Road Extraction from Aerial Images

Master's thesis in Complex Adaptive Systems

RICKARD SIREFELT

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2015

MASTER'S THESIS IN COMPLEX ADAPTIVE SYSTEMS

Road Extraction from Aerial Images

RICKARD SIREFELT

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2015

Road Extraction from Aerial Images
RICKARD SIREFELT

Master's thesis 2015:82
ISSN 1652-8557
Department of Applied Mechanics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
The so called "Judge Harry Pregerson Interchange" in Los Angeles, California.

Chalmers Reproservice
Göteborg, Sweden 2015

Road Extraction from Aerial Images
Master's thesis in Complex Adaptive Systems
RICKARD SIREFELT
Department of Applied Mechanics
Chalmers University of Technology

Abstract

Road extraction from aerial images is essential to everyday life having several useful application in for example urban planning and automotive navigation. Roads are currently extracted using methods that requires a lot of manual work conducted by humans which is both time consuming and error prone. The aim of this thesis is to develop a robust algorithm that can automatically extract roads from aerial images. It was concluded, based on a literature review, that the most suitable method for automatic road extraction is a machine learning approach based on stacked convolutional neural networks. The method was implemented and evaluated against four different road images in the vicinity of the motorway E6 in southern Sweden. The best network achieved a recall of 0.845, precision of 0.878 and quality of 0.760 over a test set of previously unseen images. Considering that the method used was relatively simple, the result is to be considered competitive compared to other published works.

Keywords: automatic road extraction, pattern recognition, machine learning, image processing, image analysis, convolutional neural networks, classified road detection, aerial image analysis

Preface

This is my master thesis which concludes five years of studying Engineering Physics at Chalmers University of Technology. It has been fun and exciting but now the time has come for it to end.

Acknowledgements

I would like to thank my supervisor and examiner Mattias Wahde for coming up with the idea for the thesis and for his guidance throughout the work. I would also like to give a big thanks to the many cited authors whose work taught me a lot about image analysis, road extraction and machine learning.

Contents

Chapter 1 Introduction

The process of detecting objects and identifying their properties from aerial images dates back almost one hundred years when humans first brought their cameras with them on airplanes. Aerial image analysis had its first major breakthrough during the first world war where the photos mainly were used for reconnaissance. Since then object recognition from aerial images have found many applications in various fields such as forest planning, urban development, disaster relief, climate monitoring and mapping.

Due to the development of high resolution imagery from satellites and drones the number of available aerial images have exploded in recent years. The need of analyzing all of this data has as a consequence grown immensely. Unfortunately the technology for analyzing all of these images has not been able to keep up and today much of the work of analyzing the images is still conducted manually by humans which is expensive, time consuming and error prone. Because of this there is a high demand for fast and reliable methods that are able to analyze all of this data automatically.

An area where the above issue is prominent is in road mapping. Road maps are essential to everyday life due to that they are an important tool in many areas such as in urban planning and automotive navigation. Since the global road network is ever changing they have to be kept-up-to date which is a problematic task with the currently used methods. The development of a new method for extracting road networks which is both automatic and reliable is therefore necessary and will be addressed in this project.

1.1 Purpose

The purpose of this thesis is to develop an algorithm that automatically extracts road networks from aerial images. The work that should be conducted in the thesis can be divided into the following three parts.

- Conduct a literature study of the available research in the field of automatic road extraction. Choose the most suitable method for automatic road extraction from aerial images.
- Implement the chosen method in the C# programming language and if possible improve it.
- Analyze and evaluate the method.

1.2 Limitations

The developed algorithm was designed to work with aerial images taken from Google Maps using a zoom level of 16. The aerial images considered was all taken from areas in the vicinity of the swedish highway E6 connecting the cities Oslo and Malmö. This limited to some extent the possible type of road scenes that needed to be considered i.e. the road scenes was mainly of rural areas taken from the Swedish countryside.

The algorithm was designed to work on a normal personal computer as opposed to a high performance computer. This imposed several performance restrictions to the algorithm to not require to much memory or computational power. However no real time requirements was opposed on it relaxing these restrictions to some extent.

Chapter 2 Literature review

The intention of this chapter is to provide an overview of the field of road network extraction. Basic knowledge of the most used methods and references to some important works where they have been applied will be given. At the end of the review a method is chosen which according to the author is the most suitable for automatic road network extraction. This method will be explored further in later chapters.

2.1 Introduction

Automatic road network extraction from aerial and satellite imagery is an active research field. Research on the area has been conducted for more than 30 years and during this period many different techniques have been developed. In a state-of-the-art review in 2003, Mena [24] described nearly 250 papers related to the topic which can serve as proof for the amount work done in the field. In the article Mena classified the different methods that have been applied to the problem and gives an exhaustive overview of the existing literature in the field. A more recent review was conducted in 2010 by Hauptfleisch [15]. In the review Hauptfleisch used the same method classification as Mena, but also provided in-depth description of the most common methods. This chapter is in many ways based on the two works mentioned above but formated more compactly and in turn is not as exhaustive.

2.2 Image sources

The aerial and satellite images, also known as remote sensed images, are mainly provided from the satellites IKONOS and QUICK BIRD as well as from various airplanes. The satellites provide images with a resolution of around 0.5 - 1 m whereas the airplane images can have a resolution as small as 0.2 m. Images from these sources are considered to have a very high resolution. Other type of images used for road network extraction are lower resolution images or images based on LIDAR[1] or SAR[2] technology.

2.3 Road extraction methods

At the top level, every road network extraction method is classified as either manual, semi-automatic or fully automatic. Manual methods require that a human manually extracts the roads. Semi-automatic methods require some kind of human input to provide important information to an algorithm such as the starting point or color of a road, whereas fully automatic methods use algorithms that require no human input. Currently there are two kind of ways that are used to retrieve road networks. The first one is ground surveying conducted by humans out in the field which is fully manual. The other one is based on semi automatic algorithms using remotely sensed imagery and relies heavily on human input, an example of such a method is project Ground Truth by Google [12]. Both of these methods are costly and time consuming which means that the development of a fully automatic method would save both money and time. Despite significant efforts, no such algorithm robust enough to replace the currently used ones exist.

In order to get a good overview of the field of automatic road network extraction it is preferable to categorize the different methods used. In accordance with [15] the following classes of operations will briefly be discussed.

Automatic seeding	Hough transform
Road tracking	Mathematical morphology
Edge detection	Snakes
Classification	Multi-resolution analysis
Segmentation	Road network construction

[1] LIDAR is a remote sensing technology that measures distance by illuminating a target with a laser and analyzing the reflected light.

[2] SAR is a form of radar which is used to create images of an object, such as a landscape.

2.3.1 Automatic seeding

Seeding is a method used in road network extraction where one marks specific points of interest called seeds in an image. The seeds can be starting points for roads, easily detected road segments, intersections etc. The method is often used as an initial step in many road extraction algorithms such as for example road tracking and snake algorithms, see Subsect. 2.3.2 and 2.3.8. Seeding can also be used to select road training sets for learning algorithms. In algorithms that rely on seeding, good seeds usually are of vital importance. Because of this, a common approach is to have a human operator provide the seeds making them semi automatic. However, many studies have been carried out to try and produce an automatic way of reliably producing good seeds in road network extraction systems. The most common approaches to achieve this are parallel edge detection and model matching.

In parallel edge detection the basic approach is to apply an edge detection filter, such as a Laplacian or Sobel filter, to the image which assigns gradient values to each pixel. The roads are then assumed to be either darker or brighter then their surroundings. Based on the assumption road seeds are selected as midpoints between opposing gradients which are within a specified road width limit. The articles Trinder and Wang [38], Baumgartner *et al.* [3], Doucette *et al.* [10] and Christophe and Inglada [8] all use parallel edge detection for seeding.

In model matching the image is first segmented, see Subsect. 2.3.5, into pixel groups with similar properties. These groups are then compared to predefined road models specifying a set of criteria such as shape and spectral properties of roads. Depending on how well they match, the pixel groups will be selected as either seed segments or not. In [20], Koutaki and Uchimura used model matching to detect intersections. They segmented the image using the Iterative Self-Organized Data Analysis Technique (ISODATA) [2]. Intersections were then obtained by comparing the pixel groups with novel intersection models comprised of rectangles. Another model matching technique for road network extraction was proposed by Shi and Zhu [35] known as the Line segment match method. They used a spectral thresholding technique to obtain a segmented binary image. Road segments were then detected by analyzing how well straight long lines in various directions fitted into the image segments.

2.3.2 Road tracking

Road tracking algorithms are basically line tracing methods that follow roads from a set of starting points. Usually they operate by iteratively doing the two operations of choosing the next road point as well as validating and refining that point. The algorithm for determining the next road point often predicts the next position based on the current and preceding road points location and orientation. The validating and refining algorithm uses properties of the road to decide if the new point is accurately placed. It also calculates the orientation of the road at the new position. In general, road trackers are based on assumptions such as that roads have homogeneous surfaces, low curvature and are interconnected to form a network. In their most simple form road trackers are prone to errors when used on real world data due to reasons such as occlusions along the road and roads that do not fit the assumptions. However, different techniques have been applied to try and make road trackers more robust.

