congregations of small, directly adjacent fields, even if they show distinct differences in texture and color. This may be related to the mask voting operation of FCIS in combination with the relatively low spatial resolution of the imagery. The segment masks of small, partially overlapping predictions could be merged unintentionally.

Table 4.2: Average precision (AP) and average recall (AR) by polygon area for the single-class experimental (validation dataset).

	AP@[0.5:0.95] IoU	**AR@[0.5:0.95] IoU**
S (< 0.025 km^2, 55.9 %)	28.1	39.5
M (0.025 km^2 - 0.1 km^2, 32.1 %)	47.3	58.9
L (> 0.1 km^2, 11.9 %)	51.0	68.6

4.1.5. Overlap

The results also show some artifacts of strongly overlapping predictions. This situation is usually characterized by a small, redundant prediction that is completely (or to a large extent) overlapped by a bigger, often correctly delineated prediction. This seems to appear most frequently near the border of image objects. As the redundant predictions do usually not enclose a full image object, some of the position sensitive bins of the RoI-inside and outside score maps in the FCIS model should not be activated. This would lead to a low detection score and the proposal being filtered out.

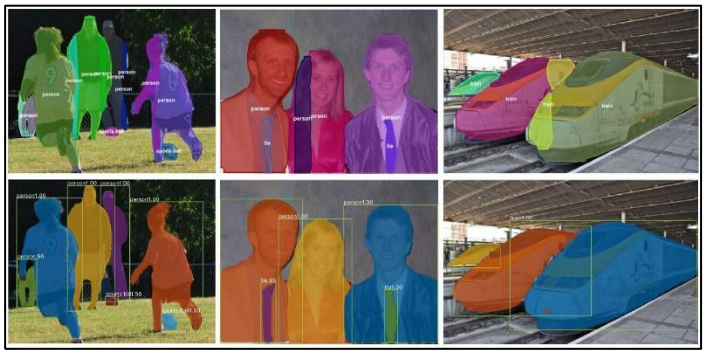


Figure 4.6: Systematic artifacts of redundant, overlapping predictions in FCIS results (top) compared to Mask R-CNN (bottom).

A similar situation was observed by the authors of Mask R-CNN, when comparing prediction results from natural photos across models, “suggesting that it is challenged by the fundamental difficulty of instance segmentation”. Neither the non-maximum suppression nor the mask voting operation of FCIS nor an additional NMS after the mask merging step are usually able to remove the redundant prediction, as two overlapping objects with distinctive size difference have a relatively low IoU score.

4.1.6. Problematic Training Data

The 2016 LPIS dataset, which was used as ground truth for the field polygons, generally matches the field structure in the satellite images that were used in this thesis. However, it was not specifically delineated from these images. Also, the main production period and exact production specifications of the dataset are not available. This causes some problems with model training and evaluation: When visually comparing the ground truth data and satellite images, definitive mismatches of declared or missing field instances can be spotted in some areas. This is due to temporal changes before or after the reporting phase, missing or inaccurate field delineations or the inclusion of property boundaries in the dataset. Mismatches can also be caused by the challenging reclassification and selection of agricultural-related field classes due to the extensive original class legend and variability of some environmental classes (e.g. also containing a minority of field objects). These challenges influence both training as well as evaluation: Although the prediction might be reasonable for the actual image content, the evaluation of these instances leads to worse results or a high training loss.

4.1.7. Empty Image Chips

To guarantee the comparability of the results, in the validation dataset, empty image chips without any field instances were discarded. This accounts for 771 land image chips in total. The scenes are mostly of urban, coastal and forest setting. Including these image chips in the evaluation slightly lowers the AP score by <0.01, but does not impair the performance of the algorithm in any significant way. For all empty image chips, only 47 additional instances with detection scores above 0.8 were predicted, distributed across 32 different image chips. These predictions count towards the overall number of false positives. However, when visualizing these false positives, it can be seen that most predictions are quite reasonable, and could potentially be confused by human annotators as well.

