DET: Data Enhancement Technique for Aerial Images

Leveraging Multiple Neural Networks to Improve Segmentation Boundary Quality

JØRGEN ÅSBU JACOBSEN, SANDER JYHNE

SUPERVISOR
Morten Goodwin

University of Agder, 2022
Faculty of Engineering and Science
Department of Engineering and Sciences

Preface and Acknowledgements

DET: Data Enhancement Technique for Aerial Images is a master thesis carried out at the University of Agder (UiA) in Grimstad, Norway. This research is conducted by Jørgen Åsbu Jacobsen and Sander Jyhne, concluding the master's program at the Department of Information and Communication Technology (ICT).

We want to thank our supervisor, Morten Goodwin, for excellent guidance and support through our master thesis. Your expertise has been tremendously helpful in keeping the research up to the expected standard. We also want to thank Ivar Oveland and Alexander Salveson Nossum for their cooperation through the KartAi project. Their outstanding domain knowledge and contributions have been invaluable for our thesis.

Grimstad,
June 3, 2022
Jørgen Åsbu Jacobsen and Sander Jyhne

Abstract

Deep learning and computer vision are two thriving research areas within machine learning. In recent years, as the available computing power has grown, it has led to the possibility of combining the approaches, achieving state-of-the-art results. An area of research that has greatly benefited from this development is building detection. Although the algorithms produce satisfactory results, there are still many limitations. One significant problem is the quality and edge sharpness of the segmentation masks, which are not up to the standard required by the mapping industry. The predicted mask boundaries need to be sharper and more precise to have practical use in map production.

This thesis introduces a novel Data Enhancement Technique (DET) to improve the boundary quality of segmentation masks. DET has two approaches, Seg-DET, which uses a segmentation network, and Edge-DET, which uses an edge-detection network. Both techniques highlight buildings, creating a better input foundation for a secondary segmentation model. Additionally, we introduce ABL(RMI), a new compounding loss consisting of Region Mutual Information Loss (RMI), Lovasz-Softmax Loss (Lovasz), and Active Boundary Loss (ABL). The combination of loss functions in ABL(RMI) is optimized to enhance and improve mask boundaries.

This thesis empirically shows that DET can successfully improve segmentation boundaries, but the practical results suggest that further refinement is needed. Additionally, the results show improvements when using the new compounding loss ABL(RMI) compared to its predecessor, ABL(CE) which substitutes RMI with Cross-Entropy loss(CE).

Contents

Glossary

ABL Active Boundary Loss [87]. ii, 41–43, 48, 49, 51

ABL(CE) Compounded loss of Active Boundary Loss [87], Lovasz-Softmax Loss [6], and Cross-Entropy Loss [35]. ii, viii, 41–43, 48

ABL(RMI) Compounded loss of Active Boundary Loss [87], Lovasz-Softmax Loss [6], and Region Mutual Information Loss [98]. ii, viii, 1, 41, 49, 51

CE Cross-Entropy Loss [35]. ii, 2, 41–43

DET Short for Data Enhancement Technique. DET has two proposed segmentation enhancement techniques, Edge-DET and Seg-DET. ii, vi, 1, 3, 26, 29, 31, 43, 45, 48–53

Edge-DET DET approach using an edge-detection network as the first model. ii, vi, viii, 29–31, 36, 38, 40, 42–44, 46–52

FKB-Bygning The map object used for building representations [32]. vi, 1, 4, 5, 7

Kartverket The Norwegian mapping authority. viii, 4, 26, 27, 40, 41, 45–47, 49

Lovasz Lovasz Softmax Loss [6]. ii, 2, 41–43

RMI Regional Mutual Information Loss [98]. ii, vi, 2, 21, 39–43, 48

Seg-DET DET approach using a segmentation network as the first model. ii, vi–viii, 29–32, 35, 36, 40, 43–46, 48, 49, 51

SFKB The databased used to store map objects, such as FKB-Bygning [50]. 1, 4, 5, 7

List of Figures

List of Tables

Chapter 1

Introduction

Deep Learning has in recent years been extensively researched, utilized, and adopted by almost all domains. The algorithms are becoming more valuable and practical due to the growth in available computing power, allowing for further usage expansion. One of the main areas in Deep Learning that has greatly benefited from extra computing power is Computer Vision, where building detection is a prominent area of research. However, a common problem is the quality and edge sharpness of the segmentation masks. The building predictions need to be precise to have practical use in real world map production, and in today's society, there are vast application areas.

Applying deep learning to achieve the quality of building masks required is challenging since the training data derives from real-world photographs that often have varying data quality. Even with high-resolution images, the photos will contain noise in different forms. Building predictions are affected by optical issues such as shadows, reflections, and perspectives. Visibility is another factor of significance where trees, powerlines, or even other buildings lead to bias [76]. Despite many advances in semantic segmentation, no state-of-the-art models produce an adequate boundary for map production.

To improve the accuracy of the edges, we introduce a Data Enhancement Technique (DET) that employs an edge-detection or segmentation network to highlight buildings. DET combines the output prediction percentages with the original images either by modifying them directly or adding the predictions as a fourth channel. A second network then receives the highlighted images as input. We also introduce a new loss combination called ABL(RMI) that encourages the alignment between predicted boundaries and ground-truth boundaries [87].

1.1 Motivation

Buildings are an essential component of information regarding population, policy-making, and city management [36]. The Norwegian mapping authority stores its building objects in a national database called SFKB [50], which contains representations of all registered map objects, their attributes, and corresponding geolocations. Within SFKB they store FKB-Bygning objects, which is representations of buildings and their properties. The objects is updated by manually measuring and annotating buildings. Precise predictions will make it possible to incorporate the building masks into FKB-Bygning objects with less human interaction compared to the traditional method. Using deep-learning to annotate buildings reduces the cost and time spent keeping FKB-Bygning objects up to date.

Furthermore, precise mask boundaries create the foundation needed to calculate the building footprint, create ridgelines or even produce 3D models of buildings. The potential gain of accurate segmentation masks is high. Therefore, the primary motivation of this thesis is to improve the quality of building masks to a level suitable for planning, object production, and quality assurance in map production.

1.2 Thesis Definition

The primary objective of this thesis is to produce a method that improves the boundary quality of segmentation masks for buildings compared to standalone segmentation models. The research is split into three goals following the thesis hypotheses.

1.2.1 Thesis Goals

Goal 1: Investigate the state-of-the-art research within the field of semantic segmentation on aerial images.

Goal 2: Explore the combination of two separate segmentation models for improved mask boundaries

Goal 3: Explore the combination of an edge-detection model and a segmentation model for improved mask boundaries

1.2.2 Thesis Hypotheses

Hypothesis 1: Substituting Cross-Entropy loss (CE) with Regional Mutual Information loss (RMI) in the compounding loss of Cross-Entropy loss(CE), Lovasz-Softmax loss (Lovasz), and Active Boundary Loss (ABL) will improve the Boundary Intersection over Union (BIoU).

Hypothesis 2: Using the compounding loss of RMI, Lovasz, and ABL will improve the BIoU of a segmentation network compared to using Region Mutual Information Loss.

Hypothesis 3: Combining (3 channels) the prediction gradients from a segmentation network with the original images as input to a secondary segmentation network will improve the BIoU compared to using the original images as input.

Hypothesis 4: Concatenating (4 channels) the prediction gradients from a segmentation network with the original RGB images as input to a secondary segmentation network will improve the BIoU compared to using the original input.

Hypothesis 5: Combining (3 channels) the prediction gradients from an edge-detection network with the original RGB images as input to a secondary segmentation network will improve the BIoU compared to using the original images as input.

Hypothesis 6: Concatenating (4 channels) the prediction gradients from an edge-detection network with the original RGB images as input to a secondary segmentation network will improve the BIoU compared to using the original images as input.

1.2.3 Thesis Outline

In Chapter 2, we introduce the domain knowledge necessary for our thesis. The chapter starts by detailing the Norwegian Building Register, which entails information regarding the building objects, databases, and their process for updating the data. Following the Norwegian Building Register, the main concepts of artifical neural networks and their use in computer vision is explained. Furthermore, Chapter 3 introduce image classification, segmentation losses, and state-of-the-art models for semantic segmentation, edge-detection, and aerial image segmentation, detailing the inner workings of each topic. Method Chapter 4 describes DETs architecture, the data enhancement process, and how we analyze the results. Additionally it provides visualizations and code examples for DET. Furthermore, Chapter 5 and 6 describes the results acquired for our compounding loss and DET, in addition to conclude and discuss our hypotheses. Lastly, Chapter 7 concludes the thesis and propose future work.

Chapter 2

Background

Within machine learning, we find the subfield of Deep Learning (DL) and Computer Vision (CV), two thriving and dominant research areas. As the available computing power has grown in recent years, it has led to the possibility of combining the approaches. Using methods from DL in CV algorithms has shown significant benefits, capable of achieving state-of-the-art results.

DL has many applications, from self-driving cars to medical image analysis. In later years, building detection has become an area of interest. An accurate record of buildings is crucial information regarding population estimation, city planning, and environmental science [71]. Although, keeping these databases up to date is currently a time-consuming, costly, and predominantly manual process that machine learning can help improve.

This chapter outlines background theory related to the research conducted in this thesis. Firstly, Section 2.1 introduces Norway's buildings register. Thereafter, Section 2.2 explains how an Artificial Neural network works and its correlation to DL, moving on to computer vision in Section 2.3. Lastly, we will describe IoU and BIoU, used to assess the performance of segmentation models.

2.1 Norway's Building Register

2.1.1 FKB-Bygning

FBK-Bygning [32] is a building object stored in SFKB [50], a database Kartverket (the Norwegian Mapping Authority) uses to keep records of map objects and their properties. FKB-Bygning builds upon a 2.5-dimensional building model and does not contain volumes or full 3-dimensional objects. Instead, it is built using vectors with height information for each object that is registered. This information makes it possible to construct simple 3-dimensional models by projecting FKB-Bygning objects onto a terrain model [32]. Additionally, FKB-Bygning objects has various information about the respective buildings, shown in Figure 2.1. The FKB-Bygning objects also provide the building masks needed to train our machine learning models.

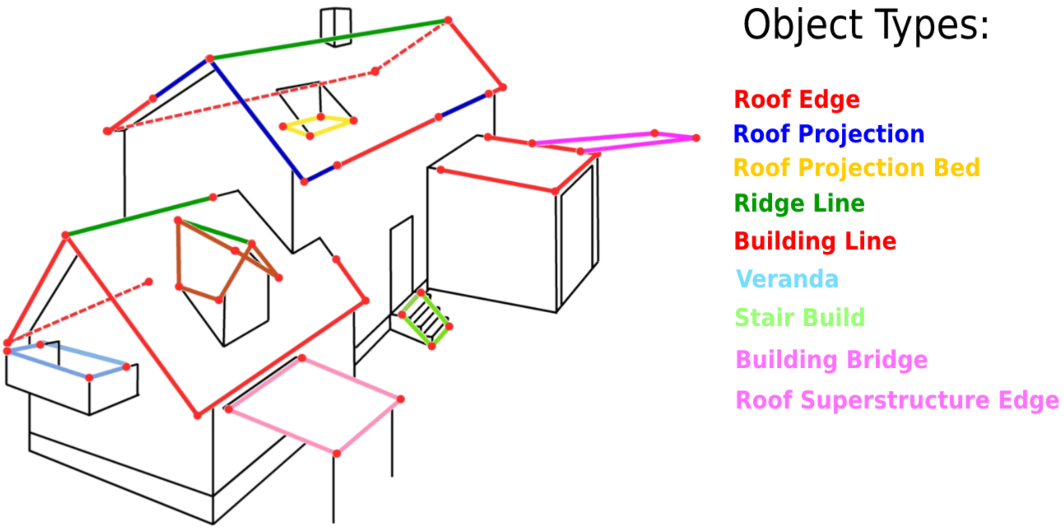


Figure 2.1: Example of registered object types in a FKB-Bygning object [32].

2.1.2 The Cadastre

In addition to SFKB [50], the Cadastre is another database containing information about buildings, properties, and areas in Norway. Compared to SFKB objects, the Cadastre has only one-dimensional, text-based objects. Both SFKB and the Cadastre are linked and should contain building representations for the same buildings [32]. If a structure exists in the Cadastre, it should have a record in the SFKB. If not, there is an inconsistency that needs to be updated. Today the national Cadastre in Norway has several deviations which motivate automating the manual updating process.

2.1.3 Photogrammetry

Photogrammetry is a method for gathering reliable information about measurements from 2D photographs, a tool widely utilized in land mapping. The most fundamental principle used in photogrammetry is called triangulation [47]. This process computes three-dimensional coordinates of points using two or more images taken at different locations, requiring information about the camera position and angle [47]. A reference scale is also needed to measure the scene accurately. Carthographers can then use the points to measure the factual distances of objects of interest. For example, they can use several points to calculate the area of a building.

Mapping land requires two or more images of the same ground feature collected from different geolocation positions. The overlapping area of the photos creates stereoscopic imagery, which the triangulation process uses as input [4]. As described in Section 2.1.1, FKB-Bygning builds upon a 2.5-dimensional building model containing vectors and their height information. The process of creating FKB-Bygning objects utilizes the triangulation process, making it possible to calculate the elevation from the stereoscopic photographs.

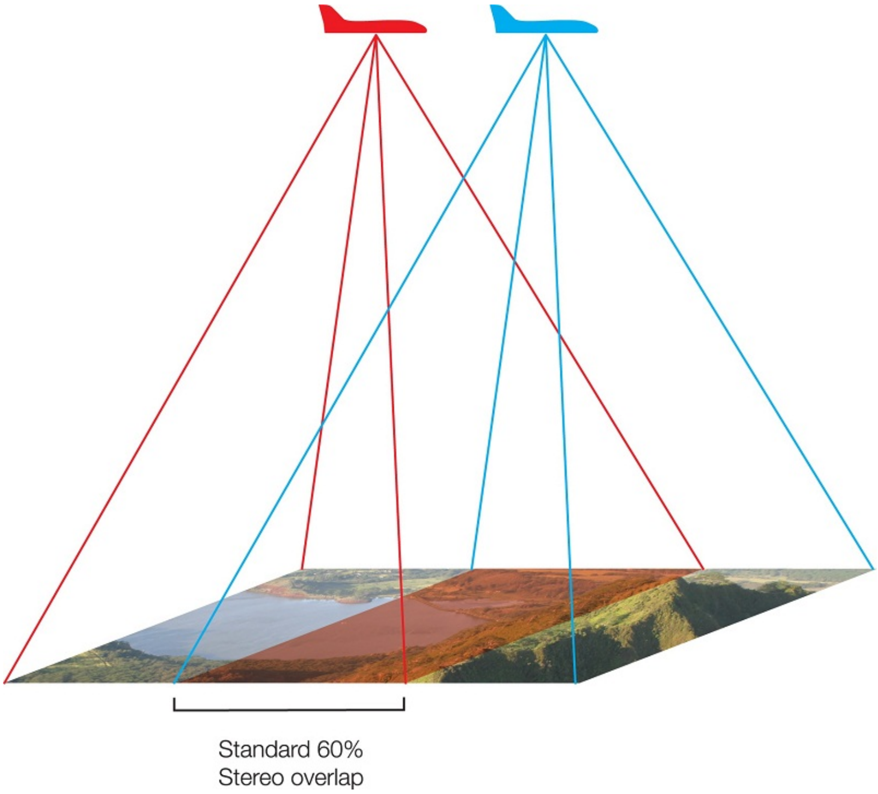


Figure 2.2: Illustration of capturing stereoscopic imagery [81].

The most typical use of the photogrammetric process is to create orthophotos and 3-dimensional models of scenes. These products utilize triangulation to generate an output corrected to the selected mapping frame, obtaining reliable information about physical objects and the environment. Orthoimages can also be edge-matched and color-balanced to produce a seamless orthomosaic map, which is accurate to a specified scale and can be used to make precise measurements [4]. We use an orthomosaic map in our machine learning models, giving us the freedom to choose the resolution of the input images.

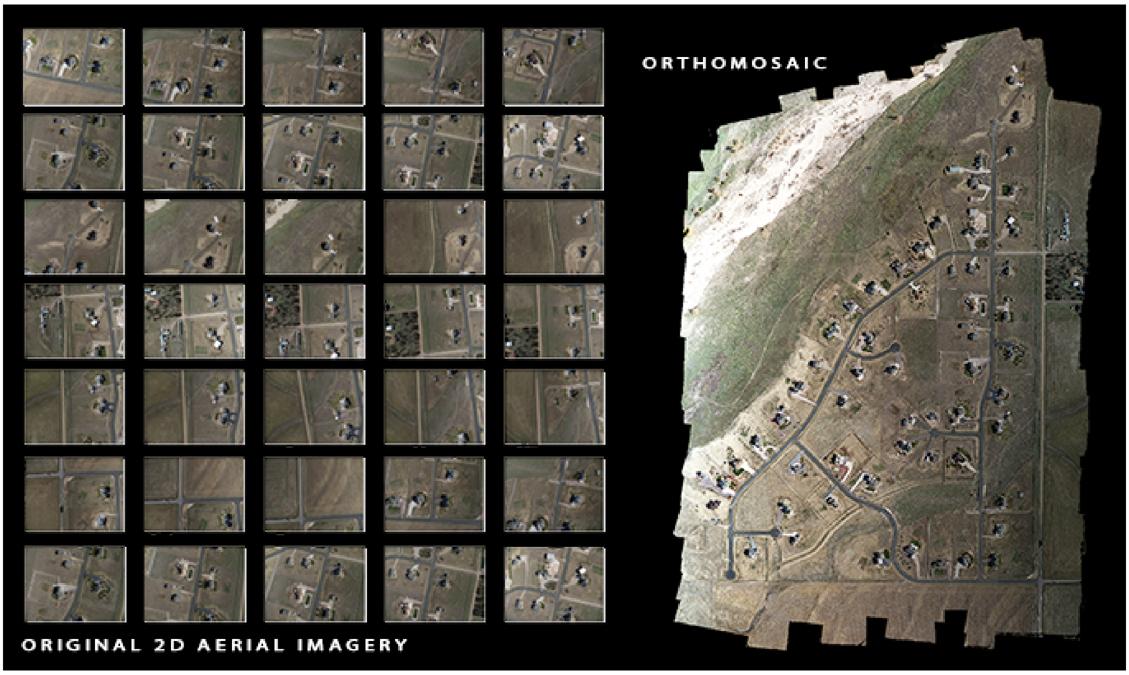


Figure 2.3: Example of an orthomosaic map [9].

2.1.4 Updating FKB-Bygning Objects

Updating the SFKB database with FKB-Bygning objects is a time-consuming and costly process. With recurring intervals, the parties in Geovekst meet, suggest, and conclude which areas in Norway need an update. Geovekst then formulates a tender and sends it to the relevant companies with their product requirements, where the company with the best counter-offer conveys the mission. The company will have to carefully construct a plan that matches the image resolutions and overlaps stated in the requirements. Before carrying out the project, it is sent back to Geovekst for approval.

The company uses calibrated cameras and global satellite navigation system during the flight to correctly register camera position and orientation. Each pixel can then be assigned a coordinate and height value using the camera orientation and position together with photogrammetrical techniques. Measuring and annotating the building objects is carried out manually, eventually constructing a complete FKB-Bygning model.

2.2 Artificial Neural Networks

Artificial Neural Networks (ANN), adaptive systems inspired by the functioning processes in the human brain, may modify their internal structure based on the objective given. The base elements in an ANN are the processing elements (PE) and the connections between them. Each PE receives input and transforms it using the internal function, producing an output [37]. The output is then possibly sent through an activation function, altering the output based on the type of the activation function. Combining interconnected ANNs is considered a Deep Neural Network and is a fundamental element of Deep Learning. Figure 2.4 show a simple Deep Neural Network with one input layer, one hidden layer, and one output layer.

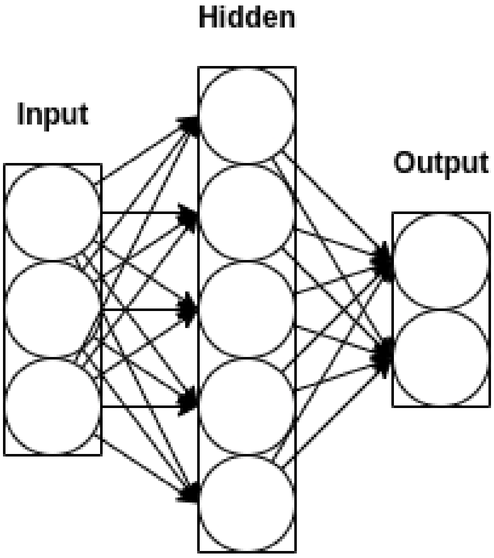


Figure 2.4: Deep Neural Network with one hidden layer.

Combining the internal function of the PE in a network of other PEs allows the DNN to represent complex functions with high-dimensionality [35]. Each connection between the PEs where information propagates is paired with a trainable weight. The input is multiplied by the weight before the PE receives it. Each input $x_1, x_2, ...x_n$ is multiplied by the corresponding weight $w_1, w_2, ..., w_3$. Figure 2.5 visualize the traversal of signals through an PE.

