Automated land use classification

Supervised segmentation of road structures on aerial images using shape regression

P. Brand

Automated land use classification

Supervised segmentation of road structures on aerial images using shape regression

by

P. Brand

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday January 23, 2019 at 2:30 PM.

Student number:	4330676	
Project duration:	February 26, 2018 – December 24, 2018	
Thesis committee:	Prof. dr. ir. M. J. T. Reinders,	TU Delft, chair
	Dr. J. F. P. Kooij,	TU Delft, supervisor
	D. van der Maas,	Rijkswaterstaat, daily supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

TU Delft Delft University of Technology

Preface

The report in front of you “Automated land use classification – Supervised segmentation of road structures on aerial images using shape regression” represents the work done for my master thesis. The research has been effected in close collaboration with TU Delft and DataLab at Rijkswaterstaat, Delft. During my search for a subject for my master thesis I got in touch with Dr. J.F.P. Kooij. He introduced me to Rijkswaterstaat in Delft, where the idea of embedding topological properties into a segmentation method for land use classification was proposed.

I am a graduate student of Computer Science at the faculty of EEMCS at TU Delft, with a keen interest in Computer Vision and Deep Learning. As I am passionate about broadening and deepening my knowledge by combining Computer Science with other technical or scientific fields, this research gave me the perfect opportunity to do so as it combines Computer Science with the field of Remote Sensing.

Although Dr. Kooij was associated with a different faculty of TU Delft, and despite his busy schedule, he was prepared to supervise me during my thesis project. I would like to thank him for this as well as for his personal guidance, his commitment throughout the research project and his constructive and critical feedback on my thesis.

My daily supervisor at Rijkswaterstaat DataLab, Daan van der Maas, who I would like to thank for his continuous support, his assistance in the execution of the experiments and the constructive and critical feedback. I really appreciated the games of table football during the breaks as well.

I would like to thank the members of the thesis committee for their interest in my research, for reading my thesis and for the critical evaluation.

Furthermore, I would like the thank all colleagues at DataLab for the pleasant cooperation and their contribution to the positive working environment and Rijkswaterstaat in general for providing me with the needed facilities to conduct my research.

Where would I be without my family and friends? Last but not least, I would like to thank all of them for their full support, love and trust throughout and my dog for the friendship and the pleasant and relaxing walks.

Without the support of all of you none of this would have been possible, for which I would like to express my sincere gratitude.

P. Brand
Delft, December 2018

Abstract

Recent advances in Artificial Intelligence and Computer Vision have been showed to be promising for automated land use classification of remotely sensed data. However, current state-of-the-art per-pixel segmentation networks fail to accurately capture geometrical and topological properties on land use segmentation, as these methods have inherently a lot of freedom. These geometrical and topological properties of land use structures are crucial for describing land usage on topographical maps, as their purpose is to present insight into topology and borders of land use structures. In order to preserve the geometrical and topological properties of land use structures, a novel segmentation method is introduced and tested on road structures in this thesis. Unlike current state-of-the-art segmentation networks, this new method performs the segmentation task by utilizing shape regression techniques as currently applied by state-of-the-art object detection networks. As modern object detection methods are only able to perform regression on simplistic shapes, and road structures generally describe complex shapes, a new topology preserving annotation generation method is introduced that subdivides a complex road structure into a set of oriented rectangular shapes. Since not many publicly available land use datasets contain both aerial images and per-pixel annotations, a new dataset based on aerial images and land use annotations, covering large areas of the Netherlands, is introduced as well. The results show that the novel segmentation method is capable of learning the newly introduced road structure representation, which preserves geometrical and topological properties. The connectedness property, however, is lost. The novel method does currently not outperform current state-of-the-art per-pixel segmentation networks, although several directions for future work are proposed to improve the segmentation performance of the shape regression based technique and preserve the connectedness property.

Contents

1

Introduction

The land use of the areas in the Netherlands are changing constantly. In about a years time, new residential areas are constructed, roads are rerouted, and green areas are created to preserve nature. Therefore, due to the ever changing environment in the Netherlands, the topographical maps that help us navigate from A to B daily, as well as provide organisations with valuable information about the land use of the country, need to be kept up to date frequently.

A topographical map is generally an abstract representation of highly detailed spatial information, so that it can be represented on a comprehensible level of detail for the scale it is presented on. Figure 1.1 shows a topographical map, created by Centraal Bureau voor de Statistiek [5] (CBS), which represents the land-use in the province of South-Holland, the Netherlands. The abstraction maps, in this case, the individual buildings of, for instance, the city of Rotterdam to larger regions that cover large collections of buildings. Consequently, these regions are not labeled as single buildings, but rather grouped together as one class named urban ('bebouwd' on the map). This is only one example of a topographical map, other examples include but are not limited to maps for navigation or vegetation charts that categorize different types of vegetation in an area.

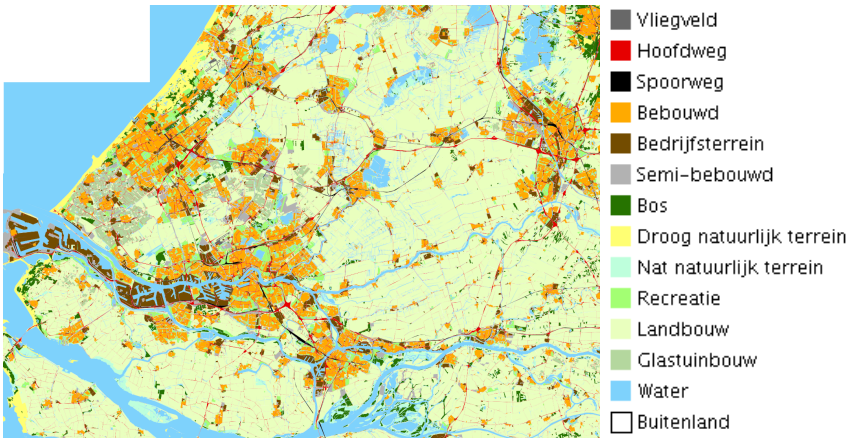


Figure 1.1: Topographical map of land-usage by CBS [5]

The construction of topographical maps of these large areas like the Netherlands are mostly manual, time consuming tasks performed by cartographers, where they assess information e.g. from aerial images and subdivide them into separate areas according to their land usage. Currently, these large topographical maps like the Bestand Grootschalige Topografie [42] (BGT), which roughly translates to "registry of large scale topography", or the Bestand Bodemgebruik (figure 1.1), which translates to "registry of land cover", of Centraal Bureau voor de Statistiek [5] (CBS) "Central Bureau for Statistics" are updated once every eighteen months in case of BGT, or even four years in case of the Bestand Bodemgebruik of CBS.

Essentially, the task of subdividing areas in aerial images (figure 1.2 *left*) according to land use could be seen as a per-pixel classification task. The topological land usage map is complete once each pixel in the image is assigned to one certain type of land usage, as for instance agriculture, water, or forest. In general, regions of land usage are defined by assigning one type of land usage to a group of neighbouring pixels, rather than individual pixels themselves. This particular per-pixel classification task, where the image is subdivided into regions of coherent pixels, is called a Semantic Segmentation task (figure 1.2 *right*).

Figure 1.2: Aerial image (left) and corresponding Semantic Segmentation (right)

Recent advances in the field of Computer Vision and Deep Learning have greatly improved performance of Artificial Neural Networks (ANN) at image classification tasks, whether it be recognizing handwritten digits, dog breeds, estimating people's poses or aiding doctors in their diagnosis of a patient.
Nowadays, Deep Learning techniques have established themselves as state-of-the-art for a variety of image classification tasks. Aside from image classification, Deep Learning has recently also been shown to achieve state-of-the-art performance for semantic segmentation tasks, e.g. in the field of autonomous driving, where a car perceives the world around it, segmenting its view of the world around it into areas that it can drive into, areas to avoid, and areas that provide information on for instance traffic rules.

Since the amount of computing power that is needed to perform these classification tasks has become available to the masses, Rijkswaterstaat [10] is interested to find out if Computer Vision and Deep Learning techniques could assist the cartographers in updating the topographical maps in order to increase the frequency of the updates released. Rijkswaterstaat's algorithm performed a two-class (binary) segmentation task of aerial images, where road pixels were separated from non-road pixels.

1.1. Challenges

This section provides an overview of the challenges that are faced in this thesis. Firstly, the challenge of defining an automated segmentation method that aims to preserve topological and geometrical properties is addressed. As this segmentation task can not be performed without data, the challenge of acquiring suitable data for this research is introduced secondly. Furthermore, the automated method should be applicable to the problems that Rijkswaterstaat aims to solve besides the preservation of topological and geometrical properties. This is described in the third subsection. In the last subsection, practical challenges imposed by hardware constraints are described as the automated segmentation task should be able to run on modern equipment.

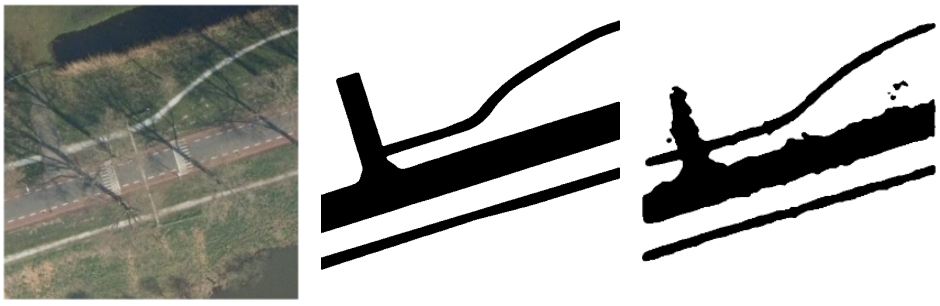
1.1.1. Topology and geometry extraction

Cartographers generally annotate roads and other types of land usage by drawing abstract polygon representations that preserve clear geometric and topological properties such as shape and genus. Modern convolutional based segmentation methods struggle to preserve an objects geometrical and

topological properties as they perform a per-pixel classification task, that inherenty has a lot of freedom. The per-pixel segmentation method tends to violate topological properties of structures, as it does not guarantee structure connectedness, nor does it prevent wrong predictions of “holes” in structures. Figure 1.3 shows an example of violation of topological properties. It depicts a binary road / non-road per-pixel segmentation task on an aerial image, where pixels belonging to the class road are depicted in black. The annotated ground truth is depicted in the middle, whereas the predicted per-pixel segmentation by an automated segmentation method is displayed on the right. Disconnected “islands” of road pixels are clearly visible in the predicted segmentations, where from the perspective of the non-road class these small islands are unrealistically small “holes” in the non-road class.

Additionally, the shape properties are not preserved in the prediction either. Where the annotation shows sharp borders for the road class, the prediction shows rough, curvy lines. This lack of ability to express sharp object boundaries is partly due to an inherent trade-off between object localization accuracy and classification performance [18], which leads to “fuzzy” borders of objects in semantic segmentation problems, as displayed in figure 1.3. Such fuzzy borders impose a problem for the automated generation of topographical maps from aerial images, since many men-made objects are defined by a rigid and straight shape. A Convolutional Neural Network generally combines information from a multitude of receptive fields from different sizes, to determine what it is looking at and in what context it should interpret it. Therefore it needs the small receptive fields to gain understanding of what it is looking at, for instance, a road section from an aerial image. On the other hand it needs a large receptive field as well, to be able to deduce that this piece of road is actually part of a park instead of a highway. However, both of these come at a cost. Gather more information from the smaller receptive fields and the network loses its ability to understand complex structures, whereas a bias towards information from larger receptive fields results in loss of descriptiveness such as detailed object borders.

Alternatively, if this problem would be defined as a regression problem on shapes instead of a per-pixel classification problem, the shape and genus properties of the structures are preserved, whereas connectedness is ensured as long as the right shapes are interconnected. Imagine for instance having the ability to place three initial rectangles of a certain width and height on the aerial image. One could then place two initial rectangles in a T-like figure at the T-junction and one rectangle on the thinner road below the large road section. By squeezing the rectangles, one could alter its shape to become longer and smaller until they fully overlap the T-junction depicted on the aerial image. Similarly the long, but thin road section could be covered by squeezing the third rectangle as well. Covering the thin curved road section is harder to achieve with this method, however it is not impossible, for instance if some non-rigid deformations are allowed to alter the shape of the box. Furthermore, arbitrarily complex road structures could be predicted by this method as long as one is allowed to place and squeeze as many rectangles as one would like.



(a) Aerial image. (b) Annotated segmentation. (c) Predicted segmentation.

Figure 1.3: Binary segmentation of roads in an aerial image [10].

This concept of translating and scaling initial boxes is embedded in modern object detection networks. These networks perform such regression tasks on rectangular shapes of different aspect ratios to attain a coarse estimate of an object’s structure, as depicted in figure 1.4. The object is subsequently classified based on, to some degree, the acquired shape information. Therefore an object that appears to be taller than it is wide, is more likely to be classified to the class ‘person’ than an object that is a bit wider than it is tall as the network learns by example to relate certain aspect ratios

to certain objects.

As the segmentation networks currently lack detailed and coherent object border representations, they could potentially benefit from the intrinsic (coarse) shape understanding trait of object detection networks, especially for objects with a fairly simplistic shape such as roads. Therefore one of the challenges in this thesis is to embed this particular trait into the segmentation network architecture.

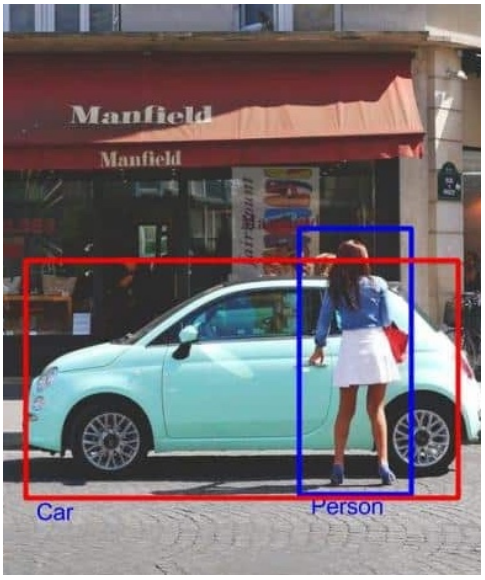


Figure 1.4: Structure representation and classification for object detection. The car is localized by the red rectangle that is wider than tall, where the person is detected by the blue, taller than wide, rectangle. Image modified from source [51]

1.1.2. Data acquisition

The Deep Learning technique used to train Neural Networks generally requires a large amount of data to learn from. One of the reasons why Deep Learning is currently state of the art for Computer Vision tasks such as image classification is due to the vast amount of example images accompanied by high quality annotations.

Unlike scene classification, a single class label per image does not suffice for the task of semantic segmentation as each pixel in an image has to be assigned to a certain class. Therefore the task of creating a large dataset with high quality annotations, is a more time consuming one. Consequently fewer semantic segmentation datasets are publicly available and generally contain fewer example images. Since Computer Vision techniques are not widely adopted for Remote Sensing purposes as of yet, gathering a sufficiently large amount of data with corresponding annotations for the semantic segmentation and shape regression task is considered to be a challenge as well.

1.1.3. Network usability

This thesis introduces a novel method for segmentation of topographical maps by representing the segmentation task as a regression task on polygonal shapes. As topographical maps can contain arbitrarily complex shapes, the scope of this research is narrowed down to one particular class, ‘Road’. Therefore, the initial network is not directly able to fulfill Rijkswaterstaat’s request of automatically extracting topographical maps from aerial images as these maps generally contain multiple types of structues besides roads.

Given that Rijkswaterstaat is currently experimenting with semantic segmentation networks for segmenting aerial images and that these networks struggle with predicting well-defined borders of roads, it is considered a challenge to find a way to utilize the newly introduced method’s potential to segment highly specialized well defined structures together with a more versatile semantic segmentation network. If succesful, this hybrid network could benefit from the more strictly defined shapes that approximate roads from the shape regression network, whilst maintaining the ability to perform the multi-class segmentation task required by Rijkswaterstaat.

1.1.4. Architecture complexity

Training semantic segmentation networks is a relatively high computationally extensive task. These networks generally consist of an encoder part, which are as complex as feature extractors of classification networks, and a decoder part often of similar complexity as the encoder part. Since Rijkswaterstaat is interested in a more accurate segmentation of road structures and multi-class segmentation, the proposed additions to incorporate shape regression would further increase the complexity of an already complex multi-class segmentation network.

As both the semantic segmentation network and the shape regression network both need a feature extractor that operates on images, defining a common basis could be beneficial. This can reduce the amount of duplication of both networks, which results in a decreased amount of learnable parameters as well. Assuming that a common basis is possible, it could also open the way for combined, end-to-end, training, which trains the semantic segmentation network and the shape regression network simultaneously.

The complexity of the algorithm influences the training and inference times, as well as physical resource requirements such as memory. Given that the algorithm should be able to both train and perform inference in a reasonable amount of time on a large dataset with samples of adequate resolution on modern hardware, it is considered a challenge to design the segmentation algorithm with efficiency in mind.

1.2. Research Questions

Per-pixel semantic segmentation networks struggle to express well defined geometrical and topological properties, such as shape and genus, due to the amount of freedom given by the per-pixel classification task. State-of-the-art object detection networks, on the other hand, inherently express geometrical properties of simplistic shapes, whereas topological properties like genus can be contained as well as they locate and classify objects via (rectangular) shape localization and regression. It is therefore proposed that the quality of semantic segmentations on aerial images can be improved by explicitly adding information about an object's geometry and topology to the segmentation process, by representing the segmentation task as a shape regression task. In order to test this hypotheses, the following research question is defined:

> *"Does introduction of geometrical and topological information to a segmentation task, attained via object localization and shape regression, improve the segmentation of road structures on Rijkswaterstaat's aerial data?"*

To address this question, several subquestions are formulated as well:

- Rijkswaterstaat aims to utilize a state-of-the-art segmentation network [49] for their multi-class land use segmentation task on large amounts of data covering the Netherlands. Since this network architecture was introduced as a solution for binary segmentation on very little data, it needs to be assessed whether it is capable of operating in Rijkswaterstaat's setting as well. Additionally, the quality of the segmentation of the newly proposed shape regression technique needs to be assessed and is therefore compared to a baseline semantic segmentation network. As the proposed shape regression technique is initially tested on a binary road / no-road segmentation task, the baseline semantic segmentation network needs to perform this semantic segmentation task as well. The semantic segmentation network is not necessarily restricted to a binary segmentation setting, although it is not certain if moving from a multi-class to a binary segmentation task influences the segmentation quality of the class that both settings have in common. In order to clear these uncertainties, the questions *"How well does the performance of Rijkswaterstaat's per-pixel segmentation network scale with the amount of data?"* and *"What is the impact on the segmentation performance of a state-of-the-art segmentation network when moving from binary to a multi-class segmentation problem?"* need to be answered.

- Rijkswaterstaat is interested in a multi-class segmentation task, whereas the proposed shape regression technique focuses on a binary segmentation task initially. One of the aims is therefore to research posibilities to combine the per-pixel segmentation task with the shape regression segmentation task in a hybrid approach. Both the state-of-the-art per-pixel segmentation network and the newly proposed shape regression network rely on a feature extractor based on

convolutions, which could potentially be shared between both networks to reduce overhead. Each method does however use a different feature extractor. Given the possibility of sharing one feature extractor for both the per-pixel and the shape regression segmentation task, it would be interesting to research the influence of utilizing a state-of-the-art feature extractor for both tasks, for which the question *"What is the influence on the segmentation performance if the feature extractor of a segmentation network is exchanged for a state-of-the art feature feature extractor?"* needs to be answered.

- Road topology can vary in complexity from a simplistic straight road section to a variety of complex shapes containing junctions, roundabouts and cross-roads. Modern day object detectors perform shape regression solely on simplistic shapes such as rectangles. Before shape regression can be applied to segmenting road topology, a topology preserving method needs to be defined that localizes roads as a series of simplistic topological structures from aerial images that a shape regression technique can segment. Thus an answer is required to the following question: *"How can an arbitrarily complex road structure be detected by locating a series of simplistic shapes such that the original topological properties are preserved?"*. Subsequently, a method that learns to segment topology preserving road structures via shape regression on a set of simplistic structures needs to be designed to perform the road segmentation task, for which the following question needs to be answered: *"How can a segmentation of a variety of pre-defined simplistic shapes segment complex road structures, whilst preserving the original topology?"*.
- Finally, a method needs to be devised to perform a multi-class segmentation task. As the new shape regression based segmentation focuses on road structures, it is expected that combining its output with a state-of-the-art multi-class per-pixel segmentation network is a suitable option to achieve a multi-class segmentation task. However, it needs to be researched whether the addition of the information provided by the shape regression method indeed preserve their geometrical and topological properties. For this reason the research question: *"How can the road segmentation, obtained from the shape regression network, be combined with a state-of-the-art per-pixel segmentation network in order to perform a geometry and topology preserving multi-class segmentation task?"*

2

Related work

The research described in this thesis combines the fields of Remote Sensing with Deep Learning and Computer Vision, as a segmentation task is performed by several Neural Networks on image data acquired through remote sensors installed on either satellites or aircrafts. This chapter contains related work from all three domains that is relevant to the research described in this thesis.

Section 2.1 focuses on the Remote Sensing domain containing related work with regard to Computer Vision on aerial or satellite imaging. As this thesis introduces a new dataset that contains aerial images accompanied with land use annotations, a set of related datasets is described in section 2.2. Previous research performed at Rijkswaterstaat resulted in a semantic segmentation network that is based on the U-net architecture. This network, as well as related semantic segmentation networks are included in section 2.3.

Since this thesis aims to find a method that solves a semantic segmantation task of road structures in aerial image using shape regression, the section 2.4 describes neural networks that utilize shape regression for another computer vision task. These networks localize objects in images by bounding them in a rectangular shape after which these objects are classified. Networks like this are called object detection networks and form an inspiration for shape regression network introduced in this work.

The preservation of topology plays and important role in this thesis, as the per-pixel classification networks do not guarantee topological properties such as genus and connectedness. In recent literature, several groups have looked into preserving topology in neural network predictions on various domains, which is described in section 2.5.

Some familiarity with neural networks, how they are constructed and the way they learn, is considered to be preliminary knowledge. For an introduction to artifical neural networks the work of Jain et. al. [31] is recommended. Their work describes the history of artificial neural networks, their relatation to biological neural networks, as well as how they are constructed and how they learn. Deep neural networks and an explanation of convolutional neural networks are described in the work of LeCun et. al. [34]. The book of Goodfellow et. al. [27] mathematically describes the tasks that neural networks attempt to solve, and what they learn in order to solve it in chapter six and is available online for free. Additionally, they provide an overview of mathematical background related to this thesis in chapters two, three and five. Furthermore, a survey on deep learning in remote sensing is provided by the work of Ball. et. al. [14], which gives an overview of Computer Vision and Deep Learning approaches to solving image classification, image segmentation and object detection problems on remote sensing data.

2.1. Remote sensing

Artificial Intelligence is widely researched in the Remote Sensing domain, where Computer Vision has been used on segmentation tasks and Deep Learning is now actively researched for image classification tasks. A land use classification task is a task where images are assigned a label based on the dominant type of land usage displayed in the image. An example is depicted in figure 2.1

Castelluccio et. al showed the potential of Convolutional Neural Networks (CNN) on the aerial images of two scene classification datasets, where their GoogleLeNet based CNN outperformed the

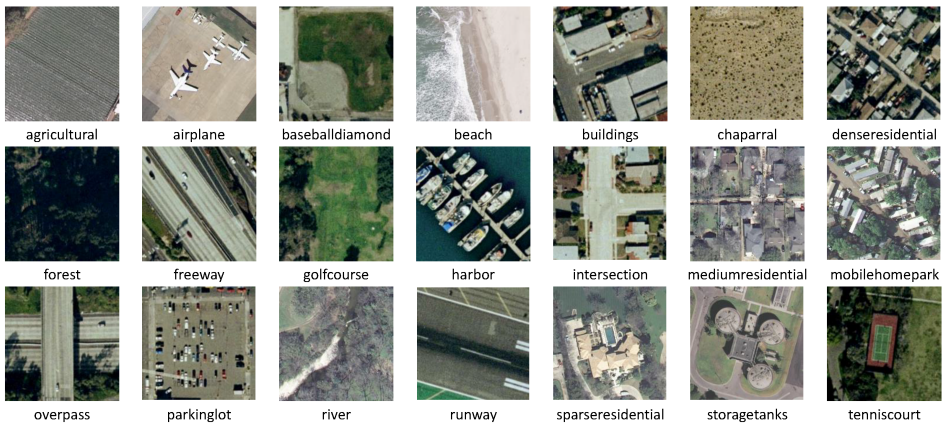


Figure 2.1: Aerial images with corresponding land use labels from the UC Merced Land Use Dataset [59]. Image acquired from "A Comparative Study of Sampling Analysis in the Scene Classification of Optical High-Spatial Resolution Remote Sensing Imagery" [29]

best reference techniques of both datasets with an accuracy of 97.10% on the UC Merced dataset [59] and an accuracy of 91.83% on the Brazilian Coffee Scenes dataset [44]. Several papers looked into feature extraction methods for Deep Learning on remote sensing data. Luus et. al. defined a data pre-processing technique for Deep Learning [39] that extracts patches of a fixed size at different locations of an image, at multiple scales of that image and combines them into a stack of views on which a CNN performs the classification task. A couple of papers looked into reusing features from pre-trained layers of CNN's that were trained on other image classification tasks outside the remote sensing domain [17, 28]. These works showed that the Deep Learning models consistently outperformed the classifiers that were trained on handcrafted features.