In [39], Vosselman and Knecht proposed a road tracking method using the recursive Kalman filter [18] and spectral value matching. A Kalman filter is an algorithm that uses a series of noisy measurements observed over time to produces estimates of unknown variables which in this case are the future position and shape of the road. To improve the tracking they also used spectral value matching to a reference road model by least squares. They found that their algorithm could track a road past obstacles such as highway overpasses and junctions. In [34], Shen *et al.* instead used a curve-fitting procedure together with the angular texture signature, see Subsect. 2.3.4, to determine the next position and direction of a road. Their approach was able to track roads through curvilinear road segments and could also handle road junctions.

2.3.3 Edge detection

Edge detection is the process of detecting points in a digital image where rapid changes or discontinuities in brightness occur. Edge detection is a fundamental tool in image analysis and it is common to include edge detection in road extraction systems. Out of the 217 articles regarding road extraction surveyed in the review

by Hauptfleisch [15], 129 used some form of edge detection to extract road networks.

There exists several edge detection techniques and most of them can be grouped into two categories, searched-based and zero-crossing based. The searched-based methods detect edges by computing a measure of edge strength often using a first-order derivative operation. The processed image is then searched for local maximas to determine edges and their direction. Zero-crossing based methods mainly uses the second-order derivative to obtain the derivative rate of change. The algorithms will then search for zero-crossings to detect edges. Edge detection algorithms are sensitive to image noise, therefore some kind of smoothing is almost always applied to the image prior to the detection. Another common approach is to apply a thinning operation to reduce gradual edges spanning over several pixels. Some of the most common edge detection methods used in road extraction systems are Canny [6], Nevatia-Babu [30], Sobel, and Laplace.

Edge detection algorithms are used for many applications in road network extraction such as in automatic seeding, road tracking and geometric classification. It is a common belief that edge detection is all that is required for road network extraction. This is however false due to several problems with such a basic approach. Except for being sensitive for image noise, edge detectors are also sensitive to occlusions such as shadows or trees blocking the roads which produces many localized edges. In [4], Baumgartner *et al.* stated that the precision of road extraction based on parallel edge detection is poor. However, research has been done on how to improve the accuracy and robustness for edge detection approaches. In [10] for example Doucette *et al.* adds a feedback loop to remove unlikely road segments based on their structural and spectral properties.

2.3.4 Classification

Classification is the process of identifying unknown patterns and organize data into categories or classes with different properties. In image classification specifically the problem is to classify different image features. There exists a vast amount of different classification methods and going through all of them would take several books. This section will therefore only serve as a short overview of some common methods and how they can be used for extracting roads.

In general, classification methods calculate probability values for each image feature and class pair. A feature is classified as belonging to the class which corresponds to its highest probability value. The classes may be specified a priori, known as supervised classification, or the features may be automatically clustered, known as unsupervised classification. In supervised classification the algorithms learn the main properties of the different classes using a training set consisting of image features whose class membership is known. In unsupervised classification the algorithms instead rely on clustering operations to automatically segment the data.

One of the most important aspects of successful image classification is to construct suitable classes which are discriminatory e.g. the overlap between the classes are as small as possible. Achieving discriminatory classes in road extraction is often difficult where for example roads can have the same color and geometrical properties as long buildings. The most common class decomposition in road network extraction is simply road and non-road features other examples of class decomposition are different kinds of road segments such as intersections and curves, other man made objects and different vegetation areas.

A common approach for classification is to create a so called feature vector $v = (x_1, x_2, ..., x_n)$ where each element x specifies a certain property of an image feature such as color, width and curvature. The probability that a certain feature belongs to a certain class is then related to a defined distance between the feature and the class in feature space. The properties that road extraction classifiers uses are of spectral, textural, geometrical or contextual nature.

Spectral classifiers uses the spectral properties of image features such as their raw RGB values to classify them. Some notable classification methods and related articles used for spectral classifications in road network extraction are statistical classifiers [25], artificial neural networks [29] and [26], fuzzy classifiers [28] and support vector machines [36]. Artificial neural networks and statistical classifiers will be further discussed below.

Textural Classifiers classifies image features based on their textural properties such as spatial structure,

contrast, roughness and orientation. A notable work to mention in the context is [14], where Haralick suggests a set of important features which can be used for texture classification known as the Haralick features. Some textural classification methods used in road extraction are texture cubes [25], see **Statistical classifiers**, and angular texture signature [9]. The angular texture signature around a pixel is found by rotating a rectangular window stepwise about the pixel. In each step the variance from the mean in the window is calculated. A histogram is then set up in which each bin represents a step in the rotation displaying the variance. By analyzing the valleys of this histogram classification of road segments can be made. For example road segments should have small variance when the rectangle is aligned with the road.

Geometrical Classifiers classifies image features based on their structural characteristics. The basis for geometrical classifiers in road extraction is that roads appear like elongated homogeneous regions in very high resolution images. Geometrical classifiers usually works on processed images where the structural information has been derived through low-level operations such as edge detectors, segmentation algorithms and the Hough transform, see Subsect. 2.3.5 and 2.3.6. Some notable geometrical classification methods are again the angular texture signature [44] and intensity terrain models [37].

Contextual classifiers uses contextual information to guide several road extraction systems. The idea is that the road extraction technique can be optimized for different kind of surrounding regions. The most common contextual classifiers determines if the region in question is a forest, rural or urban area. The contextual information used can be anything from nearby objects such as cars, road markings and tree lanes to region color. Some notable works that have used contextual classifiers are [16], where Hinz and Baumgartner used shadows, buildings and vehicles to classify urban areas and [40], where Yang and Wang used an edge density histogram model to classify urban, country, montane and hybrid regions.

Statistical classifiers

Statistical classifiers are based on classic probability theory. They are mainly used for their simplicity and computational efficiency. The most common methods applied are minimum distance matching, correlation matching and maximum likelihood. A notable work using statistical classifiers for road network extraction is [25], where Mena and Malpica used three different statistical classifiers to classify pixels based on their spectral properties. The first two methods were spectral distance measures to road models based on the RGB values of the pixel and neighboring pixels using the Mahalanobis [23] and Bhattacharyya [5] distance respectively. The third method used three dimensional texture cubes. These cubes were constructed by the HSI[3] values of the pixels in a 3x3 window around the pixel in study. These cubes were compared to road model cubes using co-occurrence distributions, Haralick features [14] and the Bhattacharyya distance. The outcome from the three statistical methods were finally fused together using Dempster-Shafer fusion [33] to get an accurate single result.

Artificial neural networks

Artificial neural networks are a family of statistical learning algorithms which are inspired by biological neural networks. Artificial neural networks are generally presented as systems of interconnected nonlinear components known as neurons. The connections between the neurons are adaptive weights which can be tuned by learning algorithms. One of the most important advantages of neural networks compared to other common classifiers is that they are distribution free which means that the significance of the sources does not have to be known in advance as for statistical classifiers.

The most common implementation of an artificial neural networks is a so called feedforward network which are taught using the backpropagation algorithm. In feedforward networks the neurons are arranged in layers and the connections between neurons are between these layers. This type of network implementation was used by Mokhtarzade and Zoej [29] for road extraction from high resolution images. Their approach was to determine if a pixel was a road pixel or not based on the RGB values of the 3x3 surrounding pixels. Mnih [26] applied a similar approach but used a different network architecture called convolutional neural network which recently has shown great potential in the field of image analysis. The network was used in a patch-based framework where pixels in a 16x16 window was classified based on the surrounding 64x64 pixels.

[3]The HSI representation of a pixel is an alternative spectral representation to the more common RGB values.

2.3.5 Segmentation

Segmentation is the process of partitioning a digital image into subsets of pixels with similar characteristics. The characteristics usually are similarities in color, intensity or, texture. The result after segmentation are collection of pixel clusters where neighboring pixel clusters have different characteristics. The goal of image segmentation is to add structure to the data which will make analyzing the image by for example a classifier both easier and faster. Usually the segmentation is of a binary kind which means that the image gets split into road and non-road segments. As for classifiers there exist a wide range of techniques to segment an image and to cover them all in this review is simply not possible. Segmentation is suitable for road network extraction because of the special characteristics of roads in remote sensed images. Therefore many road network systems utilities segmentation techniques, for example the clustering methods k-means [44] and ISODATA [42].

2.3.6 Hough transform

The Hough transform is a feature extraction technique used to find imperfect instances of objects in an image within a certain class of shapes through a voting procedure. The classical Hough transform was used to detect regular shapes such as lines, circles and ellipsoids. The later developed generalized Hough transform uses a parametric approach and is able to detect arbitrary shapes. The main benefit with the Hough transform is that it is well suited for dealing with noise and fragmentations.

The Hough transform transforms image features to a point in parameter space and vice versa. An easy illustrative example of the Hough transform is when it is used for line detection. Think of a line $y = mx + b$ which can be represented as a point (b, m) in parameter space representing all possible points on that line. Vice versa a point (x, y) in image space becomes a line in parameter space representing all possible lines that can go through that point. The general procedure is to apply an edge detection filter to an image and then use the Hough transform on each detected edge pixel. This will correspond to many crossing lines in parameter space. The points in parameter space where most interactions occur, i.e. local maximas, represents the parameters for the lines detected in image space.

