NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DIOGO ANDRÉ DE NORONHA ESTIMA

Bachelor Degree In Electrical And Computer Engineering

AUTOMATIC RURAL ROAD CENTERLINE EXTRACTION FROM AERIAL IMAGES FOR A FOREST FIRE SUPPORT SYSTEM

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
November, 2021

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

AUTOMATIC RURAL ROAD CENTERLINE EXTRACTION FROM AERIAL IMAGES FOR A FOREST FIRE SUPPORT SYSTEM

DIOGO ANDRÉ DE NORONHA ESTIMA

Bachelor Degree In Electrical And Computer Engineering

Adviser: Luís Oliveira
Assistant Professor, NOVA University of Lisbon

Co-adviser: Henrique Oliveira
Assistant IT Professor, Polytechnic Institute of Beja

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
November, 2021

Automatic Rural Road Centerline Extraction from Aerial Images for a Forest Fire Support System

I dedicate this thesis to my grandparents Maria and Aluino.
I hope this achievement will complete the dream that you
had for me many years ago when you chose to give me the
best education you could.

Acknowledgements

In the present section, I would like to express my endless gratitude towards everyone involved in the process of writing and developing this thesis.

First and foremost, I would like to thank Professor Luís Oliveira for providing me with the opportunity of developing a thesis in a field of my choice, which I am truly passionate about. I was welcomed with open arms to embrace the project, and make part of this exciting journey. Professor Luís Oliveira made sure that the right tools, resources and guidance were provided.

I am forever grateful for having Professor Henrique Oliveira as my co-adviser, and I would like to especially thank him for the trust and confidence he put into my work, allowing me to find my own path and supporting me whenever necessary. Besides his guidance, experience, and patience, he also proved to be a role model and a friend. I can proudly announce that it was an honor to learn and work side-by-side with him.

I would like to thank Direção Geral do Território (DGT) for providing all of the aerial images of the Mação district that were essential for the development of the datasets, and training of the deep learning models.

I must also thank FCT/UNL for all the knowledge, friendships, and experiences that allowed me to push beyond my limits and beliefs. Since the beginning of this academic journey, I have met many professors and colleagues that shaped my world view to help me grow not only as a person, but also as a future engineer. This University granted me priceless confidence towards solving real life problems with absence of fear.

I am very thankful for being part of the *foRESTER*[1] project, being able to do research and develop systems that will eventually help mitigate forest fires and help firefighters.

And last but not least, I would like to express my infinite gratitude to my mother, my father, my sister and especially my grandparents' efforts and sacrifices, unconditional love, and timeless life lessons that will be passed throughout generations. Without you, none of this would be possible...

[1] https://forester.pt/web/sumario-pt/

"It always seems impossible until it's done." (Nelson Mandela)

Abstract

In the last decades, Portugal has been severely affected by forest fires which have caused massive damage both environmentally and socially. Having a well-structured and precise mapping of rural roads is critical to help firefighters to mitigate these events. The traditional process of extracting rural roads centerlines from aerial images is extremely time-consuming and tedious, because the mapping operator has to manually label the road area and extract the road centerline.

A frequent challenge in the process of extracting rural roads centerlines is the high amount of environmental complexity and road occlusions caused by vehicles, shadows, wild vegetation, and trees, bringing heterogeneous segments that can be further improved. This dissertation proposes an approach to automatically detect rural road segments as well as extracting the road centerlines from aerial images.

The proposed method focuses on two main steps: on the first step, an architecture based on a deep learning model (DeepLabV3+) is used, to extract the road features maps and detect the rural roads. On the second step, the first stage of the process is an optimization for improving road connections, as well as cleaning white small objects from the predicted image by the neural network. Finally, a morphological approach is proposed to extract the rural road centerlines from the previously detected roads by using thinning algorithms like the Zhang-Suen and Guo-Hall methods.

With the automation of these two stages, it is now possible to detect and extract road centerlines from complex rural environments automatically and faster than the traditional ways, and possibly integrating that data in a Geographical Information System (GIS), allowing the creation of real-time mapping applications.

Keywords: Rural Roads, Centerline Extraction, Geographic Information System, Forest Fires, Convolutional Neural Network, Deep Learning.

Resumo

Nas últimas décadas, Portugal tem sido severamente afetado por fogos florestais, que têm causado grandes estragos ambientais e sociais. Possuir um sistema de mapeamento de estradas rurais bem estruturado e preciso é essencial para ajudar os bombeiros a mitigar este tipo de eventos. Os processos tradicionais de extração de eixos de via em estradas rurais a partir de imagens aéreas são extremamente demorados e fastidiosos. Um desafio frequente na extração de eixos de via de estradas rurais é a alta complexidade dos ambientes rurais e de estes serem obstruídos por veículos, sombras, vegetação selvagem e árvores, trazendo segmentos heterogéneos que podem ser melhorados.

Esta dissertação propõe uma abordagem para detetar automaticamente estradas rurais, bem como extrair os eixos de via de imagens aéreas.

O método proposto concentra-se em duas etapas principais: na primeira etapa é utilizada uma arquitetura baseada em modelos de aprendizagem profunda (DeepLabV3+), para detetar as estradas rurais. Na segunda etapa, primeiramente é proposta uma otimização de intercessões melhorando as conexões relativas aos eixos de via, bem como a remoção de pequenos artefactos que estejam a introduzir ruído nas imagens previstas pela rede neuronal. E, por último, é utilizada uma abordagem morfológica para extrair os eixos de via das estradas previamente detetadas recorrendo a algoritmos de esqueletização tais como os algoritmos Zhang-Suen e Guo-Hall.

Automatizando estas etapas, é então possível extrair eixos de via de ambientes rurais de grande complexidade de forma automática e com uma maior rapidez em relação aos métodos tradicionais, permitindo, eventualmente, integrar os dados num Sistema de Informação Geográfica (SIG), possibilitando a criação de aplicativos de mapeamento em tempo real.

Palavras-chave: Estradas rurais, Extração de eixos de via, Sistema de Informação Geográfica, Fogos florestais, Redes Neuronais Convolucionais, Apredizagem Profunda.