For the particular case of segmenting images according to various types of land usage, Deep Learning has seemingly not been widely adopted as of yet, possibly due to a lack of open data. In recent years many Kaggle segmentation challenges on remotely sensed data have been introduced, many of which showed active participation. Published literature on used architectures and achieved results on these challenges is on the other hand surprisingly very hard to find. The often restricted use of the provided dataset and fixed time window for challengers to submit their results, might be a possible explanation for the lack of publications on image segmentation in the Remote Sensing domain compared to for instance scene segmentation tasks, for which large datasets with high quality annotated segmentations are publicly available.

A semantic segmentation task for land use classification is described in the work of Maggiori et. al. [40], where they introduce a Fully Convolutional Neural Network (FCN) for a binary building segmentation task on satellite images. Their method is compared to a regular CNN based approach and a Support Vector Machines (SVM) classifier applied to each pixel, from which they concluded that thei FCN method outperformed the CNN and SVM method in terms of accuracy, with fewer visual artifacts and shorter inference time.

Volpi et. al. designed and compared several multi-class semantic segmentation networks for per-pixel classification of urban areas, two of which are image patch-based architectures, as well as one architecture that operates on the full image and contains an encoder and decoder-like architecture [57]. Sherrah introduced a FCN that, unlike the encoder-decodr FCN architecture, does not downsample the spatial resolution of the image [52] in the same multi-class semantic segmentation setting as Volpi et. al., where Sherrah achieved significantly improved accuracies at the expense of increased complexity, and a thus increased computation time.

2.2. Semantic segmentation datasets

This section discusses commonly used benchmarks and datasets that combine remote sensing data with per-pixel annotations such that new automated per-pixel semantic segmentation methods can be developed and subsequently compared to other reference methods. Important aspect that are discussed here, are the different channels included in the images as well as their mode of acquisition

and the distinct classes that are present in the annotations. Additionally, the spatial resolution of the images and the number of included samples are mentioned for each dataset as well as different types of acquisition modes capture images at different scales of detail.

2.2.1. ISPR

The International Society for Photogrammetry and Remote Sensing (ISPR) launched a 2D semantic labeling contest, for which they provided aerial images and digital surface models (DSM) containing height information of urban areas. The data is split in two major areas: the small village Vaihingen [3] that contains many detached buildings and small multi story buildings, and Potsdam [2] that contains large building blocks with narrow streets.

The aerial images of the Vaihingen set contain three bands consisting of near infrared, red and green channels, whereas the DSM file contains one band gray scale levels that correspond to the height information. For the Potsdam dataset, the aerial images are provided as the same infrared, red and green channel images as the Vaihingen set, or regular red green blue channel images. Additionally, this set also provides the union of the above mentioned bands, resulting in aerial images of four channels, that contain a red, blue, green and infrared channel.

The spatial resolution of one pixel is $9 \times 9cm$ for both the aerial images as the DSM images of the 33 patches of the Vaihingen set, of which 16 patches are accompanied with ground truth annotations. The spatial resolution of one pixel in the 38 patches of the Potsdam set is $5 \times 5cm$ for both the aerial images and the DSM images, of which 24 patches are accompanied with ground truth annotations. The segmentation challenge depicts a multi-class segmentation task with six defined classes: impervious surfaces, building, low vegetation, tree, car and clutter / background.

2.2.2. DeepGlobe

The DeepGlobe dataset [22] started as a Kaggle Challenge, for which challengers were provided with high resolution satellite images. Three types of annotations are provided for three distinct challenges, namely road extraction, building detection and land cover classification. Even though one of the challenges contains a per-pixel binary segmentation task of road structures, the land cover classification challenge is more closely related to Rijkswaterstaat's aim for automated land use classification.

The land cover classification challenge is a per-pixel classification task on satellite images that consist of three channels: red, green and blue. 1146 image samples of 2448×2448 pixels are available, each with a pixel-resolution of $50 \times 50cm$. These images are divided over a training, validation and testing set of respectively 803, 171 and 172 images. Land cover annotations are provided and contain 6 distinct classes and an additional Unknow class. The other six classes are: urban land, agriculture land, rangeland, forest land, water and barren land. Demir et. al. mention that they aim to make the DeepGlobe datasets publicly available [22]. Unfortunately, it has not yet been published at time of writing of this thesis.

2.2.3. Dstl Satellite Imagery Feature Detection

The Defence Science and Technology Laboratory (Dstl) designed a Kaggle challenge to classify features on satellite images [1]. They provide satellite images either in three band red, green, blue format, or in sixteen band format that contains spectral information in the ranges of multispectral and short-wave infrared wavelengths. Each image spans an area of $1 \times 1km$. Annotations are deliverd as polygonal shapes, for which ten different classes are labeled: buildings, misc. manmade structures, road, track, trees, crops, waterway, standing water, vehicle large and vehicle small. The rules on intellectual property for this dataset describe that this dataset is solely available for the purpose and duration of the Kaggle challenge, which ended in 2017.

2.2.4. Zurich Summer Dataset

The Zurich Summer Dataset, as described by Volpi and Ferrari [56] provides 20 multispectral satellite images that contain four bands: near-infrared, red, blue and green. The images have an average size of about 1000×1150 pixels and were acquired by the QuickBird satellite over the Swiss city Zurich in August 2002. The pixel resolution is $0.61 \times 0.61m$. Annotations are provided as superpixel images, that contain 8 different urban classes: roads, buildings, trees, grass, bare soil, water, rails and pools. This dataset is made publicly available and can be redistributed freely if the added acknowledgement

and license are added to the distribution as well.

2.3. Segmentation networks

Semantic segmentation networks perform a per-pixel classification task on image data, where a certain label is assigned to each pixel in the image. Currently, encoder-decoder semantic segmentation networks achieve state-of-the-art results on image segmentation tasks. These networks utilize an image classification network as a basis for the encoder, and a decoder method which utilizes an upsampling method for converting the encoded feature representations into per-pixel label predictions of the same spatial resolution as the input image.

The concept of this encoder-decoder based FCN's for semantic segmentation is introduced by Long et. al. [38]. Their network modified a VGG-16 image classification network into an encoder by discarding the classification layers, and by converting the all fully connected layers to convolution layers. In order to decode the abstract feature representations back to per-pixel predictions by three upsampling operations on feature maps via backwards convolutions, also called deconvolutions, upconvolutions or transpose convolutions.

Ronneberger et. al. developed a semantic segmentation network for Biomedical Image Segmentation [49] based on the encoder-decoder FCN architecture. Their implementation differs from the FCN introduced by Long et. al. in the sense that they remove the last pooling layer, thus reducing the number of receptive fields. Furthermore, they added an additional upsampling operation such that an upsampling operation and skip connection exists for each pooling operation. The U-net architecture consists of a regular feature extractor, based on convolutional layers and pooling layers as can be seen seen on the left side of the U-shaped architecture in figure 2.2, and an upsampler part, the right side of the U-shape in figure 2.2 that decodes the compressed layers from the feature extractor back to the original input size. Skip connections between each level of maxpool and upsamples ensure that information from various detail levels are combined for the per-pixel classification task in the final layer. The work of Kaiser et. al. applies this concept of a U-shaped segmentation network with so called

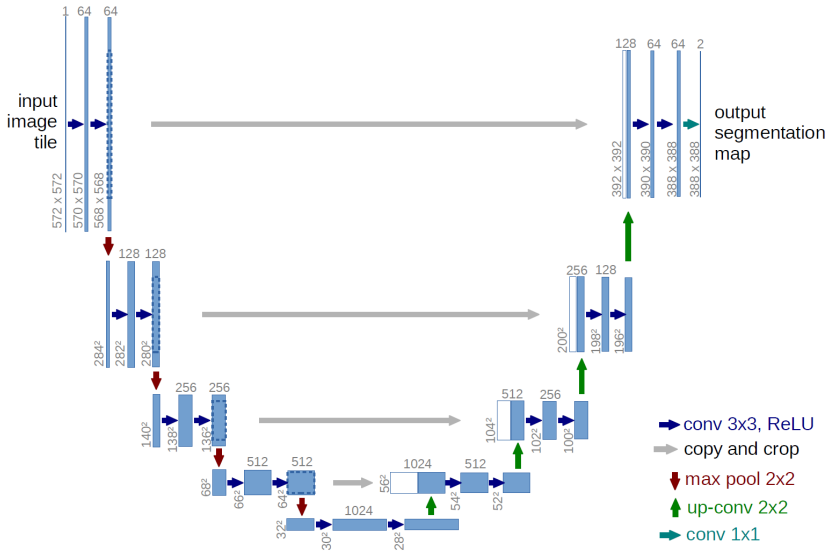


Figure 2.2: U-net architecture of Ronneberger et. al [49].

skip-connection on aerial images to segment buildings and roads [32].

The work of Badrinarayanan et. al. describes an architecture similar to U-net, although proposing a few changes [13]. Contrary to U-net, SegNet keeps the amount of feature maps constant throughout the network. Additionally, SegNet incorporates the full VGG-16 feature extractor, leading to a deeper network and a larger receptive field due to an additional pooling layer. Furthermore Badrinarayanan et. al. claim that SegNet is more memory efficient than U-net, due to the fact that SegNet only transfers pooling indices over the skip-connections, whereas U-net transfers whole feature maps over the skip-connections.

Several methods have been researched to reduce the effect of the Localisation and Detection trade-off that is inherent to these convolution and pooling based segmentation networks. One method of achieving sharper borders from Convolution Neural Network predictions is by Conditional Random Field (CRF) post processing. The CRF combines information from both local and remote neighbours and weights that according to its spatial distance, in an attempt regain shape information that was lost during during the convolution and pooling operations.

Chen et. al. reduced the inherent localization versus detection trade-off of Convolutional Neural Networks by implementing Conditional Random Fields and appending them to the Convolutional Neural Network in the form of Fully Connected Layers, resulting in an architecture called DeepLab [18].

The work in this thesis approaches this trade-off in a different way by performing regression on actual topological information from provided shape files to obtain concise border representations. This shape information could then be shared with the FCN segmentation network in order to obtain both contextual information as well as detailed shape information. This could

2.4. Object detection networks

Where semantic segmentation networks encode objects as groups of pixels with the same label, an object detection network highlights objects by encapsulating them in a rectangular shape, often referred to as bounding box, and assigning a label to each rectangle as visualized in figure 2.3. The object detection task they performed is therefore defined as a box regression task, where regression is performed on its location and width and height, combined with a box classification task.

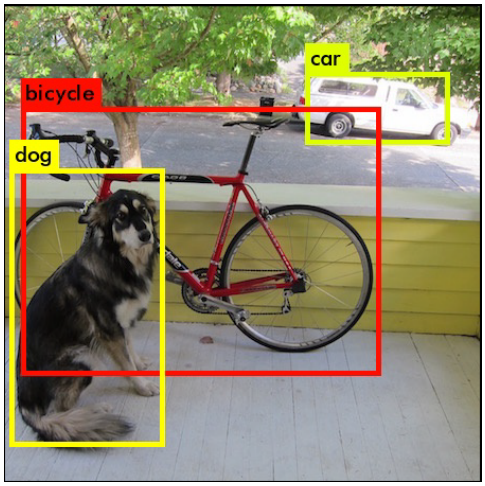


Figure 2.3: Example of object detection and localization. Modified from source [11].

The Regions with CNN features (R-CNN) as introduced by Girshick et. al. [26] forms the basis for many modern day object detectors. Their method combines a region proposal algorithm to extract regions that might contain objects. Each region is subsequently fed to a CNN that, in turn, performed the feature extraction after which a Support Vector Machines (SVM) classifier assigns a class label to the region. Fast R-CNN [25] improved the R-CNN architecture by creating one CNN that takes the whole image and all region proposals as input, and produces a feature vector per region proposal. The SVM is replaced by Fully Connected layers, that produce a region of interest feature vector. This vector is then fed to a set of Fully Connected layers that perform the classification task and to a different set of Fully Connected layers that perform the object localization task by regression on the bounding box parameters. Ren et. al. included further improvements to the Fast R-CNN architecture that they call Faster R-CNN [48]. They used a CNN to create a feature map of the whole input image, after which a newly introduced region proposal network is applied in a sliding-window fashion on that feature map to predict multiple regions of interest, called anchors, at each sliding-window location. Two sets of fully connected layers are then used to perform box regression and box classification respectively for each anchor box.

One of the down sides of the R-CNN methods is that it consists of several distinct algorithms, each with their own task in the pipeline of creating seperate box locations and box predictions. Some of

these algorithms are executed many times, resulting in relatively slow inference times when compared to more recent object detectors. Several recent works presented solutions to perform a combined box location and prediction operation [37, 45–47], where their most recent versions performed all disctinct task in one FCN [37, 45, 46].

These networks use a set of convolutional layers and max pool layers to extract features from an image similarly to the encoder part of FCN semantic segmentation networks. The feature maps provided by the encoder are of size $S \times S \times D$, where S represents the spatial width and height of the feature map and D is the depth of the feature maps, assuming that the input image had width w and height h and $w = h$ and $S < w$. Each spatial location of that feature map represents the feature vector of a cell in a coarse grid, that is overlayed on top of the input image. An example of such a grid is shown in figure 2.4.

A set of k anchor boxes of varying aspect ratios is defined for each grid cell, in order to localize and detect certain types of object. A cyclist generally has a profile that is more tall than wide, whereas a car is generally more wide than it is tall. Regression is performed on the anchor box parameters, as well as box classification by a set of convolutional layers. As these networks generally predict far more boxes ($S \times S \times k$) than there are objects present in the image, a box confidence score is predicted as well in order to predict whether an object is present in each box or not.

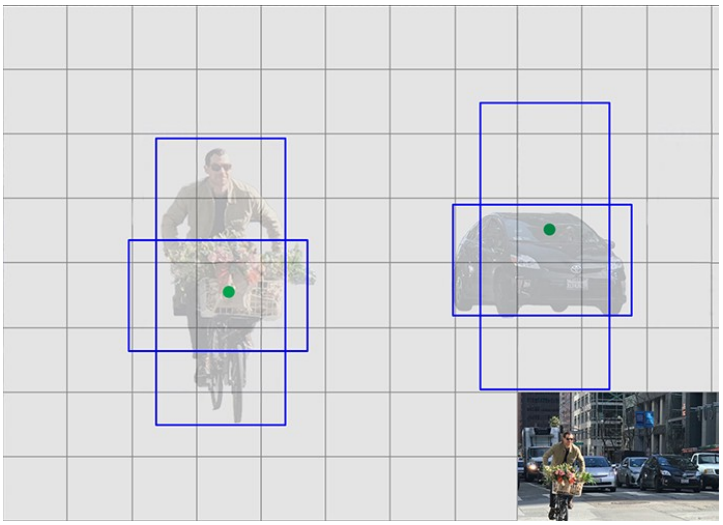


Figure 2.4: Visualization of the coarse grid of an input image. Anchor boxes of two objects are shown in blue, whereas the center of the object is shown as a green dot. Image acquired from [6].

The Single Shot Multibox Detector (SSD) [37] is one particular implementation of the method described above. Its FCN network architecture is based on a VGG-16 feature extractor, after which extra feature layers are defined that each represent features at a certain scale. Feature maps at each scale have their own set of anchor boxes. Box regression and box classification tasks are performed on all individual anchor boxes at each scale, to ensure that objects of varying sizes at various locations can be detected.

Another implementation is called You Only Look Once (YOLO) as introduced by Redmon et. al. [47]. Their second [45] and third [46] version follow a similar method as SSD, though with a few adjustments. Instead of defining anchor boxes per feature map scale, they introduce a reconnect layer that is similar to skip layer between two feature maps of different scales to obtain multi-scale features. Subsequently, they perform box regression and box classification on the last feature map. In the second version of YOLO, the anchor boxes are removed and prior boxes are introduced. These prior boxes are similar to anchor boxes, except that they are not necessarily centered around a common point. YOLO's prior boxes are derived from the provided annotated boxes by a K-means clustering algorithm. This allows for better box priors, under the assumption that object placement in the images generally follows a certain pattern.

Several methods extend general object detection networks to perform tasks beyond box regression and classification. The work of Liu et. al. extends the multibox detection by adding an orientation to the predicted bounding boxes, in order to provide tighter object localizations [36]. This is particularly

useful for object detection on images taken from a top-down perspective, such as satellite and aerial images as is shown in an example of their work, depicted in figure 2.5.

Figure 2.5: Rotable bounding boxes from Liu et. al. [36].

2.5. Utilizing topology

Including topological knowledge in Deep Learning on itself is not new. Cao et. al., for instance, introduced a Multi-Person 2D Pose Estimation [16] framework that detects people's poses in images by associating body parts, with individuals in an image. Their network predicts both part confidence maps, which detects body parts in an image, and part affinity fields, that encode the topology of the human body as part-to-part associations expressed in 2D vector fields. A post-processing step that interconnects detected body parts of each indivdual is performed by defining a spanning tree skeleton of human pose, after which the matching problem is subdivided into seperate bipartite matching problems in adjacent tree nodes. An example is shown in figure 2.6

Figure 2.6: Example of Multi-Person 2D Pose Estimation. Image modified from [16].

Several recent works introduced neural networks that extract road topology from aerial images [15, 41]. DeepRoadMapper [41] utilizes a CNN to segment roads in aerial images via per-pixel classification. Road centerlines are afterwards extracted from these segmented roads in a post-processing step that constructs a road graph. Since the road segmentation process does not guarantee road connectedness, several disjoint subgraphs could be detected, whereas they are not disjoint in the true road topology. They therefore proposed a correction step that, firstly, interconnects leaf-nodes based on inter-node distance and, secondly, chooses the best proposed connection based on the A^* shortest path algorithm. Bastani et. al. proposed several changes to DeepRoadMapper that creates a road topology extraction method that they call RoadTracer [15]. They removed the post-processing step of DeepRoadMapper. Instead, RoadTracer defines a a search method that starts at a point known to be

on the road, after which a decision function implemented in a CNN is iteratively invoked to take steps along the road whilst proposing vertices and edges that lay on the road.

2.6. Contributions of this thesis

This thesis builds on the results of the first network of Rijkswaterstaat, where it aims to increase the quality of the performed segmentation task. There are three main areas where it contributes to the application of a combination of Artificial Intelligence, Deep Learning and Computer Vision on the Remote Sensing domain.

First of all, a novel method is introduced that utilizes an object detectors capability to locate objects is designed to localize road structures in aerial images. In contrast to the localization task that the modern object detectors perform, as describe in section 2.4, this novel method performs an additional shape regression task on rectangles in order to segment detected road structures in aerial images. As this is a supervised method, a novel shape pre-processing method is proposed to create shape annotations that subdivides complex topological structures into smaller, more simplistic shapes, whilst aiming to preserve the roads topological properties. This novel method is based on the second version of the YOLO object detection network [45], as Redmon et. al. consistently showed that the YOLO architecture provides the flexibilty to change the architecture, whilst consistently achieving competitive results. The newly introduced shape regression network produces individual simplistic road shape proposals, that are conceptually very similar to the body part proposals described in the work of Cao et. al. [16] as these shape proposals need to be interconnected to preserve the connectedness property of road structures. The open pose framework introduced topological knowlegde to their neural network via part affinity fields combined a post-processing step in order to inter-connect individual body part proposals to produce estimated poses that are globally coherent. Whilst this globally coherent connection of local structures is relevant and important for road segmentation, road topologies are less restricted than the topology of the human body which makes pursuing global coherence a more complicated matter. Therefore, initially, a simpler approach that purely aims to detect individual road shapes is proposed. Individual road segment proposals are produced, in a similar fashion to the work of Liu et. al. [36], by encapsulating them by oriented rectangular shapes.

Complementary, a new semantic segmentation network based on the Darknet-19 feature extractor [45] and the U-net architecture [49], is introduced as well to enable the shape regression and per-pixel semantic segmentation network to share the same base feature extraction. Rijkswaterstaat uses a U-net architecture, which is based on the VGG-16 feature extractor. For the implementation of YOLO V2, the authors switched from using a VGG-16 feature extractor basis towards their Darknet-19 feature extractor as they claim it significantly increased the detectors speed without a significant loss in detection accuracy. In order to assess if U-net could benefit from a similar speed up, without giving in on classification performance, a new U-net like architecture is introduced that utilizes the Darknet feature extractor. This architecture, named Dark U-net, is designed to perform the same per-pixel semantic segmentation task as U-net does.

Furthermore, a new, large, dataset aimed at both multi-class semantic segmentation and object detection is made publicly available to assist further research on Artificial Intelligence for Remote Sensing. Section 2.2 shows that there are not many publicly available datasets that provide both a multi-class labelled land use segmentation task as well as a large amount of annotated samples for semantic segmentation. Luckily, Rijkswaterstaat has access to a vast amount of aerial image data and topographical maps of the Netherlands. The available topographical map covers the whole area of the Netherlands, whereas the available set of aerial images cover a large area of the Netherlands as well In this research a new dataset, based on these aerial images and the Bestand Bodemgebruik annotations by CBS, is introduced and made publicly available. This dataset contains nearly 200 aerial images with a spatial resolution of 0.5×0.5 meters, and a pixel resolution of 12500×10000 pixels, where the annotations consist of 13 individual classes. The aerial images are complemented with corresponding both per-pixel annotations and shapefile annotations, covering thirteen distinct classes. Compared to the datasets in section 2.2, this novel dataset covers an impressively large area, with a relatively high level of detail. Additionally, the annotations provide a 13 class semantic segmentation land use setting covering both urban and rural areas of the Netherlands.

3

Methodology

This chapter presents the proposed novel approach that represents semantic segmentation as a shape regression problem, instead of a per-pixel classification problem. Current state-of-the-art semantic segmentation networks perform a per-pixel classification task in order to segment an image into regions. As solving the per-pixel classification problem allows for a lot of freedom, these segmentation networks struggle to preserve the inherent geometrical and topological properties of the original structures in their segmentation. Additionally, topographical maps for land use are generally stored as shapefiles rather than regular RGB images, so the per-pixel classification methods need a post-processing step to convert the pixel representations to polygonal shapes.

Since polygon shapes are default for representing land use maps and polygons preserve topological properties such as genus and connectedness besides shape, it is interesting to see if the semantic segmentation can be defined as a task on shapes rather than pixels. Some of the object detection networks described in section 2.4 perform regression on rectangular shapes in such a way, that they localize and detect multiple objects in a scene by encapsulating them in a tightly fit box.

Let road structures be the main focus to simplify this idea. These structures are generally represented by a connected set of oriented rectangles, or lines of a certain width. If these structures could be subdivided into multiple interconnected road segments, represented by oriented rectangles, the topological and geometrical properties of road structures can be preserved. An object detector could possibly localize these individual road segments by encapsulating them in axis-aligned boxes. Now all that is left to do is to fit an oriented rectangle on the actual road segment inside that localized box. By defining road segmentation as a task of road segment localization and shape regression, complex road structures could be segmented in a way that preserves both the geometric and topological properties of a road structure.

However, this new representation only works on the binary road / no-road segmentation problem, whereas multi-class segmentation is preferred. State-of-the-art per-pixel segmentation networks are capable of performing multi-class segmentation problems. Possibly, the per-pixel segmentation networks could benefit from the information that the novel shape regression based technique provides. Therefore it is interesting to see if a hybrid system could preserve the geometrical and topological information from the shape regression based network on a per-pixel multi-class segmentation task.

Several requirements need to be fulfilled before this new concept can be tested. First of all a land use dataset is needed, which should consist of aerial images accompanied with shapefile annotations that contain road structures. As no such dataset was publicly available and a large amount of aerial and annotated data from various sources is accessible to Rijkswaterstaat, a new dataset is created. The design of a novel dataset at Rijkswaterstaat for multi-class segmentation of registered land use of the Netherlands is described in section 3.1. In that section, the methods used to acquire the data as well as its properties in terms of spatial resolution, annotated classes and number of samples are explained in further detail.

Furthermore, a per-pixel segmentation network needs to be constructed such that multi-class segmentation remains possible for the novel segmentation method. Since per-pixel segmentation is currently the state-of-the-art representation of the segmentation task, a general overview of this task, as well as how a neural network aims to learn to perform this task is explained in section 3.2.

The architecture and learning scheme of the per-pixel semantic segmentation network utilized by Rijkswaterstaat is described in 3.2.1, whereas a novel semantic segmentation network, using the state-of-the-art feature extractor as introduced by [45] is presented in 3.2.2. This network utilizes the same feature extractor as the YOLO object detector, thus opening opportunities for sharing the feature extractor of the per-pixel segmentation network and the shape regression segmentation network in the hybrid approach.

Since this thesis aims to segment roads via shape regression and object detection networks perform shape regression on rectangles, an object detection network might be a good basis for road segmentation via shape regression. Therefore, an explanation of the object detection task is provided in section 3.3. Subsequently, the YOLO V2 object detection network is briefly described, accompanied by a more detailed description of how this algorithm can be extended to potentially perform the road segmentation task via shape regression. Furthermore, a description of how the annotated data needs to be processed to accomodate the novel road segmentation technique is included in that section as well.

Lastly, a hybrid multi-class segmentation method that combines a multi-class per-pixel segmentation network with the shape regression based road segmentation network is described in section 3.4.

3.1. Dataset

Several datasets for semantic segmentation on remotely sensed data are described in section 2.2. However, most of these datasets contained very few samples with very little variation in environment, as they cover data of one city [2, 3, 56]. The DeepGlobe [22] and DSTL [1] datasets contain more data, of which deep neural networks would benefit, as well as a wider variety in scenery. Unfortunately, these datasets are not publicly available at time of writing, although there are plans to make the DeepGlobe dataset publicly available in the future [22].