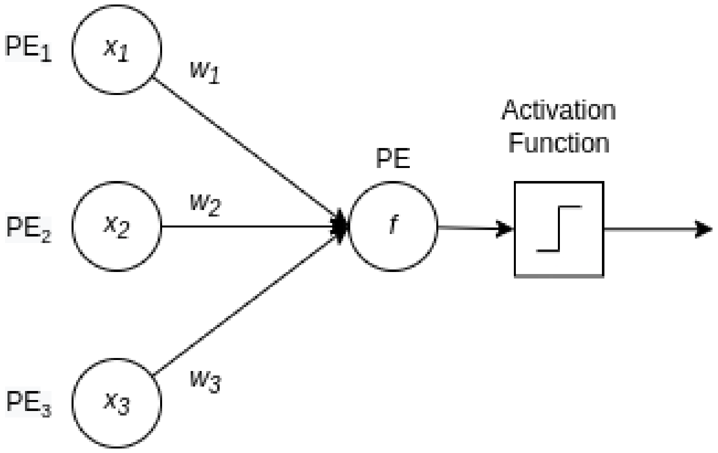


Figure 2.5: Artificial processing entity.

We can use $\sum$ to describe the internal function inside the PE, as the inputs is reduced to a sum. The result is then passed to an activation function, determining the PE's output. We mainly utilize activation functions in Deep Neural Networks to introduce non-linearity. It is necessary to introduce the activation functions in order to learn non-linear patterns. Many different activation functions exist, but none of them has proven to be superior [83].

2.2.1 Activation Functions

Even though none of the activation functions are superior, the Deep Learning Community trends display the most popular activation functions. The activation functions listed in the table below display the top 5 most used activation functions used in papers published to `https://paperswithcode.com`.

Table 2.1: Most used activation function

Papers	Activation Function	Equation
6669	ReLU	$f(x) = max(0, x)$
4540	Sigmoid	$f_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$
4244	Tanh	$f(x) = \frac{2}{1+e^{-2x}} - 1$
3861	GELU	$f(x) = x * \frac{1}{2}[1 + \text{erf}(\frac{1}{\sqrt{2}})]$
670	Leaky ReLU	$f(x) = \begin{cases} 0 & x \leq 0 \\ \frac{100-x}{100} & 0 \leq x \leq 100 \end{cases}$

Researchers do not entirely understand why and how a specific activation function works better for a particular problem and that is why the optimal match is established through trial and error [2, 57].

2.2.2 Loss Functions

Loss functions, also called objective functions, calculate the difference between the predicted value and the ground truth, measuring the model's error. The loss distills all good and bad aspects into a single number, allowing candidate solutions to be ranked and compared [72]. The loss function is a mathematical representation of the objective, so the loss function should measure the success of achieving a particular task. Many well-established loss functions exist, and each performs differently based on the classification task.

Mean Squared Error (MSE) is a well-known linear regression function calculated by averaging the square root of the difference between the prediction and the ground truth. Since the difference is squared, the loss will always be positive [35]. Equation 2.1 defines the function where n is the number of data points, x is the predicted value, and y is the true value.

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{2.1}$$

Cross-Entropy (CE), also called log loss, is another widely used loss function in traditional ML, but also in DL and CV. Whereas MSE is preferred for predicting continuous values (regression), CE is better suited for classifying discrete values (classification). CE is a loss function that applies to all kinds of classification tasks, whether image classification, sentence classification, or pixel classification [35]. Equation 2.2 defines the function where M is the number of classes, p is the predicted probability, and y indicates if the classification was correct, represented by 0 or 1.

$$-\sum_{c=1}^{M} y_{o,c}\log(p_{o,c}) \tag{2.2}$$

2.2.3 Optimization

Optimization in ANNs are algorithms that train the network by using the error from the loss function. The optimizer updates the weights to reduce the loss, improving the model's performance. There exist a wide variety of optimizers, one of the most known is called Stochastic Gradient Descent (SGD). SGD uses the first-order derivative of the loss function to calculate how the weights should be updated so that the function can reach a minimum. Calculating the gradient for each layer has a high computing cost and is why SGD is most often combined with backpropagation. The last layer of weights is the first gradient calculated in backpropagation. The approach allows partial computations from the respective layer to be used in the previous, allowing for a more efficient gradient calculation.

2.2.4 Hyperparameters

Hyperparameters are configurable variables that control the learning of an ANN. These parameters are defined before the training process starts and directly affect the model's performance. Learning rate, optimization algorithms, and loss functions are some examples of hyperparameters.

2.3 Computer Vision

Computer vision is the field of machine learning that makes it possible for a machine to see and give recommendations or make a decision based on what it observes. Using digital images or videos as input allows a deep learning model to gain meaningful information about the visual world.

2.3.1 Computer Vision Tasks

We find many different task objectives within the field of computer vision, from plainly classifying an image to identifying and localizing an object. The various goals can be divided into four fundamental categories:

Category	**Subcategories**
Image Classification	Single-label, Multi-label
Object Recognition	Object Localization, Object Detection
Image Segmentation	Instance, Semantic, Panoptic
Edge Detection	Category-Aware, Category-Agnostic

Table 2.2: Four fundamental categories and their subcategories.

Image classification is the process of assigning a class label to an image. We distinguish between single and multi-label classification. The latter is when an image can be classified into more than one class.

Object Recognition identifies an object and localizes it by drawing a box around it. Object localization detects only a single object type, whereas object detection classifies several objects and labels every box.

Image Segmentation is similar to object recognition, but instead of identifying the objects by drawing a box, each pixel associated with the object is highlighted. Semantic segmentation assigns every pixel a class label, treating multiple objects of the same class as a single entity. In contrast, Instance and Panoptic segmentation handle multiple objects of the same class as separate entities by adding an instance id to each pixel. The differences between them are how they deal with overlapping segments. Instance segmentation allows overlapping, but panoptic does not and therefore adds a new label and instance id for overlapping pixel classes.

Edge Detection also assigns every pixel a class, but only at the boundaries of the objects. For category-aware edge-detection models the edge pixels are assigned the class of the object. On the other hand, category-agnostic models only cares about edges, and will not assign any classes to the edge pixels.

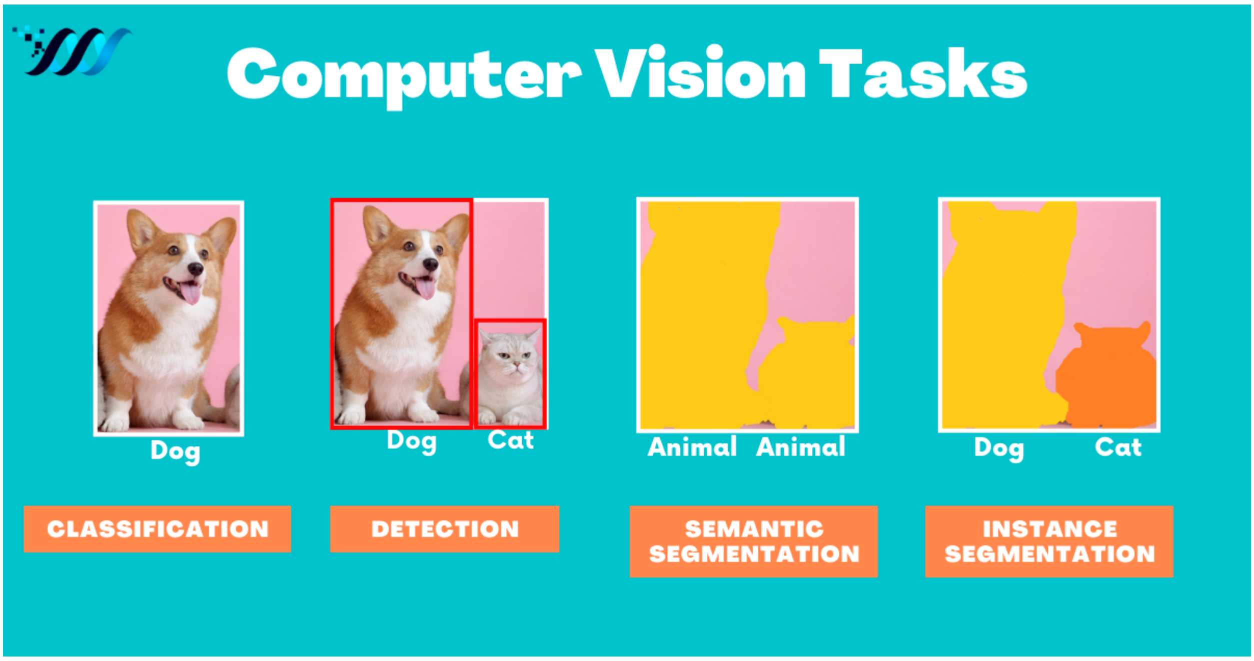


Figure 2.6: Illustration of different Computer Vision tasks [77].

2.3.2 Model Evaluation

Computer vision utilizes different metrics to quantify the model's performance based on the task objective. Since image classification tasks only assign a class to the image, the evaluation of the model is straightforward, calculated using the number of correct and incorrect predictions. In comparison, object recognition and image segmentation adds complexity by not only classifying the object but also locating it, requiring a different type of metric.

Intersection Over Union

Intersection over Union (IoU) is the most common metric for object recognition and image segmentation. IoU measures how well the prediction overlaps with the ground truth, calculated by dividing the intersecting area of their masks by the area of their union. A perfect model would result in an exact match, yielding an IoU score of 1.

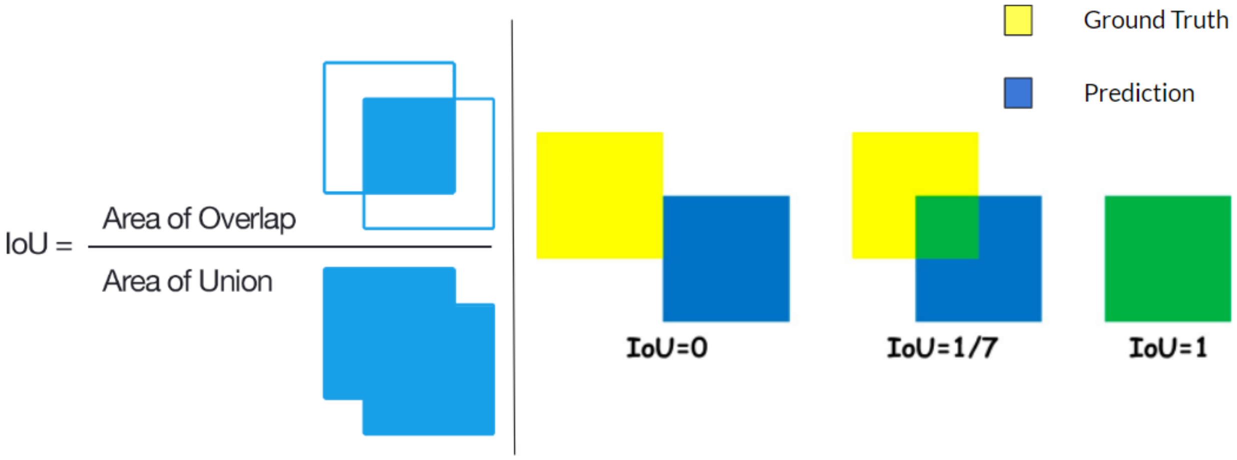


Figure 2.7: Illustrations of how IoU is calculated with visualizations of different IoU values [74] [79].

A limitation of the IoU is varying measures of boundary quality. The number of internal pixels expands quadratically with object size, considerably outnumbering the number of boundary pixels, which grows linearly [17]. This unbalance affects the IoU score, leading to a lower sensitivity to boundary quality in larger objects since all pixels are valued equally.

Boundary Intersection Over Union

The Boundary IoU (BIoU) tackles the problem above by calculating the IoU for pixels within a certain distance from the original masks, reducing the number of internal pixels. As a result, the BIoU is sensitive to boundary quality across all scales, making it more susceptible to edge errors and will better evaluate improvements in boundary quality [17].

$$\frac{\mid (G_d \cap G) \cup (P_d \cap P) \mid}{\mid (G_d \cap G) \cup (P_d \cap P) \mid} \tag{2.3}$$

Given the original masks for the ground truth G and the prediction P, the BIoU first computes a new set of masks referred to as P_d and G_d. These sets contain pixels within distance d from the edges of the masks. As seen in Equation 2.3, calculating the BIoU is similar to IoU, but utilizes P_d and G_d by intersecting them with their respective originals before computing the intersection-over-union.

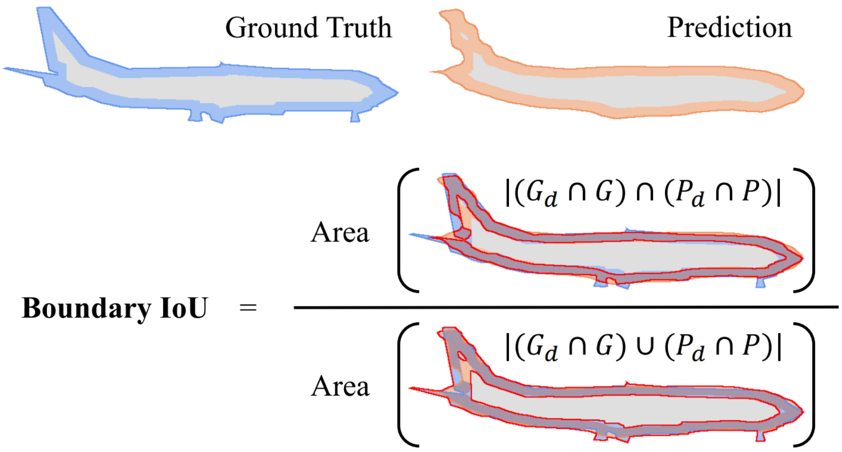

Ground Truth
Prediction
Area
$|(G_d \cap G) \cap (P_d \cap P)|$
Boundary IoU
=
Area
$|(G_d \cap G) \cup (P_d \cap P)|$

Figure 2.8: An illustration of computing the BIoU where the pixels that are within distance d from the edge are highlighted. In this example the IoU is 0.91, whereas the BIoU is 0.73 [17].

This method is substantially more sensitive to boundary errors on large objects than the standard IoU measure, and it does not over-penalize faults on smaller objects [17]. The distance d controls the sensitivity, and if large enough to include all internal pixels, the BIoU will be equivalent to the standard IoU. In contrast, a smaller distance d will ignore internal pixels, making it more sensitive to boundary pixels on larger objects [17].

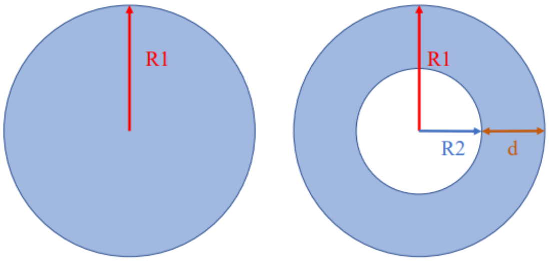


Figure 2.9: Figure illustrating the limitation of the BIoU measure where non-identical masks are given a perfect score [17].

Figure 2.9 show a limitation of the BIoU measure where two non-identical masks are given a perfect score. The reason being is that only pixels within a distance d from the edge of the ground truth or prediction will be evaluated [17]. In this example, all non-matching pixels are further away than d pixels from the boundary, resulting in a perfect score. The authors of the paper state that the BIoU in most cases (99.9%) is smaller or equal to the standard IoU, meaning the chance of a model producing a flawless edge with a large part of missing interior pixels is low.

2.3.3 Convolutional Neural Network

Many computer vision models use an algorithm called Convolutional Neural Network (CNN). The CNN is comparable to an ANN. However, the CNN learns to extract features from the image, reducing the required computing power to learn the weights and biases. The CNN is divided into three layers: *convolutional*, *pooling*, and *fully connected*. A CNN can have several layers of the same type, but the last layer will always be fully connected. If not, the model is not a CNN, but instead a Fully Convolutional Network (FCN).

Convolutional Layer

The object of the convolutional layer is to extract features from the input that can be seen as a set of *feature maps*. The two primary components of the convolutional layer needed to calculate the feature maps are the *kernel* and the *stride*. The kernel, also called a *filter* or *feature detector*, is a weight matrix that gets multiplied by the input pixels in its *receptive field*, which is the area of the image that the kernel focuses on [2]. The output from this calculation is added to a new matrix, the feature map. The filter then shifts by a stride, and the process repeats until the kernel has traveled across the entire image [26]. This procedure is known as a convolution. With a stride of 1, the kernel moves along the x-axis with x+1. When reaching the end, the kernel ascends down the y-axis with y-1. The higher the stride value, the faster the process will be. Although, a stride higher than one is not common.

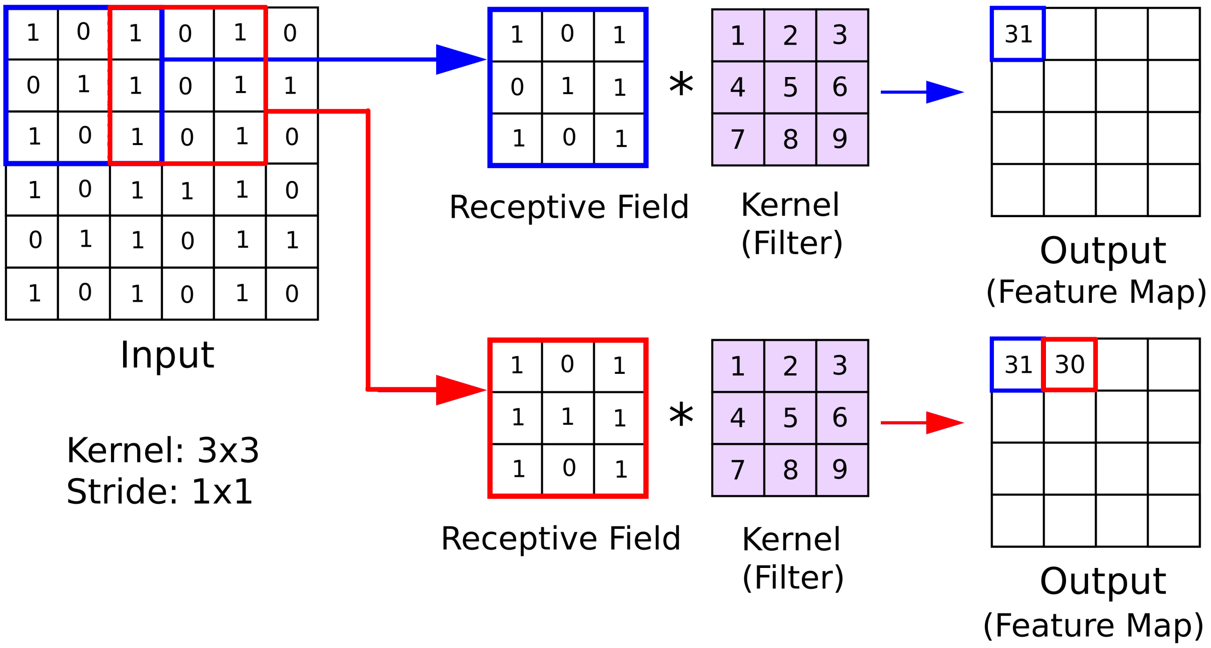


Figure 2.10: Convolution of an 6x6 image as input and a kernel of size 2x2 with a stride of 1 (extended version of [27]).

The CNN learns to recognize features of the image since it looks at the pixels in context. The network can learn patterns and objects and even identify them at different image positions. As mentioned earlier, the computing cost can be drastically lower than a regular ANN. The reason is twofold, local connectivity and shared parameters. Since the feature map does not directly connect to each pixel in the input, the convolutional layer is referred to as partially or locally connected. Each pixel connects to every neuron in a fully connected neural network, which means that every neuron receives complete information from the image. Compared to CNN, the neurons only receive a local group of pixels (reducing the input size). Secondly, the weights in the kernel matrix stay fixed as it moves across the image, resulting in parameter sharing. Consider a 4x4 image and a 2x2 kernel with a stride of 1. The filter only contains four weights, although there are 16 individual pixels. These methods help reduce the number of parameters in the network, making the computations more efficient.

Pooling Layer

Pooling is the process of downsampling the data by reducing the resolution, often added to a feature map. The operation is similar to shifting a kernel in the convolutional layer, but instead of a weight matrix, the filter applies an aggregation function to the receptive field. Pooling can be done in several ways, but the two main methods are Max and Average pooling, illustrated in Figure 2.11 [2]. In this example, we have a feature map of size 4x4 as input, a kernel = 2x2, and a stride = 1 as hyperparameters for the pooling operation, resulting in a 2x2 output volume.

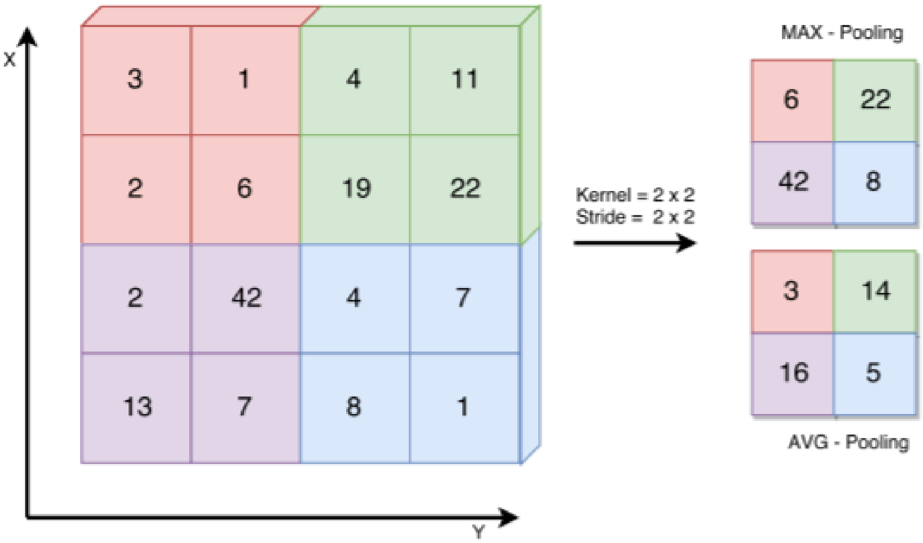


Figure 2.11: Max and avg. pooling [2].