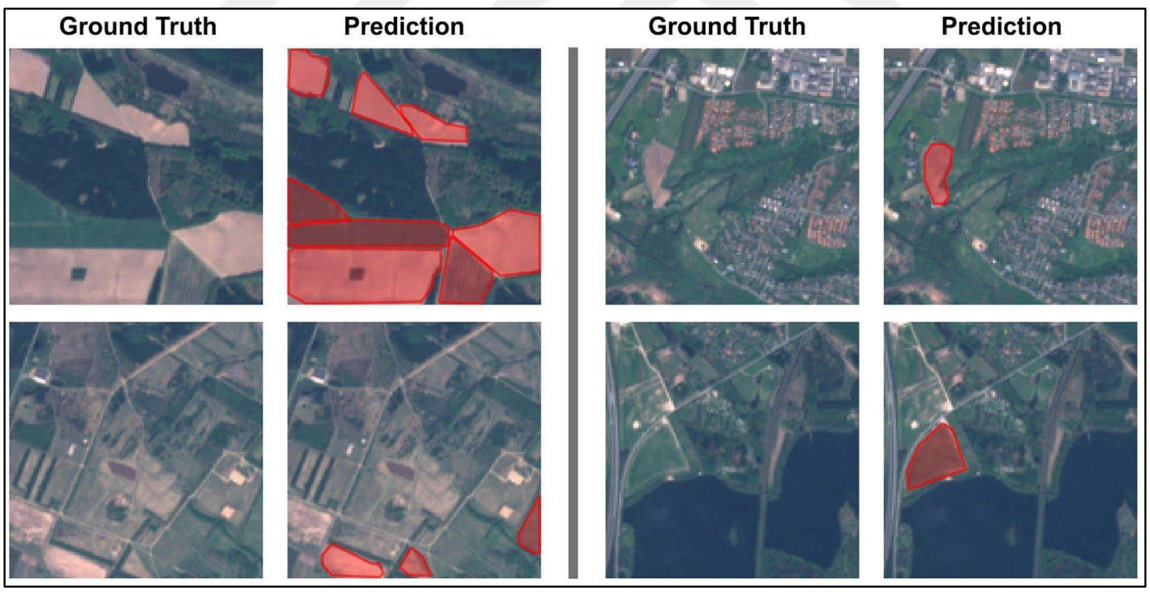


Figure 4.7: Example false positive predictions in image chips without ground truth instances.

4.2. EXPERIMENT 2: MULTI-CLASS

In the second experiment, the model was trained in a multi-class setting: Each field instance was assigned to one of five crop classes. The task was now to predict the polygons of the field instances (as in experiment 1) and simultaneously classify their crop type.

4.2.1. Metrics

Table 4.3 summarizes the mean average precision (mAP) and mean average recall (mAR; averaged over all classes) at different IoU thresholds on the validation set. Table 4 reports the class-specific AP at [0.5:0.95] IoU for each of the five agricultural crop types. Overall, the multi-class delineation was not as successful as the single-class setting: the (averaged) metrics are significantly lower than in the single-class experiment. Even with the low IoU threshold of 0.5, only 45.0 % of the predictions are correct (in contrast to 71.4 % for single-class). However, one has to note the different evaluation criteria in this experiment: False positives are not only field instances that cannot be assigned to a ground truth object, but also instances that do not match the correct ground truth class label. Classification seems to be most successful for the winter and spring cereal classes, which are also the most frequent classes in the study area. As in the single-class experiment, applying the stricter IoU thresholds 0.75 and [0.5:0.95] leads to a significant drop of the mAP and mAR metrics (about 40 to 50 % in relative numbers). Overall, the network was apparently not able to handle the classification of field instances based on the available information in the RGB images.

Table 4.3: Mean average precision (mAP) and mean average recall (mAR) at different IoU thresholds for the multi-class experiment (validation dataset).