Rijkswaterstaat would like to research the potential of automated land use segmentation for both urban and rural areas, such that it is applicable to an area covering the Netherlands. Many topological charts covering the Netherlands are visualized online at Publieke Dienstverlening op de Kaart [4] (PDOK). Several of these charts are publicly available and can serve as annotations for the required dataset. For this thesis a set of aerial images from 2012, provided by The Ministry of Agriculture, Nature and Food Quality [8], and Bestand BodemGebruik (BBG) topological chart also from 2012, provided by Centraal Bureau voor de Statistiek [5] (CBS) and Kadaster [7], are combined into an annotated dataset containing 13 classes: Road, Water, Residential area, Agriculture, Forest, Industrial area, Open and dry area, Railroad, Airport, Other terrain, Recreational, Greenhouse, Open and wet area. An overview of the class occurrences in terms of pixels is depicted in table 3.1. The BBG annotations are supplied in vector format, and are thus rasterized for direct use in segmentation networks. Additionally, a set of elevation level images of the Netherlands is included as well, acquired from Actueel Hoogtebestand Nederland (AHN), which is provided by Rijkswaterstaat [10]. These elevation levels can be used to extend the red, green, blue channel encoded aerial images with a depth channel, although this is not used for the work in this thesis. The combined aerial and topological data of 199 parcels with spatial size 0.5×0.5 meters of urban and rural areas of the Netherlands in 2012 are packaged and distributed for public use by Rijkswaterstaat [10].

All data is downsampled to the same spatial resolution of 0.5×0.5 meters, as the dataset is created from several sources (AHN, aerial images, BBG), of differing spatial resolutions. This resolution corresponds to the resolution of the coarsest data source. Another approach would be to upsample the data of all lower resolution sources to the resolution of the finest data source, however licensing agreements prevent publishing data in resolutions higher than 0.5×0.5 meters. The data of all sources are deliverd in the rijksdriehoek [9] (RD) coordinate system, of which 199 parcels of size 62.500×50.000 kilometers are cropped. The same extents are used to crop and store the BBG shapes. Additionally, these cropped shapefiles are rasterized to a pixel resolution of 0.5×0.5 meters for the per-pixel segmentation task. All images are stored in GeoTIFF format to preserve the geospatial meta-data. This dataset will be made publicly available by Rijkswaterstaat in the near future.

These parcels are of a pixel resolution of 12500×10000 pixels, which is too large to train neural networks on, as it exceeds the memory capacity of the equipment. Hence, these patches are pre-processed by cropping out and storing over 99000 patches with a pixel resolution of 500×500 pixels as shown in figure 3.1. The resolution of these patches are, however, not divisable by 32, which is a

Figure 3.1: Cropping 500 regions from one large image patch.

requirement of the used segmentation networks as described in section . These networks perform four or five max pool operations, leading to a spatial reduction of at most a factor $2^5 = 32$ and are therefore unable to handle images of dimensions that are not a multiple of 32. Consequently, these patches are cropped to 480×480 pixels before passing them to the network, to satisfy that condition. An example of such a patch is shown in figure 3.2.

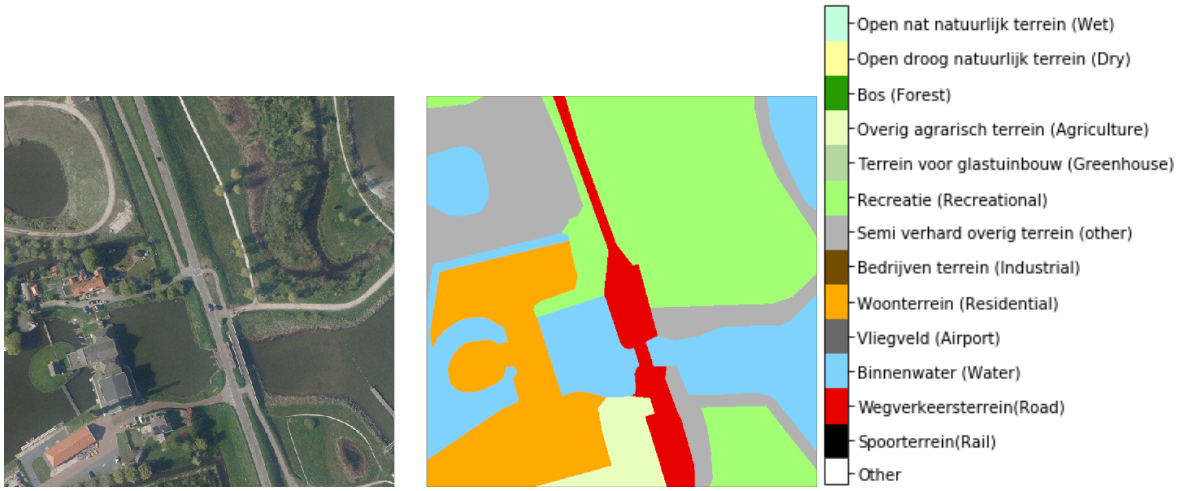


Figure 3.2: Example of a 480×480 aerial image patch (left) and corresponding annotation (right).

3.2. Semantic segmentation network

A semantic segmentation network combines a segmentation task, that subdivides an image x into separate regions, with a classification task, where all pixels belonging to a certain region are assigned to the same class. This semantic segmentation function f^* operates on an image x to produce the

Table 3.1: Overview of class occurence in terms of pixels for all annotations of the dataset.

Class	Number of pixels	Frequency (%)
Agriculture	1.30e+10	52.2
Water	3.62e+09	14.6
Residential area	2.80e+09	11.3
Forest	1.29e+09	5.2
Recreational	1.07e+09	4.3
Industrial area	8.84e+08	3.6
Road	7.66e+08	3.1
Other terrain	5.40e+08	2.2
Open and dry area	3.42e+08	1.4
Open and wet area	3.09e+08	1.2
Greenhouse	1.17e+08	0.5
Railroad	9.52e+07	0.4
Airport	3.07e+07	0.1
Total	2.49e+10	100.0

segmented image y.

Let an image x with width $w \in \mathbb{N}$ and height $h \in \mathbb{N}$ be defined as a set of $w \cdot h$ pixel values $(r, g, b) \in [0, 1]^3$, such that $x \in \{1, 2, \dots, 480\}^2 \times [0, 1]^3$ and the set of all Red,Green,Blue (RGB) channel images be X. Additionally the set of all segmented images is defined as Y. Assuming that there are c defined classes with $c \in \mathbb{N}$ and $c > 1$, then a segmented image y is defined as a set of $w \cdot h$ pixel values $y \in \{1, 2, \dots, c\}^{w \cdot h}$. A semantic segmentation network then operates as follows:

$$\begin{aligned} f^* &: \left\{\{1, 2, \dots, 480\}^2 \times [0, 1]^3 \in X\right\} \rightarrow \left\{\{1, 2, \dots, 480\}^2 \times \{1, 2, \dots, c\} \in Y\right\} \\ f^* &: x \mapsto y \end{aligned} \tag{3.1}$$

Unfortunately, this function f^* is unknown, however it can be approximated by learning from observations of RGB images and their corresponding segmented images. Let Ξ be the domain containing tuples of all possible RGB images and their corresponding segmented images. Then, given a large finite set of n examples $\mathrm{E} = \left\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\right\} \subset \Xi$, a set of k parameters θ, with $\theta \in R^k$ can be defined to approximate f^* with a function f for which $f(x; \theta) = \hat{y}$. Given the examples E a neural network can be trained to find the values of θ for which the mapping $f(x; \theta) = \hat{y}$ is the best approximation of f^* under observations E. The better E represents Ξ, the better function f will approximate f^*, under the assumption that the neural network has a sufficient large k to match the complexity of function f^*. How a neural network architecture actually translates to function f is considered to be outside the scope of this thesis. “Deep Learning” by Goodfellow et. al. [27] describes how the architecture of a neural network defines function f in further detail in the introduction of chapter 6 and section 6.1.

Supervised learning dictates that the network learns by example, and in order to do that, needs to be able to compare its predicted segmentation $\hat{y}$ with the provided true segmentation y. As the aim is to find the optimal configuration for θ in parameter space $\mathbb{R}^k$, a dissimilarity function L is defined for a certain set of values for θ that captures the difference between $f^*(x)$ and $f(x; \theta)$. Let this function, from here on defined as loss function, L^* be defined as $L^*(\theta) = f^*(x) - f(x; \theta)$, where the operator $-$ is an abstract representation of a differentiable difference measure. Since the actual function $f^*(x)$ is unknown, the set of observations E that contains a y produced by $f^*(x)$, is used for that actual loss function $L(\theta) = \sum_{x \in X} y - f(x; \theta)$. The calculation of $\hat{y}$ and the loss function L is called a forward pass through the neural network as it takes an image x as input and traverses the network in a forward direction towards the prediction $\hat{y}$.

Finding the best approximation f of f^* can be defined as an optimization problem, now that the difference between a prediction $\hat{y}$ and y is defined under a set of values for θ. The best approximation f is the set of values for parameters θ for the dissimilarity between $\hat{y}$ and y are minimal, hence the set of values for parameters θ for which function $L(\theta)$ is at its minimum. A numerical optimization algorithm, generally based on gradient optimization of L, is used to find the optimal optimal values for

the parameters θ.

New values for the parameters θ are calculated based on each neuron's contribution to the total loss. These loss contributions are calculated from the back of the network to the front, as the segmented image $\hat{y}$ is calculated on the output of the neurons in the last layer of that the network, that in turn rely on information of one or more neurons from the layer before it. As each neuron is a function that operates on input and the values of a subset $\omega^{(n)} \subset \theta$, its contribution to the loss can be influenced by changing the values of $\omega^{(n)}$ and by changing the input. Since the input of the neurons in the last layer is a combination of the output of the neurons of the layer before it, the input of the neurons in the last layer can be influenced by changing the values of the subset $\omega^{(n-1)} \subset \theta$ the the neurons in the layer before it operate on, as well as changing the input of those neurons This recursively propagates back to the neurons up to the very first layer of the network. As the input of these neurons are the provided data itself, their input is fixed and only the values of their subset of θ can be changed to influence the loss function. The method that calculates the $\nabla_\theta L(\theta)$ based on the partial derivatives of each subset ω is known as the back-propagation algorithm and is further explained in the work of Rumelhart et. al. [50] or in section 6.5 of "Deep Learning" [27]. A gradient based optimization algorithm is then used to optimize the loss function, based on its gradient provided by the back-propagation algorithm. As a result, it calculates the new set of values for parameters $\theta^{(1)}$. This procedure is known as a backwards pass through the network.

After obtaining the new values for parameters θ, another forward pass is performed to obtain $L(\theta^{(1)})$. Subsequently the gradient based optimization algorithm is run to obtain $\theta^{(2)}$. Alternating forward and backward passes through the network are performed until convergence is declared by the optimization algorithm after h iterations after which the final set of values of the parameters $\theta^{(h)}$ are presumed to set of values for which $f(x; \theta^{(h)})$ approximates function $f^*(x)$ best. The "Deep Learning" book provides a more detailed explanation of several optimization algorithms in chapter 8.

3.2.1. U-net per-pixel segmentation network

The network architecture that is used by Rijkswaterstaat to solve the initial road segmentation task follows the U-net architecture as described by Ronneberger et. al. [49]. It is a fully convolutional network as presented by Long et. al [38] that takes an image of dimensions width w, height h and depth d, where d represents the amount of channels. U-net's result is an image of dimensions of the same dimensions w and h except for the depth d, which is one and contains values in range $\{1, 2, \ldots, c\}$, where c is the number of classes.

The U-net architecture is depicted in figure 3.3 and consists of an encoder part and a decoder part. The encoder downsamples the spatial resolution of the feature maps by a factor $2^4 = 16$ in each spatial dimension, whereas the decoder upsamples the feature maps by a factor of 16 in each dimension, to supply an output with the same spatial resolution as the input image.

The encoder part is subdivided in 4 blocks, each consisting of two consecutive 3×3 convolution and rectified unit (ReLU) layers followed by a 2×2 max pooling layer with a stride of two that downsamples the width and height dimensions of the image by a factor of two. For each time the spatial dimensions are reduced, the number of feature channels, provided by the convolution layers in the next block, are doubled.

The decoder part is also subdivided in 4 blocks, where each block is the same as its corresponding encoder part, except for the max pooling layers, which are replaced by transpose convolution layers, or so called 'up-convolution' layers. These layers up-sample the feature maps to increase the spatial dimensions by a factor of two. At each block with a max pooling layer, the feature maps of the last convolution layer is concatenated with the up-sampled feature maps to counter the loss of information by the interpolation performed during up-sampling. These so called 'skip connections' are shown as grey arrows, where the concatenated feature maps are shown as white and blue boxes in figure 3.3.

The original U-net architecture describes a cropping step, to crop out the original image dimension from the pre-processed image. The authors describe an overlap-tile strategy for seamless segmentation of arbitraty large images [49]. They pad each image patch by mirroring the patch at the boundaries, to provide missing input data at the boundary for the convolution layers. this pre-processing step is not implemented in this research, hence the cropping steps in the U-net implementation are not implemented either. In the domain of of bio-informatics, where U-net was applied on images containing cells, the mirroring of patches at their boundary seems reasonable since cells are generally adjacent to other cells. This assumption arguably does not hold for aerial images, where mirroring the boundaries

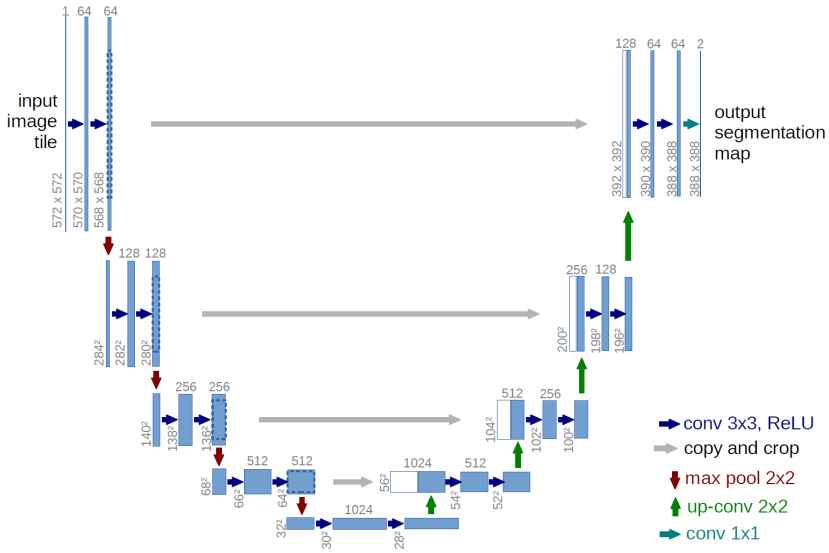


Figure 3.3: Schematic representation of U-net architecture. As this image was acquired from the work of Ronneberger et. al. [49], the width and height sizes of the feature maps, displayed vertically before each feature map, do not correspond to the width and height sizes of the architecture used in this image. Since U-net is a Fully Convolutional Neural Network, only the operations and the number of feature maps define the architecture of U-net as the width and height of the feature maps scale based on the width and height of the input image.

could lead to strange structures, e.g. a small strip of sea in between two beach lines. Since a zero-padding boundary conditions does not reuse parts of the image to add information at the borders, whether it be a cyclic or mirror pattern, it can not introduce strange and unexpected structures in the images and is therefore chosen as the padding strategy for this thesis.

3.2.2. Dark U-net per-pixel segmentation network

As this research focuses on a binary semantic segmentation task to extract roads from aerial images by defining it as a regression problem on shapes whilst Rijkswaterstaat is interested in improving their results on the multi-class segmentation task, a method to combine the binary shape regression technique with the semantic segmentation network is proposed. Since both methods rely on a convolution based feature extractor, it could be possible that both methods share the same feature extractor, which in turn, reduces the overhead of the combined method.

U-net utilizes a feature extractor that is similar to the VGG-16 architecture that is described in the work of Simonyan et. al. [53]. The YOLO V2 object detection network that serves as a basis for the road segmentation via shape regression method utilizes their own Darknet 19, of which the architecture is depicted in figure 3.4, as a feature extractor [45]. Redmon et. al. state in their work on YOLO V2 [45] that they exchanged their VGG-16 feature extractor for the original YOLO implementation [47] for Darknet 19 as they concluded that VGG-16 was needlesly complex for their task, as Darknet 19 proved to be faster than VGG-16 whilst achieving only a slightly worse accuracy. YOLO has proven to be capable of performing the object detection task well with either a VGG-16 or a Darknet 19 feature extractor.

Since the combined network incorporates U-net, which uses a VGG-16 like encoder and a similarly complex decoder, together with a YOLO based shape regression network, the combined network has a higher complexity than either individual network even if the feature extractor is shared. This high complexity potentially influences both training time and inference time drastically. As the VGG-16 architecture is stated to be needlesly complex for the object detection task, and the semantic segmentation task relies on the same feature extractor, the same statement could hold for the the semantic segmentation task. Therefore, a Darknet 19 based semantic segmentation network that follows the U-net architecture is introduced. This network architecture is called Dark U-net and is depicted schematically in figure 3.5 *bottom*.

The encoder part of the u-shaped architecture consists of the darknet 19 architucture, where the scene classification part, which is displayed under the double horizontal line in figure 3.4, is removed.

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Figure 3.4: Darknet 19 classification network as presented by Redmon and Farhadi [45]. The column “Output” represents the width and height of each feature map of the Darknet 19 classification network. Since images with a different width and height are used in this thesis, the Output column does not correspond to the width and height of each feature map of the Darknet 19 architecture used in this thesis.

The scene classification part is not suitable for the per-pixel classification task at hand, since it gives a single prediction based on the output of the last layer, whereas a class label for each pixel of the input image is required.

In order to achieve this per-pixel classification, five up-sample layers are needed to compensate for the five max pooling layers of the Darknet 19 architecture, which downscale the spatial resolution by a factor of $2^5 = 32$ in both the w and h dimension. Similar to U-net, these up-sample steps are performed by 3×3 transpose convolution layers with strides of two in both the w and h dimension. Basically, the convolution layers with 1024 and 512 filters after the last max pooling layers form the bottom ‘horizontal’ part of the u-shape, where the base of the right part is a copy of the left part. This copy is, however, modified in three ways. Firstly, the max pooling layers are replaced by the transpose convolutions. Secondly, the same, so called, skip connections that the U-net architecture uses are introduced to Dark U-net as well. These connections concatenate the output of the convolution layers before each max pooling layer of the left side of the network with the output of each transpose convolution layer on the right side of the network and are portrayed by the grey arrows and the white and blue box in figure 3.5. This ensures that features extracted from kernels with varying receptive fields flow back up into the network towards the per-pixel classification layers. Thirdly, a convolution layer with kernel size 1×1 is added on top of the last layer. This last layer converts the 32 output filters to n filter banks, where n represents the number of class labels. A softmax layer, applied to the output of the last layer, provides the estimated per-pixel a posteriori probabilities for each class. The final decision is then made by assigning each pixel to the class where its a posteriori probability is largest.

In order to train these semantic segmentation networks, a loss function needs to be defined. The U-net semantic segmentation network as described by Ronneberger et. al. utilizes a softmax cross-entropy loss [49], which is therefore used for both U-net and Dark U-net in this thesis as well.

The discrete Cross-entropy between the true distribution, an array y, containing the class annotations for all N pixels of the sample, and the estimated distribution, an array $\hat{y}$, containing class predictions for each pixel of the sample, is defined as follows:

$$L(y, \hat{y}) = -\sum_{i=1}^{N} y_i \log \hat{y}_i \tag{3.2}$$

Since y' contains class membership probabilities per pixel position, the array y must contain the

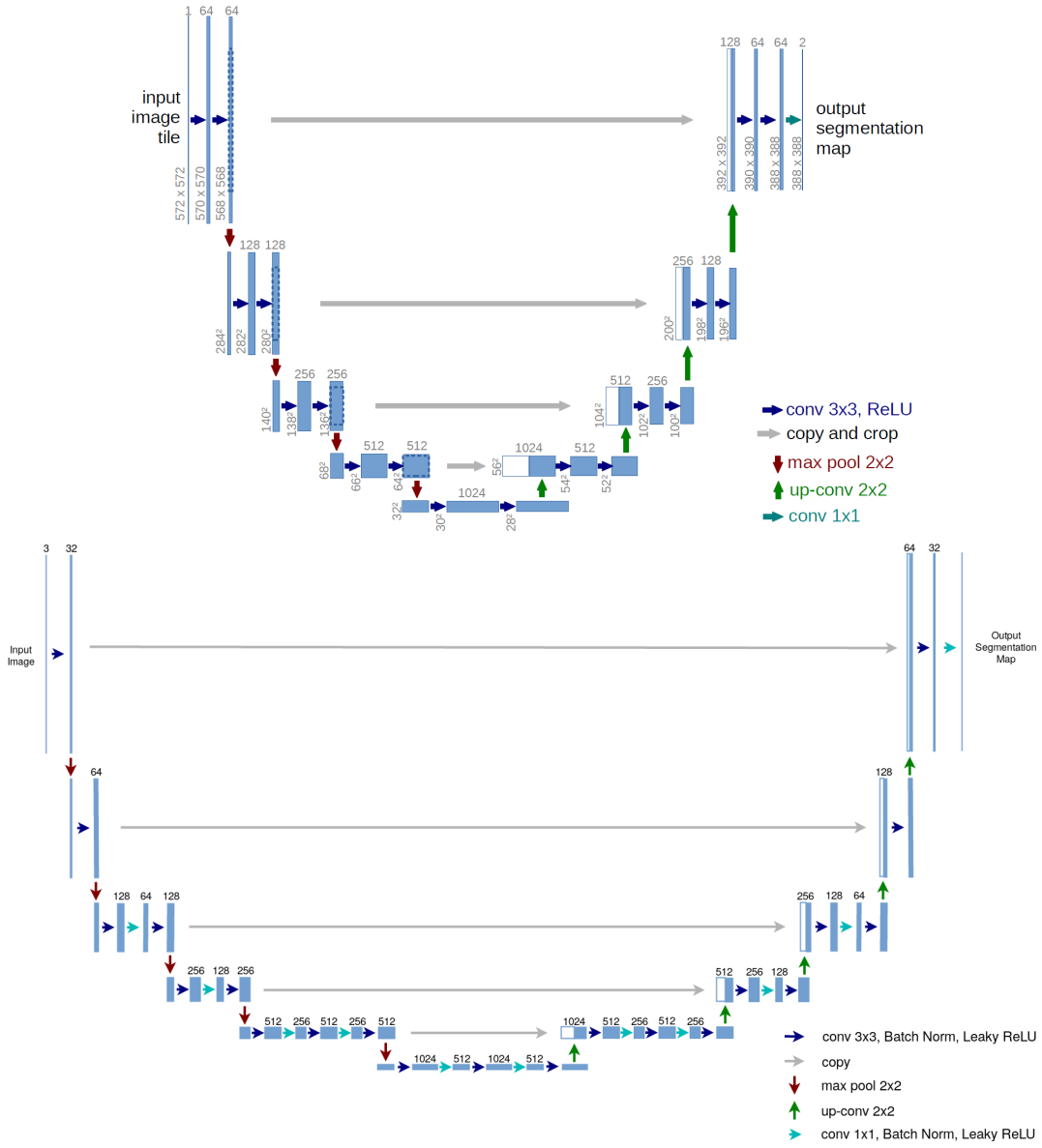


Figure 3.5: Architecture of Unet (top) and Dark Unet (bottom)

true class membership values per pixel position, which is achieved by a one-hot encoding of the class label. This one-hot encoding creates an array with M elements, where M is the number of classes. Let this bijective mapping function f, that maps a natural number n to its corresponding one-hot encoded vector $\underline{n}$, be defined as follows:

$$f : \{1, 2, \dots, M\} \to \left\{ h \in \mathbb{N}^M \;\middle|\; \forall i \; h_i \in \{0, 1\} \text{and} \sum_{i=1}^{M} h_i = 1 \right\}$$
$$f : n \mapsto \underline{n} \tag{3.3}$$

Where $\underline{n}$ is defined as:

$$\underline{n} = \left(h_i\right)_{i=1}^{\infty}, \; s.t. \; h_n = 1, \; and \; \forall i \neq n \; h_i = 0 \tag{3.4}$$

The cross entropy then becomes:

$$L(\underline{y}, \hat{y}) = -\sum_{i=1}^{N}\sum_{j=1}^{M} \underline{y_i}_j \log \hat{y}_{i_j} \tag{3.5}$$

A softmax function is applied to the output of the network, λ, in order to obtain, per pixel, class membership probabilities that sum up to 1. $\hat{y}_i$ is thus defined as follows:

$$f(x;\theta)_i = \hat{y}_i = \frac{e^{\lambda_i}}{\sum_{j=1}^{M} e^{\lambda_j}} \tag{3.6}$$

Now convert the loss function L to a function of the parameters the neural network θ on all rgb images $x \in X$ and corresponding annotations $y \in Y$, function L is rewritten as:

$$L(\theta) = \sum_{\forall x\, \forall y \text{ s.t. } (x,y)\in \mathrm{E}} -\sum_{i=1}^{N}\sum_{j=1}^{M} \underline{y_i}_j \log f(x;\theta)_{i_j} \tag{3.7}$$

For the semantic segmentation networks to be able to learn from their observations, a backward pass needs to be defined. The back-propagation algorithm is used to calculate the partial derivates of the configuration for θ, after which an optimizer is defined to minimize the loss function. Ronneberger et. al. described using Stochastic Gradient Descent (SGD) for optimizing the loss function of U-net. The loss function of YOLO V2 is also optimized using SGD as described by Redmon et. al., although they used a different set of values for the parameters of SGD. The semantic segmentation networks in this thesis use the ADAM optimzer as introduced by Kingma and Ba [33], as they describe that ADAM is computationally efficient and has little memory requirements. Their results show that the training cost of ADAM is lower than that of SGD with momentum for convolutional neural networks, which is the main reason why ADAM is used. Training the per-pixel semantic segmentation networks on the large dataset introduced in section 3.1 is expected to require a lot of time, due to the computationally expensive back-propagation algorithm. This expensive algorithm is performed on the complex network architecture for each batch of images in an epoch, which indicates that all training samples are processed exactly once. However, each batch can only contain a few images due to memory limitations of the hardware whilst the dataset contains a large amount of images. Consequently, the expensive back-propagation algorithm has to be performed many times, as it is run seperately for each batch in an epoch.