While pooling loses a lot of information, there are distinct benefits. As in the convolutional layer, pooling reduces the computational cost by reducing the number of parameters needed to be optimized. The generalization of features also decreases the risk of overfitting.

Fully Connected Layer

The final layer of a CNN is always fully connected, meaning each neuron in the output layer links to the previous. The job of this last layer is to perform the classification task based on the kernels and feature maps retrieved from the convolutional layers. The fully connected layer usually leverages a softmax activation function to produce classifications with a probability between 0 and 1.

Chapter 3

State-of-the-art

This thesis uses and focuses on several topics within deep learning and computer vision that is in active research. Since this research area advances rapidly, several new papers push the state-of-the-art further each year. This chapter introduces the advent of image classification and then focuses on four different but related areas of importance to the paper. Section 3.2 discuss different semantic segmentation models developed over the years. Further on, Section 3.3 outlines the loss functions available and used in semantic segmentation. Section 3.4 then highlights and explore relevant edge-detection models and methods available. Lastly, Section 3.5 explore the state-of-the-art models for segmentation on aerial imagery.

3.1 Image Classification

Deep learning has been around for over three decades, dating back to the late 1980s. A crucial development for deep learning has been training deep neural networks using backpropagation. [75] introduced backpropagation, a procedure that is still extensively used in neural networks. The procedure works by repeatedly adjusting the weights in the networks where the main goal is to minimize the difference between the output vector of the network and the desired output vector. The measure minimized can be different based on the task at hand.

Yann LeCun introduced [55] in 1989, where they implemented backpropagation for convolutional neural networks. The implementation successfully recognized handwritten zip code digits provided by the US. Postal Service. The paper was an essential step towards the convolutional neural networks still used in state-of-the-art research. There were scaling issues with the existing convolutional neural networks, resulting in SVMs being the better option at the time. Deep neural networks were not able to compete with SVMs for several years. Still, as the processing power exponentially increased, following Moore's Law [38], deep neural networks started to become useful. The training was still slow compared to SVMs, but they could achieve even better results using the same data. In 2006, [43] was released, showing a procedure training one layer at a time in deep neural networks, enabling the great success and advancement achieved within the realm of deep learning.

This thesis relies on the computer vision field of deep learning. Computer vision is a broad area within deep learning, specializing in image analysis and information retrieval. Common tasks in computer vision is image classification, object detection, and image segmentation, further detailed in 2.3. Furthermore, for image classification, object detection, and image segmentation the convolutional neural network (CNN) has proven to be one of the most successful architectures.

Some of the most known architectures used for computer vision tasks is AlexNet [54], VGG [80], ResNet [41], and Inception [84]. AlexNet is the deep learning architecture, composed of convolutional and maxpooling layers, that popularized CNNs for computer vision. After AlexNet, VGG was proposed, which has very similar structure as AlexNet, but it is much deeper and consists of more convolutional layers. The large size of VGG enables it to achieve new state-of-the-art results, but makes it computationally expensive to train and requires a large dataset to be able to generalize [80]. ResNet is an even deeper CNN model, but introduces a residual learning framework reducing the training load. The layers are reformulated as learning residual functions referencing to the input of the layer, allowing to train very deep CNNs without the expensive computationally training that comes with deep neural networks [41]. Inception focus on utilizing the internal computing resources efficiently, using a carefully crafted design that allowed for increase in size and depth with constant computational cost [84].

More recently, the transformer architecture is adopted for computer vision tasks. Originating from the Natural Language Processing domain, the transformer introduced in [86] is based solely on attention mechanisms. The attention mechanism can be described as mapping a query and a set of key-value pairs to an output, where the output, query, key, and value are vectors. The output is a weighted sum of the values, where the weights are computed by a compatibility function [86]. In 2020, *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale* [22], introduced the first vision transformers with state-of-the-art results on multiple small and mid-sized image recognition benchmarks (CIFAR-100 [53], ImageNet [19], etc.).

3.2 Semantic Segmentation

Semantic segmentation is finding and assigning the correct class to each pixel in an image, resulting in annotations of each class. Semantic segmentation models is used to correctly classify all pixels in an image. One example would be the classification of buildings using aerial images. A group of classified pixels that have the same class is called a mask. Using the predicted mask in combination with other information, such as the scale of the image, gives meaningful data about objects. Every year, new state-of-the-art results on semantic segmentation datasets are claimed. Each model utilizes various techniques to segment the objects of interest. Many domains benefit from advances in semantic segmentation, such as fully automated driving, medical image diagnosis and disaster prediction.

Image segmentation techniques that are not based on deep learning usually clusters pixels based on pixel values and information such as spatial distance, contours, and edge information [90, 45]. Employing techniques such as Markov processes [31] and combining contour detection using a hierarchical approach [3] has improved upon the clustering methods. Despite the improvements, the aforementioned techniques is currently obsolete in image segmentation due the great success of deep learning. Using deep neural networks in image segmentation has allowed the field to evolve and achieve excellent results. Under the umbrella of image segmentation using deep learning, several different architectures and training methods have been proposed to improve the results in general.

We will first look at the approach of using fully convolutional neural networks. Long *et al* [60], proposed one of the first fully convolutional neural network approaches for semantic segmentation. The model takes an arbitrarily sized input and produces a correspondinglysized output. They also introduced a novel feature called concatenation, where semantic information from a deep layer merges with the appearance information from a shallow layer. The model and models alike it can be seen as a milestone in the field of image segmentation, demonstrating a convolutional neural network trained for semantic segmentation in an end-to-end manner. Even though the models achieved good results, they were not efficient enough to use in real-time scenarios. In addition, it was not exploiting the global context efficiently [64].

Image segmentation have adopted the encoder-decoder structure with great success. The encoder-decoder models encodes the image first into a high-dimensional latent space, to extract and express key features of the image. The decoder extracts the important information, and the result is often a percentage-wise class prediction per pixel. A vital technique for the decoding process of these networks is the transposed convolution, introduced as a part of DeconvNet [68]. Transposed convolutions, introduced as deconvolutions, associate a single input activation with multiple outputs, using learnable upsampling filters [68]. Several well-known models utilize this type of structure, such as SegNet [5], U-Net [73] and HRNet [88]. Furthermore, SegNet introduced an improvement over the DeconvNet where the decoder uses pooling indices computed in the max-pooling step to perform non-linear upsampling. Using this technique removes the need to learn the upsampling of an image. The U-Net model adds skip-connections to help the network keep details of images through the encoder-decoder process. Skip connections pass information across the encoder-decoder structure, keeping information that is lost when image data is encoded, as seen in Figure 3.1.

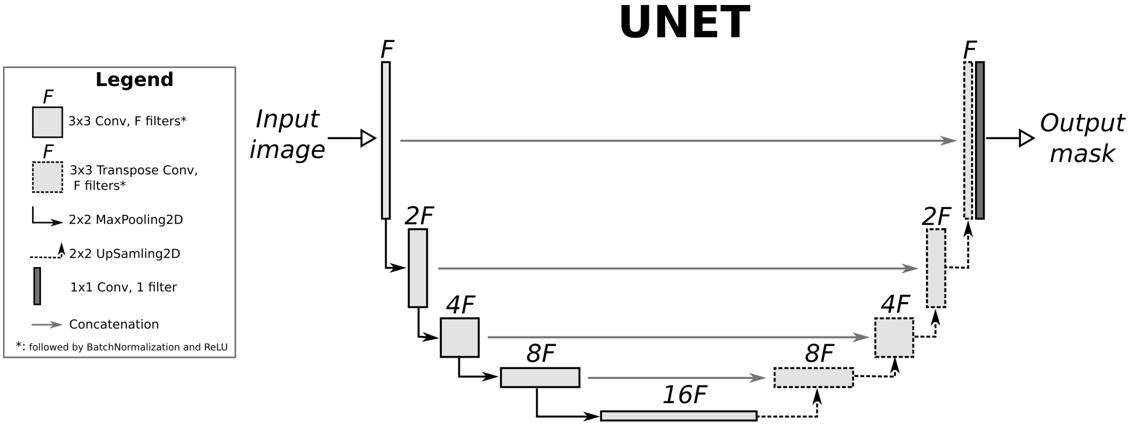


Figure 3.1: U-Net architecture [67].

HRNet differs from the other encoder-decoder structure by having more connections and convolution blocks within the network. The network has three parallel convolution streams connected repeatedly to exchange information across resolutions. The result is semantically richer and spatially more precise representations [88], however the repeating connections increases the resources needed to train models using HRNet. Figure 3.2 visualizes the interconnected structure of HRNet.

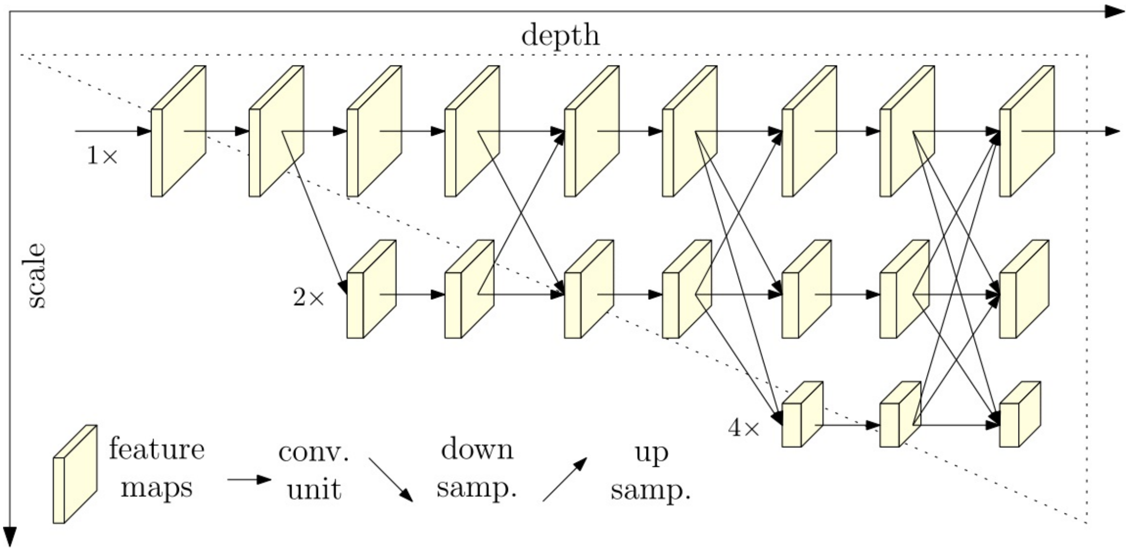


Figure 3.2: HRNet architecture [88].

Dilated convolutional neural networks utilize the concept of dilated convolutions, where the convolutions have holes between the convolution points, visualized in Fig 3.3. The idea has been around for decades under different names and was first used in the early 90s [78]. Some of the most important algorithms using dilated convolutions to this date have been the Deeplab family of models [14, 11, 13, 12], DenseASPP [95], and ENet [70].

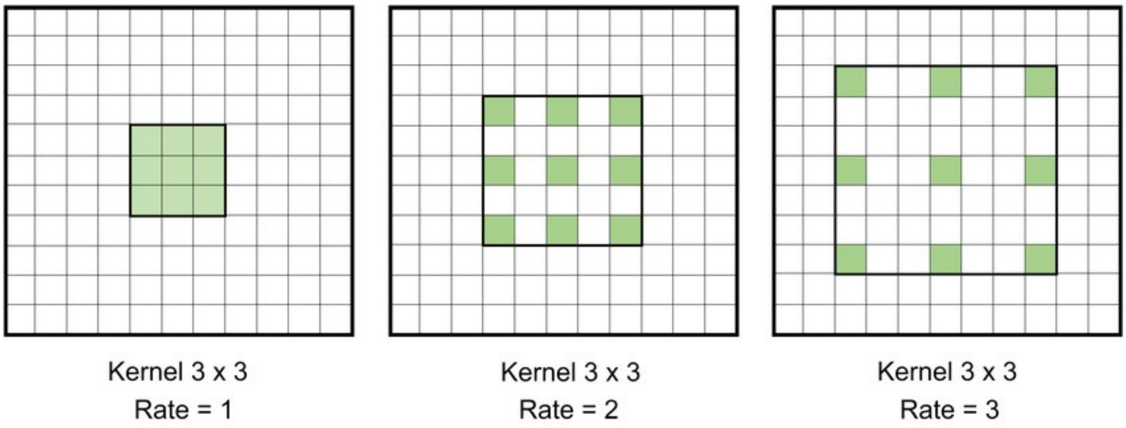
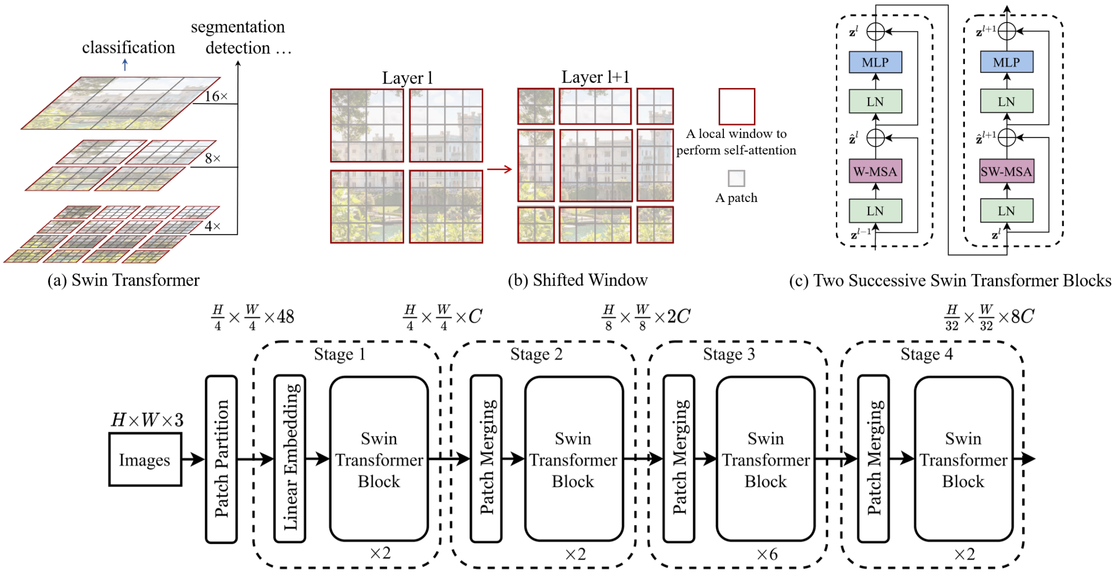


Figure 3.3: Dilated convolutions with different rates [66].

Deeplabv3+ is the best and most recent model in the Deeplab family and has been adopted and used in many domains, such as aerial image segmentation [89, 62, 42]. It utilizes improvements from the previous deeplab versions, such as atrous spatial pooling pyramid, depthwise convolutions, and an encoder-decoder structure.



(a) Swin Transformer

(b) Shifted Window

(c) Two Successive Swin Transformer Blocks

(d) Architecture

Figure 3.4: Swin transformer architecture [63].

The Vision Transformer (ViT) [23] and variants have become extremely popular and claims new state-of-the-art results of many datasets each year. The ViT architecture achieves state-of-the-art results by taking advantage of attention mechanisms and not utilizing convolutions, though it is known to be computationally expensive. One of the best-performing vision transformers is the Swin (Shifted Windows) Transformer [59]. The Swin Transformer reduces the expensive computational operation of the Vision Transformer by reducing the computation of self-attention to non-overlapping local windows. At the same time, it still allows for cross-window connections [59], acquiring global context.

Empirical Evaluation

The segmentation models mentioned in the former subsection have been developed with different goals, affecting the datasets used for evaluation. Table 3.1 show a list of empirical results for different models acquired on different datasets. It is hard to perfectly compare the models as they have not been evaluated on the same dataset under the same conditions. Note that only comparing the IoU values is insufficient without looking at what dataset they are evaluated on. An example of this would be the swin-transformer, which despite having the lowest IoU value, is ranked as the second-best model on https://paperswithcode.com for the ADE20K dataset.

Table 3.1: Comparison of state-of-the-art segmentation models. Most of the models have been tested on different datasets, making it hard to evaluate which model is best.

Model	**IoU (%)**	**Dataset**
HRNet OCR Multi-Scale [85]	86.3	CityScapes Val [29]
DeeplabV3+ [12]	79.5	CityScapes Val [29]
U-Net [73]	77.5	DIC-HeLa
DeconvNet [68]	72.5	PASCAL VOC 2012 [48]
SegNet [5]	60.1	CamVid [30]
Swin-Transformer [59]	53.5	ADE20K val [91]

3.3 Segmentation Loss

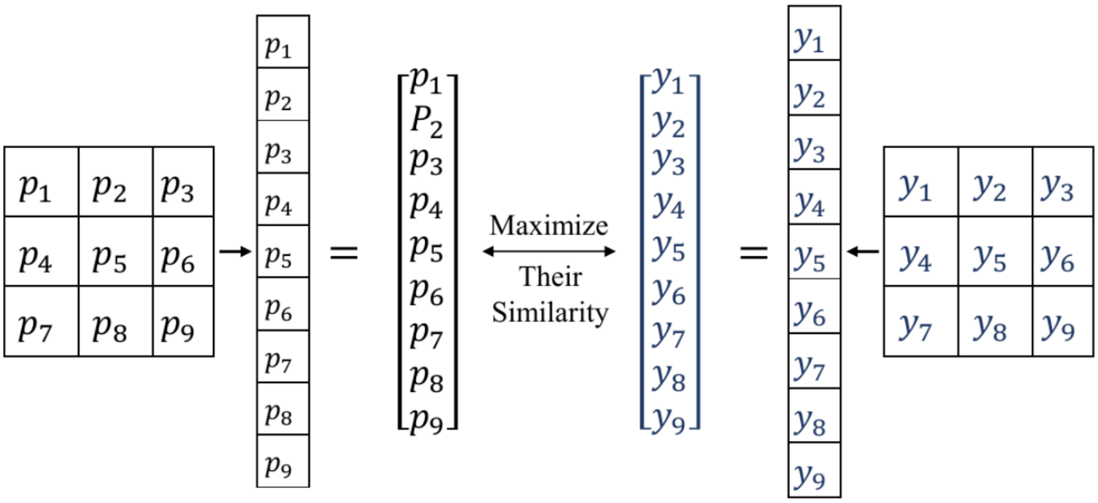
Using the correct loss function is vital for the performance of the model on the task at hand. In semantic segmentation we want to use loss functions that is able to correctly predict which class a pixel belongs to. Measuring the correctness of the prediction has different approaches, allowing us to divide the loss functions for semantic segmentation into three types: distribution-based, region-based and boundary-based. Furthermore, it should be noted that combining different types of loss functions may improve the performance and training of the model.

3.3.1 Distribution-Based

Cross-entropy is a distribution-based loss function frequently used to measure how well the prediction matches the ground truth. It assesses each pixel individually based on the prediction certainty and whether the prediction was correct or wrong. It works best when used on datasets with equal data distributions among classes [46]. Below we can see the softmax cross-entropy loss, where $y \in \{0,1\}$ is the ground truth label, $p \in [0,1]$ is the estimated probability, N is the number of pixels, and C is the number of classes [98].

$$L_{CE}(p,y) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C} y_{n,c}\log(p_{n,c}) \tag{3.1}$$

Fixing the shortcomings of cross-entropy loss can be done in a few ways. One of the ways, named weighted cross-entropy loss, is to weigh the positive examples by some coefficient, often related to the class distribution in the dataset. In [98], Zhao *et al.* introduced an improved cross-entropy loss, named Region Mutual Information(RMI) loss for semantic segmentation. The motivation for the new loss function was the lack of using natural information found in adjacent pixels when using cross-entropy loss. As seen in Eq. 3.1, cross-entropy loss evaluates each pixel as individual pixels and ignores the information inherent in the adjacent pixels. The new RMI loss uses one pixel and its neighboring pixels to represent this pixel. Each pixel in the image is represented by a multi-dimensional point, encoding the relationship between the pixels, acquiring a multi-dimensional distribution of high-dimensional points, visualized in Figure 3.5. Their experimental results show significant and consistent improvements in the performance on segmentation tasks such as PASCAL VOC 2012 and CamVid datasets.



p1 p2 p3
p4 p5 p6
p7 p8 p9
p1 p2 p3 p4 p5 p6 p7 p8 p9
=
p1 P2 p3 p4 p5 p6 p7 p8 p9
Maximize
Their
Similarity
y1 y2 y3 y4 y5 y6 y7 y8 y9
=
y1 y2 y3 y4 y5 y6 y7 y8 y9
y1 y2 y3
y4 y5 y6
y7 y8 y9

Figure 3.5: RMI loss pixel representation [98].