	AP@IoU			AR@IoU		
	0.50	**0.75**	**0.5:0.95**	**0.50**	**0.75**	**0.5:0.95**
overall	45.0	25.0	24.3	74.1	49.2	44.9

Table 4.4: Class-specific average precision at [0.5:0.95] IoU for the multi-class experimental (validation dataset).

	winter cereal	spring cereal	Grass-fields	maize	other
AP@[0.5:0.95]	44.0	32.2	22.4	19.1	3.6

4.2.2. Classification of Field Yield Instances

When comparing the results visually, the prediction of the polygons' shapes and layouts seems to match the performance of the single-class experiment. The predicted field instances closely resemble the ground-truth data, with some deviations particularly at low per chip AP. However, the crop classes are frequently misclassified.

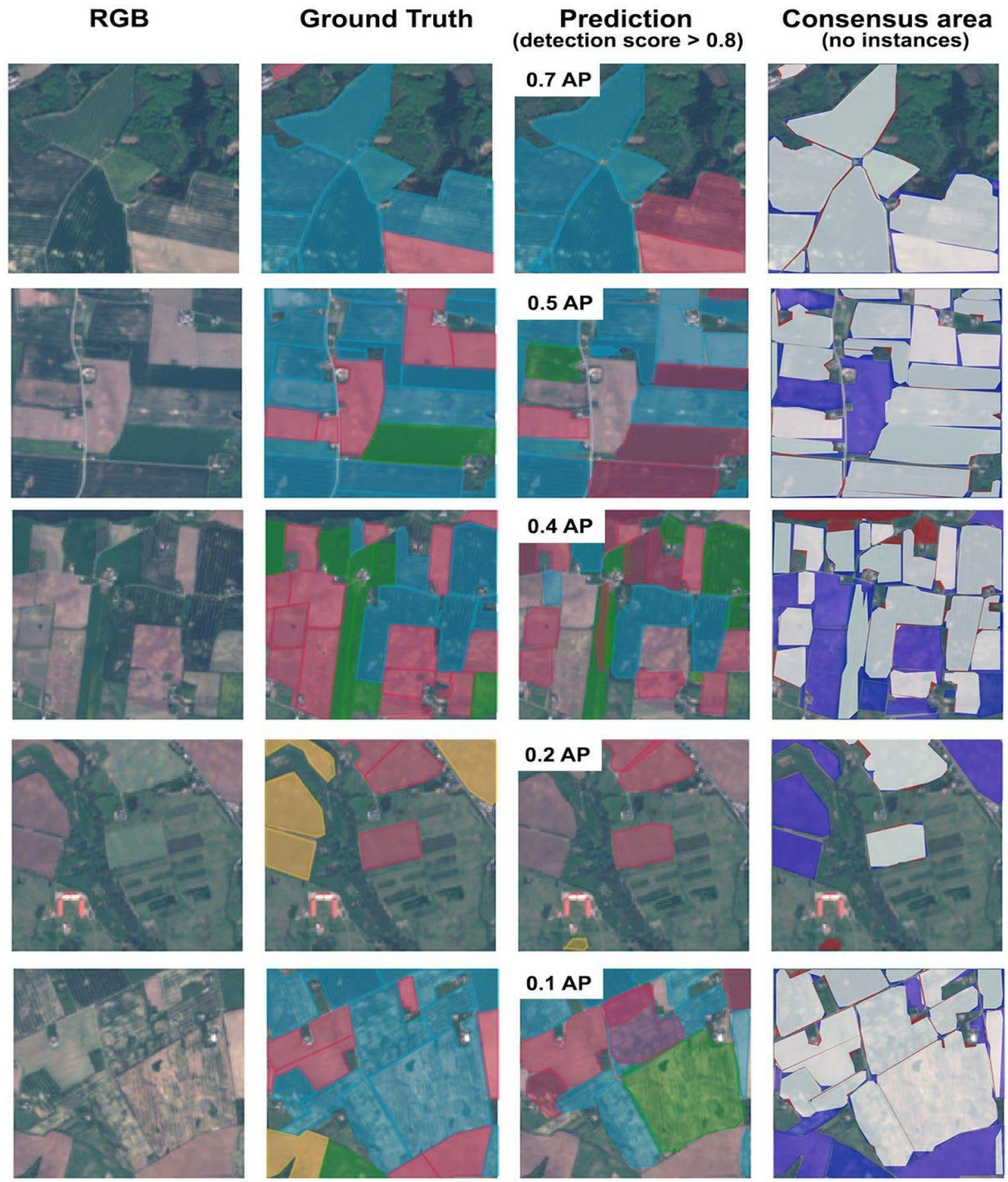


Figure 4.8: Example image chips from the validation dataset and predictions for the multi-class setting (ranked by per chip AP@[0.5:0.95]). Only field instances with detection scores over 0.8 are displayed and considered for the per chip AP calculation. The consensus area (white) shows the overlap of ground truth and predictions.

5. DISCUSSION

The methodology presented in this thesis offers potential for many improvements. Additional research could greatly benefit the results achieved for agricultural fields segmentation, the applicability of the model to other scenarios and the general understanding of deep learning instance segmentation for the use with remote sensing imagery. The four main directions for future improvement are: 1) using more accurate and heterogeneous training data, 2) exploiting the satellite image information in a better way, 3) improving model training and 4) extending post-processing.

Table 5.1: The comparison of accuracy of proposed technique with existing literature.

Article	**Algorithm**	**Accuracy**
[59]	Support Vector Machine (SVM)	92.43%
[60]	Long Short Term Memory - Recurrent Neural Network (LSTM-RNN)	97.69%
Proposed	Convolutional Neural Network (CNN)	98.92%

More Accurate and Heterogeneous Training Data: The precise parcel situation at the time of the satellite image acquisition would be the ideal training data for the field delineation task. It would take a lot of time to manually edit the data to make it exactly match the satellite scenes that were chosen, but it could be done to create general reference datasets. The use of additional and more diverse ground truth and satellite image data, such as from multiple study areas concurrently, could also enhance the general model's accuracy, robustness, and transferability in light of the growing public availability of high-quality geo-datasets. Random scaling, distortions, color offset or jitter, and the introduction of image noise are all examples of additional data augmentation that could help reduce overfitting and improve