The Hough transform has been used in several road network extraction systems for different reasons. The most common purpose is to detect straight road segments as in [17], where Jin and Davis combines the Hough transform with the pixels angular texture signature. The transform has also been used for purposes such as automatic seeding and to connect road segments [1].

2.3.7 Mathematical morphology

Mathematical morphology is a technique for analyzing and processing geometrical structures based on set theory. The basic idea is to test how a predefined shape called the structuring element or kernel fits or misses the shapes in the image. Originally mathematical morphology was used solely on binary images but was later generalized to be able to handle gray scale images. There are two basic morphological operators called dilation which causes objects to grow in size and erosion which vice versa causes objects to shrink. Based on these two operations other operations such as opening, closing and thinning of objects can be achieved.

In road network extraction systems mathematical morphology is often used in a preprocessing stage for feature detection [7] or in a post processing step to extract the final road skeleton [43]. The method has also been used to enhance or remove certain features as well as closing gaps between road segments.

2.3.8 Snakes

Snakes or active contour models are deformable lines that are adjusted to fit features of interests such as roads. They were first proposed by Kass *et al.* in 1988 [19]. The position of a snake is influenced by two forces known as the internal forces and the image forces. The image forces originates from the image and pulls the snake towards features of interests such as roads whereas the internal forces makes the snake resist deformations. By combining these forces into an energy function the problem of aligning the snake to the features is transformed to finding the minimum of the function. Before the minimization problem can be solved the snake is initialized into the image. Good initial position of the snake is of utmost importance for successful feature extraction which is a significant drawback to the method.

The original snakes developed by Kass *et al.* considers line shaped active contour models which have no width. A later addition to the concept called ribbon snakes adds a width property to the snake which allows it to align itself to the boundaries of a feature. This is of course a very good property for road network extraction from very high resolution images. In [22], Laptev *et al.* used ribbon snakes to remove spurious features detected by a line detection algorithm and to connect road segments.

2.3.9 Multi-resolution analysis

Multi-resolution analysis is a theoretical technique where one analyzes an image represented in different resolutions. In road network extraction multi resolution analysis is useful because of the fact that certain road features becomes more apparent at different resolutions. In a course resolution roads looks like long lines whereas in high resolution images roads are represented as long homogeneous regions where cars and other nearby objects are visible. The general approach is to apply either different or the same road extraction techniques at different resolutions and then combine the results to achieve an accurate road extraction. In the road network extraction literature there exists many variations of multi-resolution analysis approaches. An example is [13], where Gruen and Li used the Wavelet transform combined with dynamic programming to extract road networks.

2.3.10 Conclusion and motivation

Based on the above literature review the method which was chosen to be most suitable for detecting roads from remote sensed images was the artificial neural network approach used by Mnih [26]. Mainly the method was chosen due to the previously achieved results which were very promising. Secondly it was chosen due to the distribution free property that artificial neural networks possess, i.e. they do not rely on assumptions on how roads look like. This property makes artificial neural networks superior in many aspects compared to the other methods which usually are based on criteria for the appearance of roads. The problem with these methods is that they fail as soon as the roads do not fulfill their criteria which often happens in real world data. In the coming chapters the method developed by Mnih [26] will further be investigated.

Chapter 3 Theory

In this chapter a short introduction to artificial neural networks using supervised learning is presented. Basic knowledge about the network architecture known as convolutional neural networks which is used in the implemented road network extraction algorithm is also given.

3.1 Artificial neural networks

Artificial neural networks are a family of statistical learning algorithms which are inspired by biological neural networks. The goal of an artificial neural network is that provided a set of input values it should produce desired output values. Artificial neural networks are generally presented as systems of interconnected nonlinear components known as neurons. The connections between the neurons are known as weights which can be tuned by learning algorithms. The most common implementation of an artificial neural network is a fully connected feedforward neural network (FFNN). In a fully connected FFNN the neurons are arranged in layers and each neuron is connected to all of the neurons in the previous layer. There are three kind of layers: the initial input layer, the intermediate hidden layers, and the last output layer. The input to the network is provided by assigning values to the neurons in the input layer and the output of the network are the calculated values of the output neurons. There can be arbitrary many hidden layers in a network and if the network has more then one hidden layer it is said to be a deep neural network. See Fig. 3.1 for a schematic illustration of a fully connected feedforward artificial neural network with one hidden layer.

An artificial neural network computation is called a forward pass where values for every neuron is calculated sequentially layer by layer starting from the input layer. By numbering the layers in increasing order starting from the input layer the neuron values of a fully connected FFNN are calculated according to

$$y_i^{(l)} = \sigma^{(l)} \left(b^{(l)} + \sum_{j=1}^{N^{(l-1)}} w_{ij}^{(l-1)} y_j^{(l-1)} \right) \tag{3.1}$$

where $N^{(l-1)}$ is the number of neurons in layer $(l-1)$, $y_i^{(l)}$ is the output value of neuron i in layer (l), $y_j^{(l-1)}$ is the output value of neuron j in layer $(l-1)$, $w_{ij}^{(l-1)}$ is the weight connecting neuron j in layer $(l-1)$ to neuron i in layer (l), $b^{(l)}$ is an adaptive bias term for layer (l) and $\sigma^{(l)}$ is an activation function for layer (l). The activation function exists to add non linearity to the network which makes it able to solve non-linearly separable problems. The most common activation functions are the sigmoid function, hyperbolic tangent function and the rectified linear function.

$$\text{Sigmoid function: } \sigma(x) = 1/(1+e^{-x}) \tag{3.2}$$

$$\text{Hyperbolic tangent function: } \sigma(x) = \tanh(x) \tag{3.3}$$

$$\text{Rectified linear function: } \sigma(x) = \max(\mathrm{x}, 0) \tag{3.4}$$

To make an artificial neural network able to produce the desired outputs one has to apply a learning algorithm that adapts the weights and biases of the network accordingly. A learning algorithm trains the network using training data consisting of a set of inputs I and a corresponding set of desired outputs O. For every input-output pair (I_i, O_i) the algorithm does a forward pass through the network. The resulting network output $f(I_i)$ is then compared to the desired output O_i using an evaluation function called the loss function $L(f(I), O)$. The goal of the learning is to minimize the value of this function. Based on the result of the loss function the weights and biases are adapted to make $f(I_i)$ more similar to O_i. To avoid overfitting[1] of the training data a separate data set known as the validation set $(\hat{I}, \hat{O})$ is used to evaluate how well the neural network is doing during the training. The training process is iterative and one learning pass of the full training set is known as an epoch, usually several epochs are required to sufficiently train an artificial neural network. To avoid cycles the order in which the learning algorithm runs through the training set is randomly shuffled between each epoch. The training is terminated when no more improvements can be seen on the validation set.

[1] A neural network which overfits the training data is bad at generalizing to other data.

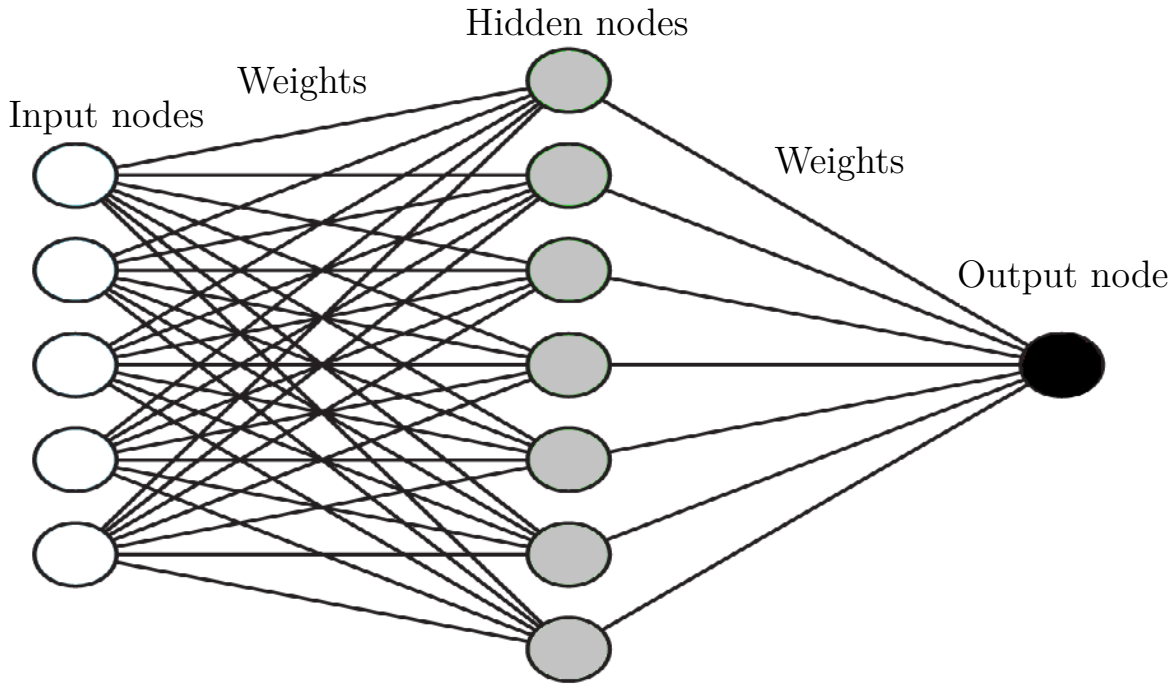

Figure 3.1: Schematic illustration of a fully connected feedforward neural network with one hidden layer. The circles represents the neurons and the lines represents the connection weights between the neurons.