3.3.2 Region-Based

Region-based loss functions tries to maximize the overlap regions between the predicted segmentation and the ground truth. Dice loss is a region-based loss function utilizing the dice coefficient to calculate the similarity between two images [46, 24, 82]. Sudre *et al* [82], show that on medical segmentation datasets, loss functions based on overlap measures, such as the dice loss, are more robust as class imbalance increases compared to cross-entropy loss. Another loss function using overlap measures is the lovasz-softmax loss [6], which directly optimizes mean intersection-over-union loss for neural networks. The authors show improved segmentation masks and IoU scores using the lovasz-softmax loss compared to the distribution-based cross-entropy loss. Furthermore, in [8], the authors compared dice loss and lovasz-softmax loss on a medical image segmentation. Both losses were found to approximate each other, and they show that there is generally no significant difference between the use of either of the metric-sensitive loss functions.

3.3.3 Boundary-Based

One of the issues both pixel-wise and region-based loss functions have is the lack of attention to the segmentation mask boundaries, resulting in rough segmentation mask edges. These rough edges have inspired the development of boundary-aware loss functions. The goal of the boundary-aware loss functions is to improve the edges of the predicted segmentation masks. One common way to use boundary information when training deep neural networks is through multi-task training, where there are additional network branches to detect boundaries. The problem with this approach is the difficulty of adequately fusing the boundary information with the predicted semantic mask[87, 16, 25]. There have also been works such as [20, 7, 15] that use the information flow through boundaries by learning pairwise pixel-level affinity. The problem with the boundary loss techniques above is that neither is model-agnostic. Standalone boundary-aware loss functions such as Boundary Loss [51] and the related Active Boundary Loss [87] solves this problem. Both loss functions focus on reducing the distance between the boundaries of the predicted and ground truth segments. Boundary loss does this by minimizing the regional integral, which may weaken the influence of pixels near the ground truth boundaries, due to the distance weights being smaller there [87]. On the other hand, active boundary loss focuses on the predicted boundary pixels, which may allow for a better alignment [87]. Active Boundary Loss cannot be used without pairing it up with other loss functions, as it is class-agnostic. In [87] they use a compounded loss consisting of Cross-Entropy Loss, Lovasz-Softmax Loss and Active Boundary Loss to achieve the best results. Combining several loss functions may make the training more complex and cause issues to learn the correct objective.

3.4 Edge-Detection

Class-agnostic edge-detection models is used to precisely delineate objects in an image, separating objects of interest from the background. The application of edge-detection models can therefore be to supervise or refine the predicted edges of a down-, or up-stream model, respectively. Before deep learning became popular, commonly used edge-detection methods include Sobel filters, and Canny edge-detection algorithm [10]. As deep learning became popular, there have been successful attempts to use it for edge-detection, acquiring better results compared to the techniques that are not using deep-learning.

Today, several deep-learning-based edge-detection algorithms have acquired superior performance to other approaches. HED [92] is one of the most famous edge-detection techniques.

The detection technique uses a trimmed VGG16 finetuned on ImageNet to generate multi-level features. Each stage produces a side output that is evaluated and backpropagated through the network using hidden layer supervision [92]. In [49], they have combined the HED technique with an encoder-decoder segmentation network. The output of HED merges with the output of the segmentation network in a boundary-enhancing module, achieving overall better results for various encoder-decoder segmentation networks [49].

A second approach, which is also VGG [80] based, is named Richer Convolutional Features (RCF). In RCF, they propose to use all of the convolutional layers in each stage, rather than just the last, to acquire a richer feature representation. They also introduce the multi-scale test technique, where they feed the network with input images of three different sizes, 0.5, 1.0, and 1.5. All result images are resized and merged to the original size to produce the output prediction.

A more recent technique is the Bi-Directional Cascade Network(BDCN). The authors of [40] makes the argument that using the same supervision for all network layers is not optimal. The reason is that different network layers depict patterns at different scales resulting in sub-optimal supervision for many of the layers. They solve the problem by training the shallow layers to focus on details while the deep layers focus on object-level boundaries. The BDCN architecture allows for the propagation of outputs between the adjacent higher and lower layers, resulting in incremental edge prediction. In other words, the bi-directional cascade structure provides for a more reasonable training procedure [40].

The previously mentioned edge-detection models struggle with detecting crisp edge maps free of localization ambiguity due to the mixing phenomenon of CNNs. In 2021, Huan textitet al [44] published a paper with two novel modules: A tracing loss performing feature unmixing by tracing boundaries to learn better using side edges, and a fusion block for side mixing and aggregation of learned side edges. The loss module and the fusion block can be paired with the previously mentioned edge-detection methods, improving edge prediction for all three models. Figure 3.6 show a visual comparison between different edge-detection models. The last column of the figure show an improved boundary quality compared to the other models.

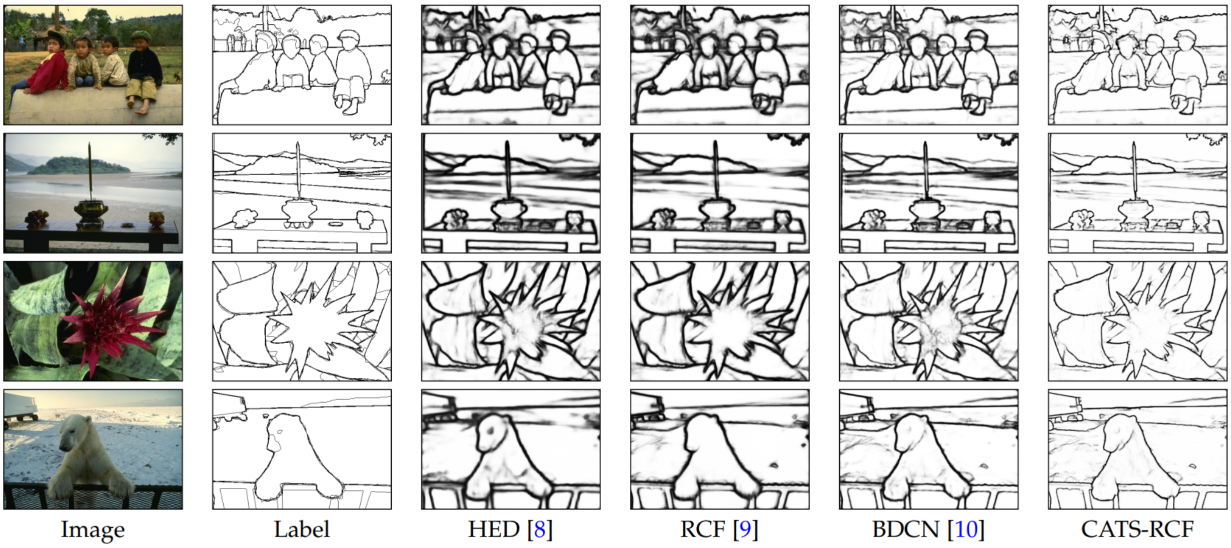


Figure 3.6: Visualized comparison of predicted edges from different edge-detection models [44].

All of the edge-detection models mentioned is based on the VGG backbone [80], which is an architecture released in 2014, nearly 8 years ago. There are many new state-of-the-art backbones currently being used in other types of neural networks, such as the HRNet [88] backbone, released in 2019. Therefore, it is plausible to assume that by adopting the HRNet backbone, or any other state-of-the-art backbone, for edge-detection would improve the results and edges obtained.

Empirical Evaluation

The edge-detection models mentioned in the former subsection have all been evaluated on the NYUDv2 (NYU-Depth V2) dataset using the ODS (Optimal Dataset Scale) measure. The ODS is calculated by iterating over all possible threshold values and choosing the threshold that gives the best F-Score [39]. In Table 3.2 the evaluation results for the edge-detection models on the NYUDv2 Dataset is displayed. The results show that CATS is currently the best performing model on NYUDv2 by a small margin.

Table 3.2: Comparison of edge-detection models on the same dataset. Using the ODS-Score it is clear that the CATS model is the best performing model on the NYUDv2 dataset.

Model	**ODS (F-Score)**	**Dataset**
CATS [44]	0.770	NYUDv2 [18]
BDCN [40]	0.766	NYUDv2 [18]
RCF [58]	0.764	NYUDv2 [18]
HED [92]	0.746	NYUDv2 [18]

3.5 Aerial Image Segmentation using Deep Learning

Section 2.1.4 outline the process of updating FKB-Bygning, where the last step was to manually annotate the buildings and their properties, ensuring an up-to-date database. However, repeatedly annotating buildings that have not changed is an unnecessary use of labor. Image segmentation models can reduce excessive work by detecting building changes. Comparing segmentation masks for the same area on data captured years apart will highlight the locations where buildings have changed, allowing the manual annotations to be location-specific. Locating and precisely segmenting buildings is essential for detecting new, extended, and demolished buildings, motivating the research for advancing the field of image segmentation on aerial images.

The most common approach for aerial image segmentation is applying a state-of-the-art model and training it to segment buildings, roads, or other objects of interest. Among them, Pan *et al.* [69], Zhang *et al.* [97], and Xing *et al.* [93] have all used a basic U-Net architecture with good results [73]. However, as with any other segmentation task, the issue of coarse segmentation boundaries occurs, especially for small and thin objects [87].

Other approaches apply modifications to the base architectures, allowing them to extract specific object features. Yue *et al.* [96] designed novel adaptive layers in a model named TreeU-Net, improving the segmentation of buildings using aerial images. In [21], Doi and Iwasaki utilized focal loss for vague aerial images to extract focused features. Kim *et al.* [52] introduced multiple pyramid pooling layers to extract multi-scale features. However, the issue with coarse segmentation boundaries still occur.

In [34] the authors propose a technique to overcome the issue and improve on the segmentation boundaries. The new approach utilizes semantic segmentation with edge-detection to improve road detection. The model encodes the features in full resolution using attention maps and feeds them into a segmentation model to produce segmentation masks. Then, they add the segmentation masks with the encoded features before feeding them into an edge-detection network, making better edges due to the additional information gained from the segmentation masks. The segmentation and edge-detection sub-models have different loss functions, where the loss is summed and backpropagated through the entire architecture. The model is rather complex but has the advantage that it trains end-to-end. However, it is more difficult to substitute the model's edge-detection and segmentation parts to test other architectures due to their tight coupling.

In 2021 Lee *et al.* [56] introduced a two-scheme approach utilizing separate segmentation networks, in addition to a novel B-Loss and USIM module, to enhance boundary predictions. The method uses two segmentation models, where the first model creates a segmentation map. Furthermore, a USIM operator combines the segmentation map with the original RGB image, creating an image twice the size of the original input images. The newly generated image passes through the second segmentation model. The two scheme method outperforms the baseline one-scheme model by a considerable margin. However, training the two-scheme method using large input images end-to-end is resource-intensive and needs large amounts of GPU ram. In addition they display the improvements of their method using older models that is not state-of-the-art. As it is easier to improve upon medium results, rather than good results, it might seem that the method works better than it really do. As a last note, they add the USIM operator inside of the decoder-network in the second segmentation network, making it hard and complex to substitute the second model with a different segmentation network.

Empirical Evaluation

The aerial image segmentation models mentioned in the former subsection have not been evaluated on the same dataset. In Table 3.3 evaluation results for the segmentation models is displayed. TreeU-Net have used F-Score, which is a different measure than the other models. Furthermore, the EdgeSeg model is evaluated on road segmentation instead of building segmentation, making it difficult to do a fair comparison between the models and therefore it is not clear which model is best.

Table 3.3: Comparison of different segmentation models and techniques used for aerial image segmentation. It is not clear which is best because the models have not been tested on the same dataset and under the same conditions.

Model	**F-Score / IoU**	**Dataset**
TreeU-Net [96]	97.3 (F-Score)	ISPRS Potsdam [33]
EdgeSeg [34]	76.6 (IoU)	Massachusetts Road Dataset [65]
B3SM [56]	71.13 (IoU)	Inria Aerial Image Dataset [61]
UnetPPL [52]	70.34 (IoU)	Self-Made Dataset

Chapter 4

Method

The primary objective of this thesis is to produce a method that improves the boundary quality of segmentation masks for buildings compared to standalone segmentation models. We utilize a Data Enhancement Technique (DET) to achieve this goal. The method highlights buildings, creating a better input foundation for a secondary segmentation network. Additionally, we introduce a new loss combination that evaluates the quality of boundaries. This chapter will present the approaches and techniques used to develop the edge improvements mentioned above. The purpose is to make the research transparent and verifiable. Figure 4.1 provides a visualization of DET, describing the general approach.

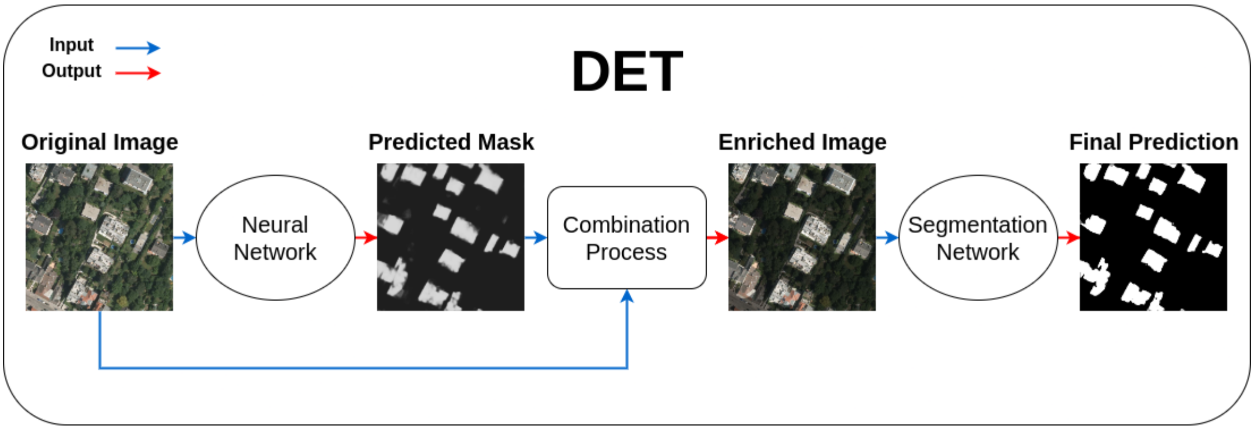


Figure 4.1: A general visualization of DET, showing the process of using the predictions of a neural network in combination with the original image data, allowing the segmentation network to leverage the learned building representations.

4.1 Datasets

For our experiments, we use two datasets. The first is a private dataset acquired and maintained by Kartverket (the Norwegian Mapping Authority). The other dataset is the public *Inria Aerial Image Labeling Dataset* [61]. Both datasets use recent orthorectified images as input data and current building-outline records as ground truths. We will dive deeper into the two datasets in the two following subsections.

4.1.1 Building Dataset from Kartverket

Our research collaborates with Kartverket, giving us access to their private data set. This data contains high-quality imagery used in map production. To make our findings more applicable and to emphasize the potential of DET, we use their dataset to test and evaluate our technique.

The dataset from Kartverket has been extracted from a large building area in the middle of Norway, as visualized in Figure 4.2. From this area we use all data available that matches the criteria where 5% of the image must be covered by buildings, resulting in 28 615 image and ground truth pairs. Each image and ground truth covers an area of 0.01 square kilometers, and the image dimension is 512x512 pixels, resulting in each pixel covering approx. 0.04 square meters. The dataset requires at least 5% of each image and ground truth pair to be buildings. We split the photos into train, validation, and test sets with 80%, 10%, and 10% of the data, respectively.

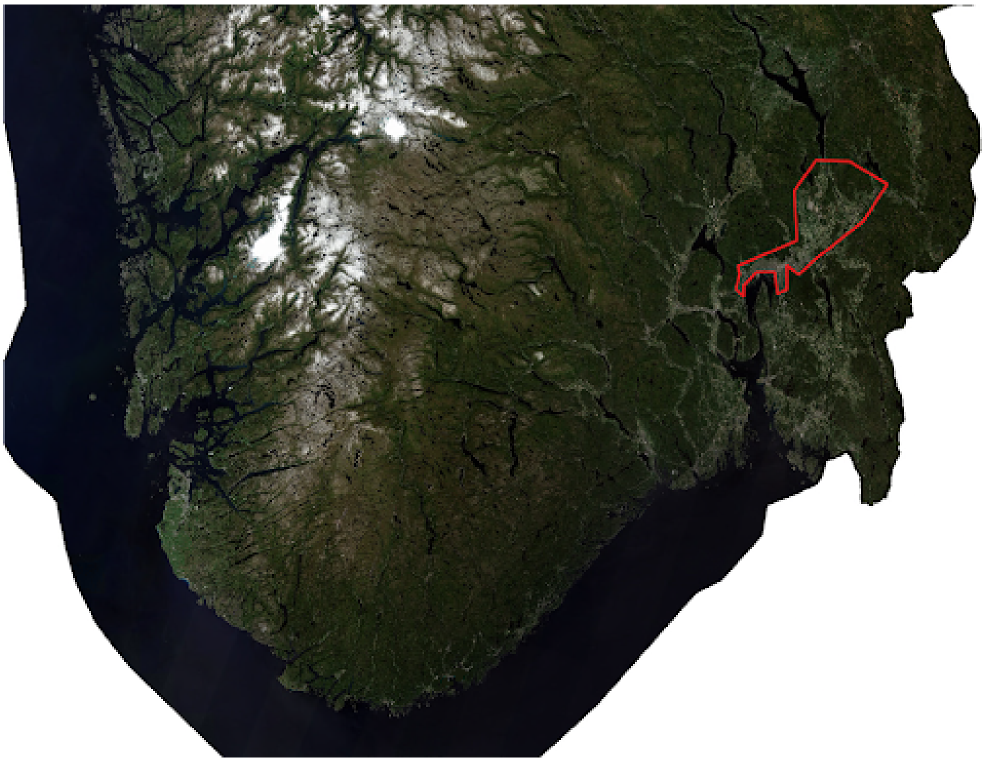


Figure 4.2: Area used to train, validate, and test the model.

An issue with the dataset is that the ground truths do not fit the buildings in the images, caused by the orthorectification process. When the raw aerial images are taken, all pixels not directly below the camera are captured at an angle. The orthorectification process converts the raw images into a form suitable for maps by removing aircraft motions and terrain-related distortions, resulting in shifted images. Figure 4.3 visualizes the shifted ground truths on top of an image, portraying the shift.

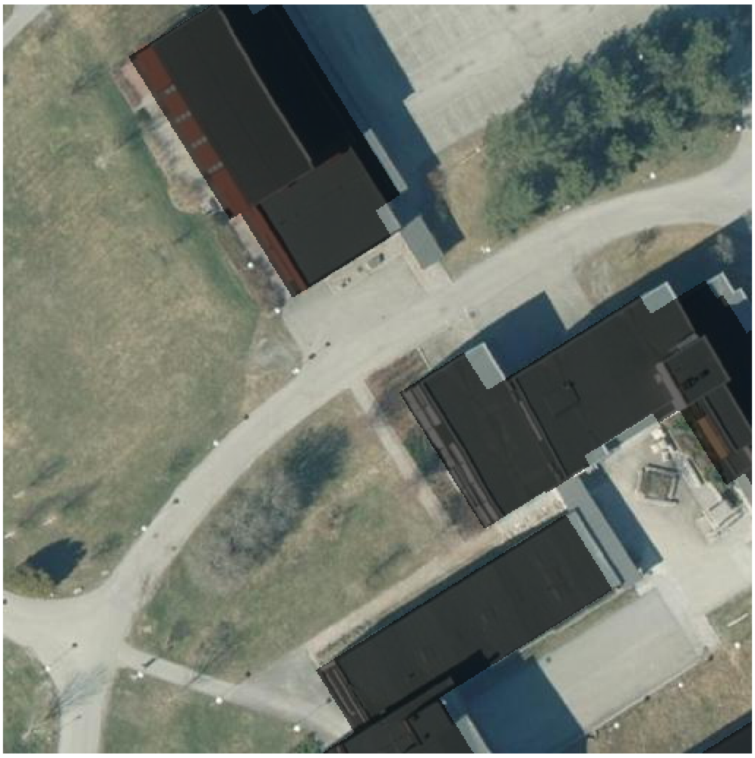
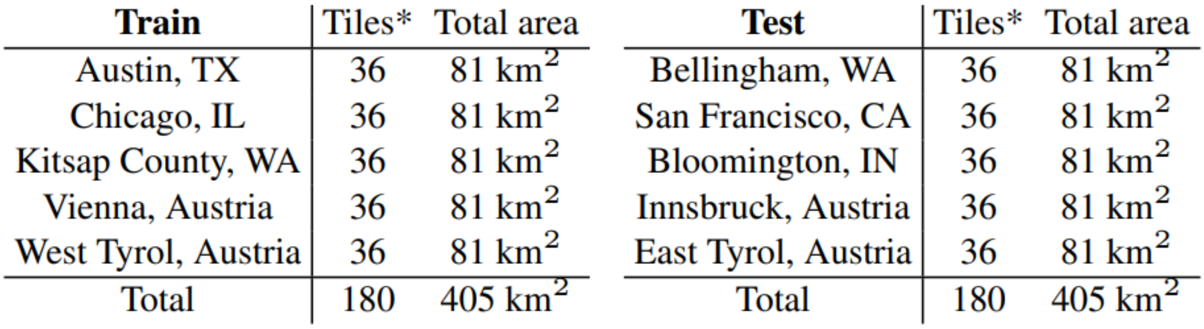


Figure 4.3: Image portraying the shifted ground truths on top of an image.

The ground truths are generated using FKB-Bygning objects from SFKB, described in 2.1.1, which is continuously updated as information is received and reported. The raw images used to create the orthophotos are created from the most recent data acquired for the area. Thereby the images and ground truths will not always be an exact match. There are buildings in the images that do not exist in the database and vice versa. The discrepancies mentioned above between ground truth and input images lead to a sub-optimal training process.