the model's adaptability to a variety of geographic settings as well as atmospheric and lightning conditions. Although the model can generalize within certain limits (local generalization), the variability of the training data primarily restricts its transferability. Furthermore, the precise delineation of the field boundaries, particularly for small objects, would unquestionably be enhanced by a higher spatial resolution. Sentinel-2 provides the highest resolution of freely available satellite imagery at the time of writing.

Better Exploitation of the Satellite Image Information: The exploitation of multi-temporal satellite image information and the full spectrum of satellite image bands could certainly improve the results. The current FCIS implementation is limited to single-temporal three band imagery. The use of more image channels of different spectral wavelengths is mainly limited by computational resources. A workaround could also be the application of principal component analysis (PCA) on the full range of the available image bands and subsequent use of the first three dimensions in replacement of the RGB image channels. The proper integration of multi-temporal information would instead require a different model architecture. The instance segmentation models that are currently available are designed for the use with natural photo images. Therefore, they do not consider a possible relation between several images. In addition to the raw image data, hand-crafted features could be integrated into the model of (e.g. band indices, edge detection preprocessing). This could possibly improve the prediction accuracy, but somewhat opposes the deep learning premise of not requiring any feature engineering.

Modified Model Training: The model's predictive power could be enhanced by adjusting the parameters and training process in a number of different ways. A tradeoff between hyper-parameter tuning effort, training time, and potential accuracy gain is frequently necessary for this kind of improvement. The use of additional RPN anchor scales and aspect ratios, additional training iterations, adjustments to the learning rate, and multi-scale training are all examples of work that could be done in the future. Predictive performance can also be improved by using model ensembles, which average the results of multiple instances of the same model architecture applied to various subsets of the training data and with different initializations. By parallelizing the model on hardware with multiple GPU cores, increased training complexity could reduce higher computational costs. Additionally, the forthcoming implementation release of the most up-to-date instance segmentation model, Mask R-CNN will make it possible to access a simpler and faster architecture.