Contents

List of Figures

List of Tables

List of Listings

Acronyms

AI Artificial Intelligence 22
ANN Artificial Neural Networks 22
API Application Programming Interface 46
ASPP Atrous Spacial Pyramid Pooling 40, 41

CAD Computer-Aided Design 9
CasNet Cascade End-to-End Convolutional Neural Network 13, 14, 21, 28
CNN Convolutional Neural Network xi, 25, 27–29
CNNs Convolutional Neural Networks 20, 21, 25, 27
COM Completeness 57
COR Correctness 57
CPU Central Processing Unit 46
CUDA Compute Unified Device Architecture 46

DBMS Database Management System 10
DCNN Deep Convolutional Neural Network 15, 21, 28
DGT Direção Geral do Território v, 33

ESA European Space Agency 9
ESRI Environmental Systems Research Institute 9, 11
ETM+ Enhanced Thematic Mapper Plus 8

F1 F1-Score 57
FAIR Facebook AI Research 59
FCN Fully Convolutional Neural Network 14, 21, 28
FN False Negatives 57, 58, 67
FNEA Fractal Net Evolution Approach 11
FP False Positives 56, 58, 67, 68
FPN Feature Pyramid Network 19, 59

GIS Geographical Information System vii, 2, 9–11, 49
GMES Global Monitoring for Environment and Security 7

GPU Graphics Processing Unit 32, 46, 59

IoU Intersection Over Union 15, 47

KDE Kernel Density Estimation 16

LiDAR Light Detection And Ranging 7, 11, 12, 21

MAnet Multi-Scale Attention Network 78
ML Machine Learning 22, 46

NASA National Aeronautics and Space Administration 7, 9

OLI Operational Land Imager 8
OLI-2 Operational Land Imager 2 8
OSM Open Street Map 12

PAN Pyramid Attention Network 78
PSPNET Pyramid Scene Parsing Network 78

Q Quality 57

RADAR RAdio Detection And Ranging 6
RAM Random Access Memory 46
ReLU Rectified Linear Unit 18, 27
ResNet Residual Neural Network 43
ResNets Residual Neural Networks 19

SCMS Subspace Constrained Mean Shift 17
SOLI Skeleton-based Object Linearity Index 12
SPP Spatial Pyramid Pooling 29, 30
SQL Structured Query Language 10
SSLF Self-supervised Learning Framework 17, 21
SVM Support Vector Machine 13, 21

TIFF Tagged Image File Format 33
TIRS Thermal Infrared Sensor 8
TIRS-2 Thermal Infrared Sensor 2 8
TN True Negatives 56
TP True Positives 56, 58

USGS United States Geological Survey 5, 7–9

VHR Very High Resolution 11–13, 21

Chapter 1

Introduction

1.1 Background and Motivation

According to the National Forestry Accounting Plan 2021-2025, Portugal is one of the most heavily forested countries in Europe. About 3% of the forest land is owned by the state and other public administration agencies, 6% is being held by local communities, and 92% by private owners. [1]

Forest fires are a frequent cyclical problem in Portugal. In the last 20 years, the country has been severely affected by large wildfires, destroying the environment, ruining residences, and claiming lives. The biggest burned area in the last 10 years happened in 2017, with over 21,000 wildfires, burning over 500,000 hectares of forest and taking over 115 human lives. [2, 3]

As rural populations have been decreasing, a lot of these private fields have been abandoned. Overtime, with the lack of management of the forests, it creates an accumulation of shrubs and detritus, that become fuel for the flames when a fire breaks out.

Despite Portugal being a warm country affected by strong winds from the Atlantic, climate change scenarios also suggest the increase of this severity in upcoming years [4].

To extinguish forest fires, firefighters need to reach the fire epicenter as fast as possible, and for that, most of the time rural roads are used to reach the spot. Rural roads are key paths for fire fighting although sometimes they are difficult to access and firefighters might reach dead-end roads, or even lose precious time finding the best route to the fire epicentre.

Rural road centerlines on aerial images provide valuable information in wildfire context. They allow the visual identification of the rural road networks, with key information to help fighting wildfires.

Road centerlines are vector line data that represent the geographic center of roadways

on transportation networks. These networks are essential elements with high importance on GIS applications because they provide the foundation for car navigation systems. This means that it is possible to develop very useful functionalities to help firefighters in the field, like: discovering the closest path, discovering the fastest emergency route, finding dead-end roads or even discovering the shortest path passing thought multiple locations.

This type of work is extremely time-consuming and tedious to manually label the road area. The motivation to solve this problem is to create a system that will help to automatically detect and extract rural road centerlines. To reach this goal, it is necessary to develop an automated system capable of extracting knowledge from aerial images and classifying road and non road pixels.

Due to complex background landscapes, multiple road materials, road occlusion by trees and shadows, many challenges will be faced in order to extract complete, smooth, and clean road centerlines. These challenges can often bring heterogeneous segments that can be further improved. [5]

Solutions should be developed keeping in mind the recent technological advancements, meaning that this particular field of study will remain relevant for years to come.

1.2 Objectives

This dissertation proposes the development of an automated system capable of detecting rural roads from VHR aerial images, and extracting their centerlines. The following methods will be focused on a particular municipality in Portugal called "Mação", which is a highly targeted zone of cyclical forest fires. The whole process will be divided into two major stages: road detection and road centerline extraction.

The road detection process must be accurate enough to ensure that all the road pixels are detected in the aerial images. With all the road pixels detected, we complete the road segmentation.

Road centerlines are vector line data that represent the geographic center of road rights-of-way on transportation networks. The road centerline will be extracted from the previous road segmentation.

The following developed work is devoted to detect and extract rural road centerlines, creating a faster and automated system, more optimized than traditional ways of extracting rural roads. Given the growing need to integrate technological means into the fire suppression, a decision support system based on low-cost technologies is proposed to leverage operational data, enabling faster, and more informed decisions in a wildfire context.

The end product should be an automated system capable of providing accurate road centerlines from multiple samples of rural roads identified from aerial images.

1.3 Main Contributions

When dealing with challenges with a high degree of complexity, like creating an automated system to extract rural roads centerlines from landscapes affected by the unpredictability of Nature, it is necessary to find an efficient and effective solution.

The importance of integrating technological means into fire suppression is growing, and by using the proposed system, mapping operators can now automate the countless hours of tedious hand-work of extracting the rural roads centerlines. By integrating this data into real-time systems, fire fighters can also make better, and more informed decisions in the field.