The most common learning algorithm is called the backpropagation algorithm using gradient descent [32]. The learning is conducted by calculating the gradient for the loss function for every weight and bias in the network i.e

$$\frac{\partial L(f(I), O)}{w_{ij}^{(l)}} \quad \text{and} \quad \frac{\partial L(f(I), O)}{b^{(l)}} \tag{3.5}$$

The weights and biases are then updated according to the update rule

$$w_{ij}(t+1) = w_{ij}(t) - \eta \cdot \frac{\partial L(f(I), O)}{\partial w_{ij}(t)} \tag{3.6}$$

$$b(t+1) = b(t) - \eta \cdot \frac{\partial L(f(I), O)}{\partial b(t)} \tag{3.7}$$

where $w_{ij}(t+1)$ and $b(t+1)$ are the new weights and biases and η is a parameter called the learning rate which controls the speed of the learning. The learning can be done using stochastic learning, full-batch learning or mini-batch learning. In stochastic learning the network weights gets updated after each single training example. In full-batch learning the weight gradients gets added together over an entire epoch before the network weights gets updated. Finally in mini-batch learning, which is a compromise of the previous two, the network weights gets update using weight gradients added together from small random subsets of the training set. The size of these subsets is known as the batch size.

Note that the introduction given in this section is only scratching the surface of the field of artificial neural networks. Many different types of network architecture and methods for improving the learning of a network exist. An example of such an architecture is a so called convolutional neural network which is used in the road extraction algorithm and will be explained more in detail in the next section.

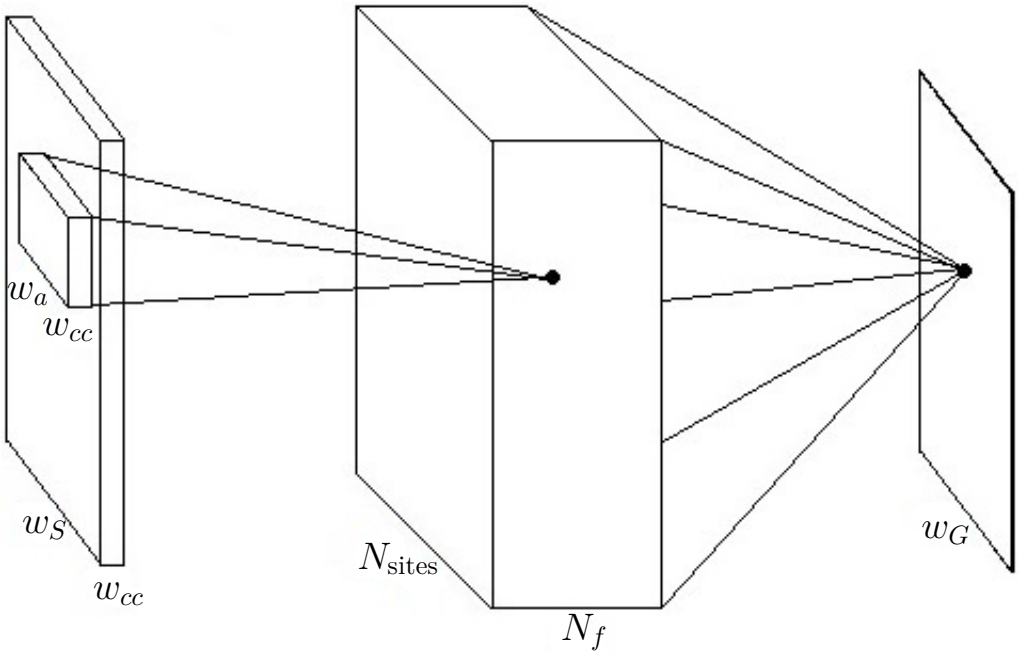

Figure 3.2: Schematic illustration of the used convolutional neural network architecture.

3.2 Convolutional neural network

In [26], Mnih tested different kinds of neural network architectures and found that an architecture known as convolutional neural networks produced the best results for road network extraction. Hence this network architecture was used in this project.

Convolutional neural networks were first proposed by Fukushima in 1980 [11] and have since been improved making the technique state-of-the-art for several image analysis tasks [21]. The inspiration for convolutional neural networks comes from the visual cortex of animals which utilizes the concept of replicated features. This means that if a feature detector is useful in one place of an image it is likely to be useful somewhere else in the image as well. Traditionally a convolutional neural network consists of three types of layers known as convolutional layers, pooling layers and fully connected layers. The network is built up by alternating between convolutional layers and pooling layers starting from the input layer. The output of the network is produced by one or more fully connected layers. In this project the network implementation included only one convolutional layer followed by a fully connected output layer and can be seen in Fig. 3.2. The reason for this relatively simple choice of network architecture is that building up a larger network would require more implementation and computation time which simply was out of the scope of this project. A deeper network would however probably produce better results, which was shown by Mnih [26].

A convolutional layer differs from a fully connected layer by utilizing the two concepts of local connectivity and weight sharing. By local connectivity it is meant that each neuron is only connected to a small subset of the neurons in the previous layer and weight sharing means that several weights to the layer are restricted to be identical. To understand the connection pattern of a convolutional layer it is beneficial to organize the layer and its previous layer as rectangular prisms of neurons as in Fig. 3.2. View the previous layer as an image of size $w_S \times w_S \times w_{cc}$. If the previous layer is the input layer the rectangular prism depth w_{cc} corresponds to the number of color channels, i.e. three in the case of a RGB images, and the neuron values are processed pixel values. The weights to a convolutional layer are also organized into a set of rectangular prisms known as filters or kernels of size $w_a \times w_a \times w_{cc}$. The biases are organized so that each filter has its own bias parameter.

To conduct a forward pass through the network each of these filters are moved, both vertically and horizontally, over the previous layer using a set step size known as the stride w_{str}. This results in a total of $((w_S - w_a)/w_{\text{str}} - 1)^2$ number of different filter sites N^2_{sites}. At each filter site the weights of the filters are con-

volved with the neurons in the previous layer resulting in the rectangular prism of neurons of the convolutional layer with the size $N_{\text{sites}} \times N_{\text{sites}} \times N_f$ according to

$$y_{ijf}^{(l)} = \sigma^{(l)}(x_{ijf}) = \sigma^{(l)}\left(b_f^{(l)} + \sum_a^{w_a}\sum_b^{w_a}\sum_c^{w_{cc}} w_{cabf} \cdot y_{(i+a)(j+b)c}^{(l-1)}\right) \tag{3.8}$$

where N_f is the number of filters, $y_{ijf}^{(l)}$ is the value of neuron ijf in the convolutional layer, $y_{(i+a)(j+b)c}^{(l-1)}$ is the value of neuron $(i+a)(j+b)c$ in the previous layer, and $b_f^{(l)}$ is the bias of filter f for the convolutional layer. One of the big advantages of a convolutional layer compared to a fully connected layer is that the number of parameters for the layer, i.e. number of weights and biases, are reduced significantly which saves both memory and speeds up the learning.

Pooling layers essentially are down sampling filters which reduces the number of neurons from one layer to the next. A pooling layer divides the previous layer into separate regions of neurons. Each neuron region in the previous layer produces one neuron in the next layer. The produced neuron value is usually calculated using max pooling where simply the maximum neuron value in each region is propagated forward or by mean pooling where the mean value of each neuron region is propagated forward. Pooling layers are usually used in deep neural networks to reduce the number of neurons from one layer to the next. The reason for that no pooling layer is used in the road extraction network in Fig. 3.2 is due the fact that they result in loss of spatial information which is not wanted in a road labeling system.

Getting meaningful network output using only convolutional and pooling layers is often very hard. Because of this the final layer or layers of a convolutional neural network are fully connected layers which can more easily produce the wanted output.

Chapter 4 Method

The method which was chosen in Chapter 2 was proposed by Mnih [26]. It is based on a patch-based framework using one or several artificial neural networks to extract road networks from an aerial image, see below.

4.1 Method overview

The patch-based framework uses the artificial neural networks to label road and non-road pixels in a remote sensed image. As described in Sect. 3.1 a training set is needed to train an artificial neural network. For the task of labeling road and non-road pixels the training set consists of a set of input remote sensed images $(S = S^{(1)}, ..., S^{(N)})$ and a corresponding set of desired road maps which in this thesis are known as the gold images $(G = G^{(1)}, ..., G^{(N)})$. The remote sensed images are usually normal RGB images but can have arbitrary number of image channels. The gold images are binary images where road pixels are labeled 1 and non-road pixels are labeled 0, see Fig. 4.1 for an example of such an image pair. To teach an artificial neural network able to handle a complex image analysis task such as this a lot of labeled training data is needed. Fortunately, in the area of road extraction there is an abundance of labeled data available. The main sources for road labeled data currently are Google Maps or Open Street Map. In this project data were taken from Google Maps. An important remark to make is that even though a lot of training data exist, the data are pretty noisy meaning that mislabeled pixels are a common occurrence. This will be discussed further in Chapter 6.