Improved Post-Processing: The fit of the field boundaries could be improved by a variety of external polygon post-processing methods, such as snake algorithms for directed line smoothing, or the number of false positive predictions could be reduced (such as the removal of strongly overlapping polygons using modified NMS or overlap criteria).Any subareas of the predicted polygons that are partially overlapping could be eliminated because the field parcel objects cannot demonstrate any direct overlap in the satellite image.

The current method's limited application in an operational setting for the region-wide delineation of field parcels is an important aspect because the use of instance segmentation with satellite images rather than natural photos presents a unique obstacle: Because the method is constrained by the memory of the GPU and requires multiple examples to iterate on, the satellite images are divided into smaller image chips. The predictions for each chip would have to be combined. However, there are at least two polygon predictions for fields that are not completely enclosed within the boundaries of a single image chip. It is difficult to easily combine these parcel sections. Using intermediate chips that correspond to a 25% area overlap of four adjacent image chips could be a possible workaround. The risk of predicting partial instances could be reduced by intelligently combining the polygon predictions close to the chip center.

6. CONCLUSION

This research examines how satellite images can be used to automatically delineate agricultural field yielding using deep learning instance segmentation. The results appear very promising in light of the insufficient precedent work on instance segmentation of remote sensing images, the non-optimal training data, and the relatively simple input-output method without extensive post-processing. The correct prediction of small field parcels, miss detections of parcels that are considered environmental areas, and similar physical characteristics between parcels of different crop classes are major obstacles for the current method. The application of deep learning instance segmentation to geospatial and remote sensing data demonstrates a significant future potential in light of the thesis's positive outcomes: The models can use raw data without needing to manually engineer features. Applying the model at test time requires less computation time than many other image segmentation algorithms, which is advantageous in light of the rapidly increasing volumes of satellite imagery and geo-data. However, the accuracy, quantity, and variability of the training samples are crucial to the predictive performance. Open source, high-quality geospatial data availability has increased, but it is still limited. For instance, provide access to their agricultural field datasets from the Land Parcel Identification System that is both comprehensive and user-friendly. Given the spatial resolution of the Sentinel-2 images and the size of the agricultural field polygons, the area-specific COCO evaluation categories do not correspond to the field delineation task using CNN. Instead, the S, M and L category area limits were selected by the distribution of field areas in the training data: S (< 162 pixel), M (162 - 322 pixel) and L (> 322 pixel). This roughly corresponds to 43.1% of training field polygons < 0.025 km2, 46.5 % from 0.025 km2 - 0.1 km2 and 10.4 % > 0.1 km2. The processing of satellite images and the training characteristics and resource requirements of deep learning models are made more difficult by the satellite image properties. The presented method has room for numerous future enhancements, such as the incorporation of additional multi-temporal and satellite image bands, modifications to the model architecture, and additional post-processing.

REFERENCES

[1] Amodei, D. & Anubhai, R. & Battenberg, Zhan, J. & Zhu, Z. (2015): Deep Speech 2: End-to-End Speech Recognition in English and Mandarin. https://arxiv.org/abs/1512.02595

[2] Arbelaez, P. & Pont-Tuset, J. & Barron, J. & Marques, F. & Malik, J. (2014): Multiscale Combinatorial Grouping. Computer Vision and Pattern Recognition (CVPR), 2014. https://arxiv.org/abs/1503.00848.

[3] Badrinarayanan, V. & Kendall, A. & Cipolla, R. (2015): SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. https://arxiv.org/abs/1511.00561

[4] Basu, S. & Ganguly, S. & Mukhopadhyay, S. & DiBiano, R. & Karki, M. & Nemani, R. (2015): DeepSat – A Learning framework for Satellite Imagery. https://arxiv.org/abs/1509.03602

[5] Butenuth, M. & Straub, M. & Heipke, C. (2004): Automatic extraction of field boundaries from aerial imagery KDNet Symposium on Knowledge-Based Services for the Public Sector. p3-4. 2004.