4.1.2 Inria Aerial Image Labeling Dataset

The motivation behind the *Inria Aerial Image Labeling Dataset* [61] is to measure and test the generalization of the segmentation models used for building segmentation. The same buildings in some regions often look different in other regions, making it a highly practical problem paying great dividends when solved. In the current literature, it is common to split data from the same area into train, validation, and test splits. However, the Inria dataset is designed so that the images in the test set are from other cities than those found in the training set [61], the exact split can be viewed in Figure 4.4. The ground truths for the test set are kept private, and users can test their scores by submitting their predictions to the competition live on their website. For the experiments, the test set is not available as the website is down for maintenance. The models is therefore evaluated on the specified validation set.



Train	Tiles*	Total area	**Test**	Tiles*	Total area
Austin, TX	36	81 km^2	Bellingham, WA	36	81 km^2
Chicago, IL	36	81 km^2	San Francisco, CA	36	81 km^2
Kitsap County, WA	36	81 km^2	Bloomington, IN	36	81 km^2
Vienna, Austria	36	81 km^2	Innsbruck, Austria	36	81 km^2
West Tyrol, Austria	36	81 km^2	East Tyrol, Austria	36	81 km^2
Total	180	405 km^2	Total	180	405 km^2

Figure 4.4: Train and test splits used in the Inria dataset [61].

We split the training data into train and validation, thus allowing us to evaluate the training of the neural network. We choose the first four out of the 36 tiles per city for the validation dataset, using approx. 10% of the training data as validation. Each tile in the Inria dataset is an image with 5000x5000 pixels, where each pixel covers approx. 0.06 square meters. We split each tile into patches of 500x500 pixels as input for the models, resulting in 18 000 images. We zero-pad the images to a dimension of 512x512 for all models to pass them through the networks. This padding is due to the encoder-decoder style and the network's depth, forcing the images to be divisible by 2, K times, where K is the depth of the network.

4.2 Architecture

As described in Section 2.3.3, CNNs obtain abstract features from the input, and in regards to building predictions, an essential component is the building edge. DET combines the strengths of two different neural networks to improve the precision and accuracy of the edges of the predicted segmentation masks. We propose two different approaches to DET: (1) Seg-DET, employing two segmentation networks, and (2) Edge-DET, utilizing an edge-detection network combined with a segmentation network. Figure 4.5 and 4.6 show visualizations of Seg-DET and Edge-DET, respectively. The goal of DET is to leverage different properties of neural networks with various architectures and combine their strengths to achieve better segmentation boundaries. Furthermore, we believe that providing the second neural network with task-specific information through the specified methods corrects errors produced by the first neural network. On a final note, the architecture allows for easy exchange of the first and second models in both techniques, enabling the user to find and use the models best fit for their domain.

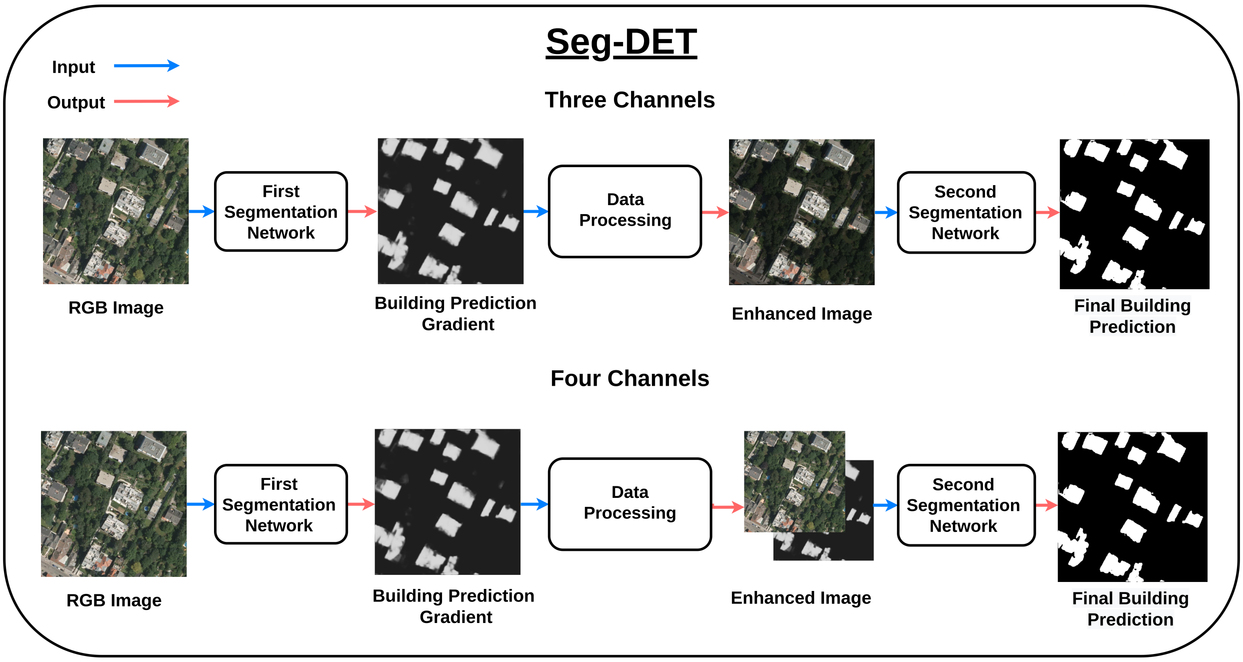

Seg-DET
Input
Output
Three Channels
RGB Image
First Segmentation Network
Building Prediction Gradient
Data Processing
Enhanced Image
Second Segmentation Network
Final Building Prediction
Four Channels
RGB Image
First Segmentation Network
Building Prediction Gradient
Data Processing
Enhanced Image
Second Segmentation Network
Final Building Prediction

Figure 4.5: Seg-DET Architecture: We enhance the data using the building predictions from the first segmentation network and the original input images through a data enhancement technique further explained in Section 4.4.1. The second segmentation network then receives the enhanced data as input, either as three or four channels.

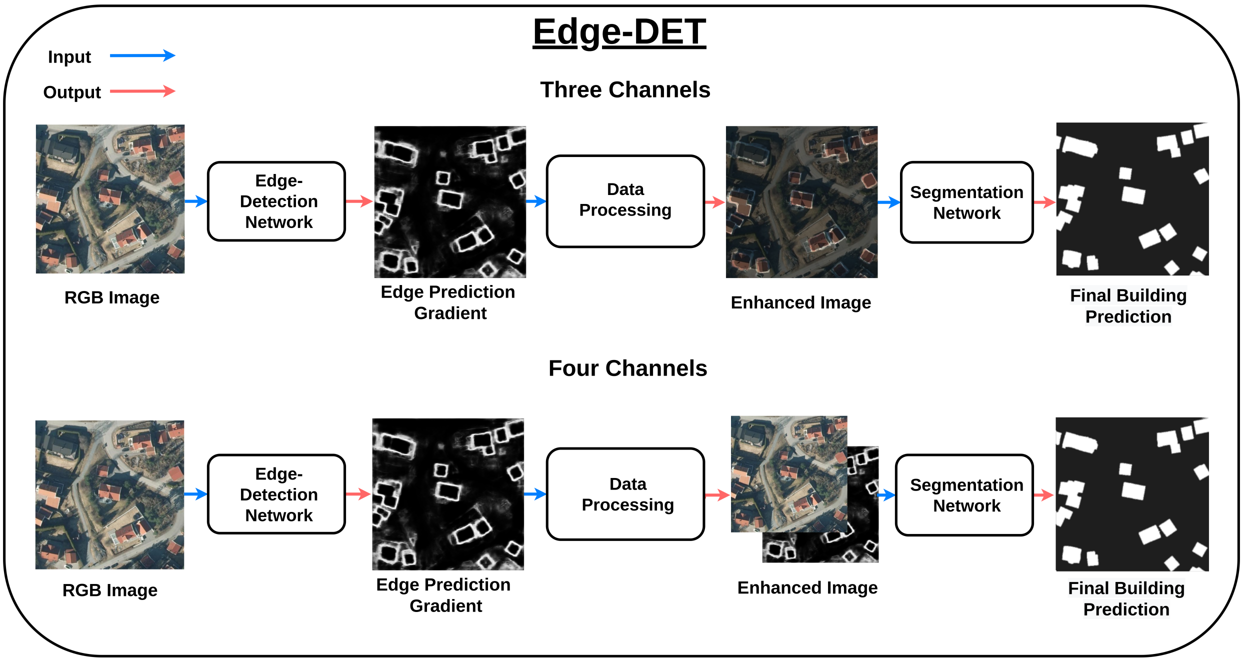


Figure 4.6: Edge-DET Architecture: We enhance the data using the building predictions from the edge-detection network and the original input images through a data enhancement technique further explained in Section 4.4.2. The segmentation network then receives the enhanced data as input, either as three or four channels.

4.3 Models and Hyperparameters

The hyperparameters we use to train the neural networks are essential for reproducibility, as they can heavily affect the model's performance. In table 4.1 we can see a list of the general parameters that we used for all models and experiments in this thesis. We have not focused our time on finetuning hyperparameters for this thesis as they are not essential for testing our hypothesis.

Table 4.1: General training hyperparameters used for all models.

Optimizer	**Learning Rate**	**Image Dim**	**Batch Size**	**Epochs**
Adam	1e-4	512x512	8	20

4.3.1 Models

For the thesis we have used three different segmentation models, namely Hierarchical Multi-Scale Attention (HMSA) [85], DeeplabV3+ [12], and U-Net [73]. Our models employ various techniques and use modified versions of the same architecture. In addition, they are all released years apart, which we assume will impact the BIoU scores achieved per model, where the oldest has the lowest score. The differences between the models described are why we use them, as it creates a diverse portfolio of models that creates a solid foundation for the testing of DET. The diverse portfolio allows us to create good evaluations of our proposed technique.

The implementation of the U-Net model and DeeplabV3+ model comes from the *Segmentation Models Pytorch* [94] package installed through pip, while the implementation of HMSA comes from the official repository for the paper. For DeeplabV3+ and U-Net, we have replaced their default encoder with the efficientnet-b0, the smallest version of efficientnet, while HMSA uses the bigger and more powerful HRNetV2 encoder. For Edge-DET we use CATS (BDCN) [44] as the edge-detection model. We use CATS (BDCN) because it is one of the best edge-detection models in the literature, as evident by the high ranking on `https://paperswithcode.com`. The CATS (BDCN) implementation comes from their official Github repository. We direct the readers to the respective papers for intricate details about each model. On a final note, each model we use in Seg-DET and Edge-DET is trained from scratch. This is due to the lack of pretrained models for data with four channels, and will allow us to make a fair comparison between the three and four channel versions.

4.4 Data Enhancement Technique

DET has two different edge improvement approaches. The first approach is Seg-DET, which employs a segmentation network to enrich and highlight important areas in the original data with the learned representations of buildings. The second segmentation model then leverages the added information to focus the training and weight optimization on the highlighted areas, and hopefully increasing BIoU scores. The second approach is Edge-DET, which employs a class-agnostic edge-detection network to enrich and highlight the edges of the buildings. The segmentation model uses the predicted edges to narrow the search space and help focus the weights on improving the predicted edges. For both approaches, the goal is that the second segmentation network will learn and use the information that is incorporated into the new and enriched training data to sharpen the edges. In the following two subsections, we elaborate on the details of both techniques.

4.4.1 Segmentation Enhanced Technique (Seg-DET)

In Seg-DET the first segmentation network receives an input image and predicts the corresponding segmentation mask, as we can see in Figure 4.7.

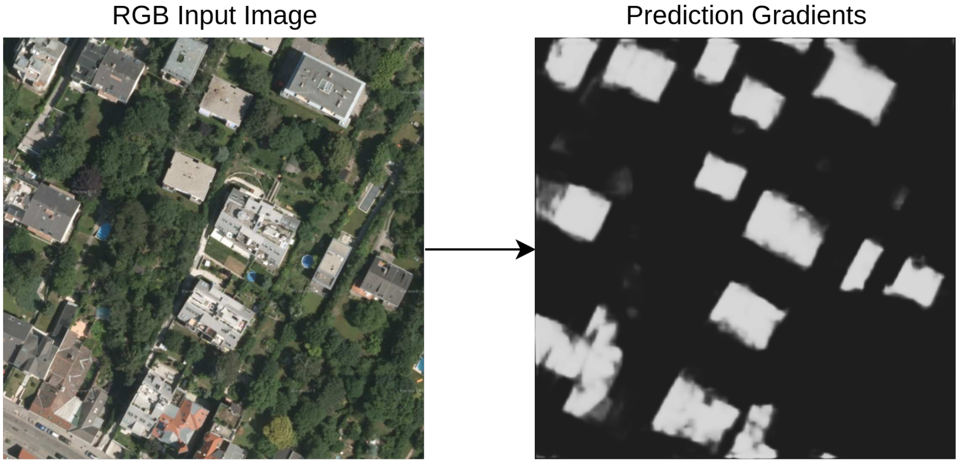


Figure 4.7: First network predicting the building gradients.

Seg-DET can then use the predicted gradients, the percentage-wise confidence that a pixel belongs to a building, in two different ways. Either add it to the input image as a fourth channel or combine it with the input image, keeping the original three channels. Adding it as a fourth channel is done by concatenating the prediction gradients to the input image. Combining the prediction gradients is a more complex process. One possible way is to multiply the prediction gradients with the input image, highlighting the predicted building pixels. However, as we can see in Figure 4.8, parts of the buildings are black due to an inaccurate prediction from the first segmentation network. A poor prediction can affect the second network if elements of the buildings are missing. These errors will not be visible to the second segmentation network and, as a result, will not have the opportunity to correct the mistakes.

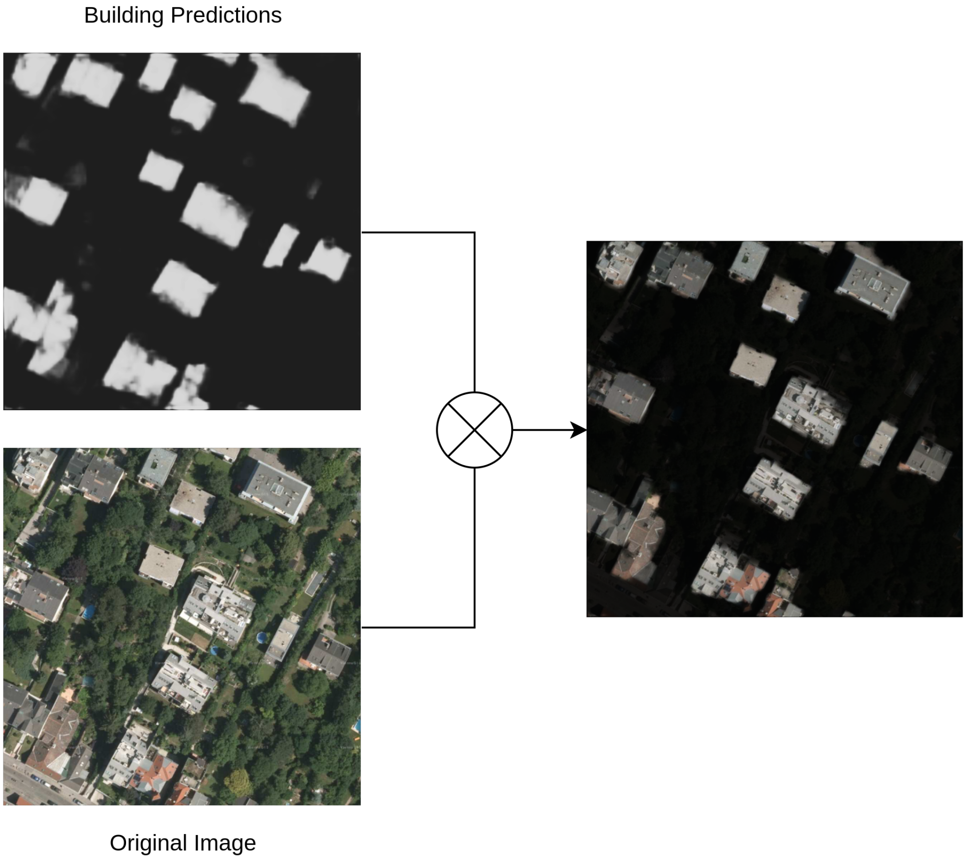


Figure 4.8: Building gradients multiplied with the original input image where the background and parts of buildings are not visible.

To mitigate the problem above, we use two different methods. Firstly, we clip all prediction gradients larger than 0.5 and set them to 1. Secondly, we dilate the remaining masks by 15 pixels in each direction, as seen in Figure 4.9. These improvements will regain possible lost information from the first network, giving the second network the possibility to correct the previous errors.

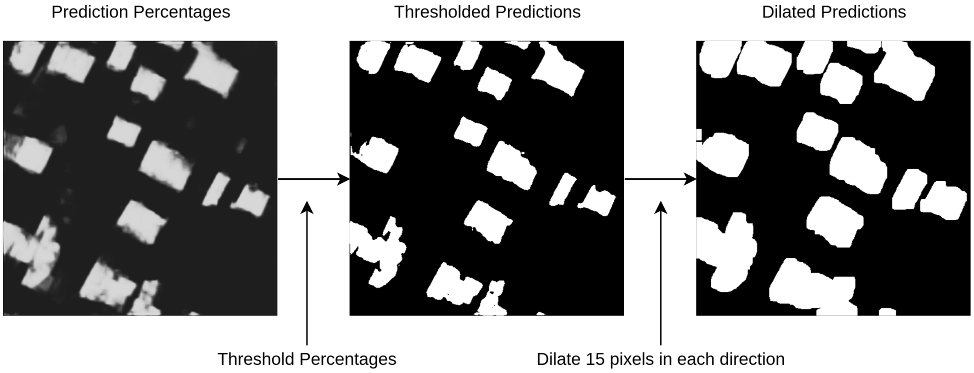


Figure 4.9: Segmentation masks thresholded at 0.5 and dilated by 15 pixels in all directions.

Furthermore, we want to give the original prediction more weight than the dilation. Therefore, we add 0.8 to all gradients within the dilated area, ensuring all gradients have a value equal to or larger than 0.8. All gradients with a value larger than 0.2 will exceed 1.0 after the addition, which would alter the color space of the input image. To prevent this problem, we clip the gradients to a maximum value of 1. The result, visualized in Figure 4.10, is a gradient matrix where the original gradients have a value of 1.0, and the gradients within the dilated area have a value between 0.8 and 1.0, depending on the original value. It should be noted that the original prediction gradients have a value of 1.0 to ensure that the second segmentation network weighs them accordingly.

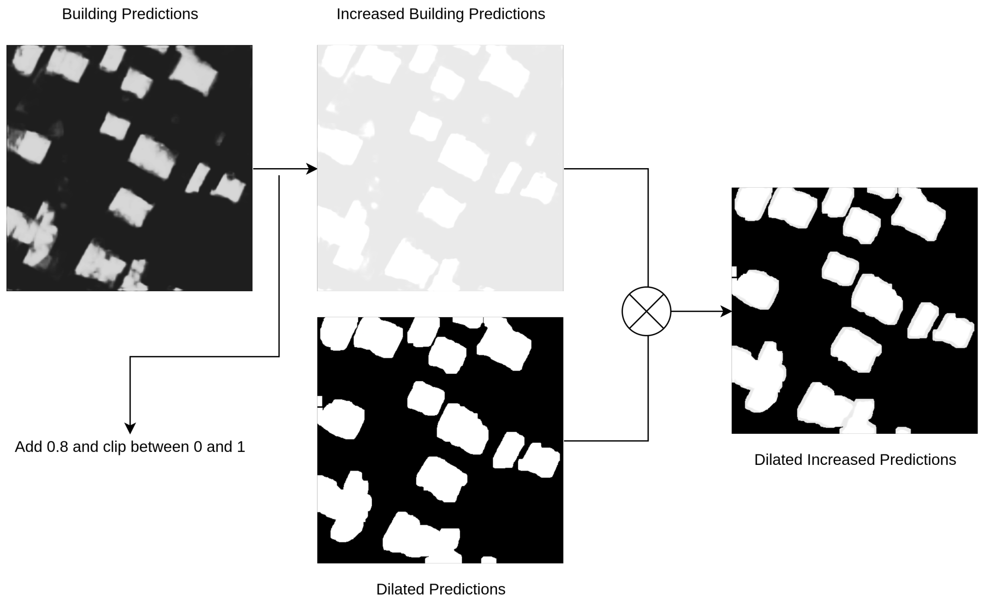

Building Predictions
Increased Building Predictions
Add 0.8 and clip between 0 and 1
Dilated Increased Predictions
Dilated Predictions

Figure 4.10: Multiplying the increased prediction gradients with the dilated prediction gradients.

As stated earlier, a poor prediction can affect the second network if elements of the buildings are missing. However, it is also possible that an entire building is missing from the prediction. Consequently, multiplying the prediction gradients with the original image will turn the missing buildings black, removing the opportunity for the second neural network to correct the mistake. Therefore, we clip all gradient values between 0.5 and 1.0 before multiplying them with the original image to prevent buildings from disappearing. The result, visualized in Figure 4.11, is an image where each pixel outside the dilation has a brightness of 50%, while pixels within the dilation have a brightness between 80% and 100%.