The proposed method allows to surpass strong shadows from trees, small bushes, and vegetation in the middle of the road. With the optimization, it is also possible to connect different types of roads like dirt, asphalt, cement, gravel, as well as overcome total road occlusions with a considerable size.

The proposed architecture provides a very practical and viable solution for accurate rural road detection as well as centerline extraction from aerial images.

To the authors knowledge is was the first time that such recent deep learning technology like DeepLabV3+ was used combined with thinning algorithms to extract rural roads centerlines filling in the gap.

1.4 Dissertation Plan Outline

The aim of this section is to provide a general sense of how this dissertation plan will unfold and how each chapter will be organized. This dissertation plan is structured in 6 different chapters, all of which will be described ahead.

The first chapter is the Introduction, where we will review Portugal current situation, the background, and the urgency to integrate technological means into forest fire suppression. The motivation behind the problem will be revealed as well as the main goals with the rural road centerline detection/extraction problem.

The second chapter contains the state-of-the-art review, where we will review and discuss the most relevant scientific articles and literature on the subject in order to achieve the proposed objectives. Various methods and the most recent researches about automatic and semi-automatic methods of road detection and road centerline extraction will be presented as well as a simplified table comparing different previous methods.

On the third chapter it will be proposed a detailed theoretical methodology involving the aspects related to the state-of-the-art aiming to solve the problem (automatically detect rural roads and extract their centerlines). The reason why the method was chosen and well as main contributions of this thesis will also be discussed..

The fourth chapter will include the practical work developed during the course of the dissertation. All stages of the process will be analysed and reviewed as well as relevant achievements and setbacks towards the final implementation. To conclude this chapter a

deep explanation of the whole system is created in order to detect and extract rural road centerlines.

In chapter 5 a validation of the proposed model with several metrics will be made. All tests made as well as the final results from those experiences will be presented as well. Comparisons between different methods will be made. Finally, the advantages and disadvantages of each method on road detection as well as road extraction will be highlighted. To complete the chapter, a general comparison between the proposed method, and other recent articles will be shown.

In the last chapter, comprehensive conclusion will be presented, followed by multiple possibilities towards future work improvements.

CHAPTER 2

State-Of-The-Art Review

2.1 Introduction

Road networks make part of a country's fundamental geographic data, being widely used in transportation [6], personal navigation [7], forest fire mitigation [8] and many other applications. On this dissertation we will be focusing on methods that can accurately detect roads and extract road centerlines. Most of the times due to occlusion of trees, vehicles, noise and shadows, many methods fail to extract clean and accurate road centerlines, resulting in heterogeneous classification results. [5] Throughout this chapter, articles and literature that served as the foundation for this dissertation will be introduced.

During this investigation, existing solutions were grouped and analyzed in order to provide an understanding about of the pros and the cons of each method aiming to solve the proposed problem. In the end, a table with the main contributions of each article will be presented.

2.2 State-Of-the-Art Review

2.2.1 Remote Sensing

According to the United States Geological Survey (USGS) [1], Remote Sensing is the process of acquisition, detecting, and monitoring physical characteristics of an area by measuring its reflected and emitted electromagnetic radiation at a distance, normally from satellite or

[1] Information extracted from:
https://www.usgs.gov/faqs/what-remote-sensing-and-what-it-used?qt-news_science_products=0#qt-news_science_products
https://earthdata.nasa.gov/learn/remote-sensors
https://www.nrcan.gc.ca/maps-tools-publications/satellite-imagery-air-photos/remote-sensing-tutorials/introduction/passive-vs-active-sensing/14639

aircraft platforms. These special cameras on satellites or aircrafts gather remotely sensed images to help scientists "sensing" electromagnetic energy reflected or emitted from objects on the Earth surface without direct physical contact with the target.

Remote sensing sensors can be classified in two distinct categories:

- **Passive:** These types of sensors are used to detect energy when the naturally existing energy is available. This energy can be electromagnetic radiation emitted or reflected by an object. One of the most common sources of reflected energy is the electromagnetic radiation from the sunlight.

    1. **Examples of Passive Sensors** (using sunlight for energy source)
        - I Low resolution (>100 meter resolution)
            - i. MODIS, AVHRR, SPOT Vegetation
        - II Moderate resolution (15 – 100 meter resolution)
            - i. Landsat TM/ETM+, SPOT, ASTER, IRS
        - III High resolution (<15 meter resolution)
            - i. IKONOS, Quickbird, OrbViewIRS, SPOT, Corona

Figure 2.1: MODIS (500m) – June – September 2001

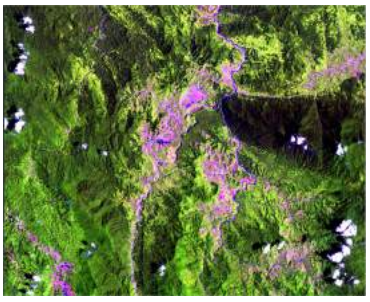


Figure 2.2: Landsat ETM+ (30m) - 2 April 2002

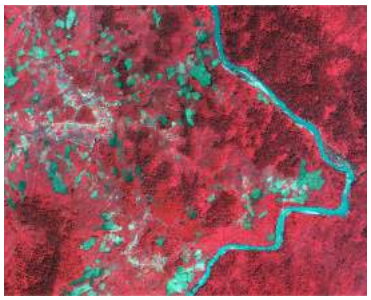


Figure 2.3: IKONOS Pan merge (1m) – 29 April 2002

In Figure 2.1, Figure 2.2, Figure 2.3 we can see some examples of different types of remote sensed images with passive sensors and their different resolutions. Images from: https://earthobservatory.nasa.gov/images

- **Active:** active sensors, in contrast to passive sensors, have their own source for illumination. Radiation is emitted from the sensor against the object to be detected. Then the sensor will measure the radiation reflected by the object.

    1. **Examples of active sensors** (generate their own energy)
        - I RADAR - It means RAdio Detection And Ranging (RADAR), and it can detect the location of objects by measuring the time delay when the signal was emitted and when it was received.
            - i. Radarsat, ERS, Envisat, Space Shuttle

II LiDAR - It means Light Detection And Ranging (LiDAR), and it measures the distance between the sensor and a target, and the signal of return. It uses wavelengths from blue through near-infrared.
   i. Mostly airborne platforms
   ii. ICESat is the only satellite LiDAR platform

2.2.2 Copernicus Program

The Copernicus Program, name inspired by the scientist Nicolaus Copernicus [2], was inaugurated in 2014. This program was first called Global Monitoring for Environment and Security (GMES), and had the goal to create the European Union observation Earth Program.