[6] Chen, T. & Xiao, T. & Xu, B. & Zhang, C. Zhang, Z. (2015): MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. In Neural Information Processing Systems, Workshop on Machine Learning Systems, 2015. https://arxiv.org/abs/1512.01274

[7] Chen, L. & Papandreou, G. & Schroff, F. & Adam, H. (2017): Rethinking Atrous Convolution for Semantic Image Segmentation. https://arxiv.org/abs/1706.05587

[8] Da Costa, J. & Michelet, F. & Germain, C. & Lavialle, O. & Grenier, G. (2007): Delineation of Vine Parcels by Segmentation of High Resolution Remote Sensed Images. Precision Agriculture 8 (1 –2): 95 – 110. https://doi.org/10.1007/s11119-007-9031-3

[9] Dai, J. & He, K. & Sun, J. (2015a): Convolutional feature masking for joint object and stuff segmentation. In CVPR, 2015. https://arxiv.org/abs/1412.1283

[10] Dai, J. & He, K. & Sun, J. (2015b): Instance-aware Semantic Segmentation via Multi-Task Network Cascades. https://arxiv.org/abs/1512.04412

[11] Dai, J. & He, K. & Li, Y. & Ren, S. & Sun, J. (2016a): Instance-sensitive fully convolutional networks. In ECCV, 2016. https://arxiv.org/abs/1603.08678

[12] Dai, J. & Li, Y. & He, K. & Sun. J. (2016b): R-FCN: Object detection via region-based fully convolutional networks. In NIPS, 2016. arXiv:1605.06409

[13] Everingham, M. & Van Gool, L. & Williams, C. & Winn, J. & Zisserman, A. (2010): The PASCAL Visual Object Classes (VOC) Challenge. International Journal of Computer Vision, 88(2), 303-338, 2010.

[14] GADM 2015: Global Administrative Areas v2.8. November 2015. http://gadm.org/

[15] García-Pedrero, A. & Gonzalo-Martín, C. & Lillo-Saavedra, M. (2017): A machine learning approach for agricultural parcel delineation through agglomerative segmentation. International Journal of remote sensing, 2017. VOL. 38, NO. 7, 1809 – 1819. http://dx.doi.org/10.1080/01431161.2016.1278312

[16] Girshick, R. & Donahue, J. & T. Darrell & Malik, J. (2014): Rich feature hierarchies for accurate object detection and semantic segmentation. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://arxiv.org/abs/1311.2524

[17] Girshick, R. (2015): Fast R-CNN. https://arxiv.org/abs/1504.08083

[18] Gorelick, N. & Hancher, M. & Dixon, M. & Ilyushchenko, S. & Thau, D. & Moore, R. (2017): Google Earth Engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment. Preprint. https://doi.org/10.1016/j.rse.2017.06.031

[19] Hariharan, B. & Arbelaez, P. & Girshick, R. & Malik, J. (2014): Simultaneous detection and segmentation. In ECCV. 2014. https://arxiv.org/abs/1407.1808

[20] Hariharan, B. & Arbelaez, P. & Girshick, R. & Malik, J. (2015): Hyper-columns for object segmentation and □ne-grained localization. In CVPR, 2015. https://arxiv.org/abs/1411.5752

[21] He, K. & Zhang, X. & Shaoqing, R. & Sun, J. (2015): Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

[22] He, K. & Gkioxari, G. & Dollar, P. & Girshick, R. (2017): Mask R-CNN. https://arxiv.org/abs/1703.06870

[23] Hinton, G. & Salakhutdinov, R. (2006): Reducing the dimensionality of data with neural networks. Science, Vol. 313. no. 5786, pp. 504 - 507.

[24] Jia, Y. & Shelhamer, E. & Donahue, J. & Karayev, S. & Long, J. & Girshick, R. & Guadarrama, S. & Darrell, T. (2014): Caffe: Convolutional Architecture for Fast Feature Embedding. https://arxiv.org/abs/1408.5093.