Moreover, the network is expected to train for many epochs as no pre-trained weights are used for any of the networks. If pre-trained weights were used, only a subset of the trainable parameters would have to be trained as the other trainable parameters are made constant. This is done by loading their optimized weights from another network that had trained these same parameters before. The networks in this thesis do unfortunately not utilize this possibility, due to time constraints given the complexity of implementing pre-trained weights in the used deep learning framework, for all networks in this chapter.

By choosing an efficient optimization algorithm, that converges in fewer epochs than a less efficient optimization algorithm, the expensive back-propagation algorithm is performed considerably less often which in turn shortens the training time considerably. Therefore ADAM is considered a suitable choice for this research.

3.3. Object detection network

An object detection network aims to localize one or more objects in images and detect to what class the object belongs. It is therefore a combination of a localization task and a classification task. Objects are generally localized by encapsulating them in a rectangle that aligns with the image's axis, which is called an Axis-Aligned Bounding Box (AABB). Additionally, the localized objects are classified by assigning a label to each bounding box.

Let an image x with width $w \in \mathbb{N}$ and height $h \in \mathbb{N}$ be defined as a set of $w \cdot h$ pixel values $(r, g, b) \in [0, 1]^3$, such that $x \in \{1, 2, \dots, 480\}^2 \times [0, 1]^3$ and the set of all Red,Green,Blue (RGB) channel images be X. Additionally an AABB b, with $b \in \{\mathbb{R}^4 \cdot \mathbb{N}\}$, is defined as a set of five parameters $(b^x, b^y, b^w, b^h, b^c)$, where b^x and b^y form the AABB's center coordinates and b^w and b^h are the width and height of the

box respectively. Assuming that there are c defined classes with $c \in \mathbb{N}$ and $c > 1$, the parameter b^c represents the class to which the object encapsulated by the AABB belongs.

Given that each image in X contains one or more objects, let k be the number of objects in an image x, with $k \in \mathbb{N}$ and $k > 1$. The localization and detection vector y contains the k boxes b such that $y \in \{\mathbb{R}^4 \cdot \mathbb{N}^2\}$. Let f^* be a function that operates on an image x to produce k AABB's. Then f^* is defined as:

$$\begin{aligned} f^* &: \left\{\{1, 2, \dots, 480\}^2 \times [0, 1]^3 \in X\right\} \rightarrow \mathbb{R}^4 \times \mathbb{N}^2 \\ f^* &: x \mapsto y \end{aligned} \tag{3.8}$$

Similar to the explanation in section 3.2, a neural network with parameters θ is defined as function $f(x; \theta) = \hat{y}$ that approximates function $f^*(x)$. In order for the network to learn the optimal set of values for parameters θ, a loss function $L(\theta)$ is optimized.

3.3.1. YOLO object detection network

In this thesis an existing object detection network architecture is extended to perform the road shape regression task on aerial images. Redmon et. al. showed that the YOLO object detection architecture is flexible as they described multiple architectural modifications that improve their original state-of-the-art architecture, described in "You Only Look Once: Unified, Real-Time Object Detection" [47], in their recent work that describe version two [45] and version three [46] of the object detector. Keeping this proven flexibility in mind, the YOLO architecture is chosen as the basis for the shape regression network for road segmentation. More specifically, version two of the YOLO architecture [45] is chosen as a basis, as it is the first version that follows a fully convolutional architecture which falls in line with the previously mentioned semantic segmentation networks. Given the aim of combining the two networks later on, choosing the fully convolutional architecture for the object detection network ensures that the main benefit of a fully convolutional architecture namely, allowing input images of an arbitraty size, is preserved for the combined architecture. In comparison with version three, the second version of the YOLO architecture is less complex as it uses the Darknet 19 feature extractor instead of the more complex Darknet 53 feature extractor that is introduced in version three.

The YOLO v2 architecture consists of the Darknet 19 feature extractor, excluding the classification part, and several convolutional layers before a 1×1 convolutional layer that produces the box and class predictions. The architecture is depicted in table 3.2. Since the architecture contains five maxpool layers, the original width and height dimensions of the input image are reduced by a factor of 32. This leads to a coarser $\frac{w}{32} \times \frac{h}{32}$ grid overlayed on the original image, where $w \in \mathbb{N}$ and $h \in \mathbb{N}$ and are both divisable by 32. The parameter λ in table 3.2 represents the number of parameters to predict per grid location. Let there be m boxes to predict and n known classes. YOLO encodes each box as four location parameters $b^{loc} = (b^x, b^y, b^w, b^h)$, with $b^{loc} \in \mathbb{R}^4$, one box confidence parameter b^p, with $b^p \in \mathbb{R}$, and a one-hot encoded class vector b^c, with $b^c \in \mathbb{N}^m$ s.t. $\forall i \; b_i^c \in \{0, 1\}$ and $\sum_{i=1}^{m} b_i^c = 1$. The total number of predicted parameters for each grid location is then defined as $m \cdot (4 + 1 + n)$.

Version two of the YOLO network does not predict the exact location, width and height of the annotated boxes, but instead predicts the offset for a set of pre-defined candidate boxes called prior boxes. These prior boxes are found by performing a K-means clustering on all annotated boxes to find a set of prior boxes that are assumed to be good starting candidates for all predictions. This K-means clustering is not performed in this thesis, as it is assumed to be a refinement step that would take additional time to implement. A set of five candidate boxes is defined instead, which follow the principle of the anchor boxes of different aspect ratios, each centered around the center of a grid cell. An example of four candidate boxes is depicted in figure 3.6. The anchor boxes used in this thesis have the following width heigh ratios which are defined relative to the width and height of the grid cell:

$$\begin{aligned} \text{anchors} = \{&(0.57273, 0.677385), \\ &(1.87446, 2.06253), \\ &(3.33843, 5.47434), \\ &(7.88282, 3.52778), \\ &(9.77052, 9.16828)\} \end{aligned} \tag{3.9}$$

These anchor boxes are not optimal for the problem at hand as several boxes have a near equal width and height even though road segments rarely have a $1 : 1$ aspect ratio. The anchor boxes are originally

designed for object detection on scene classification images, as they are used for object detection on the Microsoft COCO dataset [35]. Nevertheless, this set of boxes are assumed to be a good starting point, as they do capture a variety of aspect ratios and sizes.

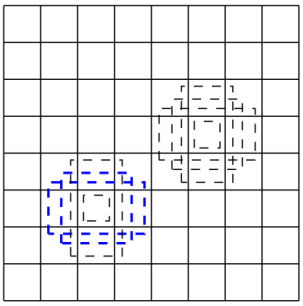


Figure 3.6: Example of anchor boxes centered around a grid cell. For visual clarity, the anchors are only drawn for two grid cells. Image modified from [37].

The values for b^{loc} are then defined the same way as introduced by Redmon and Farhadi [45], which is as follows:

$$\begin{aligned} b^x &= \sigma(t^x) + c^x \\ b^y &= \sigma(t^y) + c^y \\ b^w &= p^w e^{t^w} \\ b^h &= p^h e^{t^h} \end{aligned} \tag{3.10}$$

Where c^x and c^y are the center coordinates of a grid cell, p^w and p^h are the absolute width and height of the anchor boxes, thus the relative width and height times the width and height of the grid cell. The parameters (t^x, t^y, t^w, t^h) are the anchor box offset and scale parameters predicted by the YOLO network, where function $\sigma(a)$ is a sigmoid activation function applied to parameter a.

Redmon et. al. describe in "YOLO9000: Better, Faster, Stronger" [45] that they added a passthrough layer, similar to U-net's skip connections, from the last $3 \times 3 \times 512$ layer to the second to last convolutional layer. It is assumed that they meant the second to last $3 \times 3 \times 1024$ convolutional layer. Batch normalization is added to each convolutional layer [45]. Additionally, the leaky ReLU activation function, with $\alpha = 0.1$, is applied to each convolutional layer as explained in the first YOLO [47] publication. In order to learn how to detect objects from aerial images Redmon et. al. [47] described the loss function, depicted in equation 3.11, for YOLO version one. The coefficient $\lambda_{coord} = 5$ is a weighting coefficient to increase the penalty for box localization errors, whereas $\lambda_{noobj} = 0.5$ is a coefficient that decreases the penalty of the box confidence scores for boxes that do not contain objects, probably as a counter measure to the object / no object imbalance. These weights were deemed necessary to prevent the optimization from diverging early on.

$$\begin{aligned} Loss_{yolo} =& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ &+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\ &+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \\ &+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \tag{3.11}$$

Table 3.2: YOLO v2 [45] architecture, where λ is defined as the number of predictions (#boxes$\cdot(4+1+$#classes$)$). The horizontal line between the convolutional layers indicates the transition from the Darknet 19 feature extractor to the part that predicts object locations and class membership.

type	Filters	Size/Stride
Convolutional	32	3×3
Maxpool		$2 \times 2/2$
Convolutional	64	3×3
Maxpool		$2 \times 2/2$
Convolutional	128	3×3
Convolutional	64	1×1
Convolutional	128	3×3
Maxpool		$2 \times 2/2$
Convolutional	256	3×3
Convolutional	128	1×1
Convolutional	256	3×3
Maxpool		$2 \times 2/2$
Convolutional	512	3×3
Convolutional	256	1×1
Convolutional	512	3×3
Convolutional	256	1×1
Convolutional	512	3×3
Maxpool		$2 \times 2/2$
Convolutional	1024	3×3
Convolutional	512	1×1
Convolutional	1024	3×3
Convolutional	512	1×1
Convolutional	1024	3×3
Convolutional	1024	3×3
Convolutional	1024	3×3
Convolutional	1024	3×3
Convolutional	λ	1×1

The first version of YOLO is only able to predict one class for all boxes in a certain grid cell, which is not the case for YOLO version two. YOLO's second version is capable of detecting m boxes per grid cell, where each box has its own class confidence score. Therefore the last term of the loss function should be rewritten to:

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{3.12}$$

$\mathbb{1}_{ij}^{obj}$ is a binary mask that contains the value one at each grid cell i where an object is present and at each box j that is responsible for detecting that object and zero otherwise and $\mathbb{1}_{ij}^{noobj} = \neg\mathbb{1}_{ij}^{obj}$. Additionally S^2 represents the number of grid cells, and B the number of predicted bounding boxes per grid cell. x and y represent the box center coordinates, where w and h represents its width. C is the box confidence score that allows for non-maximum suppression during inference, and p symbolises the one-hot encoded class confidence vector. The loss is originally optimized by SGD, however, in this thesis, the ADAM optimizer is used instead of SGD.

3.3.2. Shape regression based segmentation method

The YOLO network described in subsection 3.3.1 is potentially able to localize roads or road sections, if suitable annotations are supplied. Localizing roads in aerial images is not a trivial task, as it could be solved in numerous different ways, all of which could lead to correct localizations. One could for instance either draw one large box around the whole road in the image, or split the road in smaller

segments by placing multiple boxes of varying sizes around locally coherent sections. However, these different methods might not all be suitable for the greater goal of performing shape regression for road segmentation. In case of localizing roads in an image by placing one large box around the road, the actual shape of the road inside could be rather simplistic such as one straight road section or terribly complex like several hairpin corners with several junctions or even a roundabout. As it is currently not feasible to perform regression on arbitrarily complex shapes, this particular choice of road localization is, although correct, not suitable for localizing roads for shape regression.

Therefore, an important part of this thesis is to define a method that divides a road in an image in a consistent way such that each individual segment has a rather simplistic shape. As the road localization task is aimed to be solved by a supervised deep learning method such as the YOLO object detection network, this subdivision method should preferably be automated so that the required annotations can be generated consistently on large quantities of data in a relatively short amount of time, instead of being created manually.

YOLO localizes an object by subdividing an image into separate regions, called grid cells, after which each cell detects an object of the object's center is present in that cell. Another characteristic is that it generally detects objects with a width and height larger than each individual cell.

Taking this into account, a simplistic method is created to subdivide complex road polygons in smaller, less complex sections. The complete road polygon is subdivided into separate regions, similar to YOLO's subdivision of an input image. In order to ensure that road sections can be larger than YOLO grid cells, the polygon grid is of a coarser scale of 3×3 cells. Per grid cell, the occurrence of a road subsection polygon is detected and stored. For each road subsection polygon its AABB is calculated by finding the individual minimum and maximum x and y values of the polygon p's points.

A Polygon p can always be represented by the ordered set P of N successive corner point coordinates, (x_i, y_i) with $i \in \{1, 2, \dots, N\}$, where all successive points are to be connected and the last point connects to the first point in order to form a closed polygon.

$$P = \big((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\big) \tag{3.13}$$

Let $X = (x_1, x_2, \dots, x_n)$ be a set containing all x coordinates of p and $Y = (y_1, y_2, \dots, y_n)$ be a set that represents all y coordinates of p. Then the minimum values are defined as:

$$\begin{aligned} x_{min} &= \min X \\ y_{min} &= \min Y \end{aligned} \tag{3.14}$$

Similarly the maximum values are defined as:

$$\begin{aligned} x_{max} &= \max X \\ y_{max} &= \max Y \end{aligned} \tag{3.15}$$

The axis aligned bounding box is then defined by the following two extreme points:

$$\begin{aligned} v_1 &= (x_{min}, y_{min}) \\ v_2 &= (x_{max}, y_{max}) \end{aligned} \tag{3.16}$$

This procedure is visualized in figure 3.7, where the original polygons are shown in figure 3.7 (a) with the road presented in red. The coarse 3×3 grid is depicted in figure 3.7 (b), where each road segment, after subdividing the road polygon by the 3×3 grid, is shown in a unique color in figure 3.7 (c). Their corresponding axis-aligned bounding boxes are subsequently depicted in figure 3.7 (d).

YOLO prefers its annotations to be stored as a list of; box center coordinates, width, height and class. Therefore for each axis-aligned bounding box B, its center coordinates (c_x, c_y), width w and height h are extracted. Since only segments belonging to the class road are localized, the class confidence vector p can be ignored. Given that the number of classes $n = 0$, and that there are four box parameters (c_x, c_y, w, h) and one box confidence parameter, the network's number of predicted variables per grid cell λ is then reduced to $m \cdot (4 + 1)$, where m is the number of predicted boxes per grid cell. The box confidence parameter is not stored explicitly in the annotations, as each annotated box has automatically a confidence score of one, whereas all other boxes are initiated as zero. This

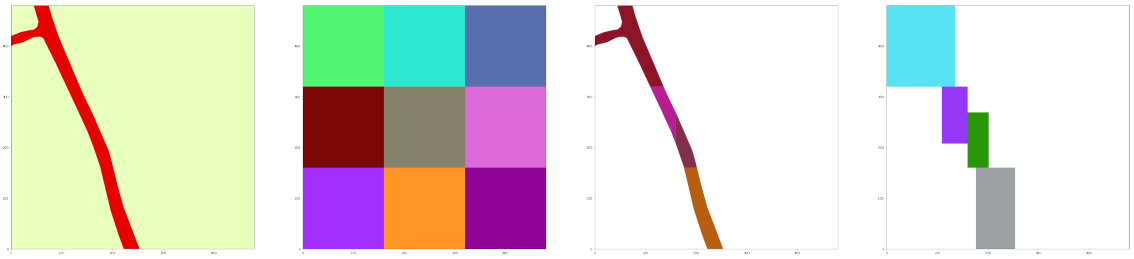
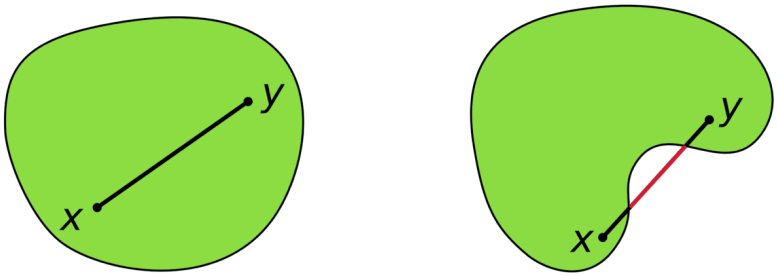


Figure 3.7: Process of annotation generation. From left to right: a) Original shape, with road in red. b) Coarse grid. c) Intersection of road shape with grid cells. d) envelope shape per road segment.



(a) An example of a convex polygon [20] (b) An example of a concave polygon [19]

Figure 3.8: Example of a convex and concave polygon shown in green, with points X and Y inside the polygon.

network is referred to as road localization network from here on.

$$\begin{aligned} c_x &= \frac{1}{2}(x_{min} + x_{max}) \\ c_y &= \frac{1}{2}(y_{min} + y_{max}) \\ w &= x_{max} - x_{min} \\ h &= y_{max} - y_{min} \end{aligned} \tag{3.17}$$

Eventually the localized road segments should be approximated, for which a straight line is used. This road segment representation assumes near straight road sections. Therefore, only patches that solely consist of straight road sections are sampled from the dataset. In order to oblige to this assumption, only road sections that are (near) convex polygons are sampled.

A polygon P is considered convex if and only if for any set of n points $V = \{v_1, v_2, ..., v_n\}$, with $n > 1$ and $\forall i\ v_i \in P$ with $i \in \{1, 2, ..., n\}$, and any set of n values $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_n\}$ such that $\forall i\ \lambda_i \in [0, 1]$ and $\sum_{i=1}^{n} \lambda_i = 1$, the following equation holds:

$$\sum_{i=1}^{n} \lambda_i v_i \ \in P \tag{3.18}$$

Figure 3.8 shows an example of a convex polygon (a) and a concave set (b). Given the green polygons and two points, X and Y, inside the polygon, the line section between these two points depict all possible linear combinations of the two points as described in equation 3.18. For the concave set depicted in 3.8 (b), the red part of the line exceeds the boundaries of the polygon, thus violating the description that any valid linear combination of $\lambda_1 X \cdot \lambda_2 Y$ should be inside the polygon. Therefore the polygon is depicted in (b) is not convex and thus concave.

The criteria of adding samples to the dataset based on all road segments being convex is a rather strict one which drastically reduces the amount of available samples. This criteria is therefore loosened a little bit, by defining convexity as a soft constraint. The soft constraint dictates that road segments are considered near-convex if the difference between its area and the area of its convex hull is less than threshold t.

The convex hull of a polygon P, consisting of points p_i for $i \in \{1, \dots, n\}$ is a polygon that describes the intersection of all convex sets containing all points p_i in P [58]. Figure 3.9 shows an example of the convex hull in blue, of a set of points in black. One could could describe the convex hull of a set of points based on an analogy, where the set points are represented by unmovable pillars and the convex hull by an elastic band. The convex hull of the set of points is then found by expanding the rubber band such that all pillars are inside its perimeter, depicted in the black outline in 3.9. Once the elastic band is released, it snaps closely along the outer pillars, representing the convex hull of the pillars.

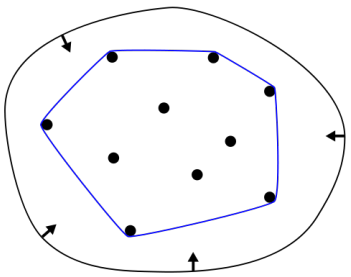


Figure 3.9: Convex hull in blue, of a set of points in black [21]. The black outline with inward arrows depict the elastic band analogy to describe the convex hull of a set of points.

Let $h(x)$ be the convex hull of x, then x is considered to be near-convex iff:

$$\begin{aligned} t &= 0.01 \cdot \text{area}(x) \\ \text{area}(h(x)) &- \text{area}(x) < t \end{aligned} \tag{3.19}$$

Since the convex hull of a convex polygon is equal to the original convex polygon, all convex polygons automatically meet this criteria. Polygons that share an overlap in area of at least 99% with their convex hull, are in this case considered to be near-convex. Consequently, the sample is added to the dataset if all other polygons in that sample meet this criteria as well.

Note that the top left road segment in 3.7 (c) is not convex, thus this example image would be discarded from the dataset.

Now that the road segment localization annotations are defined, all that is left is to add the annotated road segment shapes to the annotations. Under the assumption that all the localized segments contain (near) convex road segments, one can approximate the shape of a road segment by defining a line between the two points where the road segment hits its encapsulating box, where the width of the line is defined by the boundaries of the road segment's polygon.

The modified YOLO v2 architecture is expected to learn to represent road polygons by performing regression on the parametrization of each individual road segment's shape. Where the parametrization of each road segment line, which is encoded as two pairs of (x, y) coordinates accompanied by a road width parameter w. The road localization network, that is based on YOLO v2, is further expanded to predict the road segment parameters as well.

Continuing from the point where the road polygons are subdivided by the 3×3 grid of the previous method, depicted in the image on the left of figure 3.10. The middle image of that figure shows the road localization boxes, and the right most image shows the outline of the road segments in red, with their line approximation in blue. The road width is not taken into account for sake of the visualization. Aside from that, this example shows the importance of the convexity property of each individual road segment. The segment on the bottom right clearly violates this property, resulting in a bad road segment representation, as the blue line nearly exceeds the border of the actual segment.

The two points for each segment are derived from the intersection between the outline of the actual road segment, which is shown in red in the right most image of figure 3.10, and the outline of its corresponding localization box, depicted in the middle image of that same figure. These intersections are depicted as green lines in figure 3.10 (right). This intersection generally contains two line segments per road segment, of which their middle points p_1 and p_2 describe the line segment that approximates the road shape. These points are shown in orange in the right most image of figure 3.10.

Now that the road segment's direction is known, the width of the road needs to be estimated in order to attain a valid approximation of the segment's shape. As a line has a constant width throughout the

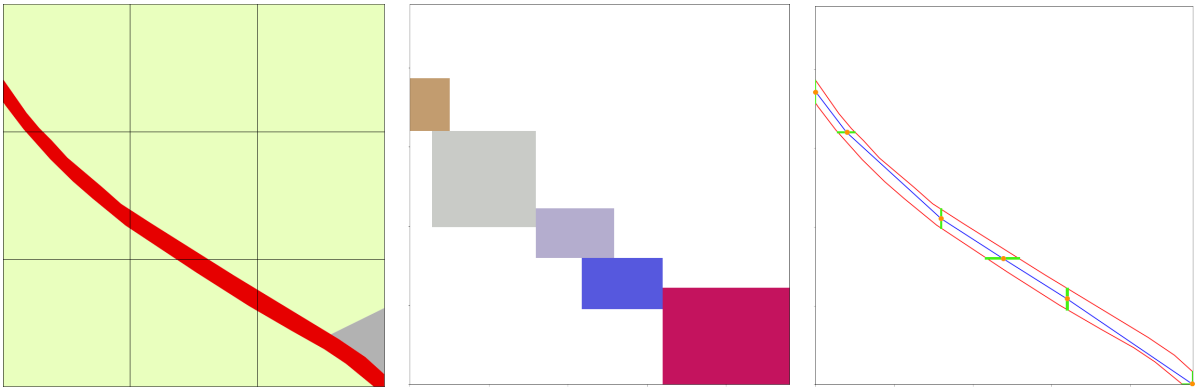


Figure 3.10: Subdivision of road polygon (in red) in the left image, with road localization annotations in the middle image, whereas the road line estimation (in blue) is depicted in the right image.

whole segment and the original road segments could still vary in width from one end to the other, a simplification is needed to estimate the road segment's width by a constant w.

For this constant one could choose for instance the smallest width, leading to a smaller road section than originally described in BBG, or the largest width, leading to a larger road section than originally provided, or something in between by taking for instance the average width. For the sake of simplicity the largest width is chosen, so that it at least covers the whole road section and, unfortunately, it covers some areas that are not annotated as road as well.

The largest width is easily calculated by the shortest side of the minimum area oriented bounding box around the original road segment, as a road segment is assumed to be longer than that it is wide. This oriented bounding box is calculated by the rotating calipers algorithm [54]. After which the width of the road is defined as the length of the shorted line segment of the outline of the oriented box. Intuitively, one could describe the rotating calipers method as iterating over each edge v_i for $i \in \{1, \dots, n\}$ of the n edges of the convex polygon. For each edge v_i, it then performs a change of basis transformation from vector space U, with the standard basis E_2 to a vector space $S_i \in \mathbb{R}^2$ of the polygon to a coordinate system with basis $D_i = \{v_i, v_i^{\perp}\}$, described by the mapping $T_i : U \mapsto S_i$ In this coordinate system, the axis aligned bounding box B_i of the polygon is calculated, of which the area A_i is stored as well. Let the axis aligned bounding box B_{min} be the bounding box with the smallest area:

$$\begin{aligned} j &= \arg\min_i A_i \\ B_{min} &= B_j \end{aligned} \tag{3.20}$$

The minimum area oriented bounding box OBB_{min} is then derived by mapping B_{min} back to U by the inverse mapping $T^{-1} : S_i \mapsto U$.

$$OBB_{min} = T^{-1}(B_{min}) \tag{3.21}$$

Visually, this could be interpreted as rotating a caliper around the polygon, while measuring the distance between the two blades each time the bottom blade aligns with one of the polygon's edges.

Road width estimation w is then defined as the length of the side l_h for $h \in \{1, 2, 3, 4\}$ of the rectangle OBB_{min} that has the shortest length.

$$w = \min_{a \in [1,\dots,4]} |l_a| \tag{3.22}$$

The annotations of each segment are then stored as a list of localization box center coordinate (c_x, c_y), localization box width w_{box}, localization box height h_{box}, road segment point one $p_1 = (x_1, y_1)$, road segment point two $p_2 = (x_2, y_2)$ and road width w_{road}, where p_1 and p_2 are normalized over the box width and height and in range $x_i \in [0, 1], y_i \in [0, 1]$ for $i \in \{1, 2\}$. Where the coordinate $(0, 0)$ corresponds to the bottom-left corner of the localization box and $(1, 1)$ corresponds to the top-right corner of the localization box.