The idea of the patch-based framework is that the satellite input images are divided into a set of square map patches $\hat{S}$ of size $w_S \times w_S$. Based on these map patches road labels are predicted for the inner most pixels located in square patches $\hat{G}$ of size $w_G \times w_G$ centered at the middle of the patches $\hat{S}$. Since information about the surrounding area to the predicted pixels is important w_S is set to be larger then w_G. This results in that the $\hat{S}$ patches will overlap and that the resulting road map image will be smaller then the input image. See Fig. 4.2 for an illustration of the patch-based framework approach. Let $\hat{g}$ and $\hat{s}$ be the vectors consisting of the pixels of a gold image patch $\hat{G}$ and the corresponding satellite input patch $\hat{S}$ respectively. The neural network uses the patch $\hat{s}$ as input to calculate output values $f_i(\hat{s}) \in [0, 1]$ representing the probability that a pixel is a road pixel. If $f_i(\hat{s}) > 0.5$ the predicted road pixel $\hat{p}_i$ is set to 1 otherwise 0.

4.2 Network details

As stated in Sect. 3.2 the used artificial neural network consisted of a single convolutional layer followed by a fully connected layer, see Fig. 3.2.

The activation function used in the convolutional layer was the rectified linear function given in Eq. (3.4). The reason for this choice is simply that it has been shown that the rectified linear function outperform the other common activation functions in convolutional neural networks [21]. Since the wanted output is probability values for road pixels the sigmoid function given in Eq. (3.2), which ensures $f_i(\hat{s}) \in [0, 1]$, was used in the fully connected output layer.

The loss function used for training the network was chosen to be the negative log likelihood function $L(S, M)$ since it has been shown to outperform other common loss functions for training an artificial neural network [27].

$$L(f(S), G) = -\sum_{\forall \text{patches}} \sum_{k=1}^{w_g^2} (\hat{g}_k \log f_k(\hat{s}) + (1 - \hat{g}_k) \log(1 - f_k(\hat{s}))) \tag{4.1}$$

Backpropagation with gradient descent using momentum, L2 weight decay and mini-batch update was used as the training algorithm for the network. Momentum is a technique used for speeding up the training of a neural network. Weight decay is an approach which is used to make a neural network generalize better to other data.

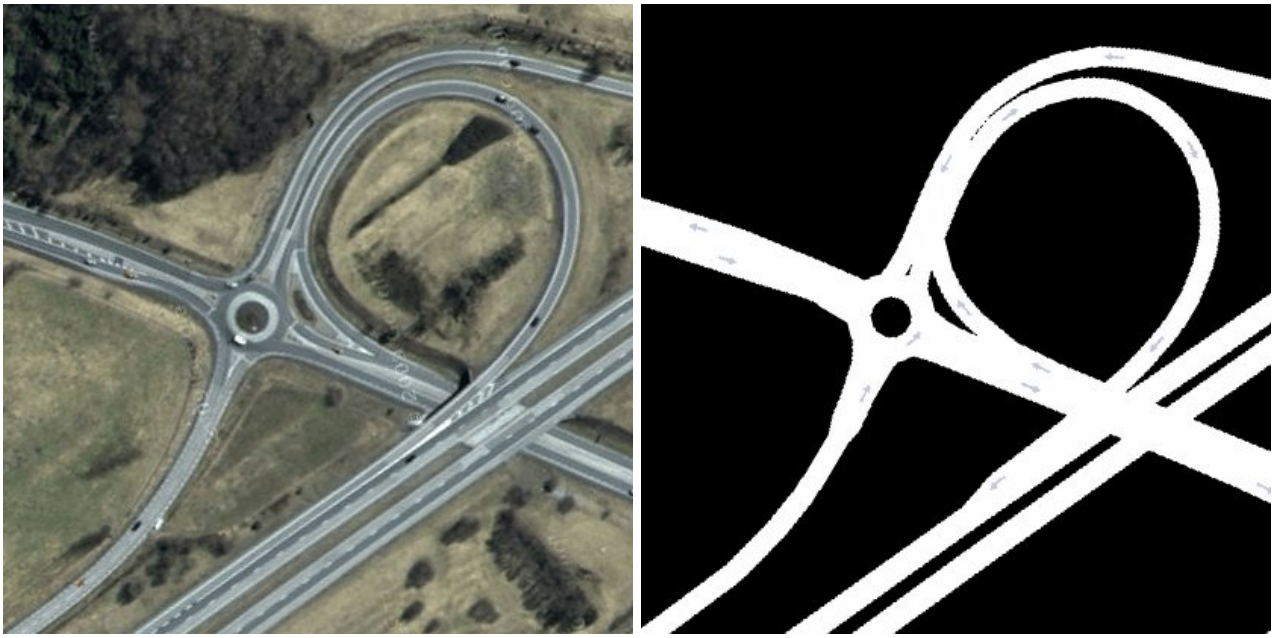

Figure 4.1: Example of a satellite image and the corresponding gold image taken from Google maps. The gray arrows seen in the gold image displaying the direction of the road are set to road pixels in the algorithm.

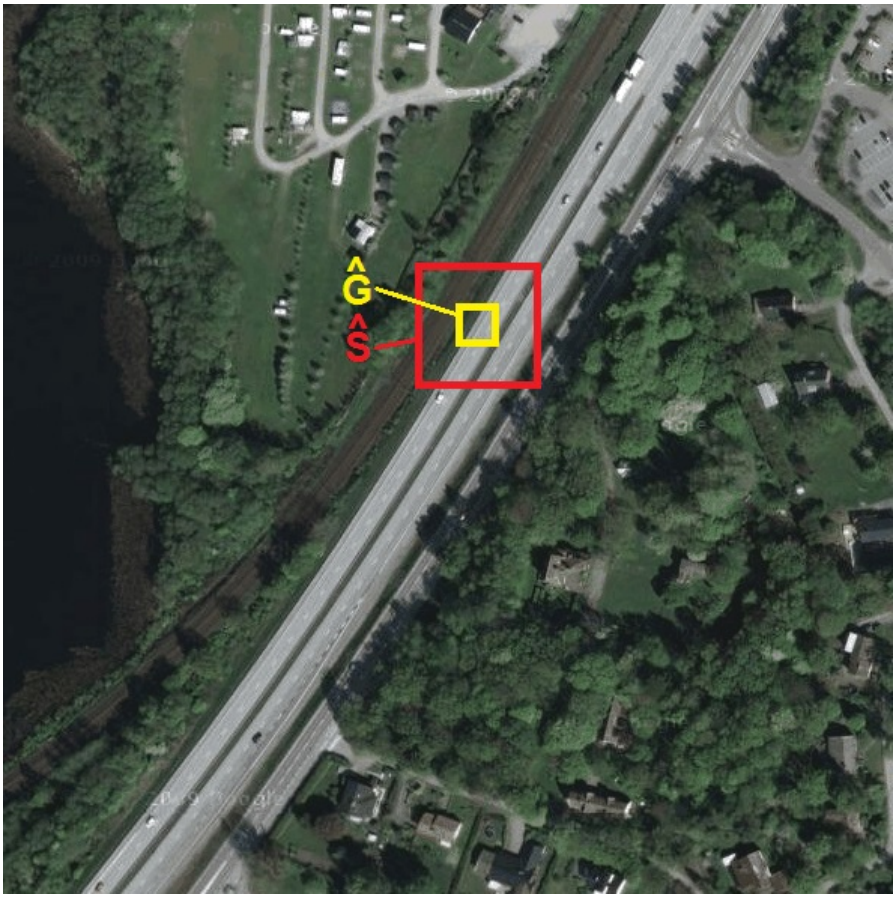

Figure 4.2: Illustration of the patch-based framework. Pixels in the outer patch $\hat{S}$ are used as input to an artificial neural network which output are road label prediction for the pixels in the inner patch $\hat{G}$.

This resulted in the weight update rules

$$w(t+1) = w(t) + \left(\alpha w(t) - \lambda \eta w(t) - \eta \frac{\partial L}{\partial w(t)} \right) \tag{4.2}$$

$$b(t+1) = b(t) + \left(\alpha b(t) - \lambda \eta b(t) - \eta \frac{\partial L}{\partial b(t)} \right) \tag{4.3}$$

where η is the learning rate, λ is the L2 weight decay, and α is the momentum parameter. The weight and bias gradients are calculate using the backpropagation algorithm according to Subsect. 4.2.1. At the end of the derivation there is a summary of the resulting set of equations.

4.2.1 Backpropagation derivation

Using the notation and knowledge from Chapter 3 the following equations are known

$$y_k^{\text{out}} = \sigma^{\text{out}}(x_k^{\text{out}}) = \sigma^{\text{out}} \left(\sum_i^{N_{\text{sites}}} \sum_j^{N_{\text{sites}}} \sum_f^{N_f} w_{kijf}^{\text{out}} y_{ijf}^{\text{con}} + b_k^{\text{out}} \right) \tag{4.4}$$

$$y_{ijf}^{\text{con}} = \sigma^{\text{con}}(x_{ijf}^{\text{con}}) = \sigma^{\text{con}} \left(\sum_a^{w_a} \sum_b^{w_b} \sum_c^{w_{cc}} w_{cabf}^{\text{con}} y_{(i+a)(j+b)c}^{\text{in}} + b_f^{\text{con}} \right) \tag{4.5}$$

where $f_k(\hat{s})$ from Sect. 4.1 has been renamed to y_k^{out} for simplicity. The superscripts in, con and out stands for input layer, convolution layer and output layer.