[25] Längkvist, M. & Kiselev, A. & Alirezaie, M & Lout, A. (2016): Classi□cation and Segmentation of Satellite Orthoimagery Using Convolutional Neural Networks. Remote Sens. 2016, 8, 329; doi:10.3390/rs8040329

[26] LeCun, Y. & Bottou, L. & Bengio, Y. & Haffner, P. (1998): Gradient-Based Learning Applied to Document Recognition. Proc. of the IEEE, Nov. 1998.

[27] LeCun, Y. & Bengio, Y. & Hinton, G. (2015): Deep Learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539

[28] Levine, S. & Pastor, P. & Krizhevsky, A. & Quillen, D. (2016): Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. https://arxiv.org/abs/1603.02199

[29] Li, Y. & Qi, H. & Dai, J. & Ji, X. & Wei, Y. (2017): Fully Convolutional Instance-aware Semantic Segmentation. https://arxiv.org/abs/1611.07709

[30] Lin, T. & Maire, M. & Belongie, S. & Bourdev, L. & Girshick, R. & Hays, J. & Perona, P. & Ramanan, D. & Zitnick, L. & Dollár, P. (2014): Microsoft COCO: Common Objects in Context. https://arxiv.org/abs/1405.0312

[31] Lin, T. & Dollar, P. & Girshick, R. & He, K. & Hariharan, B. & Belongie, S. (2016): Feature pyramid networks for object detection. https://arxiv.org/abs/1612.03144

[32] Long, J., & Shelhamer, E. & Darrell, T. (2014): Fully Convolutional Networks for Semantic Segmentation. Conference Paper.

[33] Marmanis, D. & Schindler, K. & Wegner, J. & Galliani, S & Datcu, M. & Stilla, U. (2016): Classification with an edge: improving semantic image segmentation with boundary detection. Preprint. https://arxiv.org/abs/1612.01337

[34] MFAF - Ministry for Food, Agriculture and Fisheries (2017): Marker 2016 dataset. https://kortdata.fvm.dk/download/Index?page=Markblokke_Marker

[35] Mnih, V. & Kavukcuoglu, K. & Silver, D. & Graves, A. & Antonoglou, I. & Wierstra, D. & Riedmiller, M. (2013): Playing Atari with Deep Reinforcement Learning. NIPS Deep Learning Workshop 2013. https://arxiv.org/abs/1312.5602.

[36] Mueller, M., & Segl, K. & Kaufmann, H. (2004): Edge-and Region-Based Segmentation Technique for the Extraction of Large, Man-Made Objects in High-Resolution Satellite Imagery.”Pattern Recognition 37 (8): 1619 – 1628.

https://doi.org/10.1016/j.patcog.2004.03.001

[37] Nielsen, M. (2015). Neural Networks and Deep Learning. Determination Press. http://neuralnetworksanddeeplearning.com

[38] Pinheiro, P. & Collobert, R. & Dollar, P. (2015): Learning to Segment Object Candidates. https://arxiv.org/abs/1506.06204

[39] Pinheiro, P. & Lin, T. & Collobert, R. & Dollar, P. (2016): Learning to Refine Object Segments. https://arxiv.org/abs/1603.08695

[40] Ren, S. & He, K. & Girshick, R. & Sun, J. (2016): Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. https://arxiv.org/abs/1506.01497

[41] Rossum, G. (1995): Python tutorial, Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, May 1995.

[42] Rydberg, A. & Borgefors, G. (2001): Integrated method for boundary delineation of agricultural fields in multispectral satellite images. IEEE Transactions: Geoscience and Remote Sensing 39 2514–20.

[43] Saito, S. & Yamashita, T. & Aoki, Y. (2016): Multiple Object Extraction from Aerial Imagery with Convolutional Neural Networks. Journal of Imaging Science and Technology R60(1). 2016.

[44] Shenquan, Q. & Ying, W. & Gaofeng, M. & Chunhong, P. (2016): Vehicle Detection in Satellite Images by Incorporating Objectness and Convolutional Neural Network. Journal of Industrial and Intelligent Information Vol. 4, No. 2, March 2016.