Several further modifications of the YOLO v2 architecture need to be made for the network to be able to learn the newly introduced representation of road segments. That modified network is from here

on referred to as the shape regression network. In order to accommodate the extra five parameters that have to be estimated, the number of output nodes of the network is increased, so that each predicted box now predicts a total of ten parameters instead of the five box parameters of the road localization network.

In order to perform regression on the shape of the individual road segments, the loss function from equation 3.11 has to be extended as well. Where the previously used loss function solely calculated the loss of the localization box coordinates and the localization confidence, the extended version also takes into account the loss of the road segment parameters. These road segments are defined as lines, for which a start point $p1$, an end point $p2$ and a width w^{road} are defined. Each of these two points is defined by an x and y component. These $p1^x$, $p1^y$ and $p2^x$, $p2^y$ network predictions are very similar to the prediction of b^x and b^y in equation 3.10. In fact, as these components are offsets relative to the origin of the predicted localization box, their constant is equal to 0. As the points $p1$ and $p2$ are defined similarly to the box center prediction point prediction their loss is calculated in the same way as well. Thus, the sum of squared differences is used to calculate the loss of the road segment line point predictions:

$$\begin{aligned} Loss_{p_1} &= \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(p1_i^x - \hat{p1}_i^x)^2 + (p1_i^y - \hat{p1}_i^y)^2] \\ Loss_{p_2} &= \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(p2_i^x - \hat{p2}_i^x)^2 + (p2_i^y - \hat{p2}_i^y)^2] \end{aligned} \tag{3.23}$$

Besides points $p1$ and $p2$, a road line segment is defined by a certain width w^{road} as well. Since the width of a road can vary largely, it is more similiar to the behavior of the width and height of the localization boxes than to the box center coordinate predictions. w^{road} is therefore predicted in a similar way as the width of the localization box b^w, shown in equation 3.10, is predicted. b^w is predicted as a coefficient to the width of an anchor box. The width and height of an anchor box act as some form of regularization on the width and height prediction of localization boxes. As the chosen anchor boxes do not represent the width of the road well, this term has to be subsituted. The width of a grid cell does not directly relate to the width of the road, however it could act as a decent regularization term as it is constant and the width of the road is for instance larger than a quarter of the width of a grid cell and not larger than two or three times the width of a grid cell. Since the width of the road is parametrized similar to the width and height of the localization boxes, the term for the loss of the road width parameter is treated the same as term of for the width and height of the localization boxes. Therefore, the sum of squared differences of the square root of the width estimation and annotation is used to calculate the road width loss term:

$$Loss_{w^{road}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i^{road}} - \sqrt{\hat{w_i^{road}}})^2] \tag{3.24}$$

The updated loss function, which includes the loss of the road segmentation parameters, is then as

follows:

$$
\begin{aligned}
Loss_{shaperegression} = & \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
& + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(p1_i^x - \hat{p1}_i^x)^2 + (p1_i^y - \hat{p1}_i^y)^2] \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(p2_i^x - \hat{p2}_i^x)^2 + (p2_i^y - \hat{p2}_i^y)^2] \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i^{road}} - \sqrt{\hat{w_i^{road}}})^2]
\end{aligned}
\tag{3.25}
$$

3.4. Hybrid segmentation method

A hybrid approach of the state-of-the-art per-pixel segmentation method combined with the novel shape regression based segmentation method is introduced, such that topological and geometrical information acquired from the shape regression network is fed back into the per-pixel segmentation network. The architecture of the hybrid network is depicted in figure 3.11. This hybrid network shares the Darknet 19 [45] feature extractor for both the Dark U-net per-pixel segmentation network and the shape regression based road segmentation network. The network branches at the end of the feature extractor, which is depicted at the bottom right part of the U-shape in figure 3.11. From that point onwards, the network splits in the Dark U-net network going upwards, whilst the architecture of the shape regression based road segmentation network is completed downwards. Non-maximum suppression is applied on the ouput of the shape regression network in order to filter out all low-confidence road localisation predictions in a similar way as described by Redmon et. al. [45]. The road segment predictions are subsequently rasterized to obtain a binary per-pixel segmentation map containing roads and background pixels. This binary per-pixel segmentation map has the same spatial resolution as the original input image and can thus be appended to the upper right part of the U-shape via a skip connection. Afterwards the Dark U-net segmentation network performs its per-pixel classification task on the information acquired from both the per-pixel segmentation network and the shape regression based segmentation network.

This network can be trained by training either a Dark U-net or a shape regression network first, after which the trainable parameters of the feature extractor can be exported. Subsequently the hybrid network can be constructed, where the previously acquired weights, or a subset of it, are loaded and made constant. One can then train both the rest of the Dark U-net and shape regression network architecture seperately.

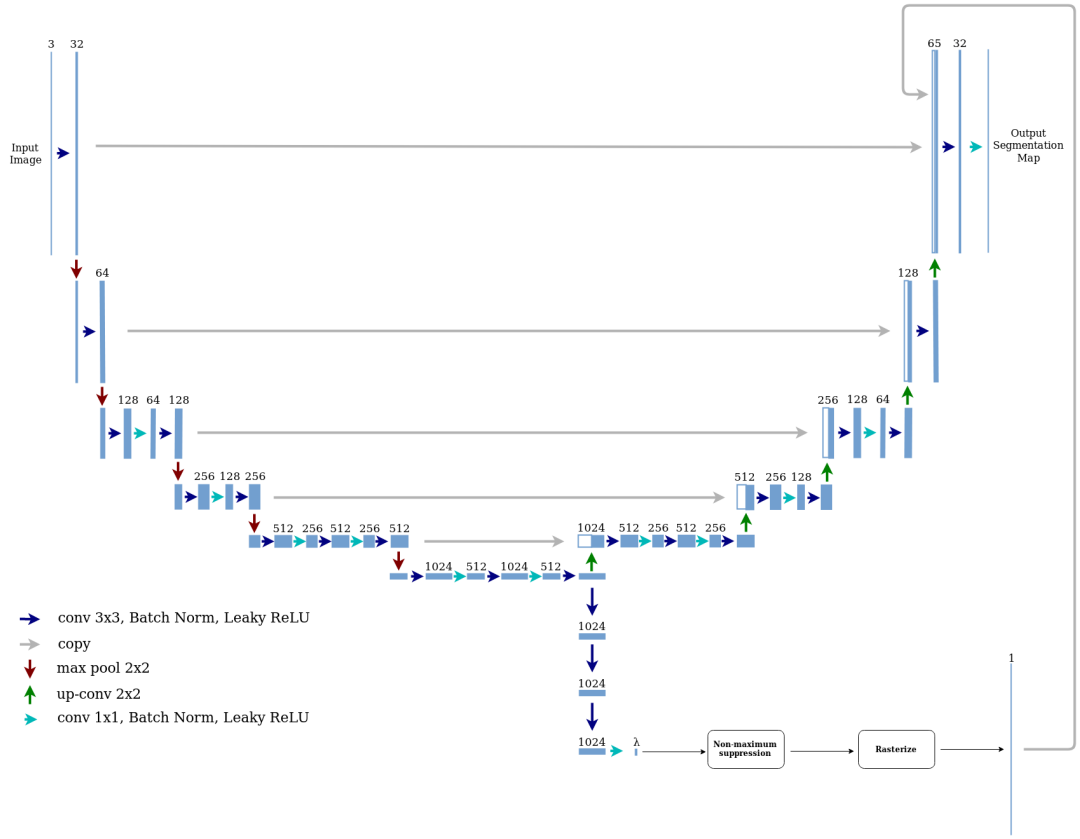


Figure 3.11: Architecture of a hybrid per-pixel and shape regression based segmentation method.

4

Experiments

The experiments described in this chapter are implemented in Python, where TensorFlow [12] is used as the deep learning framework. Scikit-Image [55] is used to proces image data, whereas Matplotlib [30] provided the necessary tools to visualize them. Shapefiles are loaded and processed using GeoPandas and Shapely [23] and rasterized using Rasterio [24]. The experiments are performed on a NVIDIA DGX-1, of which one NVIDIA Tesla V100 graphics card with 16 GB of video memory is utilized for training the neural networks.

The first experiment, described in section 4.1, assesses the potential of the U-net architecture on a multi-class semantic segmentation task, provided by the DeepGlobe benchmark that provides a large amount of satellite images combined with multi-class per-pixel annotations. Its performance is subsequently compared to the newly introduced Dark U-net semantic segmentation network. Additionally, the impact of data augmentation on the learning procedure is assessed.

As the novel shape regression technique focuses on a binary segmentation problem, the per-pixel semantic segmentation network needs to perform a binary segmentation task as well for comparison. The per-pixel semantic segmentation network is, however, able to perform a multi-class semantic segmentation task. In order to ascertain that road segmentation performance of the per-pixel network is not influenced by the total number of classes of the semantic segmentation problem, the second experiment is performed as described in section 4.2.

The experiment described in section 4.3 assesses an object detector's capability of understanding the novel road segment representation that is introduced in section 3.3.2 by training it on the generated road segment annotations.

Section 4.4 describes the experiment that is conducted to assess the performance of the road shape regression technique on images that contain (near) convex road structures. A Dark U-net per-pixel binary segmentation network is used as baseline for comparison of the networks' performance on the binary road segmentation task.

4.1. Feature extraction for segmentation

Since both the semantic segmentation network and the object detection network need a convolution based feature extractor, it would be beneficial in terms of resources if both networks could share the same feature extractor. Therefore, the performance of U-net is compared to Dark U-net on a segmentation task to assess the potential of both feature extractors. Additionally, the suitability for segmenting images according to land use is tested as well, for which a baseline is required.

This experiment aims to answer the research question *"How well does the performance of Rijkswaterstaat's per-pixel segmentation network scale with the amount of data?"*. U-net follows a similar encoder-decoder architecture that is used by the work of Long et. al. [38], which is known to perform well on multi-class segmentation tasks. Similarly, the feature extractor of U-net is based on the VGG-16 image classification network. This feature extractor is known to perform well on multi-class image classification tasks [53]. It is therefore expected that the U-net architecture also performs well on a multi-class semantic segmentation task. The networks on which U-net is based all proved to be effective on large dataset such as PASCAL VOC or ImageNet as well, thus it is expected that U-net

performs well on large quantitities of data as well.

A suitable dataset, for which a reference score is available, is needed for this experiment as there is no reference of the performance of a proven network architecture on the dataset that is introduced by Rijkswaterstaat. This lack of a benchmark renders this dataset unsuitable for the task of comparing network architectures for suitability of the segmentation task in general.

Unfortunately, not many datasets for land use segmentation are publicly available. The DeepGlobe dataset provides a similar setting, namely land use classification, on images taken from satellite images above the United States [22]. A DeepLab based network achieved a mean Intersection over Union (IoU) score of 0.433 at epoch 30 with a 512×512 patch size, where the class 'undefined' was not included in the calculation.

According to the DeepGlobe paper [22] the dataset should come available, where they hope it will become a valuable benchmark, although it is at time of writing of this thesis, not made publicly available yet. As this experiment is performed early on in the research it was expected that the DeepGlobe dataset would have been published near the end of the thesis project. The intention was to re-run the experiment on the public dataset at the end of the thesis, however the dataset is still not publicly available. As no other dataset, that provides a similar land use segmentation problem, is available, only the training set of the DeepGlobe dataset is acquired for this thesis. Due to the fact that the dataset is currently not publicly available, no qualitative results are published in this thesis.

The acquired training set contains 803 images of 2448×2448 pixels, which are split in four patches of size 608×608. As a training and validation set are required for training and a test set is required for evaluation, the original DeepGlobe training set is subdivided. The patches of width and height 608×608 are randomly added to either the training, validation or test set with a chance of $[0.8, 0.1, 0.1]$ respectively. By this random assignment of patches a split of roughly 80 percent for training, 10 percent for validation, and 10 percent for testing is achieved, whilst decreasing the chance of assigning whole images to either the training, validation or testing set.

Furthermore, the research questions *"What is the influence on the segmentation performance if the feature extractor of a segmentation network is exchanged for a state-of-the art feature feature extractor?"* is answered by the results of this experiment as well. Since both U-net and Dark U-net are based on the encoder-decoder architecture with skip connections, any difference in performance should be attributed to the different feature extractor architectures. It is expected that the Dark U-net architecture will improve the multi-class semantic segmentation task since it adds an extra set of convolutional layers and an additional maxpool layer which increases the network's receptive fields.

YOLO achieved significant speed ups in computational time by switching the VGG-16 feature extractor for Darknet-19 whilst losing very little performance in terms of object detections. U-net does not utilize the full VGG-16 feature extractor, as it contains only four maxpool operations, therefore already reducing the complexity of the original VGG-16. On the other hand, the encoder-decoder architecture results in a significant increase of complexity as the encoder is basically duplicated to serve as a basis for the decoder as well. Therefore it is expected that a smaller reduction in complexity when compared to the full VGG-16 feature extractor still becomes noticable as that smaller reduction is basically applied for the decoder part of the architecture as well.

Both a U-net and a Dark U-net architecture are constructed with a batch size of 4, a cross-entropy loss function and are trained on the training set via the ADAM optimizer with learning rate $\alpha = 1e^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e^{-8}$. Due to random initialization of the weights, it is assumed not fair to compare individual networks. Therefore the experiment is performed on five networks, where their performance metrics will be averaged over those five runs.

Additionally, the average Intersection over Union (IoU) is measured per class, as this is the benchmark metric for the DeepGlobe dataset. This metric shows how well the areas of the predicted pixels of a certain class overlap with the areas of the pixels of that same class in the ground-truth image. It measures the intersection of the predicted and ground-truth areas in terms of pixels and divides them by the union of their respective areas. In the per-pixel semantic segmentation task, this corresponds to the amount of True Positives (TP) divided by the sum of TP, False Positives (FP) and False Negatives (FN) per class. Therefore, an IoU for class j for all images $i \in \{1, 2, \ldots, n\}$, where n are the number of images as described in the DeepGlobe paper [22]. It is defined as:

$$\text{IoU}_j = \frac{\sum_i^n \text{TP}_{ij}}{\sum_i^n \text{TP}_{ij} + \sum_i^n \text{FP}_{ij} + \sum_i^n \text{FN}_{ij}} \tag{4.1}$$

The mean IoU is then the mean over the individual IoU's of all classes.

An early stopping condition is defined for the optimization scheme in order to prevent overfitting. The stopping condition takes into account the achieved mean IoU on the whole validation set, where the optimization algorithm is stopped if the best mean IoU is not improved over three consecutive validation epochs. To reduce the total training time, the validation scores are only calculated once for every five training epochs.

Data augmentation is applied on the aerial images to increase the variation in training samples as well as reduce the chance that the neural network learns to classify based on unrelated patterns. Take for example a dataset containing images of cars and trucks and a neural network is trained to distinguish cars from trucks. If by any chance, all cars are facing to the right and all trucks are facing to the left, the network could learn to classify based on vehicle orientation rather than the, for us obvious, visual differences in vehicle size and design. Augmenting these data samples by randomly flipping them, ensures that the network trains on examples of both left and right facing cars and trucks, thus avoiding classification based purely on vehicle orientation.

The previous example is an obvious visible pattern, which designers of networks should be able to notice fairly quickly. Other patterns, however, are not so easy to detect by humans especially on large datasets that contain tens of thousands of images. For this thesis, a set of commonly used random data augmentation operations is used to reduce the chance of severe overfitting of the neural networks. It is not claimed that this is the best combination of augmentation techniques for this particular domain, as this is considered to be outside the scope of the thesis.

An example of each of the individual augmentation techniques is displayed in figure 4.1. Subfigure 4.1 (a) shows the original, unmodified patch and its annotation.

A random crop of a sub-patch, figure 4.1 (b), is introduced to increase spatial variety in the images. Subsequently, by randomly picking from a variety of crop sizes for the sub-patch, the objects in the patch are displayed at different scales as well, potentially increasing the network's scale invariance. Since the network expects images of a fixed dimension during training time, the patches are up-scaled to match the dimensions of the original image. Another way to introduce more spatial variety in the image, is to perform a random rotation and / or a random flip of the input image, which is shown in figure 4.1 (c) and (d).

Variation in illumination conditions, introduced by applying random brightness, contrast and saturation, decreases the network's reliance on certain lighting conditions. Examples are shown in figure 4.1 (e-g). High-frequency features are often susceptible to noise, and are therefore less suitable for a robust classification network. The network's reliance on these high-frequent features is reduced by distorting these high-frequent features. For this reason, Gaussian noise is applied to the image, where σ is randomly chosen per input sample.

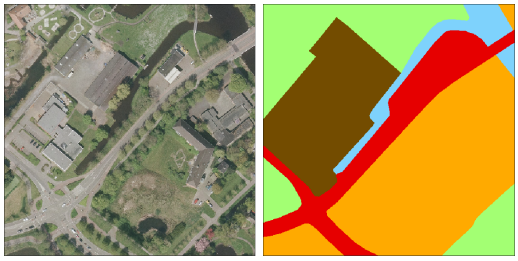
The pixel values could unfortunately exceed the bounds of [0,1] due to these augmentation steps. Therefore a clipping operation is performed, where values outside the boundaries are clipped to the boundary values.

4.1.1. Results

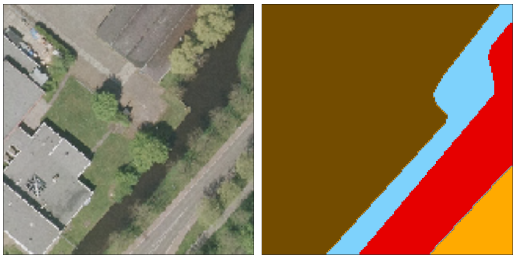
The DeepGlobe 2018 dataset is used as a benchmark to assess the potential of U-net and Dark U-net on a large multi-class dataset for the Remote Sensing domain. Figure 4.2 depicts the mean Intersection over Union of all land usage classes, except the "other" class, for consistence with the DeepGlobe paper [22]. For both U-net and Dark U-net, the mean Intersection over Union of the networks trained on augmented data are depicted. As five networks are trained for each architecture, the bars of U-net and Dark U-net actually depict the average mean Intersection over Union, whereas the black error bars indicate their standard deviation.

This graph indicates that both U-net and Dark U-net achieve a higher mean Intersection over Union than the DeepLab reference network. However, as the training, test and validation sets are not identical for the (Dark) U-net implementation and the DeepLab implementation, it is not fair to draw any conclusions on which architecture is better. Additionally, DeepLab is just trained for 30 epochs, where U-net and Dark U-net are trained for at most 305 epochs. These results do prove that both Dark U-net and U-net show potential for performing a segmentation task in the Remote Sensing domain.

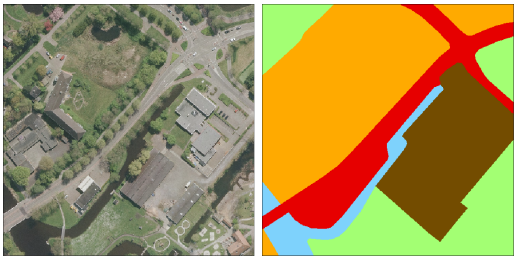
The impact of the data augmentation is measured on the test set of the DeepGlobe 2018 dataset, in terms of Intersection over Union per class. Figure 4.3 shows the results as a bar graph for the U-net architecture (a) and the Dark U-net architecture (b), where the blue bar indicates the measured



(a) The original aerial image patch (left) and corresponding annotation (right)

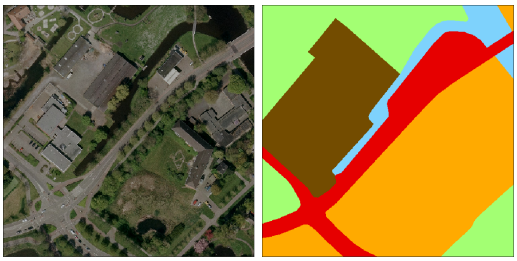


(b) Random crop and rescale.

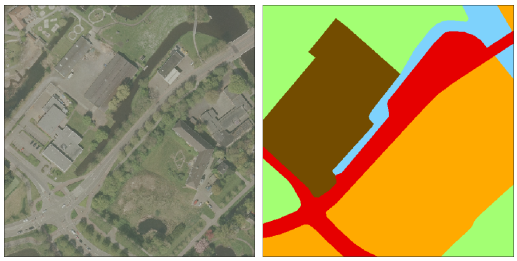


(c) The random rotation data augmentation operation.

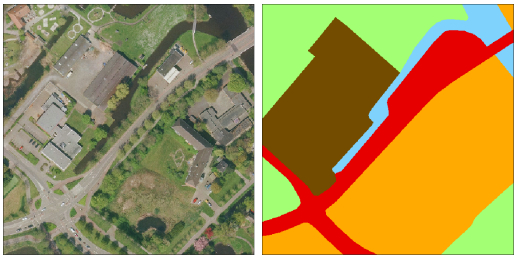
(d) The random flip, either horizontal or vertical.



(e) The random brightness.



(f) Random contrast.



(g) The random saturation data augmentation operation.

(h) Gaussian noise with random standard deviation.

Figure 4.1: Individual data augmentation steps performed on an original image patch and corresponding annotation.

Intersection over Union of the network that was trained without Data Augmentation. The red bar depicts the average performance of the five networks that were trained with Data Augmentation, where the black error bars indicate the standard deviation. The same results are listed numerically in table 4.1

When taking the variation between the five runs of the networks that are trained on augmented data into account, the overall performance in terms of IoU appears to have a slight advantage over the network that is trained without augmented data. When comparing the standard deviation on the mean IoU values of the networks trained on augmented data with the networks trained on not augmented data, depicted in the bottom row of table 4.1, it becomes apparent that the standard deviations on the networks trained on augmented data is significantly lower.

Additionally, the difference in learning curves between a network that is trained on augmented data versus a network with the same architecture trained on non-augmented data shows that training on augmented data reduces the chance of severe overfitting on the DeepGlobe 2018 dataset. Figure 4.4 shows the largest difference and smallest difference between training curves in subgraph (a) and (b) respectively. As this behaviour was apparent for both the U-net and Dark U-net network based

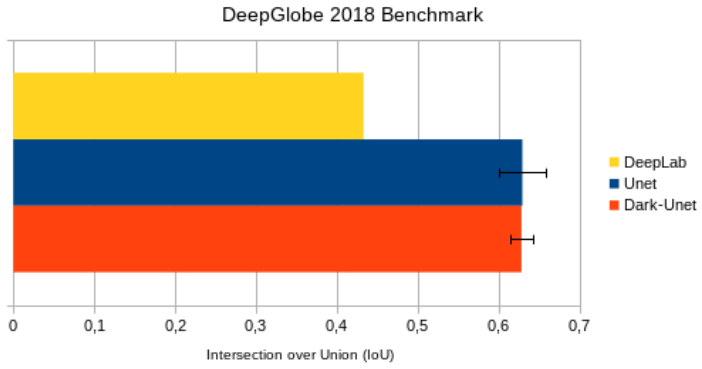


Figure 4.2: Average Intersection over Union of U-net, Dark U-net and DeepGlobe reference DeepLab

Table 4.1: Comparison of segmentation performance in terms of Intersection over Union per class for both U-net and Dark U-net when trained on the DeepGlobe dataset either with or without data augmentation. The results on augmented data are averaged over five runs and their standard deviations are added as well. The Mean results on augmented data are calculated over the average IoUs of each run.

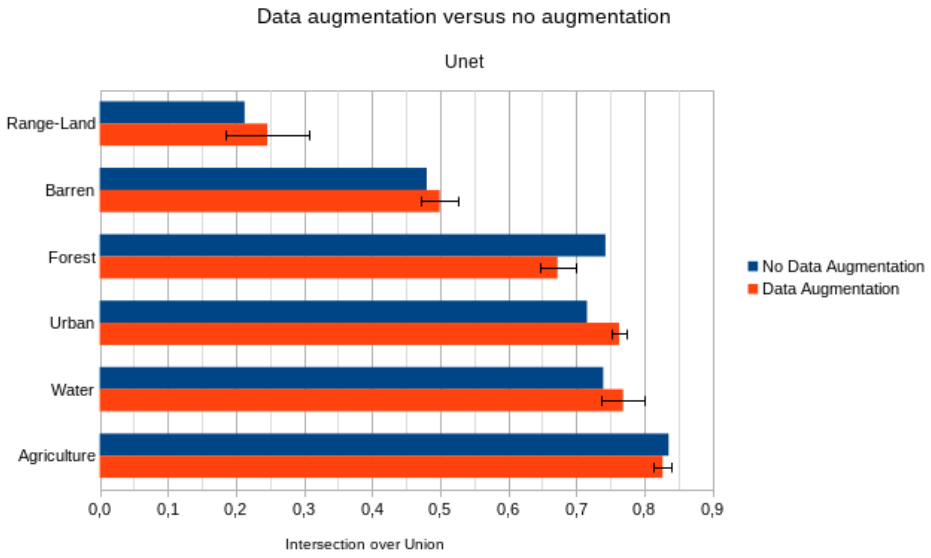
	U-net		*Dark U-net*	
Labels	Pristine Data	Augmented Data	Pristine Data	Augmented Data
Water	0.743	0.769 ± 0.032	0.686	0.777 ± 0.007
Forest	0.716	0.673 ± 0.026	0.694	0.679 ± 0.025
Range-Land	0.213	0.246 ± 0.060	0.192	0.239 ± 0.009
Agriculture	0.836	0.827 ± 0.013	0.819	0.823 ± 0.010
Urban	0.740	0.763 ± 0.011	0.744	0.766 ± 0.008
Barren	0.481	0.500 ± 0.027	0.464	0.520 ± 0.028
Mean	0.621	0.630 ± 0.026	0.600	0.628 ± 0.010

networks, therefore the network architecture is not named for this graph. The red lines indicate the learning curve on the training set of the network that is trained without data augmentation, whereas the orange lines indicate the learning curve on the training set of the network that is trained with data augmentation. Similarly, the light blue line shows the learning curve on the validation set of the network that is trained without data augmentation, whereas the dark blue line shows the learning curve on the validation set of the network that is trained on augmented data.