Output layer:
Using the chain rule for the bias and weight gradients one gets

$$\frac{\partial L}{\partial w_{kijf}^{\text{out}}} = \frac{\partial L}{\partial x_k^{\text{out}}} \frac{\partial x_k^{\text{out}}}{\partial w_{kijf}^{\text{out}}} = y_{ijf}^{\text{con}} \frac{\partial L}{\partial x_k^{\text{out}}} \tag{4.6}$$

$$\frac{\partial L}{\partial b_k^{\text{out}}} = \frac{\partial L}{\partial x_k^{\text{out}}} \frac{\partial x_k^{\text{out}}}{\partial b_k^{\text{out}}} = \frac{\partial L}{\partial x_k^{\text{out}}} \tag{4.7}$$

where

$$\frac{\partial L}{\partial x_k^{\text{out}}} = \frac{\mathrm{d}L}{\mathrm{d}y_k^{\text{out}}} \frac{\partial y_k^{\text{out}}}{\partial x_k^{\text{out}}} = \frac{\mathrm{d}L}{\mathrm{d}y_k^{\text{out}}} \sigma^{\text{out}'}(x_k^{\text{out}}) = \delta_k^{\text{out}} \tag{4.8}$$

This gives the following bias and gradient equations for the output layer

$$\frac{\partial L}{\partial w_{kijf}^{\text{out}}} = y_{ijf}^{\text{con}} \delta_k^{\text{out}} \qquad \frac{\partial L}{\partial b_k^{\text{out}}} = \delta_k^{\text{out}} \tag{4.9}$$

By inserting the sigmoid function, given in Eq. 3.2, as the activation function and the negative log likelihood function, given in Eq. 4.1, as the loss function it follows that

$$\delta_k^{\text{out}} = [\sigma^{\text{out}'}(x_k^{\text{out}}) = y_k^{\text{out}}(1 - y_k^{\text{out}})] = -\left(\frac{\hat{g}_k}{y_k^{out}} - \frac{1 - \hat{g}_k}{1 - y_k^{out}} \right) y_k^{\text{out}}(1 - y_k^{\text{out}}) = -(\hat{g}_k - y_k^{\text{out}}) \tag{4.10}$$

Convolutional layer:
Using the same technique as for the output layer it follows that

$$\frac{\partial L}{\partial w_{cabf}^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \frac{\partial L}{\partial x_{ijf}^{\text{con}}} \frac{\partial x_{ijf}^{\text{con}}}{\partial w_{cabf}^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \frac{\partial L}{\partial x_{ijf}^{\text{con}}} y_{(i+a)(j+b)c}^{\text{in}} \tag{4.11}$$

$$\frac{\partial L}{\partial b_f^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \frac{\partial L}{\partial x_{ijf}^{\text{con}}} \frac{\partial x_{ijf}^{\text{con}}}{\partial b_f^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \frac{\partial L}{\partial x_{ijf}^{\text{con}}} \tag{4.12}$$

where

$$\frac{\partial L}{\partial x_{ijf}^{\text{con}}} = \frac{\partial L}{\partial y_{ijf}^{\text{con}}} \frac{\partial y_{ijf}^{\text{con}}}{\partial x_{ijf}^{\text{con}}} = \frac{\partial L}{\partial y_{ijf}^{\text{con}}} \sigma^{con'}(x_{ijf}^{\text{con}}) \tag{4.13}$$

$$\frac{\partial L}{\partial y_{ijf}^{\text{con}}} = \sum_{k}^{w_G^2} \frac{\partial L}{\partial x_k^{\text{out}}} \frac{\partial x_k^{\text{out}}}{\partial y_{ijf}^{\text{con}}} = \sum_{k}^{w_G^2} \frac{\partial L}{\partial x_k^{\text{out}}} w_{kijf}^{\text{con}} \tag{4.14}$$

Setting $\delta_{ijf}^{\text{con}} = \frac{\partial L}{\partial x_{ijf}^{\text{con}}}$ gives the following bias and gradient equations for the convolutional layer

$$\frac{\partial L}{\partial w_{cabf}^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \delta_{ijf}^{\text{con}} y_{(i+a)(j+b)c}^{\text{in}} \qquad \frac{\partial L}{\partial b_f^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \delta_{ijf}^{\text{con}} \tag{4.15}$$

Inserting that the activation function for the convolutional layer is the rectified linear function, given in Eq. 3.4, gives

$$\delta_{ijf}^{\text{con}} = \begin{cases} \sum_{k}^{w_G^2} \delta_k^{\text{out}} w_{kijf}^{\text{out}} & \text{if} \quad x_{ijf}^{\text{con}} > 0 \\ 0 & \text{if} \quad x_{ijf}^{\text{con}} <= 0 \end{cases} \tag{4.16}$$

Summary:
The above derivation lead to the following weight and bias gradient equations

$$\frac{\partial L}{\partial w_{kijf}^{\text{out}}} = \delta_k^{\text{out}} \cdot y_{ijf}^{\text{con}} \qquad \frac{\partial L}{\partial w_{cabf}^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \delta_{ijf}^{\text{con}} \cdot y_{(i+a)(j+b)c}^{\text{in}}$$

$$\frac{\partial L}{\partial b^{\text{out}}} = \delta_k^{\text{out}} \qquad \frac{\partial L}{\partial b_f^{\text{con}}} = \sum_{i,j}^{w_S - w_a} \delta_{ijf}^{\text{con}}$$

$$\delta_k^{\text{out}} = -(\hat{g}_k - y_k^{\text{out}}) \qquad \delta_{ijf}^{\text{con}} = \begin{cases} \sum_{k}^{w_G^2} \delta_k^{\text{out}} w_{kijf}^{\text{out}} & \text{if} \quad x_{ijf}^{\text{con}} > 0 \\ 0 & \text{if} \quad x_{ijf}^{\text{con}} <= 0 \end{cases}$$

where in, con and out stands for input layer, convolution layer and output layer.

4.3 Implementation

The road network extraction algorithm was implemented using *C#*. Since a neuron calculation is independent of the other neuron calculations in the same layer in both the forward and backward propagation step, high parallelism can be achieved in neural networks. By using the Task Parallel Library in the .NET framework the road extraction algorithm was implemented to run on parallel processor cores. This increased the performance of the algorithm almost linearly by the number of available processor cores. Even higher parallelism could have been achieved by implementing the network on a graphical processing unit (GPU) instead of an central processing unit (CPU). A project was started implementing this using NVIDIAs compute unified device architecture (CUDA). However due to the time limit of the project and the relatively small network used this implementation was not finished.

The neural network was trained using a training set consisting of 172 images and a validation set consisting of 10 images of size 416×416 taken from areas surrounding motorways in the southwestern part of Sweden. The size of the validation set might seem small but one should remember that 10 images of size 416×416 corresponds to 5290 validation patches. Since the images mainly were taken of highways the roads in the different images often had the same general heading. Using these images directly would probably teach the network to detect roads better if they had a certain direction. To avoid this effect the images in the training set were all randomly rotated before used for training. The maximum image size available from Google Maps is 640×640. The cropping effect of rotating an image of this size is the reason for the seemingly arbitrary used input image size of 416×416. To try and teach the network many different road types the training set was chosen to be as large as possible. Therefore, the full training set could not be loaded into the main memory all at once. In order to still achieve a mini-batch training scheme, the training set was in the beginning of each epoch randomly divided into

several smaller sets. An epoch consisted of loading these image sets one at a time into memory. For each loaded image set, the algorithm could then perform stochastic mini-batch learning, as described in Sect. 3.1 and Sect. 4.1.

During training it is beneficial to have a decreasing learning rate to limit the search range and an increasing momentum to speed up the learning. Thus, the learning rate and the momentum were adapted during the training to achieve better performance. The learning rate η was halved and the momentum α was increased by 0.1 for every other epoch until the final parameter values were reached. The following neural network parameter values were used during the training.

w_S :	64	η :	$2.5 \cdot 10^{-6} \rightarrow 3.125 \cdot 10^{-7}$
w_G :	16	α :	$0.6 \rightarrow 0.9$
w_a :	12	λ :	0.0001
N_f :	80	batch size :	100
w_{str} :	4		

Table 4.1: Parameters used for the convolutional neural network seen in Fig. 3.2.

The classification error of the validation set, i.e. the number of incorrectly classified pixels, of the validation set was calculated after each epoch. The parameters of the network which achieved the lowest error were saved. The training was conducted on a laptop with a Intel Core i7-2650M (2.80 Ghz) processor with four cores. A full network training took around 24 hours to complete using this computer.

4.3.1 Incorporating structure

Many of the methods discussed in Chapter 2 used a post-processing step to incorporate structure in their road extraction algorithms. Since the patch-based framework method classifies individual pixels, such a post-processing step which smooths and cleans up the result of the neural network would probably improve the extracted road network. To achieve this effect a second artificial neural network known as the structuring network was used. The structuring network had the same architecture as the first road extraction network but it was trained using different input data. Instead of using the raw Google Map images as input the structuring network was trained using the road probability maps produced by the fully trained road extraction network P while keeping the original gold images G as the desired output. Note that the road probability maps were not thresholded before they were used as input to the structuring network. A flowchart of the entire road extraction system can be seen in figure 4.3.