[45] Shrivastava, A & Gupta, A. & Girshick, R. (2016): Training Region-based Object Detectors with Online Hard Example Mining. https://arxiv.org/pdf/1604.03540.pdf

[46] Simonyan, K. & Zisserman, A. (2014): Very Deep Convolutional Networks for Large-Scale Image Recognition. https://arxiv.org/abs/1409.1556

[47] Szegedy, C. & Liu, W. & Jia, W. & Sermanet, P. & Reed, S. & Anguelov, D. & Erhan, D. & Vanhoucke, V. & Rabinovich, A. (2014): Going Deeper with Convolutions. https://arxiv.org/abs/1409.4842.

[48] Tiwari, P. & Pande, H. & Kumar, M. & Dadhwal, V. (2009) Potential of IRS P-6 LISS IV for Agriculture Field Boundary Delineation. Journal of Applied Remote Sensing 3 (1): 033528 – 033528. https://doi.org/10.1117/1.3133306

[49] Torre, M. & Radeva, P. (2000): Agricultural Field Extraction from Aerial Images Using a Region Competition Algorithm, International Archives of Photogrammetry and Remote Sensing, Amsterdam, Vol. XXXIII, No. B2, pp. 889-896.

[50] Turker, M. & Kok, H. (2013): Field-based sub-boundary extraction from remote sensing imagery using perceptual grouping. ISPRS Journal of Photogrammetry and Remote Sensing. Volume 79, May 2013, Pages 106-121.

[51] Uijlings, J. & van de Sande, K. & Gevers, T. & Smeulders, A. (2012): Selective Search for Object Recognition.

[52] Vakalopoulou, M. & Karantzalos, K. & Komodakis, N. & Paragios, N. (2015): Building detection in very high resolution multispectral data with Deep Learning features. IGARSS 2015 - 2015 IEEE International Geoscience and Remote Sensing Symposium.

[53] Wu, Y. & Schuster, M. & Chen, Z. & Le, Q. & Dean, J. (2016): Bridging the Gap between Human and Machine Translation. CoRR, vol. abs/1609.08144 (2016). http://arxiv.org/abs/1609.08144

[54] Yalcin, H. & Günay, B. (2016): Delineation of parcels in agricultural lands in remote sensing. 2016 Fifth International Conference on Agro-Geoinformatics. Tianjin, China. https://doi.org/10.1109/Agro-Geoinformatics.2016.7577601

[55] Yan, L. & Roy, D. (2014): Automated crop field extraction from multi-temporal Web Enabled Landsat Data. Remote Sensing of Environment 14442–64.

[56] Yu, F. & Koltun, V. (2015): Multi-Scale Context Aggregation by Dilated Convolutions. Conference Paper ICLR 2016. https://arxiv.org/abs/1511.07122

[57] Zagoruyko, S. & Lerer, A. & Lin, T. & Pinheiro, P. & Gross, S. & Chintala, S. & Dollar, P. (2016): A MultiPath Network for Object Detection. https://arxiv.org/abs/1604.02135

[58] Zeiler, M. & Fergus, R. (2013): Visualizing and Understanding Convolutional Networks. https://arxiv.org/abs/1311.2901

[59] Zhong, Y. & Fei, F. & Liu, Y. & Zhao, B & Jiao, H. & Zhang, L. (2017) SatCNN: satellite image dataset classification using agile convolutional neural networks. Remote Sensing Letters, 8:2, 136-145, DOI: https://doi.org/10.1080/2150704X.2016.123529

[60] Brdar, Sanja & Marinković, Branko & Crnobarac, Jovan & Crnojević, Vladimir. (2011). Support Vector Machines with Features Contribution Analysis for Agricultural Yield Prediction.

[61] Patel, Jigisha & Vala, Brijesh & Saiyad, Makhduma. (2021). LSTM-RNN Combined Approach for Crop Yield Prediction On Climatic Constraints. 1477-1483. 10.1109/ICCMC51019.2021.9418231.