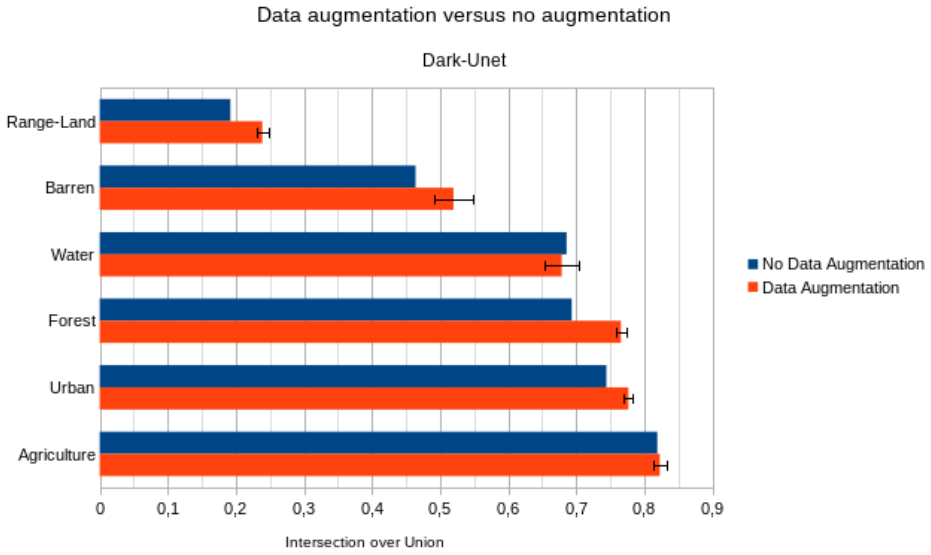
Figure 4.4(a) shows severe overfitting of the network without data augmentation on the training set, whereas the network that is trained on augmented data shows a more natural learning curve. Figure 4.4(b) shows similar signs of overfitting, althoug less severe than in graph (a). It is interesting to point out that, for the network that is trained on augmented data, the performance on the validation set is generally above the performance on the training set. This is possibly due to the rather invasive data augmentation techniques which are only applied to the training samples. As these techniques are not applied to the validation images, these images are potentially more pristine, thus making the segmentation task more clear on these images.

Figure 4.5 shows the per class IoU averaged over the five runs per network architecture, where their respective standard deviations are displayed as black error bars. Dark U-net and U-net appear to have similar performances, where some classes are slightly in favor of U-net and others are slightly in favour of Dark U-net. Figure 4.2 and table 4.1 show that, on average, Dark U-net performs similarly well as U-net as well, where the slight difference in performance is not deemed significant. Both figures show that the U-net networks shows a larger standard deviation in segmentation performances, both on average, and per class than the Dark U-net networks, which shows that Dark U-net is a more stable architecture for this dataset.

From figure 4.6(a) it is apparent that U-net generally converges towards a (local) minimum quicker than Dark U-net, as the average amount of epochs is significantly lower for U-net when compared with



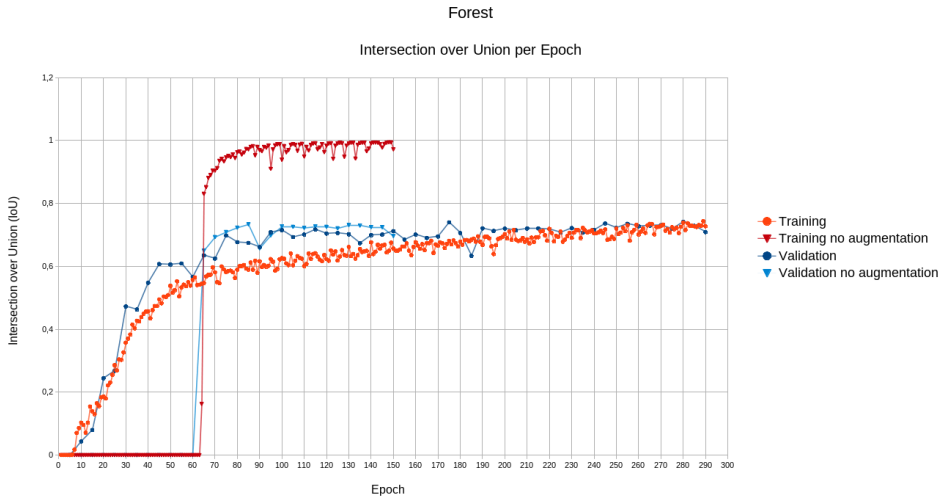
(a) U-net



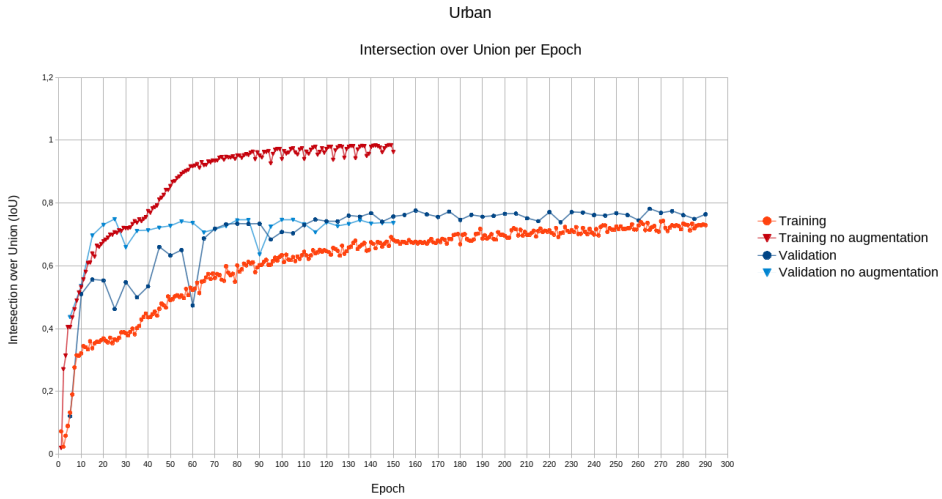
(b) Dark U-net

Figure 4.3: Comparison of both Dark U-net and U-net in terms of per class IoU with and without data augmentation on DeepGlobe test set

Dark U-net. On the other hand, figure (b) shows that Dark U-net trains quicker in terms of elapsed time when compared to U-net, even though Dark U-net has more trainable parameters than U-net. A possible explanation for this could be that Dark U-net has very few trainable parameters in the top few layers when compared to the deeper layers, whereas U-net has a more even distribution of parameters along all layers. When the loss is calculated, the filters contributing to the error are updated via backpropagation. However, the calculation of the gradients for backpropagation consume quite a lot of time. Since the top layers create general purpose feature maps, that deeper layers make use of to create more specialized features, the top layers contribute to many different filters in the deeper layers. Thus an error attributed to one deeper filter, propagates back to many filters in the top layers, leading to many parameter updates in the top layers. A network architecture with more trainable parameters in the top layers, could therefore train slower than an architecture that has more trainable parameters in total, whilst containing fewer parameters in the top layers, due to the amount of gradients that have



(a) Learning curve of Dark U-net with and without data augmentation for the class Forest.



(b) Learning curve of Dark U-net with and without data augmentation for the class Urban.

Figure 4.4: Train and validation learning curve for the classes Forest and Urban

to be calculated during backpropagation.

4.1.2. Discussion

As both the U-net architecture and Dark U-net architecture achieved better scores than the benchmark score of the segmentation network on the DeepGlobe dataset, it can be concluded that both the U-net and the Dark U-net architecture are able to perform a multi-class segmentation task that is competitive with modern day standards of automated image segmentation on aerial images.

Unfortunately the results of Unet and Dark U-net show a mean IoU of below 70 percent on the test set, which indicates that both these networks are not able to perform the intended segmentation task on new unseen data perfectly. This could however be partly explained by the quality of the supplied annotations, as well as the seemingly loosely defined classes. It is for instance rather complex to understand what the actual criteria is for a patch of land to be considered as belonging to the class ‘Range Land’. Arguably, most humans would not achieve a perfect score in terms of IoU when given the original aerial images either. As The benchmark semantic segmentation network presented in the DeepGlobe paper, did not achieve near perfect scores either, and since the DeepGlobe dataset was purely used to verify the hypothesis that U-net and Dark U-net are adequately complex to learn to perform a segmentation task on aerial images, the actual reason for why these networks did not obtain

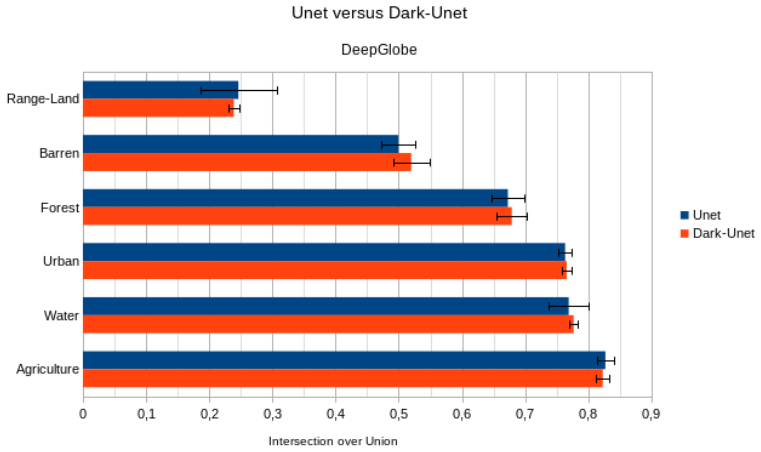
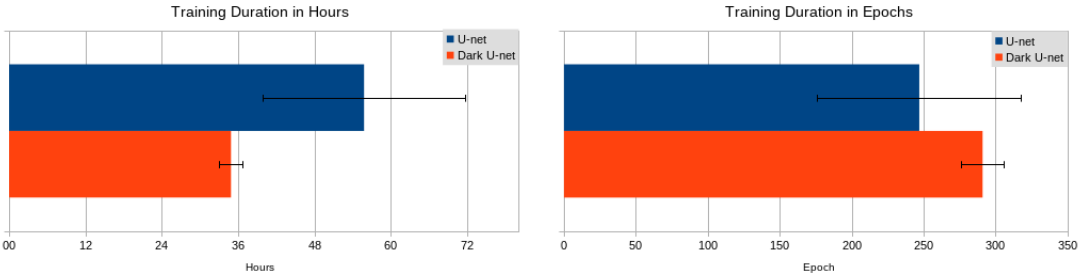
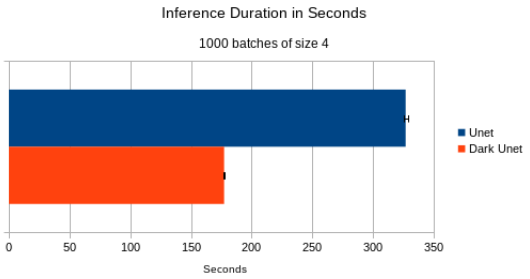


Figure 4.5: Comparison of U-net and Dark U-net, with data augmentation on DeepGlobe test set.



(a) Train duration of U-net and Dark U-net in terms of hours

(b) Train duration of U-net and Dark U-net in terms of epochs



(c) Inference duration in seconds of U-net and Dark U-net on 1000 batches, where each batch contains four images.

Figure 4.6: Training and inference duration measurements in terms of epochs and time of U-net and Dark U-net.

a perfect score was not pursued.

From the resuls on data augmentation one can conclude that the use of data augmentation results in a smaller performance gap between predictions on the training set and the validation set. This indicates that the network that is trained on the augmented data learned a more general representation of the segmentation than the network that was not trained on augmented data. Therefore the network that was trained on augmented data should be able to cope with new unseen data better than the network that was not trained on augmented data. This is also shown in figure 4.3, where the network that was trained on augmented data performed better overall in terms of IoU than the network that was not trained on augmented data. The class ‘Forest‘, did not seem to benefit from data augmentation and, in fact, was predicted less well by the network trained on augmented data. Both U-net and Dark U-net networks that were trained on augmented data showed a drop in performance on predicting regions that belong to the class forest. As forests, viewed from above, do not have any clear orientation or shape, it

is deemed unlikely that the random rotations had any influence on the drop of performance. Similarly, the brightness and saturation do not influence the distinct green and brown hues that characterize a forest either. Possibly, the random crop and scale operations could distort the patches in such a way that the network is unable to learn a good representation. The individual trees, branches or leaves could be magnified extremely by this operation, which potentially results in an unnatural representation of a forest.

When comparing the segmentation performance of U-net with those of Dark U-net, it is apparent that they perform similarly in terms of IoU. However, there are indications that the Dark U-net network is more stable than the U-net architecture, as after training several networks independently, the Dark U-net networks seem to show a lower variance between the individual predictions compared to the U-net networks. Aside from that, if training time is valuable, Dark U-net seems to learn the segmentation task quicker than the U-net network despite generally needing more epochs to converge.

4.2. Influence of binary versus multi-class segmentation

This experiment aims to find an answer to the research question: *"What is the impact on the segmentation performance of a state-of-the-art segmentation network when moving from binary to a multi-class segmentation problem?"*. As a per-pixel semantic segmentation network predicts an individual class confidence score for each label at each spatial position, it is expected that the segmentation performance of class k is not influenced when additional classes are added to the segmentation task, assuming that these classes do not change the annotations of class k.

In order to measure the impact of switching from a binary classification task towards a multi-class classification task, thus increasing the range $\{2, 3, ..., n\}$ with $n > 2$, five datasets were created. These datasets all contained the same aerial images, however the class annotations were edited. The first dataset merged all classes except the class road into one class, "other", to create a binary annotation. The second dataset merged all but three classes, to create a four class annotation. Similarly, a six and eight class annotation was created, whereas the last dataset kept all the original classes for a thirteen class annotation. Table 4.2 shows an overview of which classes were included in which dataset.

For this experiment, the semantic land use dataset introduced by Rijkswaterstaat is used. a set of 24662 image patches with a resolution of 480×480 pixels were divided in a train, validation and test set according to a random $[0.8, 0.1, 0.1]$ split respectively. A separate Dark U-net network is trained per dataset, of which its performance is measured in IoU per class. These networks are configured in the same way as in the previous experiment except for the batch size which is increased to six. Additionally, the data augmentation and early stopping criterea from the previous section are used for this experiment as well. Since the class road is included in all datasets, the performance of *Road* classification in terms of IoU can be compared across the five networks. Similarly, the classification performance for classes *Water* and *Residential area* can be evaluated across the networks trained on CBS 4 to CBS 13, whereas the performance for classes *Forest* and *Industrial area* can be compared for the networks trained on CBS 6, 8 and 13.

One important factor that could potentially influence this experiment is class imbalance. Especially in the binary annotation, the amount of pixels that belong to the class *Road* are either zero, or a small fraction of the total amount of pixels in the image. Therefore, a huge imbalance is introduced where, generally, over 90% of the pixels belong to the class *Other* in an image. The influence of the class imbalance gradually decreases as more classes are introduced, and the dominance of the *Other* class disappears, however the original class imbalance of the original dataset will always remain. Semantic segmentation datasets, especially in the domain of Remote Sensing, are hardly ever balanced, due to geographical properties. The Netherlands is a, mostly flat, country with many waterbodies and areas dedicated to agriculture, hence pixels belonging to *Water* or *Agriculture* are in abundence, whereas there are only a few, and relatively small, Airports resulting in very few samples of pixels belonging to the class *Airport*.

Several solutions for solving class imbalances exist in literature, where simply resampling the dataset such that all classes are equally represented is one of the easiest solutions. However, this solution would drastically reduce the amount of data samples in this case, as it could only sample as many pixels from every class, as the amount of pixels belonging to the least well represented class.

For this experiment, a class balancing scheme is utilized such that each occurring class in an image contributes to the loss evenly. A weighted loss is defined where all pixels in a batch belonging to a class

	Dataset				
Class	CBS 2	CBS 4	CBS 6	CBS 8	CBS 13
Road	×	×	×	×	×
Water		×	×	×	×
Residential area		×	×	×	×
Agriculture			×	×	×
Forest			×	×	×
Industrial area				×	×
Open and Dry area				×	×
Railroad					×
Airport					×
Other terrain					×
Recreational					×
Greenhouse					×
Open wet area					×
Other	×	×	×	×	

Table 4.2: Included classes per dataset.

i with $i \in \{1, 2, \ldots, m\}$ are weighted by $\frac{1}{n_i}$, where m represents the number of classes. n_i represents the number of pixels present in the batch that belong to class i.

4.2.1. Results

This section contains the quantitative and qualitative results of the impact of the amount of available classes on the segmentation performance of a Dark U-net network. These results were attained on the publicly available aerial image dataset provided by Rijkswaterstaat.

Figure 4.7 shows the results of the performance in terms of IoU of five instances of a Dark U-net network, each trained on the same aerial data, however on different annotations. These annotations varied from binary class labels to a multitude of multi-class annotations of different amounts of classes. For each subplot, the bars show no significant difference in IoU when increasing the amount of classes of the multi-class segmentation problems. However, a large difference is noticeable for the binary class segmentation problem. The class 'Road' has an IoU of 0.0 for the binary class segmentation problem, whereas the IoU for all the other multi-class problems containing 'Road' showed an IoU above 0.3 as displayed in table 4.3. This is caused by the class imbalance in the dataset, which is especially evident in the binary class problem.

Therefore, a weighted loss function is introduced to counter for this class imbalance, for which the results are displayed in Figure 4.8 and table 4.4. The problem, where the class 'Road' is not detected in the binary segmentation problem is solved as it is in line with the rest of IoU's for the multi-class problems. On the other hand, the weighted loss function introduced a larger variance between different networks, as the difference in IoU's per class between multi-class annotations is increased.

	Dataset					
Class	CBS 2	CBS 4	CBS 6	CBS 8	CBS 13	Mean ±std
Road	0.000	0.337	0.331	0.342	0.348	0.272 ± 0.152
Water	N/A	0.747	0.739	0.746	0.752	0.746 ± 0.001
Residential area	N/A	0.682	0.673	0.684	0.682	0.680 ± 0.005

Table 4.3: Intersection over Union of Dark U-net trained without class weights on a binary segmentation task (CBS2), and several multi-class segmentation tasks with 4,6,8 and 13 labels in CBS 4, CBS 6, CBS 8 and CBS 13 respectively.

Figure 4.9 depicts several examples of Dark U-net's predicted segmentations on the test set. For each example, the left most image depicts the aerial image fed to the network, the middle image its prediction, and the right image the annotation assumed to be the ground-truth. The network has not

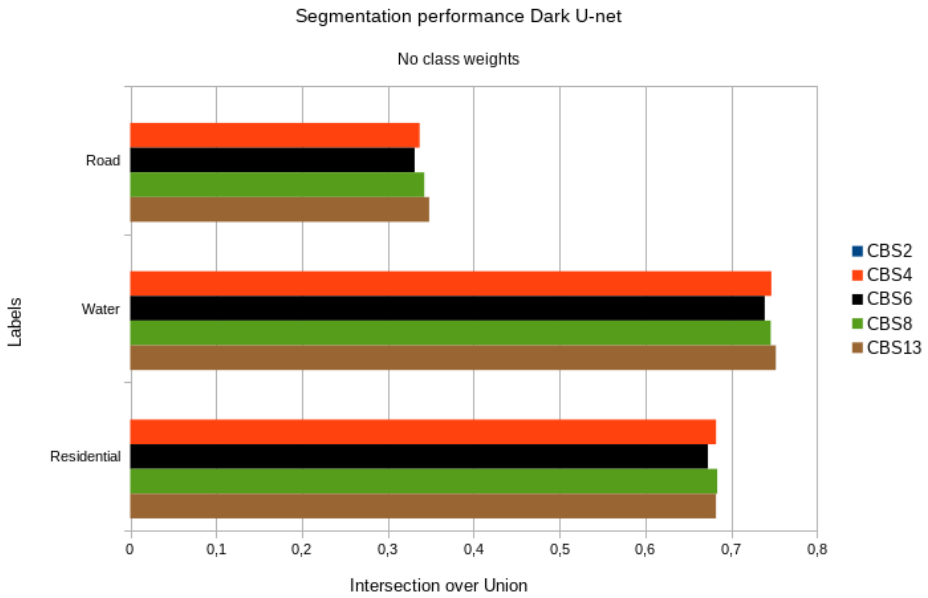


Figure 4.7: Comparison of Intersection over Union scores of Dark U-net network on various classes, on binary or a multitude of multi-class annotations. No class weights are used in the loss function.

	Dataset					
Class	CBS 2	CBS 4	CBS 6	CBS 8	CBS 13	Mean ±std
Road	0.351	0.350	0.356	0.345	0.334	0.347 ± 0.009
Water	N/A	0.712	0.711	0.728	0.702	0.713 ± 0.011
Residential area	N/A	0.644	0.647	0.651	0.619	0.641 ± 0.014

Table 4.4: Intersection over Union of Dark U-net trained with class weights on a binary segmentation task (CBS2), and several multi-class segmentation tasks with 4,6,8 and 13 labels in CBS 4, CBS 6, CBS 8 and CBS 13 respectively.

seen the annotations for these examples, and are purely added for visual comparison.

Two examples are shown per trained network. Figure 4.9 (a) shows the results trained on the binary segmentation set, whereas (b) depicts results on the 4-class segmentation set. subfigures (c), (d) and (e) show results on the 6-class, 8-class and full 13-class segmentation sets respectively.

When comparing the results vertically, it is clearly visible that all networks learned to predict roads in a similar way, although not exactly similar. The same can be said for the residential areas depicted in orange, although the residential areas on the left side of the right examples show more variation. When comparing the predictions of the class 'Road', it is noticeable that the network trained on the binary segmentation task only predicted one large road section in the right example, whereas the networks trained on the multi-class segmentation task did predict the second road section on the left of the water body as well.

4.2.2. Discussion

From this experiment it is concluded that class imbalance is potentially a serious issue for segmentation networks. This becomes apparent in the binary segmentation task, where an image is subdivided in regions belonging either to the road class or other. Since roads occupy a relatively small area of a total image, the misclassification of "road" pixels as "other" contributes far less to the overall loss function than the misclassifying pixels belonging to "other" as "road" pixels if each pixel contributes to the loss equally. This results in a network that simply classifies all pixels as belonging to class other due to the optimization of the loss function's convergence to a local minimum, as mistakes of misclassifying road pixels are hardly punished by the loss function. By leveling the contribution of misclassifications between all classes to the total loss, the network is forced to take the minority class into account as

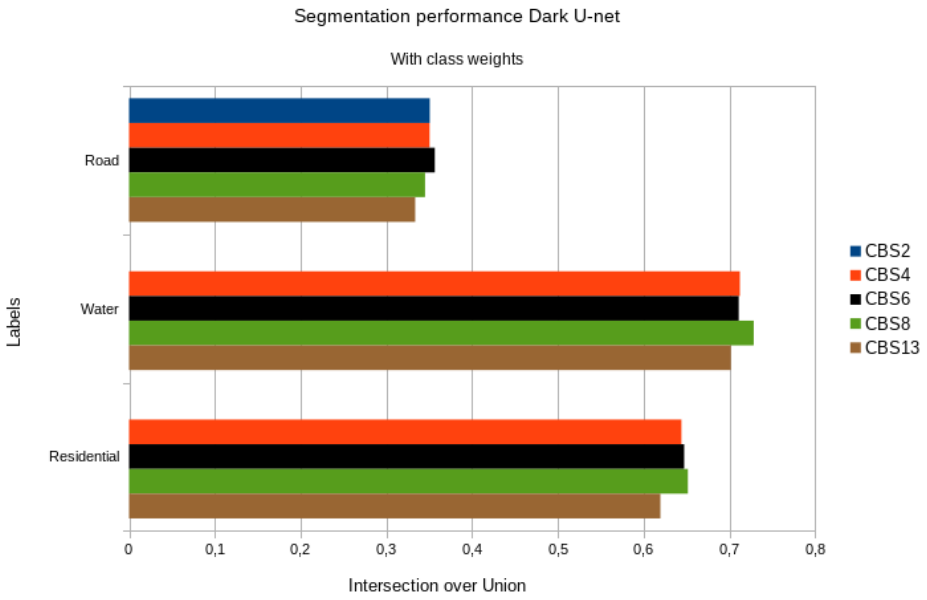


Figure 4.8: Comparison of Intersection over Union scores of Dark U-net network on various classes, on binary or a multitude of multi-class annotations. Class weights are used in the loss function

well as it gets punished just as much for misclassifying the pixels belonging to the major class as ones of the minor class as it gets punished for misclassyfing pixels belonging to the minor class as ones belonging to the major class.