Based on satellite Earth Observation data, the Copernicus Program monitors Earth environment for the overall benefit of the European citizens. This information also helps to improve environmental understanding and management.

The Copernicus Program comprises a family of satellites that are responsible for monitoring Earth's subsystems and retrieve information used by service providers, authorities and international organizations.[2]

There are six main services that the Copernicus Program focus on:

- Atmosphere Monitoring Service CAMS
- Marine Environment Monitoring Service CMEMS
- Land Monitoring Service - CLMS
- Climate Change Service - C3S
- Emergency Management Service - EMS
- Security Service

2.2.3 Landsat Program

In 1972 the Earth Resources Technology Satellite was launched. This program was later called the Landsat Program, as a joint effort between the USGS, and National Aeronautics and Space Administration (NASA).

Since then, the data acquired by Landsat satellites have been crucial for helping scientists and decision makers managing and providing essential and accurate information about multiple activity sectors. Some of them are: forest monitoring, urban growth, water quality, agricultural productivity, response to natural disasters and even ice sheet dynamics.

[2] Astronomer who proposed the heliocentric universe theory, that the planets orbit around the Sun

[2] Information extracted from:
https://www.copernicus.eu/sites/default/files/Brochure_Copernicus_2019%20updated.pdf

As we can see, in Figure 2.4 between the first launch of the first satellite in 1972, called Landsat 1, and the time of writing this document, a total of 8 missions have been launched and Landsat 9 is expected to be launched in September 2021.

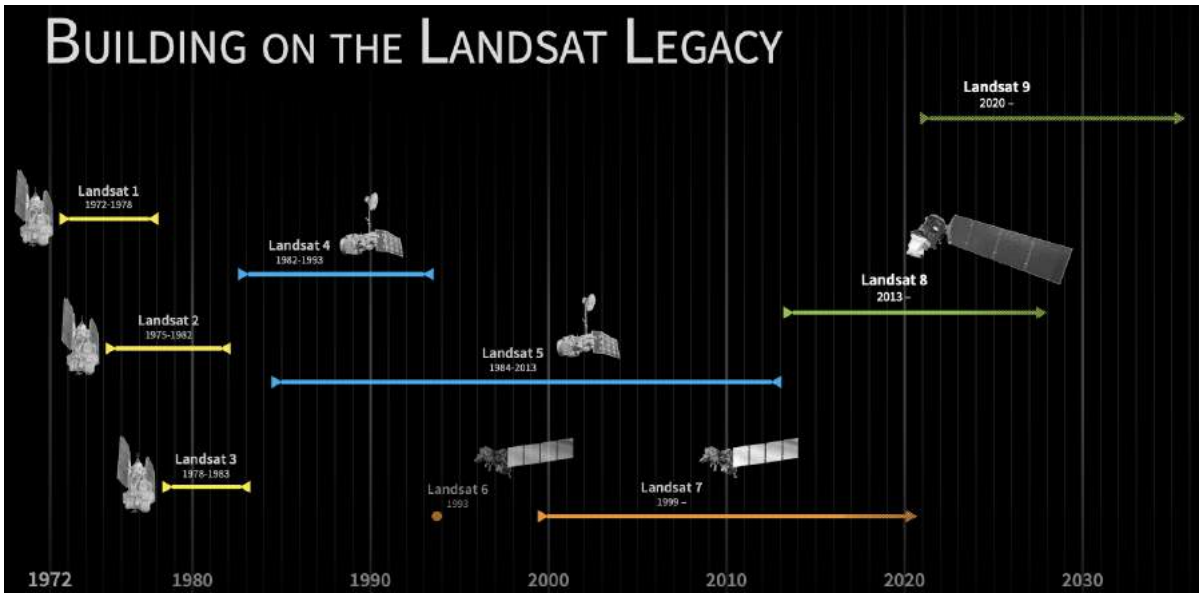


Figure 2.4: Landsat program chronology. Image from: https://landsat.gsfc.nasa.gov/

Nowadays only two missions are active, Landsat 7 and Landsat 8. These satellites are equipped with Enhanced Thematic Mapper Plus (ETM+) that will reproduce the previous capacities of acquiring high resolution imagery. Landsat 8 uses two types of sensors, the Thermal Infrared Sensor (TIRS) for thermal bands and an Operational Land Imager (OLI) for optical bands.

Once Landsat 9 is launched into orbit, it's going to replace the Landsat 7. The Landsat 9 satellite carries advanced image sensors when compared to previous Landsat missions, with both presenting a geometric and radiometric higher level of performance.

Landsat 9 will carry two instruments replicating a more advanced version of Landsat 8. The first sensor will be an Operational Land Imager 2 (OLI-2) that is a multispectral sensor that captures near infrared and shortwave infrared electromagnetic energy. The improved OLI-2 version will lead to better observation of darker regions, for example in forests. [9]

The second sensor will be a Thermal Infrared Sensor 2 (TIRS-2) that can detect the heat emitted as radiation through the Earth surface in order to track evapotranspiration and other parameters like the soil moisture, and detect the health of plants.

To help Landsat users to learn and to study Earth surface phenomenons for large periods of time, the USGS archive owns today more than 8 million Landsat scenes, image processing software, cloud computing systems and advanced Geographic Information Systems. [10][3]

[3]Information extracted from: https://landsat.gsfc.nasa.gov/

2.2.4 Satellite Imagery

In the last years satellite imagery has been one of the pillars that human beings use to track vegetation health, droughts, reverse engineer natural and unnatural disasters, and track human activity. [11–13] These types of images have a very large field of view being able to gather a lot of information and generally cover a much wider area than other types of imagery from ranges of 3500 to above 30,000 sq.km.

Satellite images are taken from orbiting sensing devices producing digitally Earth representations. These images are usually taken from 600-900Km of altitude. In satellite imagery the radiance reconstruction is done over a region by multiple detectors each one collecting information over small tiles of the entire region.