As stated above the hope is that this second network will learn to clean up and improve the results from the first network. One reason it is able to do this is that the structuring network is able to incorporate higher level dependencies between the Google Map pixels then the first network. The reason is that the probability map pixels already are based on surrounding pixel information which gives that information from a larger patch is used for the second network. More precisely, if the structuring network uses the same patch sizes as the first network, patches of size $(2w_S - w_G) \times (2w_S - w_G)$ is used to predict the pixels in the second network.

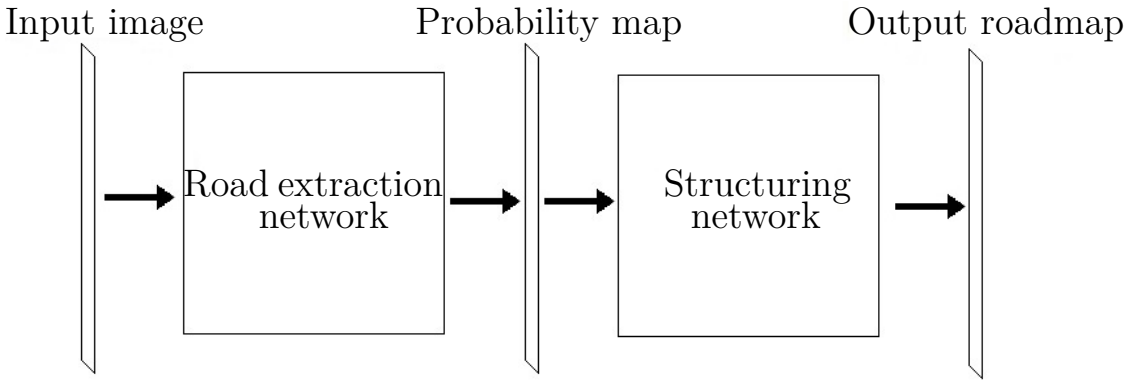

Figure 4.3: Flowchart of the road extraction algorithm using a structuring network.

Chapter 5 Results

The road extraction system was evaluated using four previously unseen remote sensed images taken from areas around the motorway E6 connecting the cities Malmö and Oslo, see Fig. 5.1. The images will be referred to as from top to bottom in Fig. 5.1 ”Lindome”, ”Lomma”, ”Malmö” and ”Helsingborg”. The evaluation images where chosen to display the algorithms performance on some common type of urban Swedish road scenes. In ”Lindome” a motorway can be seen running through a tree covered area, in ”Lomma” a motorway is running through a field area with buildings, in ”Malmö” a big motorway exit can be seen and in ”Helsingborg” a suburban area in the vicinity of a motorway can be seen.

To evaluate road extraction systems the following quantities are calculated.

- True positive (TP): Number of correctly predicted road pixels.
- True negative (TN): Number of correctly predicted non road pixels.
- False positive (FP): Number of incorrectly predicted road pixels.
- False negative (FN): Number of incorrectly predicted non road pixels.

These are then combined into the evaluation metrics precision, recall and quality where the first two also are known as correctness and completeness.

Precision is the fraction of predicted road pixels which was correctly labeled i.e. the percentage of the extracted road network which was covered by the road network taken from the gold images.

$$precision = \frac{TP}{TP + FP}$$

Recall is the fraction of true road pixels that were correctly labeled i.e. the percentage of the gold image road network that was extracted.

$$recall = \frac{TP}{TP + FN}$$

Quality is a combination of of the precision and recall metrics .

$$quality = \frac{TP}{TP + FP + FN}$$

The following results for the road extraction algorithm with and without a structuring network were found for the four evaluation images in Fig. 5.1. The numbers in parenthesis are the results for the algorithm using no structuring network.

Image name:	Recall:	Precision:	Quality:
Lindome	0.86 (0.88)	0.93 (0.93)	0.81 (0.82)
Lomma	0.78 (0.61)	0.92 (0.93)	0.73 (0.59)
Malmö	0.90 (0.85)	0.91 (0.92)	0.83 (0.79)
Helsingborg	0.84 (0.74)	0.75 (0.75)	0.66 (0.59)
Average	0.845 (0.77)	0.878 (0.883)	0.76 (0.70)

Table 5.1: Results of the road extraction algorithm for the four images which can be seen in Fig. 5.1. The numbers in parenthesis are the results for a single network algorithm which used no structuring network.

Based on visual analysis of the evaluation images and the table above it can be concluded that the algorithm does fairly well on most type of road images. Even though the extracted roads are a bit jagged the algorithm manages to extract the major part of them while at the same time filter out the different type of backgrounds such as forests, fields and buildings. The algorithm is even able to handle suburban areas fairly well, as can be seen in the ”Helsingborg” image. This is a bit surprising since only a small amount of training images contained such road networks. The algorithm is far from perfect though. For example an area where the algorithm fails can be seen in the ”Lomma” image where a road blocked by heavy shadows from nearby trees is

Figure 5.1: The four aerial images used for evaluating the road extraction system. In the left column are the unprocessed Google Map images, in the middle column are the gold images taken from Google Maps and in the right column are the predicted road maps. The images will be referred to as, from top to bottom, ”Lindome”, ”Lomma”, ”Malmö” and ”Helsingborg”.

not extracted at all. The algorithm also has some problems in the vicinity of roads that crosses over the motorway.

Comparing the results achieved with and without a structuring network it can be seen that the preprocessing step made a significant improvement for the recall and quality metrics while roughly performing equally on the precision metrics. This means that by using the structuring network the algorithm labeled more pixels as road pixels which made it extract a higher percentage of the road network giving it a higher recall value. At the same time a few of those newly labeled road pixels were mislabeled making the precision metric stay the same. Overall however the quality of the road map increased by quite a margin. This shows that the structuring network achieved the wanted result and managed to incorporate some structure to the data by filling out various holds in the road network.

The choice of training set and evaluation set greatly impacts the results of a road extraction system. The images which are shown in Fig. 5.1 where chosen to give a fairly accurate evaluation of the algorithm even though both better and worse examples exists. In fact for most road extraction systems almost any results can be achieved depending on which type of road images are chosen for evaluation. Since different road extraction systems in almost all cases are trained and evaluated against different data sets a fair comparisons between the systems are almost impossible to make. Despite this a few published results of fully automatic road extraction system compared to this road extraction system (Sirefelt) can be seen in table 5.2.

Methods:	Recall:	Precision:	Quality:
Hauptfleisch (Spectral classifier) [15]	0.71	0.70	0.54
Yuan (LEGION) [41]	0.77	0.66	*
Mnih (DCNN) [26]	0.90	0.90	*
Moktarzade (ANN) [29]	*	0.75	*
Baumgartner (Multi-Scale) [3]	*	0.76	*
Poullis (Tensor Cuts) [31]	0.68	0.64	0.62
Sirefelt (CNN)	0.845	0.878	0.76

Table 5.2: Results of various published road extraction algorithms. A * symbol means that no result was published for that metric.

Chapter 6 Discussion and Conclusion

In table 5.2 it can be seen that this thesis road extraction system (Sirefelt) outperforms all of the other methods except the one developed by Mnih [26], on which it was based. Except for the fact that it is hard to compare road extraction systems there are a few probable reasons why Mnih got a better result. One is that Mnih trained a deeper neural network with several layers which could detect more complex features. He also used more training data which probably made his network able to generalize better to different road scenes. Both of these factors do however increase the computation time by a significant margin. It is noticeable that the Sirefelt road extraction system almost has the same performance results as Mnih using a simpler algorithm which could be trained on a normal laptop. Since both of the methods, Sirefelt and Mnih, which was based on convolutional neural networks outperformed the others by a significant margin it is pretty safe to say that the currently best method for automatic extraction of road networks was chosen.

Despite the good result the method does have a few downsides which should be addressed. First of all training a convolutional neural network requires a large amount of correctly labeled training data. As already stated in the field of road extraction it is easy to get a hold of a large amount of labeled data but it is often noisy and not perfect. The gold images taken from Google Maps for example have several mislabeled road pixels. This mislabeling error gives rise to two sources of error for the algorithm. The obvious one is that the neural network sometimes are taught that a non-road pixel is a road pixel and vice versa which makes it perform worse. How much and which type of harm this does to the performance of the algorithm is hard know for certain. A good guess is that it has no effect on the algorithms ability to extract the centerline of a road but has an effect on its ability to extract road pixels that are close to the road edge making them a bit jagged. The reason is simply that the road pixels close to the centerline of roads is almost never mislabeled compared to the road pixels close to a road edge. The other factor is that the evaluation of the predicted road maps gets inaccurate. This can for example be seen in the "Helsingborg" gold image where roads at certain places have the wrong width when compared to the satellite image. Driveways is another problem since it is debatable if they are roads or not and are not always labeled as such. There will therefore sometimes be a discrepancy between the algorithm and gold image if a driveway is a road or not which can be seen in the "Helsingborg" images. Because of the above it would be interesting to see how much better the algorithm would perform if all of the gold images were perfectly labeled.