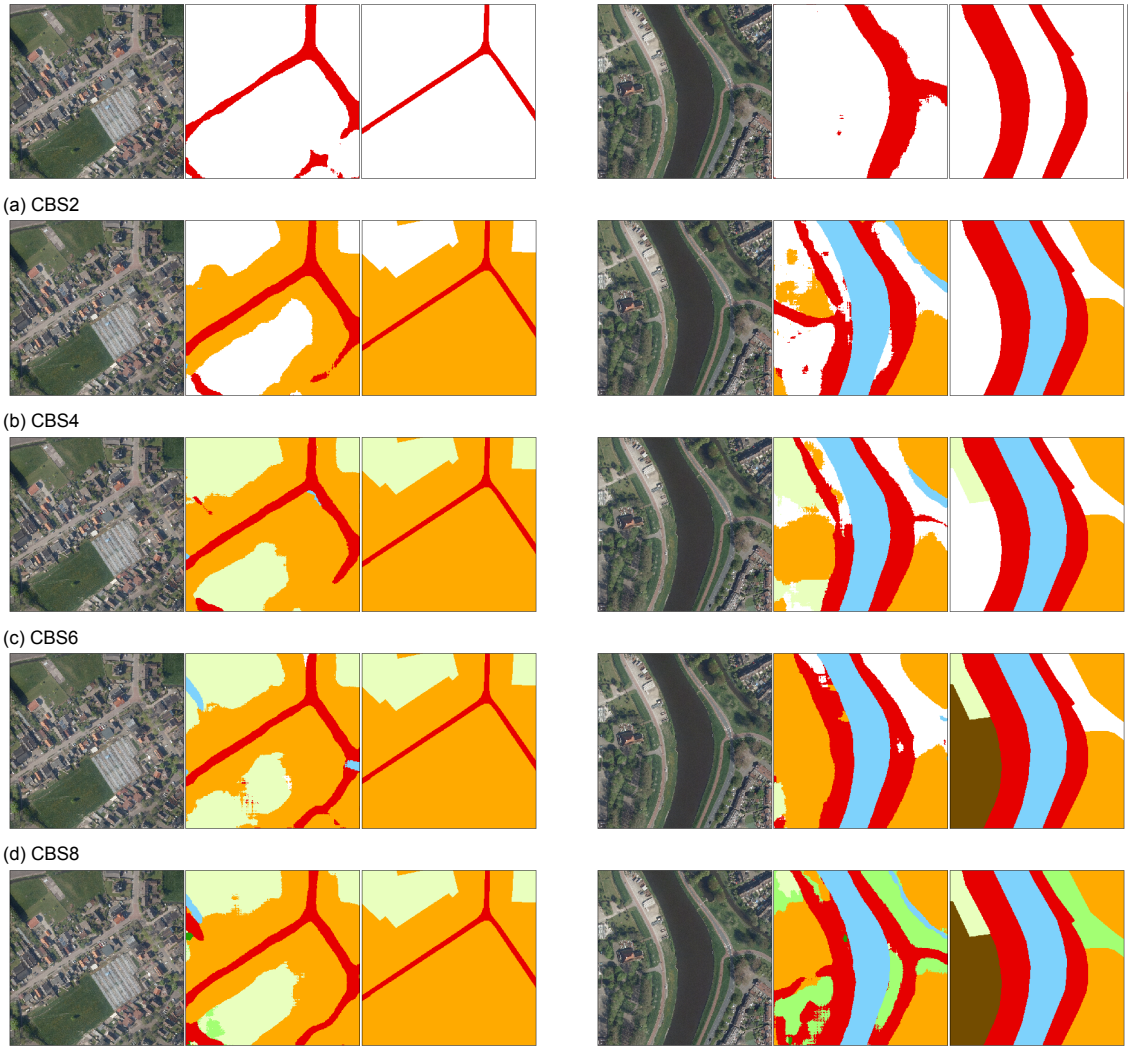
Assuming that all class labels are independent of each other, the information on which the network learns to discriminate between pixels that either belong to a certain class or do not belong to that class stays the same. Combining this with the network's inherent ability to predict a confidence score per individual class, it should be able to achieve the same confidence score, for example of class road, when trained on the same data, regardless of how many other classes are available in the dataset. Based on the results depicted in this chapter, this hypothesis is confirmed as it shows that a difference in number of classes does not influence a network's discriminative ability, meaning that there is no significant difference in segmenting roads in a binary segmentation setting in comparison with segmenting roads in a multi-class setting.

The small deviations in performance between the binary and multiple multi-class segmentation tasks can potentially be addressed to the random instantiation of each network's trainable parameters. As the previous chapter showed, training one particular network architecture several times on the same data, results in a small variance in segmentation performance. The variation between the segmentation performance shown in figure 4.8 are relatively small and show no correlation with the amount of classes included in the segmentation task. These variations are therefore deemed more likely to be caused by the random initialization as shown in the previous chapter, rather than due to the difference in the number of available classes of the segmentation task.

4.3. Road Segment Localization

This experiment aims to find an answer to the research question: *"How can an arbitrarily complex road structure be detected by locating a series of simplistic shapes such that the original topological properties are preserved?"*. Modern object detection networks perform shape regression on simplistic shapes, such as rectangles, to detect objects in images. By modifying an object detection network to just a localization network as described in section 3.3, and subsequently training it on samples from the dataset as described in section 3.1 with generated road localization annotations as described in section 3.3.2, the localization network is expected to learn the proposed road segments representation of complex road structures.

The following experiment is performed to measure an object detector's capability to locate road



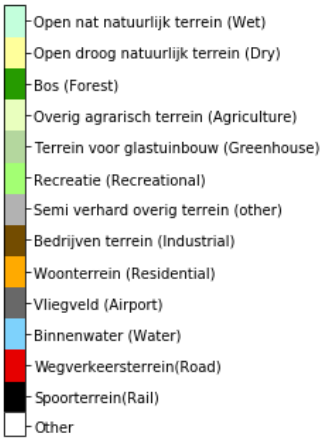
(a) CBS2

(b) CBS4

(c) CBS6

(d) CBS8

(e) CBS13



(f) Legend

Figure 4.9: Examples of segmented images from the test set on a binary or a multitude of multi-class annotations. Two example images are shown in the columns, where each row indicates the same sample from the CBS 2,4,6,8 and 13 dataset respectively. Each example consists of an aerial image, the predicted segmentation and the annotated segmentation respectively. The color white is used for pixels that belong to the class “other” due to the merging of several classes in order to create the CBS 2,4,6 or 8 datasets. The model is trained with a class weighted loss.

	Metrics		
Labels	IoU	Precision	Recall
Road localization	0.626	0.760	0.781
Other	0.944	0.973	0.969

Table 4.5: Intersection over Union, Precision and Recall of rasterized road segment localization boxes.

segments in an image. As this is only a part of the algorithm that should learn to detect roads by performing shape regression, it is crucial that this algorithm learns a representation of roads that allows for shape regression in latter stages. Since roads could be localized in images in various different ways, which might not all suit the goal of segmenting roads by shape regression, this localization task is defined as a supervised learning task. This ensures control about the representation of roads that the algorithm attempts to learn.

An object detection network, inspired by YOLO, is used to learn the road segment localization task. Originally, YOLO localizes objects and assigns them to a certain class. The classification part of the network is discarded, as all detections belong to class road.

The network is trained on patches with a width and height of 480×480 pixels that all contain road sections. They are sampled from the original dataset with aerial images and corresponding annotations of regions of the Netherlands, taken in 2012. This dataset with aerial images and annotations is subsequently randomly split in a train, test and validation set with ratios $[0.8, 0.1, 0.1]$ respectively. Since this architecture is more memory efficient than the semantic segmentations networks from earlier experiments, the batch size could be increased to a size of 32 samples. Thanks to that, a train set consisting of 20.000 images is trained in just a couple of hours. The validation and test set each consist of 5120 samples, so that they are dividable by 32 as well.

The network is then trained on the train set with the adapted YOLO v2 loss functions as depicted in equation 4.2, for which the ADAM Optimizer is used with the same parametrization as in the previous experiments. Additionally the early stopping condition is also used for this experiment to prevent overfitting, where the stopping criteria is based on the YOLO loss instead of the IoU of the previous experiments. Non-maximum suppression is used on the predicted localization boxes, where a simple threshold $\tau = 0.25$ is introduced. All boxes with a box confidence score lower than τ are discarded. This type of non-maximum suppression is rather simplistic and the value of τ is not actively researched due to time constraints.

$$\begin{aligned} Loss_{adapted} =& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ &+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\ &+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \end{aligned} \tag{4.2}$$

The validation loss is calculated once every 3 epochs of training on the train set. The network is trained until the best validation loss has not been improved for 3 consecutive validation iterations. The model is then run on the test set, for which the loss is measured as well.

4.3.1. Results

Figure 4.10 displays road segment localization by predicting boxes in red, whereas underlying ground truth annotations are displayed in green. The numbers next to the class indication “Road” and the separator symbol “-” shows the box prediction confidence, which is a measure for the localization confidence. It is a number in range $[0.0, 1.0]$. However, no confidence lower than 0.25 is generally showed, as it predicted boxes with a lower confidence score than 0.25 are filtered out. This is a form of

non-maximum suppression which ensures that localizations with a low confidence are not considered as valid predictions as they are unlikely to be a good candidate for the localization of a road segment.

Table 4.5 depicts the IoU, precision and recall for the road localization predictions on the test set. These measurements are taken on the rasterized output of the network and are thus performed per-pixel. Precision and recall for class j for image i with $i \in \{1, 2, \dots, n\}$ are then defined as:

$$\begin{aligned} \text{Precision}_j &= \frac{\sum_i^n \text{TP}_{ij}}{\sum_i^n \text{TP}_{ij} + \sum_i^n \text{FP}_{ij}} \\ \text{Recall}_j &= \frac{\sum_i^n \text{TP}_{ij}}{\sum_i^n \text{TP}_{ij} + \sum_i^n \text{FN}_{ij}} \end{aligned} \tag{4.3}$$

The figure shows that most predictions in red are similar if not equal to their annotated counterparts, where the predictions are generally a bit larger than the annotations. In the bottom two images, there is more than one prediction for the annotated road segment localization box, which shows that the currently used non-maximum suppression scheme is not strict enough as of yet.

In general the predicted boxes that show high similarity with their corresponding annotations also have a high box confidence score. An exception however, is the image on the bottom right, where box confidence scores are lower than 0.5 whilst they are still similar to their annotations. Moreover, there is a false positive in that same image in the top left corner with a rather high confidence above 0.8. It is up for debate whether the false positive is indeed a false positive, or that the annotations are not correct here. Contrarily it could be stated that, if the network indeed believes that the false positive is an actual road segment, the road segment below it should have been localized as well. Arguably that is considered to be more consistent and more logical as well, as road sections are generally connected to other road segments.

Aside from that, the road segment that is falsely predicted is a concave segment, which even violates the earlier assumption of exclusively localizing convex road segments. All annotated concave shapes were removed form the dataset. On the other hand, there were many examples of road like patches, which were not annotated as roads by CBS. Therefore non-convex road-like patches are still in the dataset.

Many false positives occurred in regions where clearly visible road like structures are indeed present, as is shown by figure 4.11. Aside from that, the network sometimes fails to detect road segments, where there is an actual annotated road segment. A few examples of these false negatives are depicted in the top left and right images of figure 4.11. The top left image is a rather peculiar example of this, as the network “falsely” predicts a road segment in the top right of that image, whilst failing to predict a seemingly similar road segment at the bottom of the image. In the top right image of figure 4.11, the network fails to detect a partly obfuscated road segment, whilst falsely detecting a road segment in the top left and bottom right of that same images.

Not all false positives are patches that actually look like road segments, as depicted in figure 4.12. Occasionally patches that contain straight lined edges, such as a line of trees, small canals, or even a line of crops, are falsely detected as road segments. These false positives still have high box confidence scores, sometimes exceeding 0.9, and can thus not be prevented by simply increasing the threshold of the non-maximum suppression step. It is noticeable that these false positives generally follow a strict pattern. They all seem to represent structures that are visually distinct from its surroundings and are shaped as a straight line, which to be fair, is a characteristic of a convex road segment. It is however, definitely not the full description of a road segment, as road segments are generally grey, could contain markers and might have cars on top of them.

4.3.2. Discussion

Based on these results, it is shown that a localization network is able to learn a predefined representation of road segments on aerial images through regression on the parameters of rectangular boxes. The visual representation shows that the network predicts boxes that show high similarity to their annotated counterparts on many occasions.

The results are far from perfect though, as the images in figures 4.11 and 4.12 suggest. This is partly due to the quality of the annotations, where sometimes road like structures that even people would directly recognize as roads are not annotated as road. In some cases this is understandable, as roads in a residential area are part of the residential area and are thus annotated as residential

Figure 4.10: Aerial images with predicted localizations of road segments and annotated localizations. Red boxes are predictions, whereas annotated boxes are displayed in green. The values near each box display the confidence of the prediction of the box, which 1 for all annotated boxes.

area rather than road. In other cases the choices made by CBS seem less obvious to the human eye. Aside from the sometimes confusing looking annotation choices made by CBS, the automatic annotation generation method that provides the annotation of individual road segments by subdividing road polygons is far from perfect either. Figure 4.13 shows an example of a seemingly odd subdivision of a road in smaller segments. As the road polygon is located near the borders of the grid cells, the road polygon is even split horizontally, as can be seen in the right most picture where each segment is depicted as a box with a distinct color. In such cases it would probably make more sense to merge the horizontally adjacent boxes. Unfortunately there are more corner cases, where for instance very small portions of road polygons cross a certain cell, resulting in a very small road section that sticks out from the rest of the sections. Fixing most of these corner cases is not impossible, although it would take time to tackle and implement them leading to a more complex annotation method altogether.

Aside from these problems, one structural problem becomes apparent as well. This network localizes road segments on a local scale, which does not guarantee inter-segment connections. Therefore the predicted boxes are, unlike the annotated boxes, not always connected to each other perfectly. This means that sometimes they either slightly overlap each other, or that they are disjointed.

Figure 4.11: False positives and false negatives of road-like segments

That particular trait is undesirable, as road segments belonging to the same road polygon should be connected to each other. In this case one could argue that an overlap is less bad than segment disjointness, as the intersection points could be seen as the true beginning and end points of neighbouring road sections. Disjointed segments on the other hand are more problematic, as simply connecting them to the nearest road segment could lead to wrong attachments, of which the chances get higher the further the segments are apart. This indicates that any attempt at correcting overlap or disjoint segments on a local scale does not necessarily guarantee a better approximation of the whole road polygon, as the information of which segment should attach to which other segment, is not available on this scale.

As this network lacks any constraints on global coherence of road segments, and instead focuses on localizing individual segments on a local scale, it is suggested to utilize this network solely for localizing road segment candidates on a local level. An extension can be devised to take into account the global coherence of the candidate localizations of road segments, however in the end one is only interested in reconstructing the final road polygon, rather than localizing all segments with boxes that cover the whole polygon. Therefore the discussion on the global coherence of the prediction is continued in the next experiment where, aside from road segment localization, the shape and orientation of the

Figure 4.12: False positives of non-road-like segments

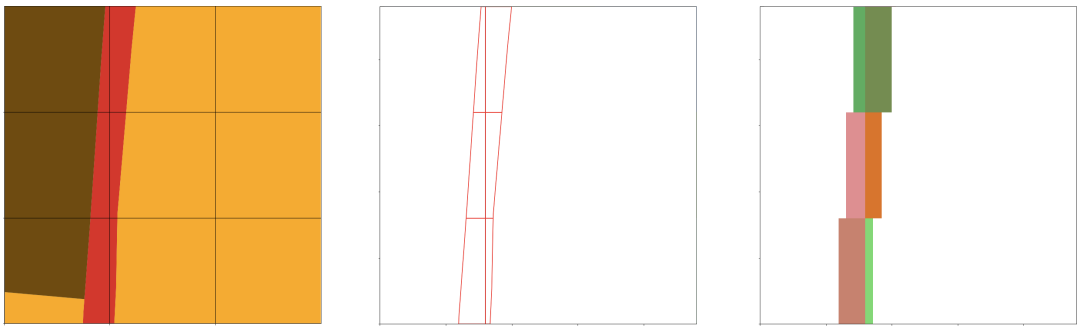


Figure 4.13: Illogical subdivision of road segment

individual road segments shapes are approximated as well.

4.4. Road Segmentation

The last experiment seeks an answer to the research question: *"How can a segmentation of a variety of pre-defined simplistic shapes segment complex road structures, whilst preserving the original topology"?*.

Where the previous experiment focusses on the localization of road segments by encapsulating them with boxes, this experiment attempts to approximate the actual shape of the road segment within each box. Under the assumption that all the localized segments contain (near) convex road segments, one can approximate the shape of a road segment by defining a line between the two points where the road segment hits its encapsulating box, where the width of the line is defined by the two boundaries of the road segment.

A neural network is expected to learn to represent road polygons by performing regression on the parametrization of each individual road segment's shape. Where the parametrization of each road segment line, which is encoded as two pairs of (x, y) coordinates accompanied by a road width parameter w. The road localization network, that is based on YOLO v2, is further expanded to predict the road segment parameters as well, as described in section 3.3.2.

In order to assess the quality of the road polygon approximations, the predicted segment lines are rasterized to achive a per-pixel binary segmentation of road and not-road. The intersection over union is then measured to provide a quantitative score of the per-pixel road segmentation, which is to be compared to the same score of the Dark U-net per-pixel binary segmentation network. This Dark U-net network is trained on the same data as the shape regression network with the exception that rasterized versions of the original road polygons are used as annotations. As the results described in section 4.2 showed no significant difference between predicting roads in a binary segmentation setting and road predictions in a multi-class segmentation setting, the baseline for this experiment is performed on the binary setting for easier visual comparison between the results of the semantic segmentation network and the shape regression network. The Dark U-net network requires per-pixel annotations and since it does not depend on the subdivision of road polygons, the original road polygons are used as this experiment aims to assess the performance of the shape regression network against the standard segmentation network.

The standard Dark U-net network architecture, trained on the same dataset as the one used in the road localization experiment, to solve the binary road / non-road segmentation of the data. It is trained with a categorical cross entropy loss, where convergence is declared if the intersection over union on the validation set does not improve in five consecutive validation runs. The ADAM optimizer is used with the same parameters as the previous experiments.

The shape regression network for road segmentation is trained on the same train set, after which its predicted line segments on the test set are rasterized and evaluated via a per-pixel IoU metric in order for comparison with the Dark U-net reference network. The loss is the adapted YOLO loss as described at the end of section 3.3.2. It is optimized with the same ADAM optimizer, using the same parameters as the networks in the previous experiments, where the optimization stops if the best loss is not improved for five consecutive validation epochs.

4.4.1. Results

For comparison with Segmentation networks, the IoU with the raster annotations is calculated on the rasterized output of the shape regression network and on the output of the Dark U-net network. The results are displayed in table 4.6.

There is a clear difference between the IoU scores of the Shape Regression network and Dark U-net network on the Road class. Dark U-net achieves an IoU score more than twice as high as the Shape Regression network. The difference in IoU of the Other class is rather small, leading to a smaller gap between both network's mean IoU scores. As the goal of this experiment is to measure the road segmentation performance of both networks, only the IoU scores on the road class should be compared, which is highly in favor of the Dark U-net semantic segmentation network.

Table 4.6: Binary segmentation performance in terms of Intersection over Union per network type per class.

	IoU		
Network	Road	Other	Mean
Shape Regression	0.319	0.954	0.637
Dark U-net	**0.787**	**0.975**	**0.787**

Table 4.7: Road segment box localization performance in terms of Intersection over Union, Precision and Recall.

	Road Localization Boxes		
Network	IoU	Precision	Recall
Shape Regression	0.581	0.672	**0.812**
Localization	**0.626**	**0.760**	0.7806

In order to visually assess why the Shape Regression networks performs significantly worse than the semantic segmentation network in terms of IoU, several shape regression predections are to be inspected visually.

Figure 4.14 shows a few examples of an aerial image with ground truth annotation boxes shown in green and predicted boxes and road line estimations in red. These images show that, in general, the box localization parts are fairly similar to the annotated boxes.

An exception of that is visible in the top left image, where for the beginning and end of the road section, the network opted for a larger segment localization box, whereas the annotations split that section in two section both containing one relatively large segment and one small segment. Although the network did differ from the annotation, the road segment prediction does not seem to suffer much from its decision, with a rather accurate estimation of the road's direction. Another observation is that, even though the roads in these images seem to have a rather constant width, the predicted width per segment is less consistent.

The segments do hardly ever seem to be connected either which is, as mentioned earlier, an undesired trait of this network. Sometimes the road line predictions even seem to start and end in the middle of the localization box, as is depicted in the top most segment of the bottom-left image of figure 4.14. Unfortunately, the network has too much freedom to predict the start and end points of the road line, which makes these predictions possible.

As the IoU measurements of the Shape Regression network indicates, many road line estimates were actually not as good as figure 4.14 shows. Many examples were more in line with the bottom most line estimation of the bottom-right image of that figure. Figure 4.15 shows many false positives and false negatives, and overall quite terrible road line estimates. The results in this image show a worse result, when compared to the false positives and negatives of the localization network described in the previous chapter. This shows that the combined task of road segment localization and shape estimation as a line consisting of two points and a width is a much harder task than solely localizing road segments in an image as even the road segment localization seems to be worse when compared to the results of the previous network.

Aside from that the road direction estimation seems to be inconsistent, as they can either be near perfect or almost perpendicular.

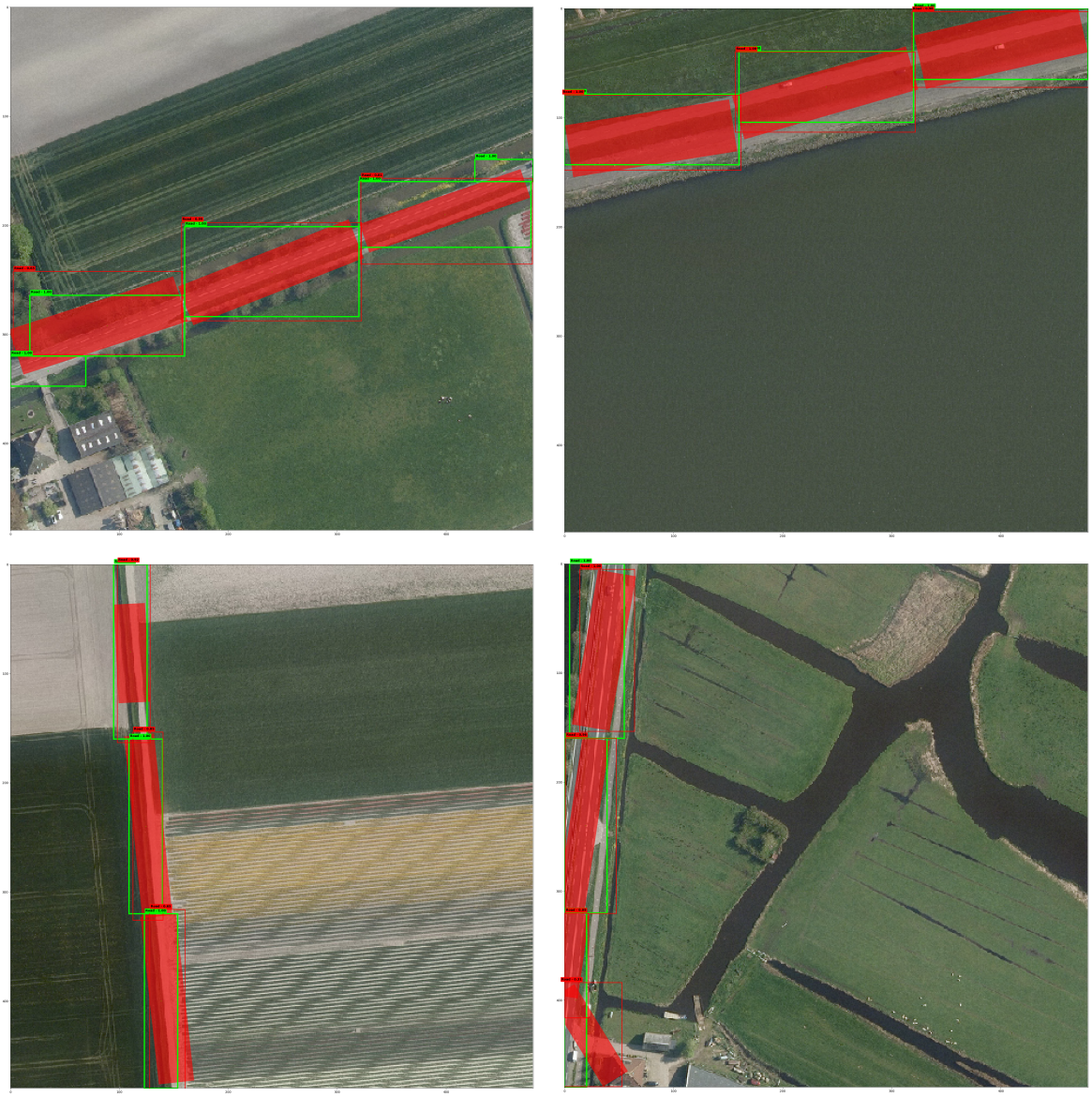


Figure 4.14: Aerial images with predicted localizations of road segments and annotated localizations combined with road segment line estimations in red and annotated localization boxes in green.

Similarly to the previous experiment, the network seems to generalize in surprising situations at times, of which examples are depicted in figure 4.16. At times, the network arguably predicts the road polygon better than the provided annotations, as is visible in the top-left picture. The annotation only provided the middle section of a road polygon, whereas the network did in fact predict the whole road. In the top-right picture, it fails to predict the bottom-right most section. However, it does predict a road-like structure on the left hand side of that image. Although, to be fair, it is understandable that the annotations do not include these kinds of roads, as they seem to be part of an area of agriculture. The same holds for the example in the bottom-left image of figure 4.16, where the network wrongly detects road segments on small canals.

4.4.2. Discussion

Initial limitations of this approach is that road polygons that consist solely of convex shapes can be approximated, which is only a small subset of all possible road polygons. As this is a novel method, the initial scope is set to just this simplified case in order to assess the method's potential. Once proven successful, this method allows the required freedom to expand to representations of more complex

Figure 4.15: A selection of various failure cases of predicted localizations of road segments and annotated localizations combined with road segment line estimations.

shapes, such as corners and roundabouts, by adding parameters of for instance additional points inside the localized box to estimate a corner via spline fitting, or a point and a radius for predicting a circle in case of a roundabout.