Another advantage of satellite images is that they are taken digitally, meaning that they can be later enhanced and improved. Some of the most up to date sources of satellite imagery are: Google Earth, NASA, USGS and European Space Agency (ESA).

2.2.5 Aerial Imagery

Aerial imagery is the name attributed to the images taken from an airborne craft like airplanes, balloons or drones. This type of imagery is an essential tool in topographical mapping and the acquiring knowledge about of places, features, and objects.

In aerial imagery, the field of view is considerably smaller when compared to satellite images, as this imagery focuses on smaller niche areas, usually a few tens of square kilometers to a few hundred square kilometers. It can provide an in depth spatial resolution up to ranges of 1-5 centimeters per pixel.

There are three main types of aerial photographs: vertical aerial photographs, low oblique aerial photographs, and high oblique aerials.

In this thesis it will be given importance to vertical photographs as they play an important role in mapping, object detection, and image segmentation.

Another relevant advantage of aerial imagery is its ability to integrate as a base layer into several applications that allow operators to work with them at scale. The most common applications used are Geographical Information System (GIS) and Computer-Aided Design (CAD).

2.2.6 Geographic Information System

We can start by asking *what is a geographical information system?* The answer to this question can have multiple definitions [14], but according to Environmental Systems Research Institute (ESRI) [15], a GIS, it is a framework for gathering, analyzing, and managing geographical and attribute data.

GIS is based mainly on the science of geography and combines many types of data like: spatial data, attribute data, and metadata. It organizes layers of information into visualizations using maps and 3D scenes and analyzes spatial location. This means that it

can show many different kinds of data in one map, such as streets, vegetation and buildings. On top of these functionalities GIS is also capable of revealing deeper insights into data, such as relationships, patterns, and enables people to easily see, analyze, and understand their relationships.

The architecture of geographic information systems usually consists essentially of three separate layers, namely: the presentation layer, the application logic layer, and the data layer.

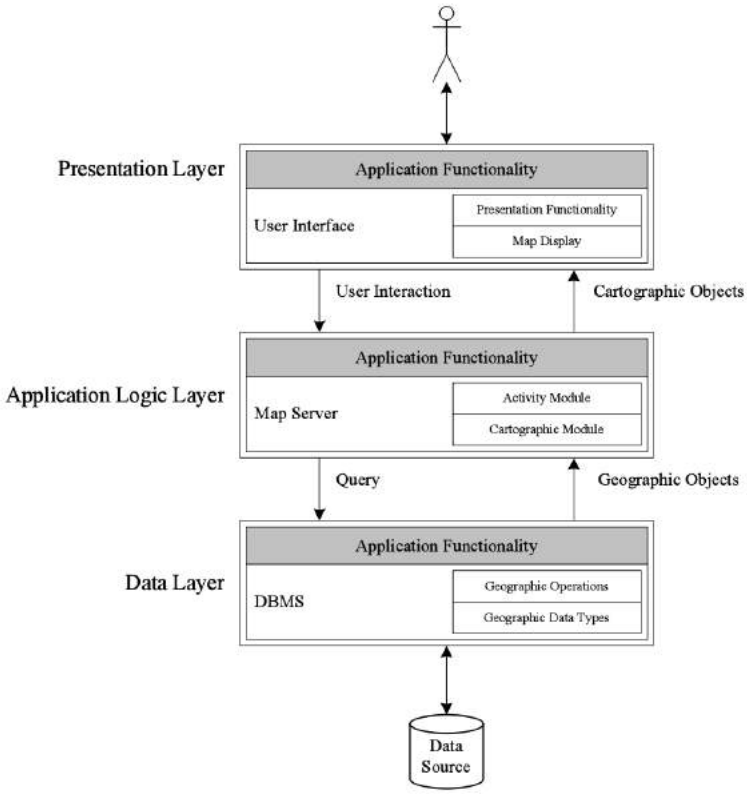


Figure 2.5: Generic architecture of a GIS. Image extracted from [16]

In Figure 2.5 the first layer, Presentation Layer, it is used to implement the user interface of the system, showing the maps and providing some functionalities over them. In particular it will receive the user requests from the Presentation Layer and translates them to the corresponding queries for the Data Layer. Later, it will translate the geographic objects returned by the Data Layer into the cartographic objects that are shown by the Presentation Layer.

The Data Layer allows data management independent from the software technology, this means that this layer provides data storage abstraction, and query language like Structured Query Language (SQL) or any query language defined by the Database Management System (DBMS).

The Application Logic Layer, implements the functionality of the system, using the Data Layer to answer the user requests and converting the geographic objects from the

Data Layer to cartographic objects to be displayed by the Presentation Layer. [16]

2.2.7 ArcGIS

ArcGIS is one of the top leaders of GIS software for industrial, research, commercial and governmental use. ArcGIS was created by ESRI, and it was first released in 1999.

This software has a plethora of utilities from being able to create, manage and share maps, layers, apps, and is able to manipulate geographic data. There are multiple variations of the software, each of them having their own functionalities and purpose.

The central application of ArcGIS is considered to be ArcMap, which is very useful when it comes to working with map layouts, display images, create and edit datasets, and document geographic information. ArcGIS includes ArcScene and ArcGlobe, that are very powerful tools because they allow three-dimensional representation of the Earth. ArcCatalog also allows users to manage data and handle tasks.

In order to simplify the user experience and create an intuitive journey, ArcGIS has integrated toolboxes, such as The ArcToolbox, that comprises a broad compilation of scripts, models and geoprocessing tools that users can take advantage.[4]

2.3 Recent Methodologies

Throughout the literature research, multiples approaches were found. This section presents some of the most recent approaches that demonstrate the achievement of relevant results, aiming the detection of rural roads as well as the extraction of its centerlines.

2.3.1 Method based on road connectivity from LiDAR data

In 2018, a road centerline extraction method using very-high-resolution aerial images and LiDAR data was proposed by Zhiqiang, (Zhiqiang Zhang, Xinchang Zhang, Ying Sun and Pengcheng Zhang) [17].

According to this article road networks provide valuable information for multiple applications such as urban planning, urban management, and map navigation. Road networks extracted from remote sensing images are likely affected by shadows, making the road map irregular and inaccurate.

This article intends to improve the extraction of road centerlines using Very High Resolution (VHR) images, and LiDAR for road connectivity. A flowchart on this method can be seen in Figure 2.6.