Another downside to the method which was noticed is that it has a problem to generalize to roads that are located far from the roads which it was trained on. For example a network trained to extract Swedish roads performs quite poorly on American roads and vice versa. The reason is simply that roads have different appearances on separate places on earth. The lightning conditions of satellite images taken of different places can also vary quite a bit making the roads look even more different. A solution to the problem is to have more training data that teaches the network to detect roads in as many different places and lightning conditions as possible. This however, would require enormous amounts of training data and computation time. It would also probably require a quite large network. Another solution to this problem is to add a little training on an already trained network whenever it is required to extract roads from a new location. According to Mnih, only a small amount of extra training is required using roads from the new location to make it perform just as well for the new place as for the place it was originally trained on.

Since the deep neural network used by Mnih loses spatial information due the use of pooling layers and that the shallow neural network used in this thesis is not able to detect complex features it would be interesting for future work to combine the two. A network architecture where a deep neural network which can detect complex features is combined with simpler single layer convolurional neural network which preserves spatial information would hopefully outperform the road extraction algorithm by Mnih and Sirefelt. Due to several reasons such as time and resource constraints this was not tested in this project.

A final conclusion is that the currently best method for detecting road network from aerial images is using some type of machine learning technique which do not depend on previous knowledge on how roads look like e.g. one which uses convolutional neural networks. Even though the algorithm developed in this thesis is quite good it is not good enough to replace the currently used manual methods. However, since a very large amount of research is currently conducted in the fields of machine learning and image analysis an algorithm which outperforms the manual road extraction method is not far from being a reality.

References

[1] V. Amberg et al. "Structure extraction from high resolution SAR data on urban areas". *Geoscience and Remote Sensing Symposium, 2004. IGARSS'04. Proceedings. 2004 IEEE International*. Vol. 3. IEEE. 2004, pp. 1784–1787.

[2] G. H. Ball and D. J. Hall. *ISODATA, a novel method of data analysis and pattern classification*. Tech. rep. DTIC Document, 1965.

[3] A. Baumgartner et al. Automatic road extraction based on multi-scale, grouping, and context. *Photogrammetric Engineering and Remote Sensing* **65** (1999), 777–786.

[4] A. Baumgartner et al. Multi-resolution, semantic objects, and context for road extraction. *Semantic Modeling for the Acquisition of Topographic Information from Images and maps* (1997), 140–156.

[5] A. Bhattacharyya. On a measure of divergence between two multinomial populations. *Sankhyā: The Indian Journal of Statistics* (1946), 401–406.

[6] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 6 (1986), 679–698.

[7] J. Chanussot, G. Mauris, and P. Lambert. Fuzzy fusion techniques for linear features detection in multitemporal SAR images. *Geoscience and Remote Sensing, IEEE Transactions on* **37**.3 (1999), 1292–1305.

[8] E. Christophe and J. Inglada. "Robust Road Extraction for High Resolution Satellite Images". *Image Processing, 2007. ICIP 2007. IEEE International Conference on*. Vol. 5. Sept. 2007, pp. 437–440. DOI: 10.1109/ICIP.2007.4379859.

[9] G. Dial, L. Gibson, and R. Poulsen. IKONOS satellite imagery and its use in automated road extraction. *Automatic extraction of man-made objects from aerial and space images (III)* (2001), 357.

[10] P. Doucette, P. Agouris, and A. Stefanidis. Automated road extraction from high resolution multispectral imagery. *Photogrammetric Engineering & Remote Sensing* **70**.12 (2004), 1405–1416.

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

[12] *Google Ground Truth*. https://developers.google.com/events/io/2013/sessions/383278298. Accessed: 2015-04-04.

[13] A. Gruen and H. Li. Road extraction from aerial and satellite images by dynamic programming. *ISPRS Journal of Photogrammetry and Remote Sensing* **50**.4 (1995), 11–20.

[14] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE* **67**.5 (1979), 786–804.

[15] A. C. Hauptfleisch. "Automatic road network extraction from high resolution satellite imagery using spectral classification methods". PhD thesis. University of Pretoria, 2010.

[16] S. Hinz and A. Baumgartner. Automatic extraction of urban road networks from multi-view aerial imagery. *ISPRS Journal of Photogrammetry and Remote Sensing* **58**.1 (2003), 83–98.

[17] X. Jin and C. H. Davis. An integrated system for automatic road mapping from high-resolution multispectral satellite imagery by information fusion. *Information Fusion* **6**.4 (2005), 257–273.

[18] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering* **82**.1 (1960), 35–45.

[19] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. English. *International Journal of Computer Vision* **1**.4 (1988), 321–331. ISSN: 0920-5691. DOI: 10.1007/BF00133570. URL: http://dx.doi.org/10.1007/BF00133570.

[20] G. Koutaki and K. Uchimura. "Automatic road extraction based on cross detection in suburb". *Electronic Imaging 2004*. International Society for Optics and Photonics. 2004, pp. 337–344.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks". *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[22] I. Laptev et al. Automatic extraction of roads from aerial images based on scale space and snakes. English. *Machine Vision and Applications* **12**.1 (2000), 23–31. ISSN: 0932-8092. DOI: 10.1007/s001380050121. URL: http://dx.doi.org/10.1007/s001380050121.

[23] P. C. Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)* **2** (1936), 49–55.

[24] J. Mena. State of the art on automatic road extraction for {GIS} update: a novel classification. *Pattern Recognition Letters* **24**.16 (2003), 3037–3058. ISSN: 0167-8655. DOI: http://dx.doi.org/10.

1016/S0167-8655(03)00164-8. URL: http://www.sciencedirect.com/science/article/pii/S0167865503001648.

[25] J. B. Mena and J. A. Malpica. An automatic method for road extraction in rural and semi-urban areas starting from high resolution satellite imagery. *Pattern Recognition Letters* **26**.9 (2005), 1201–1220.

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

[27] V. Mnih and G. E. Hinton. "Learning to detect roads in high-resolution aerial images". *Computer Vision–ECCV 2010*. Springer, 2010, pp. 210–223.

[28] A. Mohammadzadeh, A. Tavakoli, and M. J. V. Zoej. ROAD EXTRACTION BASED ON FUZZY LOGIC AND MATHEMATICAL MORPHOLOGY FROM PAN-SHARPENED IKONOS IMAGES. *The Photogrammetric Record* **21**.113 (2006), 44–60.

[29] M. Mokhtarzade and M. V. Zoej. Road detection from high-resolution satellite images using artificial neural networks. *International Journal of Applied Earth Observation and Geoinformation* **9**.1 (2007), 32–40. ISSN: 0303-2434. DOI: http://dx.doi.org/10.1016/j.jag.2006.05.001. URL: http://www.sciencedirect.com/science/article/pii/S0303243406000171.

[30] R. Nevatia and K. R. Babu. Linear feature extraction and description. *Computer Graphics and Image Processing* **13**.3 (1980), 257–269.

[31] C. Poullis. Tensor-Cuts: A simultaneous multi-type feature extractor and classifier and its application to road extraction from satellite images. *ISPRS Journal of Photogrammetry and Remote Sensing* **95** (2014), 93–108.

[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling* **5** (1988).

[33] G. Shafer et al. *A mathematical theory of evidence*. Vol. 1. Princeton university press Princeton, 1976.

[34] J. Shen et al. "Knowledge-based road extraction from high resolution remotely sensed imagery". *Image and Signal Processing, 2008. CISP'08. Congress on*. Vol. 4. IEEE. 2008, pp. 608–612.

[35] W. Shi and C. Zhu. The line segment match method for extracting road network from high-resolution satellite images. *IEEE Transactions on Geoscience and Remote Sensing* **40**.2 (2002), 511–514.

[36] M. Song and D. Civco. Road extraction using SVM and image segmentation. *Photogrammetric Engineering & Remote Sensing* **70**.12 (2004), 1365–1371.

[37] C. Steger. An unbiased detector of curvilinear structures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **20**.2 (1998), 113–125.

[38] J. C. Trinder and Y. Wang. Automatic road extraction from aerial images. *Digital Signal Processing* **8**.4 (1998), 215–224.

[39] G. Vosselman and J. de Knecht. "Road tracing by profile matching and Kaiman filtering". *Automatic Extraction of Man-Made Objects from Aerial and Space Images*. Springer, 1995, pp. 265–274.

[40] J. Yang and R. Wang. Classified road detection from satellite images based on perceptual organization. *International Journal of Remote Sensing* **28**.20 (2007), 4653–4669.

[41] J. Yuan et al. LEGION-based automatic road extraction from satellite imagery. *Geoscience and Remote Sensing, IEEE Transactions on* **49**.11 (2011), 4528–4538.

[42] C. Zhang. Towards an operational system for automated updating of road databases by integration of imagery and geodata. *ISPRS Journal of Photogrammetry and Remote Sensing* **58**.3 (2004), 166–186.

[43] C. Zhang et al. *Road network detection by mathematical morphology*. Vol. 8093. Citeseer, 1999.

[44] Q. Zhang and I. Couloigner. Benefit of the angular texture signature for the separation of parking lots and roads on high resolution multi-spectral imagery. *Pattern Recognition Letters* **27**.9 (2006), 937–946.