Another limitation is the global coherence of the segments that is mentioned in the chapter on road segment localization as well. By increasing the freedom of the network to predict the segments actual entry and exit points on the box, and the roads width per individual segment, the global coherence of the segments is disrupted even further. Since the prediction of entry and exit points of the road contains too much freedom, the points can actually be predicted inside the box, leading to further disjointness if the localization boxes are already disjoint. Additionally, even if the localization boxes were perfectly joined, the individual angles of the predicted road lines can still cause the predicted segments to be disjoint. Therefore the need for a method that aims to achieve global coherence is of increased essence for this method. However, due to time constraints no further research is done on such a method. Some initial thoughts for a method that maintains global coherence are added tot he future work section of this thesis.

Based on the results in the previous section, it has to be concluded that the shape regression

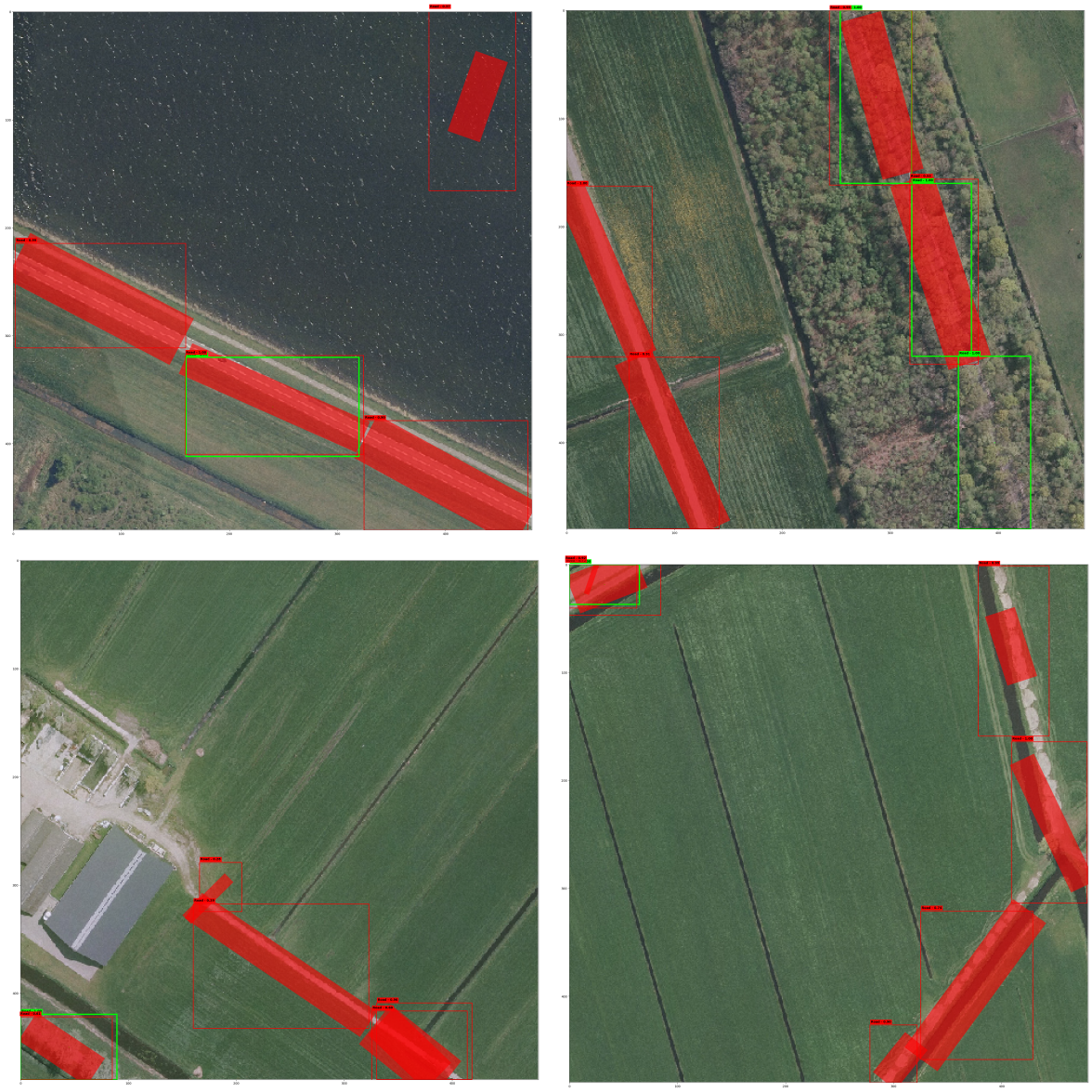


Figure 4.16: A selection of various false positive cases of predicted road segment localizations and annotated localizations combined with road segment line estimations.

network is not able to learn the provided representation of road structures well. It could possibly help to add a restriction that ensures that the predicted points are always on the border of the segment localization box.

As in some cases, the network does predict the road segment shapes rather well, it can be concluded that a neural network is able to learn a representation of road segments by performing shape regression. Although it is questionable whether this particular representation of road segments with this particular network can lead to consistently accurate predictions.

When compared to the qualitative results of the previous chapter, even the localization part of the network seems to suffer from the added task of performing shape regression. The precision and recall results shown in table 4.7 confirm this, as the precision of the segment localization part of the shape regression network decreases compared to the precision of the segment localization network of the previous experiment. This indicates that the amount of false positives have increased. Additionally, the recall of the localization part of the shape regression network increases when compared to the recall of the segment localization network of the previous experiment. That shows that the shape regression network does produce more relevant road segment localization predictions, however when combined

with the decreased precision, it indicates that this comes at a cost of more false positives. This phenomena can be addressed to the combined loss function, where localization and shape regression are contributing to the same loss function. The original YOLO network, introduced separate weights for the localization and detection parts of the loss, which indicates that this architecture needs fine-tuning before it can successfully perform both tasks well. Aside from that, it makes intuitively more sense to first learn how to localize road segments well before attempting to predict the road entry and exit points on the localized box. At this time both localization and shape regression tasks are learned at the same time, due to the combined loss function, which could explain not only the the degradation in localization performance, but the inconsistent predictions of road entry and exit points as well.

As mentioned earlier, the width estimation of individual road sections is inconsistent as well. After visually inspecting the road polygons of several images, it is concluded that the width of the annotated roads are rather complex. Sometimes the width of the road aligns well with the actual visible road, other times it aligns well with the road shoulders, whereas some roads are not even annotated at all. For an average person, without inside knowledge on how road annotations are defined by CBS, it is near impossible to give a consistently accurate prediction of the width of the road when provided the same aerial image as the network, whereas it is sometimes even impossible to predict which roads should be annotated as road and which roads should not. Figure 4.17 shows an example of three aerial images with roads and their annotations. Even though the roads on the images seem to be of similar width, the annotated widths of the road polygons differ immensely.

Figure 4.17: Seemingly inconsistent road width annotations of three samples of aerial images with their corresponding annotations. Roads are depicted in red on the annotations.

5

Conclusion

In section 4.1 Rijkswaterstaat's U-net showed to be capable of achieving a competitive segmentation performance in terms of average IoU, when trained on the large six class segmentation problem of the DeepGlobe dataset of nearly 2600 images with pixel resolution 608×608 as it achieved a significantly higher score than the reported reference network. Unfortunately the DeepGlobe benchmark has not been made publicly available at time of writing, therefore U-net could not be trained nor tested on the full DeepGlobe benchmark. Consequently, no claims are made about whether the tested per-pixel segmentation networks outperform the reference network. Assuming that the acquired DeepGlobe training data is a good representation of the test data, it is concluded that the U-net architecture performs well on both large amounts of data and in a relatively large multi-class segmentation setting. The answer to the research question: *"How well does the performance of Rijkswaterstaat's per-pixel segmentation network scale with the amount of data?"* is therefore that Rijkswaterstaat's U-net scales well with the amount of data. However, in order to measure how well it performs on a, for Rijkswaterstaat relevant, dataset a public benchmark dataset is needed. Until DeepGlobe becomes publicly available, or any other multi-class per-pixel land use classification dataset becomes available, this part of the question will remain unanswered.

Additionally, the newly introduced Dark U-net architecture is compared to U-net as the influence of a network's feature extractor is measured. The Dark U-net architecture utilizes a Darknet-19 feature extractor, which is more complex than U-net's VGG-16 based feature extractor. The results show that the U-net performs slightly better in terms of mean IoU when trained on pristine data, whereas the difference in segmentation performance between U-net and Dark U-net is negligible when trained on augmented data. On augmented data the Dark U-net network tends to be more stable than U-net in terms of training as it shows a significantly lower standard deviation on both mean IoU, training time and number of training epochs. Changing the feature extractor to a more compex feature extractor does not necessarily increase the network's segmentation performance, however it can significantly influence both training time and stability as well as inference time, which answers the following research question: *"What is the influence on the segmentation performance if the feature extractor of a segmentation network is exchanged for a state-of-the art feature feature extractor?"* For the YOLO object detector, Redmon and Farhadi experienced a slight drop in performance in exchange for faster inference [45]. However it has to be noted that Redmon and Farhadi compared their Darknet-19 feature extractor to the full VGG-16 feature extractor for YOLO version two, whereas U-net uses a part of VGG-16. Redmon and Farhadi therefore concluded that their Darknet-19 is less complex than the full VGG-16, whereas it has to be concluded in this thesis that Darknet-19 is slightly more complex than U-net's VGG-16 based feature extractor. Since the shape regression based segmentation network is an extension of YOLO version two, the described state-of-the-art feature detector is already incorporated. As only the amount of output nodes and the loss function of the YOLO architecture are modified, the influence of the exchange of the feature extractor is expected to be the same as its influence on the original YOLO architecture. The conclusion in terms of performance, therefore, remains unchanged, as segmentation performance is similar when compared to U-net, whereas Dark U-net's training and inference time is significantly better.

Besides these research questions, the effect of data augmentation is assessed in section 4.1 as

well, where it is tested on both the U-net and the Dark U-net architectures. From that experiment it is concluded that, although it does not necessarily increase the segmentation performance in terms of IoU, it does increase the overall stability of the training process as it shows fewer signs of severe overfitting as well as a significantly lower standard deviation of the mean IoU when comparing the performance of five seperately trained U-net and Dark u-net networks. It is not ruled out that data augmentations can increase the segmentation performance, although further research has to be conducted to assess the potential of the individual techniques for land use classification.

From section 4.2 it is concluded that the amount of classes of multi-class segmentation problems do not influence the segmentation performance of the classes these segmentation problems have in common. The little variances in the performance do not show a consistent pattern when observed for the different multi-class segmentation problems and are therefore addressed to random initialization of the variables and variations in the training data induced by the augmentation techniques, for which the random crop and scale operations is assumed to be most influental. This provides an answer to the research question: *"What is the impact on the segmentation performance of a state-of-the-art segmentation network when moving from binary to a multi-class segmentation problem?"*.

The experiments described in section 4.3 and section 4.4 focus on the shape regression network for road segment localization and detection. Road segment localization performance is assessed in section 4.3, where a YOLO version two object detection network is stripped of its classification task to perform road segment localization. As road segment localization annotations, the newly introduced annotation generation method is utilized to subdivide a complex road polygon in simplistic road segments. These individual road segments preserve the original topology, as both the genus and connectedness property is preserved. From the results of this section it is concluded that the road segment localization network is able to learn the newly introduced road polygon representation using individiual road segments. This partially answers the research question: *"How can an arbitrarily complex road structure be detected by locating a series of simplistic shapes such that the original topological properties are preserved?"* as the results prove that road structures can be detected by locating a series of simplistic shapes. This network was solely trained on convex road structures, thus it is unknown if arbitrarily complex road structures can be detected. However, the annotation generation method for localization does work on complex road structures. Given the fact that the feature extractor is able to learn the representation of the annotations, combined with the fact that the feature extractor is able to detect more complex road structures (as observed from the per-pixel segmentation network), it is expected that the road segment localization network is able to localize arbitrarily complex road structures. Due to time constraints this hypothesis has not yet been tested, and is thus considered to be future work.

Additionally it has to be concluded that the road localization performance is not yet optimal. The recall measurement indicates that the network fails to localize a significant amount of annotated road segments. Furthermore, the precision measurements shows that the network predicts a significant amount of road segments, where no road segment is present in the annotations. From a visual inspection of the patches of aerial images and their original annotations for this experiment, it is concluded that the newly introduced dataset contains many patches where road-like structures in the aerial image are not annotated in the shapefiles. Based on the context visible in the image patches it is often not clear to the human eye which road structures in the aerial image do actually appear in the annotations, without having seen the annotations first. These, sometimes counter intuitive, annotations could influence the precision and recall of the network as the network receives contradictory feedback for visually similar looking patches. Further research should be conducted in order to back up this hypothesis. One could for instance replicate this experiment on a dataset that contains more consistent road annotations in both urban and rural areas, for which OpenStreetMap [43] data might be a suitable candidate.

Another factor of influence is the newly introduced annotation generation method that splits a road polygon into smaller sections. Due to its naive splitting condition, subdividing the annotation space in a 3×3 grid, the road polygon can get subdivided in an illogical way. Counter measures should be researched to correct illogical annotations. A possible research direction could be to utilize the current method to produce candidate segments, whereafter a more sophisticated merging operation is performed to merge candidates. This merge operation could for instance take into account the area of a segment as well as whether adjacent segments share their shortest or longest edge, which is an important indication of whether a road segment is split along the road or over the width of the road.

As a result, merging based fixes probably violate the condition that a road segment is fully inside one of the defined grid cells, thus changing the representation of a road polygon. Whether this change of representation is beneficial for the road segment localization network is considered to be future work.

Aside from these observations, the localization network only operates on a local scale where individual road segments are localized. Even though the annotation generation method does preserve the topological connectedness property, it has to be concluded that the network is not able preserve this property in its predictions. Since the loss function only takes into account the parameter values of the individual segments, it does not incorporate any measure for global coherence of the predicted segments. Therefore, global coherence is not guaranteed by this shape localization network, which in turn violates the topological connectedness property. For future work it could be interesting to see if topology can be preserved by encoding global coherence into the loss function of the network. It could be possible to add a term to the loss function that encodes the sum of the distance between predicted segments for each annotated pair of connected segments. This might improve connectedness of localization boxes. On the other hand, since this part solely includes segment localization, the actual shape and orientation of the predicted road segments are not taken into account and thus connectedness of the segmented road sections are still not guaranteed.

The road segmentation results, where road segments are approximated by a line segment per localized box, show that this interconnectedness is not guaranteed at all by the shape regression based road segmentation network. Other neural networks that deal with the same connectedness problem apply post-processing steps that aim to operate on the local structures to achieve global coherence. Body part predictions combined with part affinity fields are post-processed in order to interconnect the corresponding body parts in the pose estimation framework [16], whereas Deeproadmapper attempts to correct disconnected road sections by interconnecting nodes based on a shortest-path criteria.

It is believed to be crucial for the road segmentation network to achieve global coherence for which further research needs to be conducted. Based on the current implementation, the post-processing step utilized by Deeproadmapper is assumed to be the easiest option to implement, as the current representation of road segments are lines, for which two nodes and an edge are defined. This is in line with the graph representation of Deeproadmapper. However, the current results show that simply interconnecting the disjoint roads is too naive as the predicted segments do not consistently align well with the annotated road section. For that method to work, the road segmentation prediction needs to improve, for which the parametrization of the road segment should be revised. The current parametrization allows for too much freedom as the nodes of the road segment are allowed to be inside the localization box, whereas the annotations only allow these nodes to be on the outline of the localization box. One possible solution is to keep the current parametrization and apply a ‘snapping’ function as post processing, that snaps the predicted points to the nearest point on the perimeter of the predicted box before calling the loss function. Possibly, splitting the network in two successive networks could be beneficial as well, as road segment shape predictions rely on road segment localizations.

On the other hand, there are also some positives as the shape regression based segmentation network shows to be capable of learning a representation of individual road segments, although it has to be concluded that its performance is far from stable. This does however show that shape regression has the potential to learn simplified simplifications of complex structures. Therefore, the research question: *“How can a segmentation of a variety of pre-defined simplistic shapes segment complex road structures, whilst preserving the original topology?”* is partially answered.

Similarly to the previous research question, this research question can not be fully answered as the network has only been tested on rather simplistic road structures, namely convex road segments. The current parametrization does not allow for segmenting complex road segments, as only straight lines are predicted. Further research has to be done on more complex road segments. For a start, curved roads could be approximated by predicting extra two points inside the road localization box, after which a spline can be fitted through the points to approximate a curved road segment, whereas crossroads could be approximated by predicting two extra points on the road localization box. Let these parameters be called template parameters, that include the current two points and width parameter as well as a set of additional parameters for the extra points introduced by the new templates. The next problem would be to be able to predict which type of template should be used for which localized box. This basically introduces the classification task again, where the predictions can be extended by one-hot encoded vector that predicts which template suits the structure at a certain localization box best after which the template parameters are used to construct the actual road segment.

Rijkswaterstaat is interested in solving their multi-class segmentation problem, for which it was initially intended to combine the shape regression based road segmentation network with the per-pixel segmentation network in order to assess whether the road segmentation would improve by the newly added information. However, due to the current state of the network, the issues specified as future work in the paragraphs above should be given a higher priority before an attempt at combining the networks, described in section 3.4, should be made. At this point in time it is unlikely that the per-pixel segmentation network would benefit from the current output of the shape regression based road segmentation network. Therefore, the research question: *"How can the road segmentation, obtained from the shape regression network, be combined with a state-of-the-art per-pixel segmentation network in order to perform a geometry and topology preserving multi-class segmentation task?"* can currently not be answered as the combined shape regression and per-pixel network is not constructed. For future work, it could be interesting to construct the hybrid network if the before mentioned issues are solved. It could then be interesting to look into end-to-end training of the hybrid network. In order to do this, the loss function of both the Dark U-net network and the shape regression network can be combined into one loss function. Since the shape regression network utilizes an aggregated loss of the individual components, depicted in equation 3.25, it could easily be extended by a per-pixel loss function as well. As the other terms in that equation are calculated via a sum of squared differences between predicted values and annotated values, this function could be applied to the one-hot encoded class values to calculate the per-pixel loss as well. Possibly, a weight coefficient has to be defined for the per-pixel loss term in order to bring its priority in line with the other terms, similar to the λ values used in equation 3.25.

Alternatively, instead of the hybrid network described in section 3.4, the networks could also be combined in a sequential way, where the rasterized output of the shape regression network could be concatenated to one of the first feature maps of the Dark U-net network so that the newly provided information is processed by the full network. It would be interesting to see whether this method would suffer from the same loss of shape and topology information as the original Dark U-net network, since the new information follows the same path through the network as the original RGB image data. On the other hand, the combined network where Dark U-net and the shape regression network are put in parallel, is not expected to suffer from this loss of shape, as the new information is only appended at the end, in its original resolution.

The main research question of this work is:

> *"Does introduction of geometrical and topological information to a segmentation task, attained via object localization and shape regression, improve the segmentation of road structures on Rijkswaterstaat's aerial data?"*

Unfortunately the results in section 4.4 show that the shape regression network performed significantly worse than the Dark U-net segmentation network on the same dataset, meaning that the introduction of geometrical and topological information, attained using road segment localization and shape regression, does not improve the performance of road segmentation of Rijkswaterstaat's aerial data at this point in time. However, the results showed that the network is capable of learning the newly introduced road polygon representation. It is expected that a more consistent and natural looking annotation, combined with a more restricted shape parametrization, will increase the shape regression's network performance although it is only fair to assume that still many steps have to be taken until it will perform close to or better than state-of-the-art segmentation networks, especially when taking into account global coherence of the road segments and road segment complexity.

Representing segmentation as a shape regression task is an interesting problem. The proposed method looks promising, despite its inability to beat the state-of-the-art per-pixel segmentation methods at this moment in time. New insights into solving segmentation problems on remotely sensed data have been gained and engaging opportunities are unveiled in order to extend and refine this novel method. It is believed that the proposed direction has the potential to be a great step forward in Rijkswaterstaat's objective to utilize automated land use classification for either monitoring land usage or maintaining topographical maps.

Bibliography

[1] Dstl satellite imagery feature detection. URL https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection/data. Last visited on 19/11/2018.

[2] Isprs 2d semantic labeling - potsdam, . URL http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html. Last visited on 19/11/2018.

[3] Isprs 2d semantic labeling - vaihingen, . URL http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html. Last visited on 19/11/2018.

[4] Pdok publieke dienstverlening op de kaart. URL https://kaart.pdok.nl/. Last visited on 14/12/2018.

[5] Centraal bureau voor de statistiek. URL https://www.cbs.nl/. Last visited on 14/12/2018.

[6] Real-time object detection with yolo, yolov2 and now yolov3. URL https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088. Last visited on 14/12/2018.

[7] Kadaster. URL https://www.kadaster.nl/. Last visited on 14/12/2018.

[8] Ministry of agriculture, nature and food quality. URL https://www.government.nl/ministries/ministry-of-agriculture-nature-and-food-quality. Last visited on 14/12/2018.

[9] Geodetic reference frames in the netherlands, . URL https://www.ncgeo.nl/downloads/43Referentie.pdf. Last visited on 14/12/2018.

[10] Rijkswaterstaat, . URL https://www.rijkswaterstaat.nl/. Last visited on 14/12/2018.

[11] Yolo: Real-time object detection. URL https://pjreddie.com/darknet/yolo/. Last visited on 14/12/2018.

[12] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

[13] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, Dec 2017. ISSN 0162-8828. doi: 10.1109/TPAMI.2016.2644615.

[14] John E. Ball, Derek T. Anderson, and Chee Seng Chan. A Comprehensive Survey of Deep Learning in Remote Sensing: Theories, Tools and Challenges for the Community. 2017. ISSN 1931-3195. doi: 10.1117/1.JRS.11.042609.

[15] Favyen Bastani, Songtao He, Sofiane Abbar, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Sam Madden, and David DeWitt. Roadtracer: Automatic extraction of road networks from aerial images. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[16] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.

[17] Marco Castelluccio, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva. Land use classification in remote sensing images by convolutional neural networks. *arXiv preprint arXiv:1508.00092*, 2015.

[18] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. pages 1–14, 2016. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2699184. URL http://arxiv.org/abs/1606.00915.

[19] Wikimedia Commons. A concave set, 2007. URL https://commons.wikimedia.org/wiki/File:Convex_polygon_illustration2.png. File:Convex polygon illustration2.png.

[20] Wikimedia Commons. A convex set, 2007. URL https://commons.wikimedia.org/wiki/File:Convex_polygon_illustration1.png. File:Convex polygon illustration1.png.

[21] Wikimedia Commons. Svg redraw of image:convexhull.png, 2008. URL http://commons.wikimedia.org/wiki/Image:ConvexHull.png. File:ConvexHull.svg.

[22] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. *ArXiv e-prints*, 2018.

[23] Sean Gillies et al. Shapely: manipulation and analysis of geometric objects, 2007–. URL https://github.com/Toblerity/Shapely.

[24] Sean Gillies et al. Rasterio: geospatial raster i/o for Python programmers, 2013–. URL https://github.com/mapbox/rasterio.

[25] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[26] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

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

[28] Fan Hu, Gui-Song Xia, Jingwen Hu, and Liangpei Zhang. Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery. *Remote Sensing*, 2015. ISSN 2072-4292. doi: 10.3390/rs71114680.

[29] Jingwen Hu, Gui-Song Xia, Fan Hu, and Liangpei Zhang. A comparative study of sampling analysis in the scene classification of optical high-spatial resolution remote sensing imagery. *Remote Sensing*, 7(11):14988–15013, 2015.

[30] John D Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9 (3):90–95, 2007.

[31] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[32] Pascal Kaiser, Jan Dirk Wegner, Aurélien Lucchi, Martin Jaggi, Thomas Hofmann, and Konrad Schindler. Learning Aerial Image Segmentation from Online Maps. *IEEE Transactions on Geoscience and Remote Sensing*, 55(11):6054–6068, 2017. ISSN 01962892. doi: 10.1109/TGRS.2017.2719738.

[33] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

[34] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[36] Lei Liu, Zongxu Pan, and Bin Lei. Learning a rotation invariant detector with rotatable bounding box. *CoRR*, abs/1711.09405, 2017. URL http://arxiv.org/abs/1711.09405.

[37] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[38] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[39] Francois PS Luus, Brian P Salmon, Frans Van den Bergh, and Bodhaswar Tikanath Jugpershad Maharaj. Multiview deep learning for land-use classification. *IEEE Geoscience and Remote Sensing Letters*, 12(12):2448–2452, 2015.

[40] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55 (2):645–657, Feb 2017. ISSN 0196-2892. doi: 10.1109/TGRS.2016.2612821.

[41] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[42] Ministerie van Infrastructuur en Milieu. Basisregistratie grootschalige topografie gegevenscatalogus bgt. Catalogue, July 2013. URL https://www.geonovum.nl/sites/default/files/BGTGegevenscatalogus111.pdf.

[43] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.

[44] Otávio AB Penatti, Keiller Nogueira, and Jefersson A dos Santos. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 44–51, 2015.

[45] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016. URL http://arxiv.org/abs/1612.08242.

[46] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. URL http://arxiv.org/abs/1804.02767.

[47] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[48] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[50] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[51] Ankit Sachan. Zero to hero: Guide to object detection using deep learning: Faster r-cnn,yolo,ssd, 2017. URL https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/.

[52] Jamie Sherrah. Fully Convolutional Networks for Dense Semantic Labelling of High-Resolution Aerial Imagery. 2016.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL http://arxiv.org/abs/1409.1556.

[54] Godfried T Toussaint. Solving geometric problems with the rotating calipers. In *Proc. IEEE Melecon*, volume 83, page A10, 1983.

[55] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

[56] Michele Volpi and Vittorio Ferrari. Semantic segmentation of urban scenes by learning local class interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2015.

[57] Michele Volpi and Devis Tuia. Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 2017. ISSN 01962892. doi: 10.1109/TGRS.2016.2616585.

[58] Eric W. Weisstein. Convex hull. From MathWorld–A Wolfram Web Resource. URL http://mathworld.wolfram.com/ConvexHull.html. Last visited on 14/11/2018.

[59] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279. ACM, 2010.