The proposed method used a Fractal Net Evolution Approach (FNEA), to segment images. To classify objects like cars, shadows and trees, the adopted approach was the random forest classification. For the Road Network Construction it was used the minimum

[4]Information extracted from: https://desktop.arcgis.com/en/documentation/
https://developers.arcgis.com/documentation/mapping-apis-and-location-services/

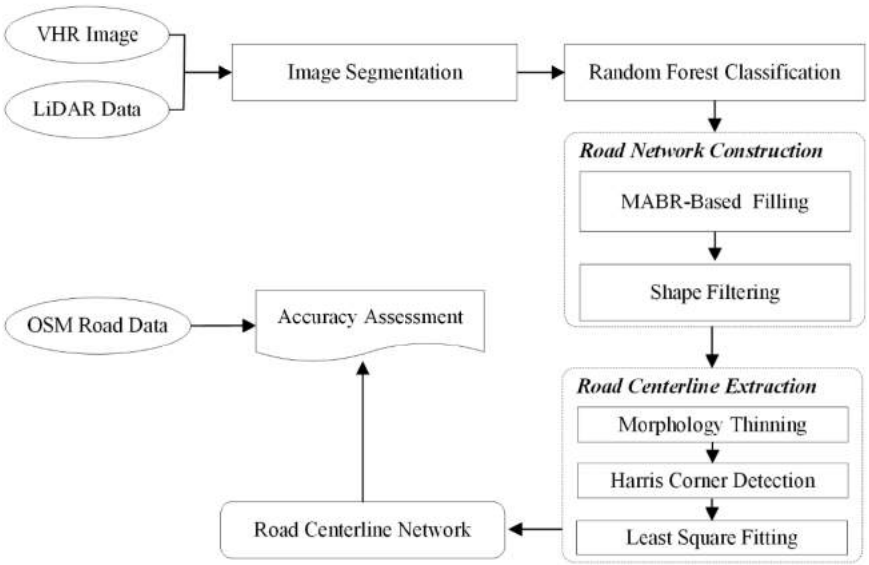


Figure 2.6: Flowchart of the proposed method for extracting the road centerline network using LiDAR, Open Street Map (OSM), and VHR images. Image extracted from [17]

area bounding rectangle, MARB-base filling approach, to remove negative influences of shadows, trees, cars, and to link the discrete road segments to obtain a complete road network. After the filling process, the Skeleton-based Object Linearity Index (SOLI) it was used to remove the false road segments that were left after the filling process.

For the road centerline extraction the chosen approach was the morphology thinning, Harris corner detection, and least square fitting algorithm to accurately extract the road centerlines from the road network.

The proposed model was applied to three datasets and then compared with two states-of-art methods (Huang's method and Miao's method). Experimental results showed that the proposed model obtained the highest completeness, correctness, and quality of the three datasets. For straight road sections, it was able to eliminate effectively the influence of negative shadows, trees, and obtain accurate centerlines. However, for curved road segments that are severely obstructed by shadows, and trees, the proposed method needs further improvement. Despite the limitation mentioned, the proposed methodology was able to reduce effectively the workload associated with road mapping and it remains to be an effective solution for road centerline extraction in complex scenarios.

2.3.2 Method using multi-level thresholds

In 2020 Dorathi, (J. D. Dorathi Jayaseelia, D. Malath) presented an upgraded cuckoo optimization algorithm using multi-level thresholding in order to effectly extract road regions from high resolution satellite images [18].

First, the histogram for the given image was found. Then a multi-level threshold approach was used, which means that defined multiple threshold values are defined by

which the processed image is segmented into a bigger number of segments, in order to create more outlined objects.

To implement the multi-level thresholding, an Otsu's bi-level method with a self-emphasized process was used, in order to find the number of threshold values recommended to segment the image.

The cuckoo algorithm was inspired by cuckoo birds as each egg represents a solution, and a cuckoo egg represents a new solution. The goal of this algorithm is to keep searching for new optimized solutions and get rid of old and less good solutions in order to keep the best solutions possible.

Finally a Support Vector Machine (SVM) is used to obtain the optimal edge between possible outcomes. This will allow the SVM classifier to identify the road region from the given image. This method can be further improved, but overall it shows high accuracy when compared with other existing road region extraction methodologies.

2.3.3 Methods using CNNs

In 2017, a novel deep learning architecture based on a cascaded end-to-end convolutional neural network was proposed by Guangliang, (Guangliang Cheng, Ying Wang, Shibiao Xu, Hongzhen Wang, Shiming Xiang, Chunhong Pan) [19].

The author proposed a Cascade End-to-End Convolutional Neural Network (CasNet) to simultaneously extract consistent road area and smooth road centerline from VHR remote sensing images.

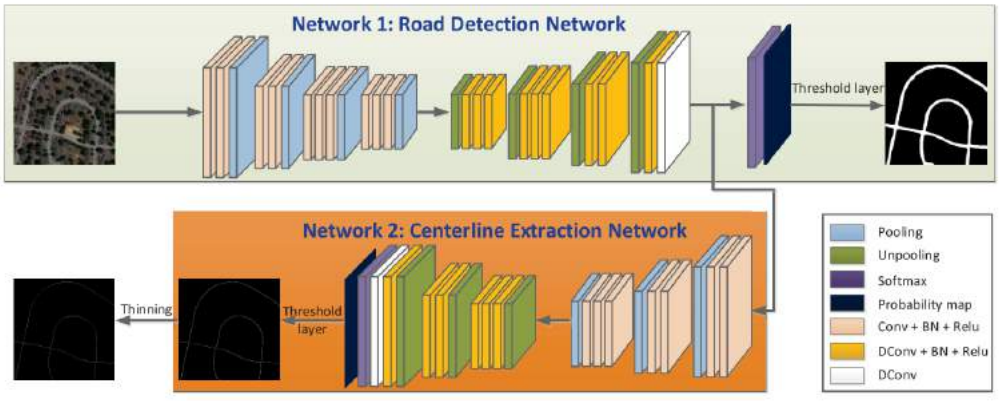


Figure 2.7: Flowchart of the CasNet. Contains two convolutional neural networks: road detection network and centerline extraction network. Image extracted from [19]

The method consists of having two convolutional neural networks, the road detection network, and centerline extraction network. The convolutional neural network consists of having an encoder network, a corresponding decoder network and a softmax layer. The encoder network is used to extract features that transform the input image into multidimensional feature maps. The feature maps contain semantic information of the input image. Later, using unpooling and deconvolutional operations, the feature maps are

upsampled and extracted from the encoder network into the original image size. Lastly a softmax classifier is used to reach the final image output. The output will be two probability maps showing the probability of each pixel belonging to the road and non-road class respectively.