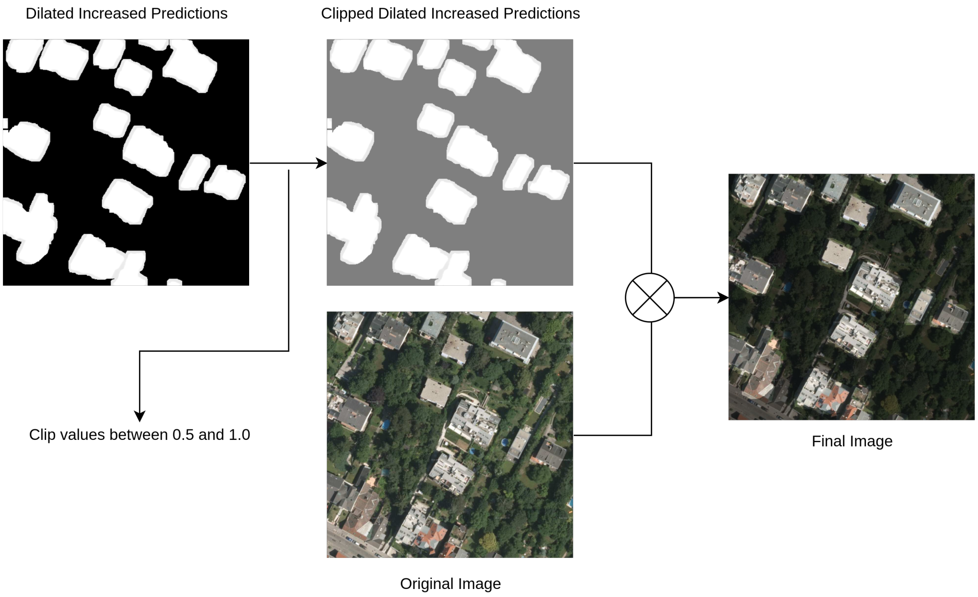

Dilated Increased Predictions
Clipped Dilated Increased Predictions
Clip values between 0.5 and 1.0
Final Image
Original Image

Figure 4.11: Clip mask between 0.5 and 1.0 and multiply with the original RGB input.

As we can see in the final image, the buildings and the surrounding area have brighter colors than the rest. We include parts of the buildings that the first segmentation model missed through the dilation. In addition, by clipping the gradients between 0.5 and 1.0, we allow the second segmentation model to see the background and possibly detect previously missed buildings. Pseudo code implementations of Seg-DET with three and four channels are found in Listing 1 and 2, respectively.

```
prediction = softmax(model(image), dim=0)
prediction_mask = argmax(gradients, dim=0)

# C, H, W
gradient = prediction[1, :, :]

dilated_mask = dilate(prediction_mask, ones((15, 15)))
dilation_gradient = clip(gradient + 0.8, 0.0, 1.0)
gradient_mask = clip(dilated_mask * dilation_grad, 0.5, 1.0)

enhanced_image = gradient_mask * image
```

Listing 1: Pseudo code for creating the enhanced images with three channels using Seg-DET.

```
prediction = softmax(model(image), dim=0)
prediction_mask = argmax(gradients, dim=0)

# C, H, W
gradient = prediction[1, :, :]

enhanced_image = concatenate((image, gradient), axis=-1)
```

Listing 2: Pseudo code for creating the enhanced images with four channels using Seg-DET.

4.4.2 Edge Enhanced Technique (Edge-DET)

In Edge-DET the edge-detection network receives an input image and predicts the corresponding building edges, as seen in Figure 4.12.

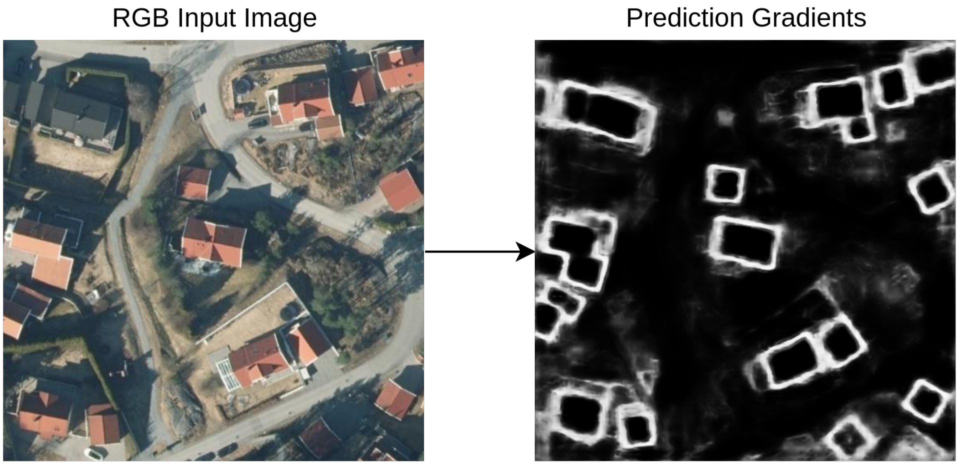


Figure 4.12: Edge-detection network predicting building edges.

Edge-DET can then use the predicted gradients, the percentage-wise confidence that a pixel belongs to the edge of a building, in two different ways. Either add it to the input image as a fourth channel, or combine it with the input image, keeping the original three channels. Adding it as a fourth channel is done by concatenating the prediction gradients to the input image. Combining the prediction gradients is a more complex process. One possible way is to multiply the prediction gradients with the input image, highlighting the predicted building edge pixels. However, as we can see in Figure 4.13, we can only see the building edges, which removes a lot of valuable information.

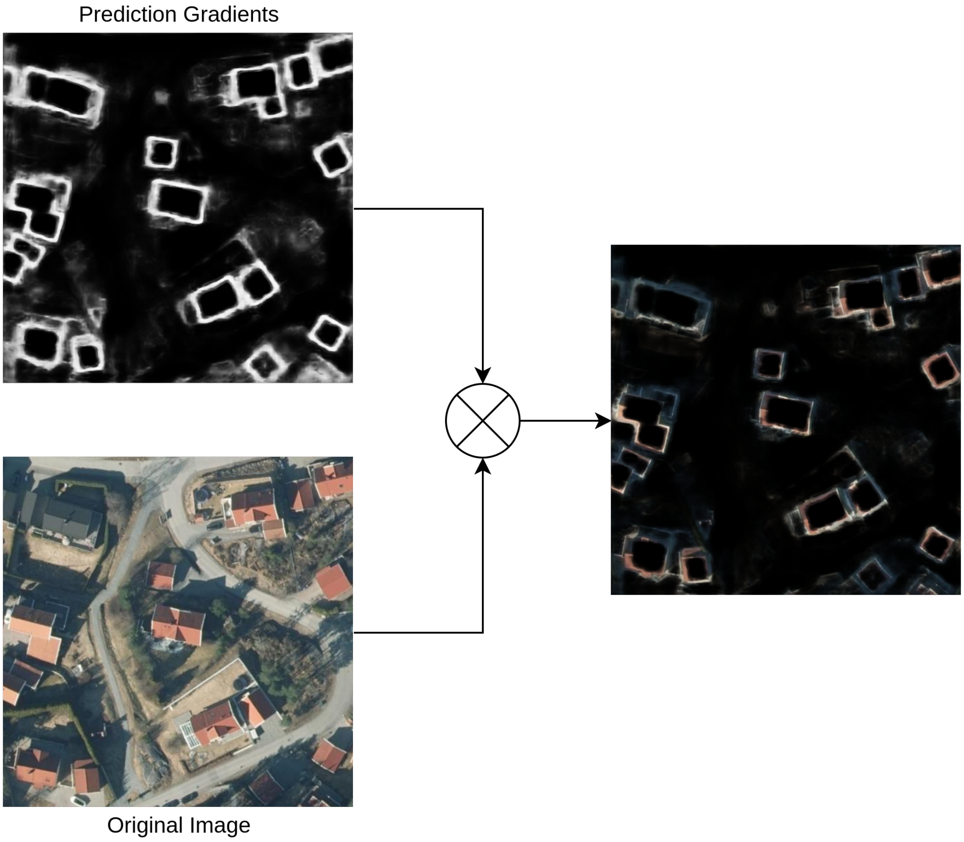


Figure 4.13: Edge gradients multiplied with the original image.

To mitigate the problem we clip the predicted gradients between 0.5 and 1.0, as seen in Figure 4.14. Clipping the gradients brings back the lost information, enabling the second segmentation network to locate possible missing buildings.

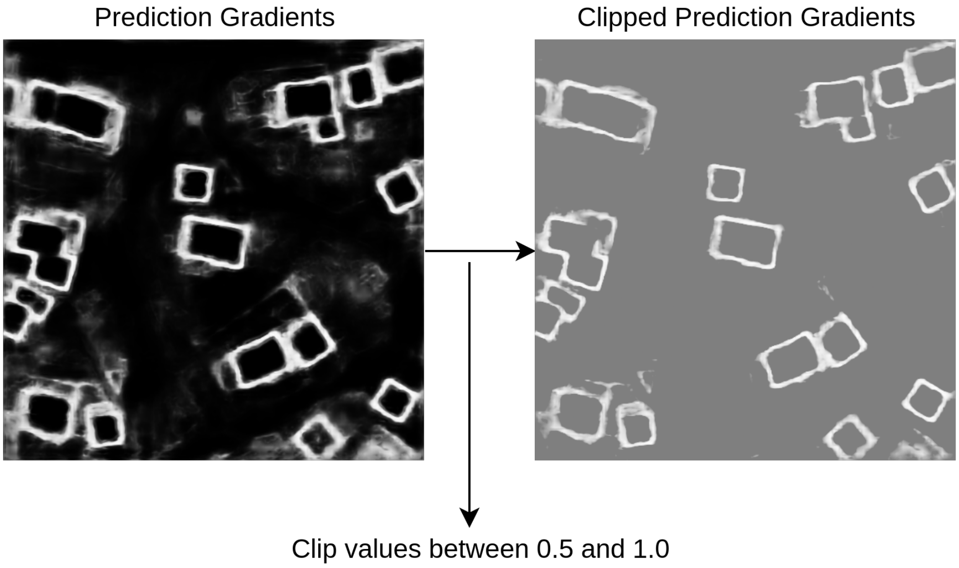


Figure 4.14: Clipped gradients between 0.5 and 1.0.

Multiplying the clipped prediction gradients with the original image provides a background with 50% brightness, while the predicted edges have a brightness of approximate 100%.

The predicted edges focuses the attention of the network on the building edges and passes on information about the predicted shape of the buildings. Figure 4.15 visualizes the final image, embedding task-specific information by highlighting building edges.

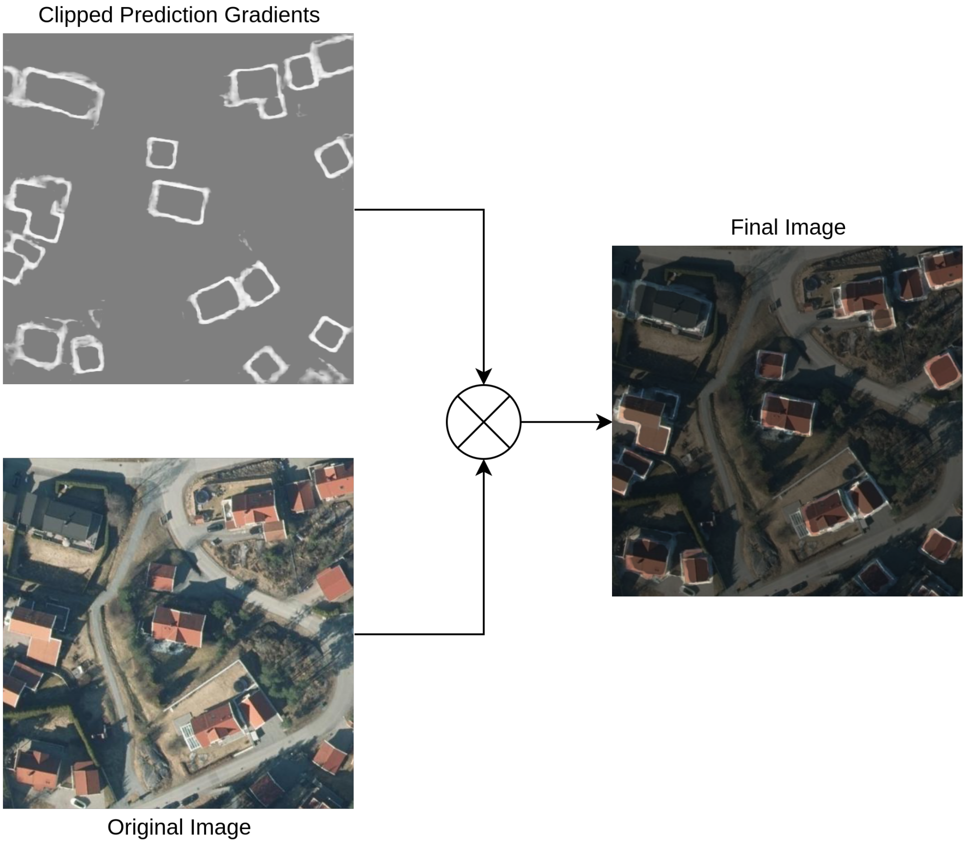


Figure 4.15: Clipped edge gradients multiplied with the original image.

Pseudo code implementations of Edge-DET with three and four channels are found in Listing 3 and 4, respectively.

```
predictions = sigmoid(model(image))
predictions = clip(predictions, 0.5, 1.0)
predictions = transpose(1, 2, 0)

enhanced_image = predictions * image
```

Listing 3: Pseudo code for creating the enhanced images with three channels using Edge-DET.

```
predictions = sigmoid(model(image))
predictions = transpose(1, 2, 0)

enhanced_image = concatenate((image, prediction), axis=-1)
```

Listing 4: Pseudo code for creating the enhanced images with four channels using Edge-DET.

4.5 Data Analysis

To test our hypotheses stated in Section 1.2.2, we need to be able to quantify the performance of our segmentation models. Therefore, we utilize two metrics to analyze the data output: Intersection over Union (IoU) and Boundary Intersection over Union (BIoU). Since evaluation metrics can be sensitive or insensitive to different errors, we hope to better evaluate the performance by incorporating several measures.

IoU is a metric used to evaluate how well the prediction and ground truth masks overlap. However, it measures all pixel values equally, making it less sensitive to boundary quality in larger objects [17]. To better evaluate the edge sharpness of our segmentation masks, we additionally utilize the BIoU metric. It calculates the IoU for pixels within a certain distance from the edges of the original masks, making it more susceptible to edge errors and better at assessing improvements in boundary quality [17]. Both of these metrics are further detailed in Section 2.3.2.

4.6 Loss Functions

Using the correct loss functions is vital for a neural network to optimize toward the preferred objective. Since this thesis concerns improvements in segmentation edges, it is important to use a loss function that evaluates the quality of the boundaries. However, it is also important to correctly classify the pixels in the image. This condition encourages the use of multiple loss functions focusing on different but important parts of the segmentation task. In this paper, we propose a compounding loss consisting of RMI loss [98], Active Boundary loss [87], and Lovasz-Softmax [6]. The compounding loss is similar to [87], but we remove the Cross-Entropy loss [35] and add the RMI loss, for improved BIoU scores. The change is based on the results in paper [98], where the authors show that RMI loss is superior to Cross-Entropy loss. Additionally, RMI loss utilizes the underlying information of adjacent pixels, which may further help the precision of the segmentation edges.

Chapter 5

Experiments and Results

This chapter details the results of the compounding loss Edge-DET, Seg-DET, and Edge-DET. We also investigate the effect of shifted labels on several different loss functions. Furthermore, to quantify the results, we use IoU and BIoU as stated in Section 4.5. The tables in experiments 2, 3, 4, and 5 use red or green values adjacent to the scores indicating negative or positive differences between the second segmentation network and its baseline, unless stated otherwise.

To conduct our experiments we use an NVIDIA Tesla V100 SXM3 32 GB to train and evaluate the models. We program in Python 3.9 and use Pytorch 1.11 as the machine learning framework. Furthermore, we use Github for version control.

5.1 Baseline

In this section, we will provide the baseline results for both datasets using each segmentation model in 4.3.1. Table 5.1 and 5.2 display the results with and without a pretrained encoder for U-Net, DeeplabV3+, and HMSA. We use the baseline results to compare Seg-DET and Edge-DET.

Table 5.1: Baseline results for Kartverket dataset.

Model	Loss	Acc.	IoU	BIoU	Best Epoch	Pretrained
U-Net	RMI	0.9547	0.7657	0.6279	20	No
U-Net	RMI	0.9613	0.8049	0.6841	11	Yes
DeeplabV3+	RMI	0.9580	0.7850	0.6586	16	No
DeeplabV3+	RMI	0.9653	0.8231	0.7157	20	Yes
HMSA	RMI	0.9667	0.8291	0.7284	9	No
HMSA	RMI	0.9699	0.8495	0.7596	14	Yes

Table 5.2: Baseline results for the Inria dataset.

Model	Loss	Acc.	IoU	BIoU	Best Epoch	Pretrained
U-Net	RMI	0.9505	0.5007	0.4453	9	No
U-Net	RMI	0.9548	0.5343	0.4715	1	Yes
DeeplabV3+	RMI	0.9525	0.7064	0.4486	20	No
DeeplabV3+	RMI	0.9635	0.7753	0.5181	20	Yes
HMSA	RMI	0.9603	0.7636	0.5078	20	No
HMSA	RMI	0.9700	0.8093	0.5605	20	Yes

From Table 5.1 and 5.2 we conclude that HMSA is the superior model, achieving the highest scores, followed by DeeplabV3+ and U-Net. It is important to note the performance of the U-Net model, where there is an apparent discrepancy between the two datasets. Comparing the scores indicates that U-Net struggles with the Inria dataset. Analyzing the IoU scores for all models on both datasets, we see that U-Net decreases approx. 20%, while DeeplabV3+ and HMSA decrease approx. 8% and 6%, respectively. The results indicate that the discrepancy in U-NETs evaluation scores for both datasets is due to the geolocation of the data collection. The Kartverket dataset contains data from a single location in Norway, while the Inria dataset includes data from five different locations worldwide. The contrasting architecture, vegetation, and building density in the Inria dataset might be the cause of the significant difference in the U-Net model.

5.2 Experiment 1: Comparing ABL(RMI) and ABL(CE)

Hypothesis 1 states that substituting CE with RMI in the compounding loss ABL(CE) will improve the BIoU. To test hypothesis 1, we evaluate both combinations using the pretrained DeeplabV3+ model. Table 5.3 presents the results, which show that substituting CE with RMI increases the performance of the model in terms of IoU and BIoU substantially.

Table 5.3: ABL(RMI) and ABL(CE) comparison.

Model	Loss	Acc.	IoU	BIoU
DeeplabV3+	ABL(CE)	0.9630	0.8110	0.6909
DeeplabV3+	ABL(RMI)	0.9647 (+0.0013)	0.8179 (+0.0069)	0.7060 (+0.0151)

Table 5.3 shows that RMI can successfully be used in a compounding loss with Lovasz and ABL to achieve better model performance compared to its predecessor, ABL(CE). The results show that the compounding loss is able to benefit from the improved performance reported in [98]. Thereby, confirming hypothesis 1.

5.3 Experiment 2: Baseline + Edge-DET

Hypothesis 2 states that using the compounding loss of RMI, Lovasz, and ABL will improve the BIoU of a segmentation network compared to using RMI. To test hypothesis 2, we evaluate U-Net, DeeplabV3+, and HMSA with RMI and ABL(RMI). Table 5.4 presents loss function abbreviations and their corresponding loss functions used in the following experiments.

Table 5.4: Loss function abbreviations and loss functions.

Loss Function Abbreviation	**Loss Function**
ABL(CE)	Cross-Entropy loss [35], Lovasz-Softmax loss [6], and Active Boundary Loss [87]
Edge-DET	Region Mutual Information loss [98], Lovasz-Softmax loss [6], and Active Boundary Loss [87]
RMI	Region Mutual Information Loss [98]
CE	Cross-Entropy Loss [35]
ABL	Active Boundary Loss [87]
Lovasz	Lovasz-Softmax Loss [6]

Table 5.5: Results for Edge-DET for U-Net, DeeplabV3+, and HMSA.

Model	**Loss**	**Acc.**	**IoU**	**BIoU**
U-Net	RMI(ABL)	0.9561 (+0.0014)	0.7792 (+0.0135)	0.6486 (+0.0207)
DeeplabV3+	RMI(ABL)	0.9577 (-0.0023)	0.7839 (-0.0011)	0.6565 (-0.0021)
HMSA	RMI(ABL)	0.9704 (-0.0001)	0.8470 (-0.0025)	0.7539 (-0.0057)

Table 5.5 show inconclusive results. U-Net demonstrates a profound increase in IoU and BIoU scores for Edge-DET, while DeeplabV3+ and HMSA decrease. Furthermore, all hyperparameters for this experiment are the same, suggesting that U-NETs deviation is inherent in its architecture, but this needs further research. The results indicate that RMI is better than the compounding loss Edge-DET. However, as explained in Section 3.3.3, ABL optimizes towards the edges of the ground truths. Meaning, faulty feedback is sent to the model when the ground truths are shifted, negatively impacting the compounding loss. Therefore, we contribute the reduced IoU and BIoU scores for DeeplabV3+ and HMSA with Edge-DET to the shifted ground truths, which we investigate in the next subsection.

5.3.1 Orthorectification Impact

To test the impact of the shifted ground truths, detailed in Section 4.1.1, we conduct an experiment where we compare different combinations of loss functions on an adjusted and unadjusted dataset. The adjusted dataset is manually corrected, resulting in accurate building masks. The following results use the DeeplabV3+ ABL(CE) as the baseline comparison in the adjacent red and green values.