To extract the road centerline, similar to the road detection network, it consists of an encoder network, a decoder network, and a softmax layer, but an architecture with less layers is used. To train both neural networks, a learning algorithm that minimized both learning loss parameters was adopted. It uses two different types of datasets to train each network independently. In Figure 2.7 a representation of CasNet architecture can be seen.

The CasNet shows a lot of advantages in terms of quantitative and visual performances. It achieves a very smooth and consistent road detection, as well as better results in terms of centerline extraction if compared to other state-of-the-art methods in the respective article. However, for continuous large areas of occlusions, the CasNet might face some difficulties detecting the road precisely.

In 2020 a simultaneous road surface and centerline extraction from large-scale remote sensing images using CNN-Based segmentation and tracing was proposed by Yao Wei, (Yao Wei, Kai Zhang and Shunping Ji) [20]. The proposed method consisted of three main stages Figure 2.11.

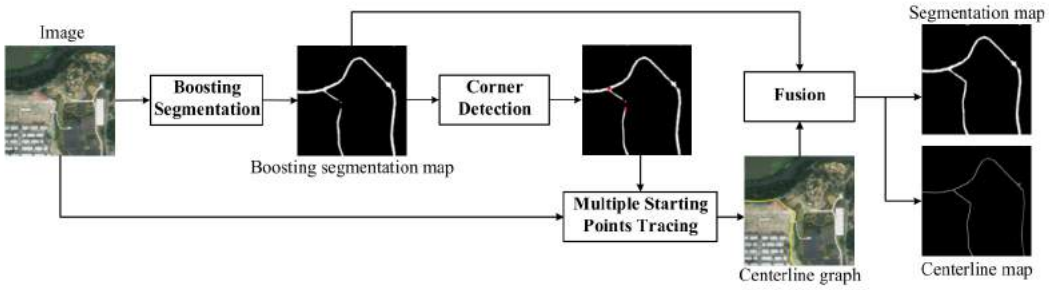


Figure 2.8: Workflow of the proposed framework for simultaneous road surface and centerline extraction. Image extracted from [20]

On the first stage a boosting segmentation was executed, where the remote sensing image was initially segmented by a common Fully Convolutional Neural Network (FCN) method, followed by a series of boosting segmentation steps with another FCN. To discover road intersections, road mask maps were adopted from the predicted boosting segmentation.

The second stage, multiple starting points tracing, was developed for tracing topographical road centerline networks from the extracted road points.

Finally, the last step is the fusion process, where the results of the road surface segmentation and the centerline tracing were merged, to produce a fine segmentation and centerline maps.

The advantages of this method are the increased connectivity of road segments by learning the complementary information of previous labels and segmentation maps with an efficient encoder-decoder. This method also proved to have advantages in terms of

accuracy, road connectivity and completeness than other recent road segmentation and road centerline extraction methods.

In 2018 Buslaev (Alexander Buslaev Mapbox, Selim Seferbekov Veeva, Vladimir Iglovikov, Alexey Shvets) [21] proposed a fully convolutional network, based on the U-Net family, for automatic road centerline extraction from satellite imagery.

This architecture enables precise pixel location, and the particular reason for this behaviour is the fact that U-Net uses skip connections to associate low-level feature maps with high level feature maps.

For the dataset, over six thousand satellite images from DigitalGlobe satellite were used. For the training data set, each image had its own mask with the labeled road pixels. The white pixels correspond to the road pixels and the black ones correspond to the background.

To make the road segmentation, a fully convolutional neural network from the U-Net family was used. To increase the resolution of the output, pooling operations to increase the number of feature maps per layer, downsampling, were used. Then it uses the process of upsampling the features maps, followed by convolutional layers. This network resides its encoder from the pre-trained ResNet-34 on ImageNet and the decoder from vanilla U-Net family.

The metric chosen to evaluate the training performance was the Intersection Over Union (IoU), which can evaluate the similarity between sets. To increase the dataset size algorithms of data augmentation were also used.

To summarize everything that has been stated so far, this method allowed the system to make fast predictions with a simple architecture, becoming one of the top results on the DEEPGLOBE - CVPR 2018 road extraction sub-challenge.

In 2019 Shikai sun, (Shikai Sun, Wei Xia , Bingqi Zhang ,Ying Zhang China) [22] proposed a method to extract road centerlines from high resolution remote sensing images. This method focused on the phenomenon that complex backgrounds produce spurs on road centerlines.

First, to extract the road region, a Deep Convolutional Neural Network (DCNN) architecture Figure 2.9 was used. Later edge smoothing and block connection were used to process the extracted results. And, lastly, to generate the final road centerline, the road centerline was segmented and smoothed. This method does not focus on the material of the road, so the road material can be gravel, asphalt, or even cement.

On the proposed method, to produce the training dataset, the images were separated in two sizes at the same time, 256*256px and 500*500px. To increase the dataset size, augmetation was done by performing multiple random transformations like: segmentations, scalling and mirroring operations.