Table 5.6: Results for different loss function combinations using the adjusted dataset.

Model	Loss Comb.	Building IoU	Building BIoU
DeeplabV3+	ABL(CE)	0.7102	0.6144
DeeplabV3+	Edge-DET	0.7204 (+0.0102)	0.6284 (+0.0140)
DeeplabV3+	ABL, RMI	0.7117 (+0.0015)	0.6226 (+0.0082)
DeeplabV3+	ABL, Lovasz	0.7079 (-0.0023)	0.6101 (-0.0043)
DeeplabV3+	Lovasz, RMI	0.7215 (+0.0113)	0.6288 (+0.0144)
DeeplabV3+	**RMI**	**0.7243 (+0.0141)**	**0.6359 (+0.0215)**
DeeplabV3+	Lovasz	0.7161 (+0.0059)	0.6212 (+0.0068)
DeeplabV3+	CE	0.7044 (-0.0058)	0.6080 (-0.0064)

Table 5.7: Results for different loss function combinations using the unadjusted dataset.

Model	Loss Comb.	Building IoU	Building BIoU
DeeplabV3+	ABL(CE)	0.6616	0.5477
DeeplabV3+	Edge-DET	0.6684 (+0.0068)	0.5613 (+0.0136)
DeeplabV3+	ABL, RMI	0.6612 (-0.0004)	0.5543 (+0.0066)
DeeplabV3+	ABL, Lovasz	0.6487 (-0.0129)	0.5401 (-0.0076)
DeeplabV3+	Lovasz, RMI	0.6760 (+0.0144)	0.5633 (+0.0156)
DeeplabV3+	**RMI**	**0.6838 (+0.0222)**	**0.5806 (+0.0329)**
DeeplabV3+	Lovasz	0.6568 (-0.0048)	0.5478 (+0.0001)
DeeplabV3+	CE	0.6676 (+0.0060)	0.5578 (+0.0101)

Comparing the results in Table 5.6 and 5.7 confirms the impact of the shifted ground truths. The results for the adjusted dataset have the lowest BIoU score when using CE. In contrast, ABL(CE) has the lowest BIoU score for the unadjusted dataset. The result demonstrate the impact of shifting ground truths and provide evidence for our claim of the negative effect. Additionally, RMI considers all adjacent pixels during evalution, making it more resilint towards label shift, which is evident in the results.

Table 5.6 shows that when ABL is part of a compounding loss, the IoU and BIoU decrease compared to the identical loss without ABL. These results indicate that ABL introduces noise regardless of label shift, but we cannot conclude that RMI exceeds RMI(ABL) since U-NET manages to improve in Table 5.5. Therefore, making it hard to either confirm or refute hypotheses 2. This predicament is further discussed in Chapter 6.

5.4 Experiment 3: Empirical Analysis of DET

To show the viability of DET, we carry out an experiment where ground truths replace input predictions. Using the ground truths enables us to simulate a situation where the first network locates all buildings, allowing the second network to focus on refining the segmentation edges. Figure 5.1 shows a sample input from the Seg-DET three-channel dataset.

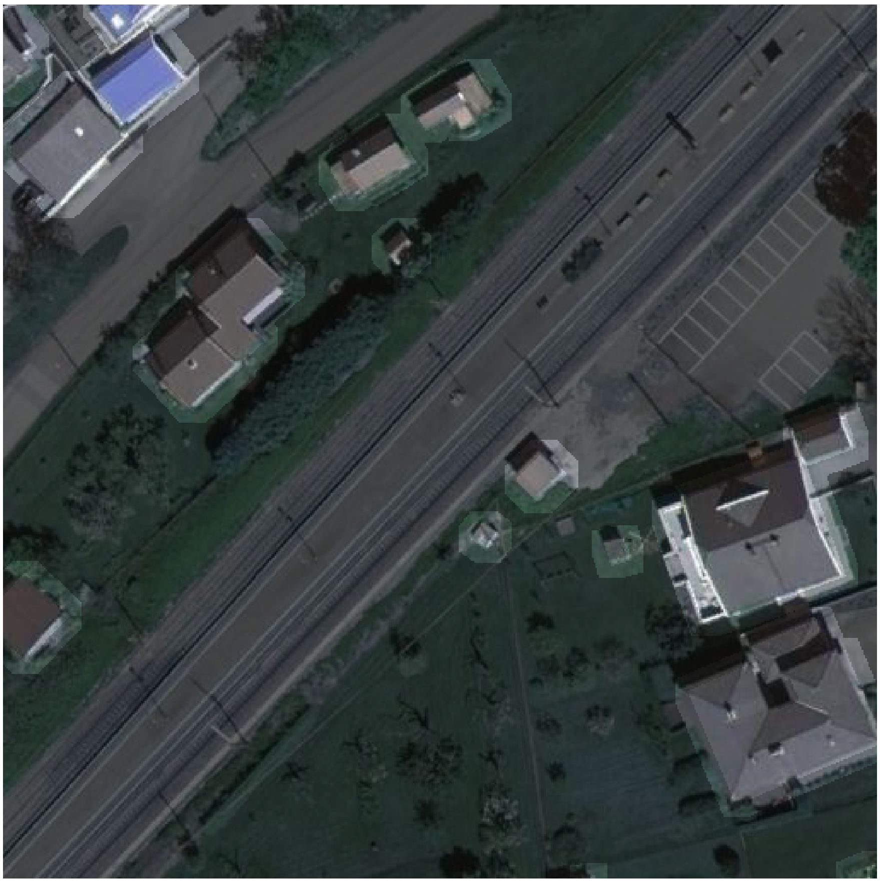


Figure 5.1: Sample input from the 3-channel Seg-DET dataset.

Table 5.8: Seg-DET generated dataset with perfect predictions from the first segmentation network.

Model	**Channels**	**Acc.**	**IoU**	**BIoU**
U-Net	3	0.9822 (+0.0275)	0.8949 (+0.1292)	0.7815 (+0.1536)
U-Net	4	0.9845 (+0.0298)	0.9063 (+0.1406)	0.7041 (+0.0762)
DeeplabV3+	3	0.9946 (+0.0366)	0.9674 (+0.1824)	0.9359 (+0.2773)
DeeplabV3+	4	0.9991 (+0.0411)	0.9945 (+0.2095)	0.9900 (+0.3314)
HMSA	3	0.9984 (+0.0317)	0.9908 (+0.1617)	0.9801 (+0.2517)
HMSA	4	0.9996 (+0.0329)	0.9976 (+0.1685)	0.9937 (+0.2653)

Table 5.9: Edge-DET generated dataset with perfect predictions from first edge-detection network.

Model	**Channels**	**Acc.**	**IoU**	**BIoU**
U-Net	3	0.9802 (+0.0255)	0.8925 (+0.1268)	0.6940 (+0.0661)
U-Net	4	0.9822 (+0.0275)	0.8939 (+0.1282)	0.7492 (+0.1213)
DeeplabV3+	3	0.9987 (+0.0407)	0.9919 (+0.2069)	0.9850 (+0.3264)
DeeplabV3+	4	0.9991 (+0.0411)	0.9945 (+0.2095)	0.9900 (+0.3314)
HMSA	3	0.9995 (+0.0328)	0.9972 (+0.1681)	0.9945 (+0.2661)
HMSA	4	0.9996 (+0.0329)	0.9976 (+0.1685)	0.9950 (+0.2666)

Table 5.8 and 5.9 displays major improvements for all models on all metrics. DeeplabV3+

and HMSA both acquire almost perfect scores, while U-Net reaches the limitation of the model around 90% IoU and between 69-75% BIoU. The results provide a strong empirical argument for the potential of DET. It is important to remember that the model receives perfect predictions in the fourth channel when using 4-channel datasets. This predicament allows the models to achieve perfect scores by disregarding the first three channels. Therefore, when using 4-channel datasets the empirical evidence might not be reliable. Although, in our experiments, they do not predict perfect segmentation masks. The results suggest that the encoder-decoder structure of the models alters the data in the fourth channel, affecting the final prediction.

5.5 Experiment 4: Combining and Concatenating Segmentation Predictions (Seg-DET)

Hypotheses 3 and 4 state that combining (3 channels) or concatenating (4 channels) the prediction gradients from a segmentation network with the original images as input to a secondary segmentation network will improve the BIoU compared to using the original images as input. To test hypotheses 3 and 4, we train and evaluate each model on a Seg-DET generated dataset. In this experiment, Seg-DET employs the baseline DeeplabV3+ model to produce enhanced data.

Table 5.10: DeeplabV3+ Seg-DET generated dataset on Kartverket dataset.

Model	**Channels**	**Acc.**	**IoU**	**BIoU**
U-Net	3	0.9544 (-0.0003)	**0.7685 (+0.0028)**	**0.6343 (+0.0064)**
U-Net	4	0.9554 (+0.0007)	**0.7730 (+0.0073)**	**0.6416 (+0.0137)**
DeeplabV3+	3	0.9567 (-0.0013)	0.7802 (-0.0048)	0.6535 (-0.0053)
DeeplabV3+	4	0.9562 (-0.0018)	0.7778 (-0.0072)	0.6492 (-0.0094)
HMSA	3	0.9621 (-0.0046)	0.8103 (-0.0188)	0.7003 (-0.0281)
HMSA	4	0.9605 (-0.0062)	0.7997 (-0.0294)	0.6860 (-0.0424)

Table 5.11: DeeplabV3+ Seg-DET generated dataset on Inria dataset.

Model	**Channels**	**Acc.**	**IoU**	**BIoU**
U-Net	3	0.9419 (-0.0086)	0.4628 (-0.0379)	0.4056 (-0.0397)
U-Net	4	0.9475 (-0.0030)	0.4903 (-0.0104)	0.4301 (-0.0152)
DeeplabV3+	3	0.9528 (+0.0003)	**0.7151 (+0.0087)**	**0.4605 (+0.0119)**
DeeplabV3+	4	0.9510 (-0.0015)	0.7071 (+0.0007)	0.4452 (-0.0034)
HMSA	3	0.9599 (-0.0004)	0.7473 (-0.0163)	0.4970 (-0.0108)
HMSA	4	0.9587 (-0.0016)	0.7574 (-0.0062)	0.4942 (-0.0136)

Table 5.10 and 5.11 presents the results obtained in experiment 4. The outcome reveals subpar evaluation scores, only showing improvements in two models. It is hard to specify why the practical results deviate from the empirical analysis. However, the results suggest that the performance relies on the difference between the baseline evaluation scores for the first and second segmentation network. Since DeeplabV3+ generates the enhanced dataset, U-NET improves, while both DeeplabV3+ and HMSA decrease on the Kartverket dataset.

On the other hand, U-NET and HMSA decrease on the Inria dataset, while DeeplabV3+ increases. The discrepancy between the results suggests U-NET struggles with the Inria dataset, strengthening the statement in Section 5.1. Furthermore, the improvement of DeeplabV3+ indicates that this combination of Seg-DET is a better fit for this specific dataset.

The outcome of this experiment show limited improvements, suggesting that the flaws of the first segmentation network negatively impacts Seg-DET. Prediction errors, such as false negatives and false positives, will affect the generated dataset, consecutively influencing the secondary segmentation network. Therefore, DeeplabV3+ might not be a good universal fit for the first segmentation network. However, depending on the second segmentation network and dataset, the results show that DeeplabV3+ can be applicable as the first segmentation network.

In order to find complementing models to have a successful Seg-DET approach, it is essential to experiment with the first and second segmentation model. A scenario where the first network is proficient at locating buildings and the second network is capable of achieving accurate segmentation masks would be ideal. Based on this experiment, the evidence show mixed results, suggesting it is difficult to find correct model combinations. Therefore, making it hard to either confirm or refute hypotheses 3 and 4. This predicament is further discussed in Chapter 6.

5.6 Experiment 5: Combining and Concatenating Edge-Detection Predictions (Edge-DET)

Hypotheses 5 and 6 state that combining (3 channels) or concatenating (4 channels) the prediction gradients from an edge-detection network with the original RGB images as input to a secondary segmentation network will improve the BIoU compared to using the original images as input. To test hypotheses 5 and 6, we train and evaluate each model on an Edge-DET generated dataset. In this experiment, Edge-DET employs the CATS model to produce enhanced data.

Table 5.12: CATS Edge-DET generated dataset on Kartverket dataset.

Model	**Channels**	**Acc.**	**IoU**	**BIoU**
U-Net	3	0.9552 (+0.0005)	0.7742 (+0.0085)	0.6445 (+0.0166)
U-Net	4	0.9553 (+0.0006)	0.7739 (+0.0082)	0.6480 (+0.0201)
DeeplabV3+	3	0.9580 (+0.0000)	0.7882 (+0.0032)	0.6658 (+0.0072)
DeeplabV3+	4	0.9581 (+0.0001)	0.7890 (+0.0040)	0.6691 (+0.0105)
HMSA	3	0.9597 (-0.0070)	0.8076 (-0.0215)	0.6968 (-0.0316)
HMSA	4	0.9613 (-0.0054)	0.8057 (-0.0234)	0.6975 (-0.0309)

Table 5.13: CATS Edge-DET generated dataset on Inria dataset.

Model	Channels	Acc.	IoU	BIoU
U-Net	3	0.9463 (-0.0042)	0.4787 (-0.0220)	0.4212 (-0.0241)
U-Net	4	0.9484 (-0.0021)	0.4958 (-0.0049)	0.4397 (-0.0056)
DeeplabV3+	3	0.9510 (-0.0015)	0.7071 (+0.0007)	0.4452 (-0.0034)
DeeplabV3+	4	0.9521 (-0.0004)	0.7060 (-0.0004)	0.4495 (+0.0009)
HMSA	3	0.9587 (-0.0016)	0.7574 (-0.0062)	0.4942 (-0.0136)
HMSA	4	0.9562 (-0.0041)	0.7323 (-0.0313)	0.4785 (-0.0293)

Table 5.10 and 5.11 presents the results obtained in experiment 5. The outcome reveals promising evaluation scores on the Kartverket dataset, where Edge-DET is able to improve upon the baseline scores for both U-Net and DeeplabV3+. The results reflects the performance of CATS, making the building delineations adequate for U-Net and DeeplabV3+. However, for HMSA the results suggest that it instead introduces noise, negatively affecting the performance.

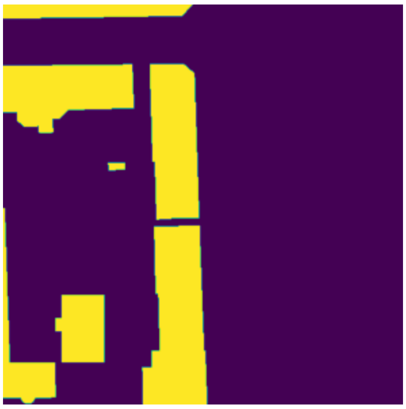


Figure 5.2: Ground truth mask.

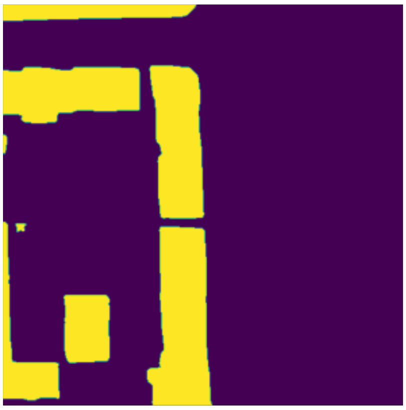


Figure 5.3: HMSA segmentation prediction.

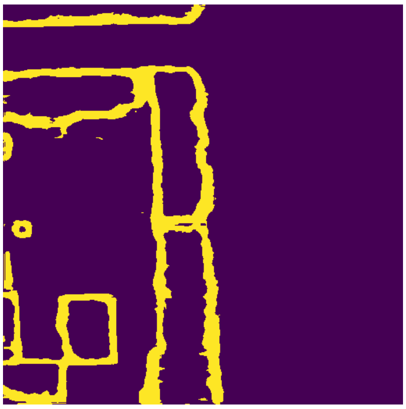


Figure 5.4: CATS edge prediction.

Figure 5.2, 5.3, and 5.4 illustrates the superior boundary quality of HMSA. The visual example shows that HMSA will not benefit from the edge prediction, suggesting that the generated dataset brings noise instead of guidance. The evaluation scores of the Inria dataset do not portray the same progress, showing no significant improvements. A possible explanation is the flaws of the first neural network, which becomes inherent in the generated data. Therefore, the performance of CATS is crucial for the success of Edge-DET.

Table 5.14: CATS IoU scores on the Kartverket and Inria dataset.

Model	Dataset	IoU
CATS	Kartverket	0.7519
CATS	Inria	0.5355

Table 5.14 advocate that the lack of improvement on the Inria dataset is due to CATS' poor IoU performance. The practical results of Edge-DET show considerable potential but are restricted by the outdated VGG16 model, which is a core part of CATS. Since edge-detection is a deprived research area, there are significant opportunities for improvements. The evidence suggests that further development of edge-detection models will benefit Edge-DET, possibly achieving higher quality boundaries. Similar to experiment 4, it is hard to

either confirm or refute hypotheses 5 and 6 since the results vary depending on the datasets and model combinations. This predicament is further discussed in Chapter 6.

5.7 Summary

In our first experiment, we show the superiority of Edge-DET compared to ABL(CE). However, Edge-DET was not able to improve upon the standalone RMI loss. Our assessment of the orthorectification impact reveals that the shift in ground truths harm the performance of ABL.

Furthermore, the results for Seg-DET and Edge-DET did not follow the trend of the empirical analysis, as the minority of the models managed to improve. The practical evidence also indicate that the subpar results originate from the first neural network, as the predicted errors become inherent in the generated dataset.

The majority of experiments show contradicting results, making it difficult to confirm or refute the hypotheses. Nevertheless, we show DET's potential through the empirical analysis, demonstrating significant improvements in a controlled environment. The next chapter discuss the unproven hypotheses in depth.

Chapter 6

Discussions

While most of the practical results did not improve, we cannot conclude that the approach is a failure. The reason is backed by the evidence showing increased performance for particular model combinations, further indicating that the search of an effective combination is crucial for the success of DET. Thus, additional testing of models and different domains is necessary to derive a definite conclusion.

The new compounding loss ABL(RMI) was not able to improve upon standalone ABL(RMI) loss for the majority of models. The results show inconclusive evaluation scores, as U-NET has a significant increase in IoU and BIoU, while HMSA and DeeplabV3+ decrease. A possible explanation for the deviation is the noise shown in the orthorectification impact experiment. Based on the results in Table 5.5, ABL introduces noise instead of improvements when the models reach a certain level of BIoU score. This indication suggests that ABL struggles when the predicted segmentation masks reach a certain proximity to the ground truth, consecutively affecting ABL(RMI).

The results achieved in the empirical analysis of Seg-DET and Edge-DET show great potential. However, the practical evidence does not show an overall increase in evaluation scores. Because the empirical analysis uses perfect predictions as input, the datasets have no false positives (FP) or false negatives (FN). Therefore, reviewing the practical results and empirical analysis suggest that a crucial factor for the success of DET is the ability to reduce FN and FP in the first network. In hindsight, we should have conducted additional experiments changing out the first network, which could have revealed a pattern. A solution would be to create an evaluation metric to assess the performance of building discoveries. Assigning a number to every case of false and true positives, and false and true negatives would ease the process of evaluating and choosing the first segmentation network. Additionally, we should have negatively altered the boundary quality of the perfect masks to strengthen the evidence of the empirical analysis.

Furthermore, an inherent trait of DET is the ability to change the first and second segmentation models easily. The architecture allows for relatively rapid testing of multiple networks in various combinations. The results indicate Seg-DET is more reliant on a good network combination, as evident in the Seg-DET experiment where DeeplabV3+ combined with DeeplabV3+ on the Inria dataset improves significantly. In contrast, the results from the Edge-DET experiment indicate that it is more reliant on the performance of the edge-detection network, and not the specific combination. Since CATS predicts better edges than U-NET and DeeplabV3+, we see an increase in their performance on the Kartverket dataset. Therefore, the edge-detection network needs to be better at delineating buildings to improve the boundary quality. Currently, CATS is one of the best class-agnostic edge-detection models, though the results show it is not good enough to improve upon the HMSA. Thus, a solution to improve Edge-DET is to further advance edge-detection as a field.

DET can generate 3- and 4-channel datasets, which have different contrasting qualities. 3-channel DET enables pre-trained models, possibly improving the generated dataset. However, an inherent weakness is the lack of multiclass segmentation support. Therefore, it is only possible to use the current approach with binary segmentation tasks. Although 4-channel DET supports multiclass segmentation, it excludes the use of pre-trained models. Thus, DET cannot use pre-trained models on multiclass segmentation tasks. A solution would be to tint the highlighted areas in 3-channel DET with class-dependent colors, resolving the issue above. On a final note, we cannot conclude whether 3- or 4-channel is best because the results indicate that it is highly dependent on the model and dataset.

These arguments lay the basis for further research and investigation to determine why the majority of the practical results deviate from the empirical analysis, which will help to derive a definite conclusion to our unresolved hypotheses.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis investigates state-of-the-art research for the segmentation of aerial images. Additionally, exploring the feasibility of combining two neural networks to improve the predicted edges of segmentation masks for buildings. In our work, we introduce the Data Enhancement Technique (DET), which uses either a segmentation (Seg-DET) or edge-Detection network (Edge-DET) to enhance the original data with inherent information about their predictions. A secondary segmentation network then uses the enhanced data to increase the accuracy of the predicted edges. In addition, we propose a new compounding loss (ABL(RMI)) for edge improvement.

The proposed compounding loss successfully improved upon its predecessor but could not exceed standalone RMI loss. The results suggest that ABL struggles when the predicted segmentation masks reach a certain proximity to the ground truth, negatively impacting ABL(RMI). However, our results show that DET successfully improved upon the baseline scores, which demonstrates that for Seg-DET, it is crucial to find the right model combinations for a specific segmentation task. Additionally, Edge-DET relies on the further development of edge-detection models for improved performance.

7.2 Future Work

This thesis lays the foundation for further exploration of using a neural network to enhance the input to a secondary model. Our work motivates the use of two neural networks for enhanced performance. In addition to the experiments we conduct, other possible approaches might lead to improved edge accuracy. The following paragraphs introduce potential future work.

Object detection models focus on finding objects in an image, but instead of creating an object mask, they place them in a bounding box. The results suggest that utilizing object detection models as the first neural network in DET would be a logical next step. The reason is that the network's sole goal is to locate objects, and it is likely it would do a better job locating buildings than segmentation networks. Using DET with object detection would then highlight the predicted bounding boxes and reduce the brightness of the rest of the image. A possible issue with this approach is when the image portrays a dense building area. The bounding boxes are often larger than the actual object, which could highlight the entire image and thereby fail to provide information for the second segmentation model. Therefore, as we show in our experiments, the performance of the first neural network is crucial for the success of the technique.

Furthermore, as mentioned in the thesis, we believe that improving upon the edge detection models would increase the performance of Edge-DET. Thereby, we advocate that it would be beneficial to invest in research advancing edge detection as a field. The edge detection model used in this thesis relies on the VGG16 [80] network released in 2014. We assume it would be relatively simple to replace the VGG16 core of CATS and replace it with a new state-of-the-art encoder, such as HRNet [88]. In addition, other architectures such as the transformer could be interesting to combine with the edge detection techniques to enhance the results further.

Another technique that could be worth testing is the exploration of different color spaces. Papers such as [1, 28] demonstrate the significant impact color spaces have on segmentation. Therefore, we believe it could be beneficial to explore and combine the use of different color spaces with DET to improve the accuracy of the predicted edges.

Appendix A

Implementation

The code for our implementation of DET can be found at https://github.com/Sjyhne/SequeNet.

Bibliography

[1] Dena A. Abdelsadek et al. “Impact of Using Different Color Spaces on the Image Segmentation.” In: *The 8th International Conference on Advanced Machine Learning and Technologies and Applications (AMLTA2022)*. Ed. by Aboul Ella Hassanien et al. Cham: Springer International Publishing, 2022, pp. 456–471. ISBN: 978-3-031-03918-8.

[2] Per-Arne Andersen. “Deep Reinforcement Learning using Capsules in Advanced Game Environments.” In: (2018).

[3] Pablo Arbelaez et al. “Contour Detection and Hierarchical Image Segmentation.” In: *IEEE transactions on pattern analysis and machine intelligence* 33 (May 2011), pp. 898–916. DOI: 10.1109/TPAMI.2010.161.

[4] ArcGis. *Introduction to ortho mapping*. URL: https://pro.arcgis.com/en/pro-app/latest/help/data/imagery/introduction-to-ortho-mapping.htm.

[5] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.” In: *CoRR* abs/1511.00561 (2015). arXiv: 1511.00561. URL: http://arxiv.org/abs/1511.00561.

[6] Maxim Berman and Matthew B. Blaschko. “Optimization of the Jaccard index for image segmentation with the Lovász hinge.” In: *CoRR* abs/1705.08790 (2017). arXiv: 1705.08790. URL: http://arxiv.org/abs/1705.08790.

[7] Gedas Bertasius et al. “Convolutional Random Walk Networks for Semantic Image Segmentation.” In: *CoRR* abs/1605.07681 (2016). arXiv: 1605.07681. URL: http://arxiv.org/abs/1605.07681.

[8] Jeroen Bertels et al. “Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory & Practice.” In: *CoRR* abs/1911.01685 (2019). arXiv: 1911.01685. URL: http://arxiv.org/abs/1911.01685.

[9] Tim Boucher. *What is an orthomosaic photo?* 2015. URL: https://medium.com/new-farmer/what-is-an-orthomosaic-photo-11140c0601df.

[10] John Canny. “A Computational Approach to Edge Detection.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (1986), pp. 679–698. DOI: 10.1109/TPAMI.1986.4767851.

[11] Liang-Chieh Chen et al. “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4 (2018), pp. 834–848. DOI: 10.1109/TPAMI.2017.2699184.

[12] Liang-Chieh Chen et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation.” In: *CoRR* abs/1802.02611 (2018). arXiv: 1802.02611. URL: http://arxiv.org/abs/1802.02611.

[13] Liang-Chieh Chen et al. “Rethinking Atrous Convolution for Semantic Image Segmentation.” In: *CoRR* abs/1706.05587 (2017). arXiv: 1706.05587. URL: http://arxiv.org/abs/1706.05587.

[14] Liang-Chieh Chen et al. "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs." In: *CoRR. arXiv* (Dec. 2014).

[15] Liang-Chieh Chen et al. "Semantic Image Segmentation with Task-Specific Edge Detection Using CNNs and a Discriminatively Trained Domain Transform." In: *CoRR* abs/1511.03328 (2015). arXiv: 1511.03328. URL: http://arxiv.org/abs/1511.03328.

[16] Xier Chen et al. "Supervised Edge Attention Network for Accurate Image Instance Segmentation." In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Cham: Springer International Publishing, 2020, pp. 617–631. ISBN: 978-3-030-58583-9.

[17] Bowen Cheng et al. "Boundary IoU: Improving Object-Centric Image Segmentation Evaluation." In: (June 2021), pp. 15334–15342.

[18] Camille Couprie et al. "Indoor Semantic Segmentation using depth information." In: (Jan. 2013).

[19] Jia Deng et al. "Imagenet: A large-scale hierarchical image database." In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[20] Henghui Ding et al. "Boundary-Aware Feature Propagation for Scene Segmentation." In: *CoRR* abs/1909.00179 (2019). arXiv: 1909.00179. URL: http://arxiv.org/abs/1909.00179.

[21] Kento Doi and Akira Iwasaki. "The Effect of Focal Loss in Semantic Segmentation of High Resolution Aerial Image." In: *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*. 2018, pp. 6919–6922. DOI: 10.1109/IGARSS.2018.8519409.

[22] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929. URL: https://arxiv.org/abs/2010.11929.

[23] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929. URL: https://arxiv.org/abs/2010.11929.

[24] Michal Drozdzal et al. "The Importance of Skip Connections in Biomedical Image Segmentation." In: *Deep Learning and Data Labeling for Medical Applications*. Ed. by Gustavo Carneiro et al. Cham: Springer International Publishing, 2016, pp. 179–187. ISBN: 978-3-319-46976-8.

[25] Ionut Cosmin Duta et al. "Pyramidal Convolution: Rethinking Convolutional Neural Networks for Visual Recognition." In: *CoRR* abs/2006.11538 (2020). arXiv: 2006.11538. URL: https://arxiv.org/abs/2006.11538.

[26] IBM Cloud Education. "Convolutional Neural Networks." In: (2020).

[27] N. Elyasi and Mehdi Moghadam. "Tda in classification alongside with neural nets." In: (Aug. 2020).

[28] P. Ganesan et al. "A Comprehensive Review of the Impact of Color Space on Image Segmentation." In: *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. 2019, pp. 962–967. DOI: 10.1109/ICACCS.2019.8728392.

[29] Alberto Garcia-Garcia et al. "A Review on Deep Learning Techniques Applied to Semantic Segmentation." In: *CoRR* abs/1704.06857 (2017). arXiv: 1704.06857. URL: http://arxiv.org/abs/1704.06857.

[30] Alberto Garcia-Garcia et al. "A Review on Deep Learning Techniques Applied to Semantic Segmentation." In: *CoRR* abs/1704.06857 (2017). arXiv: 1704.06857. URL: http://arxiv.org/abs/1704.06857.

[31] Stuart Geman and Donald Geman. "Geman, D.: Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6(6), 721-741." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 6 (Nov. 1984), pp. 721–741. DOI: 10.1109/TPAMI.1984.4767596.

[32] Geovekst. *SOSI-standardisert produktspesifikasjon: FKB-Bygning 5.0.* 2022. URL: https://sosi.geonorge.no/produktspesifikasjoner/FKB-Bygning/#trueinnledning-historikk-og-endringslogg.

[33] Markus Gerke et al. "ISPRS Semantic Labeling Contest." In: Sept. 2014. DOI: 10.13140/2.1.3570.9445.

[34] Hamza Ghandorh et al. "Semantic Segmentation and Edge Detection; Approach to Road Detection in Very High Resolution Satellite Images." In: *Remote Sensing* 14.3 (2022). ISSN: 2072-4292. DOI: 10.3390/rs14030613. URL: https://www.mdpi.com/2072-4292/14/3/613.

[35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[36] Carmen Grecea, Alina Bălă, and S. Herban. "Cadastral requirements for urban administration, key component for an efficient town planning." In: *Journal of Environmental Protection and Ecology* 14 (Jan. 2013), pp. 363–371.

[37] Enzo Grossi and Massimo Buscema. "Introduction to artificial neural networks." In: *European journal of gastroenterology & hepatology* 19 (Jan. 2008), pp. 1046–54. DOI: 10.1097/MEG.0b013e3282f198a0.

[38] John L. Gustafson. "Moore's Law." In: *Encyclopedia of Parallel Computing*. Ed. by David Padua. Boston, MA: Springer US, 2011, pp. 1177–1184. ISBN: 978-0-387-09766-4. DOI: 10.1007/978-0-387-09766-4_81. URL: https://doi.org/10.1007/978-0-387-09766-4_81.

[39] David J. Hand, Peter Christen, and Nishadi Kirielle. "F*: An Interpretable Transformation of the F-measure." In: *CoRR* abs/2008.00103 (2020). arXiv: 2008.00103. URL: https://arxiv.org/abs/2008.00103.

[40] Jianzhong He et al. "Bi-Directional Cascade Network for Perceptual Edge Detection." In: *CoRR* abs/1902.10903 (2019). arXiv: 1902.10903. URL: http://arxiv.org/abs/1902.10903.

[41] Kaiming He et al. "Deep Residual Learning for Image Recognition." In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[42] Michael R. Heffels and Joaquin Vanschoren. "Aerial Imagery Pixel-level Segmentation." In: *CoRR* abs/2012.02024 (2020). arXiv: 2012.02024. URL: https://arxiv.org/abs/2012.02024.

[43] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. "A Fast Learning Algorithm for Deep Belief Nets." In: *Neural computation* 18 (Aug. 2006), pp. 1527–54. DOI: 10.1162/neco.2006.18.7.1527.

[44] Linxi Huan et al. "Unmixing Convolutional Features for Crisp Edge Detection." In: *CoRR* abs/2011.09808 (2020). arXiv: 2011.09808. URL: https://arxiv.org/abs/2011.09808.

[45] Dana Ilea and Paul Whelan. "Image segmentation based on the integration of colour-texture descriptors - A review." In: *Pattern Recognition* 44 (Oct. 2011), pp. 2479–2501. DOI: 10.1016/j.patcog.2011.03.005.

[46] Shruti Jadon. "A survey of loss functions for semantic segmentation." In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)* (Oct. 2020). DOI: 10.1109/cibcb48159.2020.9277638. URL: http://dx.doi.org/10.1109/CIBCB48159.2020.9277638.

[47] Pekka J. Saukko Jay A. Siegel and Max M. Houck. *Encyclopedia of Forensic Sciences (Second Edition)*. Elsevier, 2013. ISBN: 9780123821669.

[48] Longlong Jing and Yingli Tian. "Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey." In: *CoRR* abs/1902.06162 (2019). arXiv: 1902.06162. URL: http://arxiv.org/abs/1902.06162.

[49] Hoin Jung, Han-Soo Choi, and Myungjoo Kang. "Boundary Enhancement Semantic Segmentation for Building Extraction From Remote Sensed Image." In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–12. DOI: 10.1109/TGRS.2021.3108781.

[50] Kartverket. *Sentral Felles Kartdatabase (SFKB)*. 2022. URL: https://www.kartverket.no/geodataarbeid/sfkb.

[51] Hoel Kervadec et al. "Boundary loss for highly unbalanced segmentation." In: *Medical Image Analysis* 67 (Jan. 2021), p. 101851. DOI: 10.1016/j.media.2020.101851. URL: https://doi.org/10.1016%5C%2Fj.media.2020.101851.

[52] Jun Hee Kim et al. "Objects Segmentation From High-Resolution Aerial Images Using U-Net With Pyramid Pooling Layers." In: *IEEE Geoscience and Remote Sensing Letters* 16.1 (2019), pp. 115–119. DOI: 10.1109/LGRS.2018.2868880.

[53] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[55] Y. LeCun et al. "Backpropagation Applied to Handwritten Zip Code Recognition." In: *Neural Computation* 1.4 (1989), pp. 541–551. DOI: 10.1162/neco.1989.1.4.541.

[56] Kyungsu Lee et al. "Boundary-Oriented Binary Building Segmentation Model With Two Scheme Learning for Aerial Images." In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–17. DOI: 10.1109/TGRS.2021.3089623.

[57] Chunhui Liu et al. "DO CONVOLUTIONAL NEURAL NETWORKS ACT AS COMPOSITIONAL NEAREST NEIGHBORS?" In: *arXiv preprint arXiv:1711.10683* (2017).

[58] Yun Liu et al. "Richer Convolutional Features for Edge Detection." In: *CoRR* abs/1612.02103 (2016). arXiv: 1612.02103. URL: http://arxiv.org/abs/1612.02103.

[59] Ze Liu et al. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows." In: *CoRR* abs/2103.14030 (2021). arXiv: 2103.14030. URL: https://arxiv.org/abs/2103.14030.

[60] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation." In: *CoRR* abs/1411.4038 (2014). arXiv: 1411.4038. URL: http://arxiv.org/abs/1411.4038.

[61] Emmanuel Maggiori et al. "Can Semantic Labeling Methods Generalize to Any City? The Inria Aerial Image Labeling Benchmark." In: *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE. 2017.

[62] Mat Nizam Mahmud et al. "Road Image Segmentation using Unmanned Aerial Vehicle Images and DeepLab V3+ Semantic Segmentation Model." In: *2021 11th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. 2021, pp. 176–181. DOI: 10.1109/ICCSCE52189.2021.9530950.

[63] Microsoft. *Swin Transformer Architecture*. [Online; accessed March 17, 2022]. 2021. URL: https://github.com/microsoft/Swin-Transformer/blob/main/figures/teaser.png.

[64] Shervin Minaee et al. "Image Segmentation Using Deep Learning: A Survey." In: *CoRR* abs/2001.05566 (2020). arXiv: 2001.05566. URL: https://arxiv.org/abs/2001.05566.

[65] Volodymyr Mnih. "Machine Learning for Aerial Image Labeling." PhD thesis. University of Toronto, 2013.

[66] Giorgio Morales et al. "Automatic Segmentation of Mauritia flexuosa in Unmanned Aerial Vehicle (UAV) Imagery Using Deep Learning." In: *Forests* 9 (Nov. 2018), p. 736. DOI: 10.3390/f9120736.

[67] nchlis. *Image segmentation with Monte Carlo Dropout UNET and Keras*. [Online; accessed 28 February, 2022]. URL: https://nchlis.github.io/2019_10_30/architecture_unetV2.png.

[68] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. "Learning Deconvolution Network for Semantic Segmentation." In: *CoRR* abs/1505.04366 (2015). arXiv: 1505.04366. URL: http://arxiv.org/abs/1505.04366.

[69] Zhuokun Pan et al. "Deep Learning Segmentation and Classification for Urban Village Using a Worldview Satellite Image Based on U-Net." In: *Remote Sensing* 12 (May 2020). DOI: 10.3390/rs12101574.

[70] Adam Paszke et al. "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation." In: *CoRR* abs/1606.02147 (2016). arXiv: 1606.02147. URL: http://arxiv.org/abs/1606.02147.

[71] John Quinn. *Mapping Africa's Buildings with Satellite Imagery*. 2021. URL: https://ai.googleblog.com/2021/07/mapping-africas-buildings-with.html.

[72] Russell Reed and Robert J Marks II. *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks (A Bradford Book) Illustrated Edition*. MIT Press; Illustrated edition, 1999. ISBN: 0262527014.

[73] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[74] Adrian Rosebrock. *Intersection over Union (IoU) for object detection*. URL: https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

[75] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Representations by Back-Propagating Errors." In: *Neurocomputing: Foundations of Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 696–699. ISBN: 0262010976.

[76] Aletta Dóra Schlosser et al. "Building Extraction Using Orthophotos and Dense Point Cloud Derived from Visual Band Aerial Imagery Based on Machine Learning and Segmentation." In: *Remote Sensing* 12.15 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12152397. URL: https://www.mdpi.com/2072-4292/12/15/2397.

[77] Computer Vision — Making the machines see. *EDGENeural.AI*. URL: https://medium.com/@edgeneural.ai/computer-vision-making-the-machines-see-361a3d0cfc3f.

[78] M.J. Shensa. "The discrete wavelet transform: wedding the a trous and Mallat algorithms." In: *IEEE Transactions on Signal Processing* 40.10 (1992), pp. 2464–2482. DOI: 10.1109/78.157290.

[79] Oleksii Sheremet. *Intersection over union (IoU) calculation for evaluating an image segmentation model*. URL: https://towardsdatascience.com/intersection-over-union-iou-calculation-for-evaluating-an-image-segmentation-model-8b22e2e84686.

[80] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[81] Yashwant Singh. *Concepts of Photogrammetry*. URL: https://www.satpalda.com/blogs/concepts-of-photogrammetry.

[82] Carole H. Sudre et al. "Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations." In: *CoRR* abs/1707.03237 (2017). arXiv: 1707.03237. URL: http://arxiv.org/abs/1707.03237.

[83] Tomasz Szandała. "Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks." In: Jan. 2021, pp. 203–224. ISBN: 978-981-15-5494-0. DOI: 10.1007/978-981-15-5495-7_11.

[84] Christian Szegedy et al. "Going Deeper with Convolutions." In: *CoRR* abs/1409.4842 (2014). arXiv: 1409.4842. URL: http://arxiv.org/abs/1409.4842.

[85] Andrew Tao, Karan Sapra, and Bryan Catanzaro. "Hierarchical Multi-Scale Attention for Semantic Segmentation." In: *CoRR* abs/2005.10821 (2020). arXiv: 2005.10821. URL: https://arxiv.org/abs/2005.10821.

[86] Ashish Vaswani et al. "Attention Is All You Need." In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[87] Chi Wang et al. "Active Boundary Loss for Semantic Segmentation." In: *CoRR* abs/2102.02696 (2021). arXiv: 2102.02696. URL: https://arxiv.org/abs/2102.02696.

[88] Jingdong Wang et al. "Deep High-Resolution Representation Learning for Visual Recognition." In: *CoRR* abs/1908.07919 (2019). arXiv: 1908.07919. URL: http://arxiv.org/abs/1908.07919.

[89] Yanheng Wang et al. "Mask DeepLab: End-to-end image segmentation for change detection in high-resolution remote sensing images." In: *International Journal of Applied Earth Observation and Geoinformation* 104 (2021), p. 102582. ISSN: 0303-2434. DOI: https://doi.org/10.1016/j.jag.2021.102582. URL: https://www.sciencedirect.com/science/article/pii/S0303243421002890.

[90] Daniel Weinland, Remi Ronfard, and Edmond Boyer. "A survey of vision-based methods for action representation, segmentation and recognition." In: *Computer Vision and Image Understanding* 115.2 (2011), pp. 224–241. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2010.10.002. URL: https://www.sciencedirect.com/science/article/pii/S1077314210002171.

[91] Weihao Xia, Zhanglin Cheng, and Yujiu Yang. “SR-Net: Cooperative Image Segmentation and Restoration in Adverse Environmental Conditions.” In: *CoRR* abs/1911.00679 (2019). arXiv: 1911.00679. URL: http://arxiv.org/abs/1911.00679.

[92] Saining Xie and Zhuowen Tu. “Holistically-Nested Edge Detection.” In: *CoRR* abs/1504.06375 (2015). arXiv: 1504.06375. URL: http://arxiv.org/abs/1504.06375.

[93] Zhao Xing et al. “Use of Unmanned Aerial Vehicle Imagery and Deep Learning UNet to Extract Rice Lodging.” In: *Sensors* 19 (Sept. 2019), p. 3859. DOI: 10.3390/s19183859.

[94] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.

[95] Maoke Yang et al. “DenseASPP for Semantic Segmentation in Street Scenes.” In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3684–3692. DOI: 10.1109/CVPR.2018.00388.

[96] Kai Yue et al. “TreeSegNet: Automatically Constructed Tree CNNs for Subdecimeter Aerial Image Segmentation.” In: *CoRR* abs/1804.10879 (2018). arXiv: 1804.10879. URL: http://arxiv.org/abs/1804.10879.

[97] Zhengxin Zhang and Qingjie Liu. “Road Extraction by Deep Residual U-Net.” In: *IEEE Geoscience and Remote Sensing Letters* PP (Nov. 2017). DOI: 10.1109/LGRS.2018.2802944.

[98] Shuai Zhao et al. “Region Mutual Information Loss for Semantic Segmentation.” In: *CoRR* abs/1910.12037 (2019). arXiv: 1910.12037. URL: http://arxiv.org/abs/1910.12037.