To extract the road centerline, one used algorithms that, given the initial direction and the starting point of the road, the algorithm can explore the next point in order to

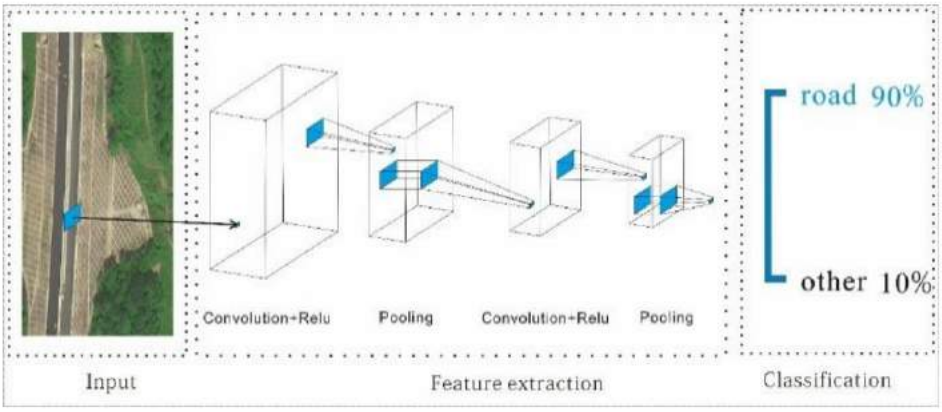


Figure 2.9: Road detection with a deep convolutional neural network. Image extracted from [22]

resolve problems like: trees closing the visibility of the road, and other road interruptions. This algorithm works by storing multiple road sub-intervals, using the collinearity formula, respecting a given threshold, and storing the average road width.

The proposed method achieved very good results by overcoming the problem of having complex backgrounds, effectively overcoming the phenomenon of "burr". It also achieved very high speed and accuracy. This method can be later improved by increasing the number of dataset samples and road materials.

2.3.4 Other relevant methodologies

In 2014, Miao (Zelang Miao, Bin Wang, Wenzhong Shi, and Hao Wu) proposed a novel accurate road centerline extraction method from classified images. This method integrates tensor voting, principal curves, and the geodesic method. From this, it was created a way to extract centerlines from classified images with accuracy. [23]

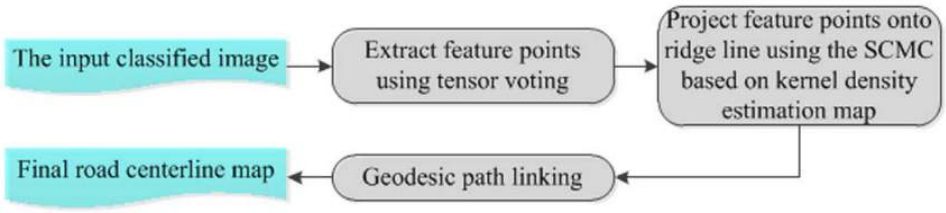


Figure 2.10: Architecture using tensor voting, principal curves, and the geodesic method. Image extracted from [23]

The method is divided into three steps: the first step is the extraction of feature points from the classified image with the tensor voting method. A vote analysis is then performed to extract two types of points, the junction points, and the end points. The second step is to use the Kernel Density Estimation (KDE) method to calculate the probability of each

pixel being located on the road centerline. Afterward using the Subspace Constrained Mean Shift (SCMS) method the feature points are projected onto ridge lines. Finally, the last step is to link the projected features points with the geodesic method to create the central line to formulate the road network. This workflow can be observed in Figure 2.10. The proposed method was compared with the widely used thinning algorithm and SCMS method. The experiment results show that both SCMS retain smoother centerlines than the thinning algorithm and do not produce spurs. The proposed method also solves the SCMS limitation of unbiased centerlines with higher computational efficiency. This method provides a good solution for efficient road centerline delineation from classified images. The downside of the proposed method is that it can't process segments with closed-form like circles.

In 2020 Guo, (Qing Guo and Zhipan Wang) proposed a Self-supervised Learning Framework (SSLF) for high-resolution remote sensing images to extract automatically road centerlines. This approach does not require to manually select the training samples and other optimization processes. [24]

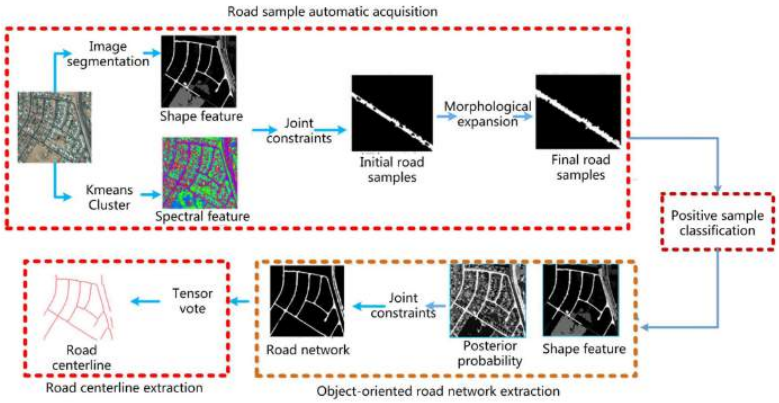


Figure 2.11: Workflow of the proposed SSLF road extraction algorithm. Image extracted from [24]

The first step on this architecture, Figure 2.11, was to make the automatic road acquisition through the image segmentation which is achieved with the joint constraints of the spectral clustering, and the road shape feature. The next step was to make the positive sample classification considering only positive road pixels samples. The third step associates the shape feature from the first step and the next probability road object from the second step to obtain the road network. The last step uses the tensor voting algorithm to join broken lines in order to achieve the ultimate road centerline.

The proposed SSLF method achieves higher accuracy when compared with the state-of-the-art Miao's method. The SSLF method also gets better results quantitatively and visually while compared with the traditional supervised road extraction algorithms. Another advantage of this method is it achieves superior noise resistance than previous

unsupervised algorithms, becoming a promising method for future improvements.

In 2018 a method for image segmentation using an Encoder-Decoder with atrous separable convolution was proposed by Chen, (Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam) [25]. The proposed method gathers the advantages of spatial pyramid pooling combined with the encode-decoder structure and extends the method of DeepLabV3 to DeepLabV3+ in a way that can improve and refine the boundaries of semantic segmentation.

A new approach of the Xception model is studied in order to diminish the number of parameters and computational cost, culminating in a stronger and faster encoder-decoder model.

Without any post processing, this method achieves a performance of 89.0% and 82.1% on cityscapes dataset, and the PASCAL VOC 2012 model becoming a new state-of-the-art architecture.

In 2017 Badrinarayanan (Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla) presented an architecture for semantic pixel segmentation called SegNet. [26] This model, Figure 2.12, has a matching encoder network for each decoder network, subsequently followed by a layer of pixel classification. The encoder is composed by 13 convolutional layers which are equal to the VGG16 network. This network is composed of a smaller and easier to train encoder. The decoder also has 13 convolutional layers and its output is connected to a soft-max classifier in order to give the probability of each pixel corresponding into a certain class.

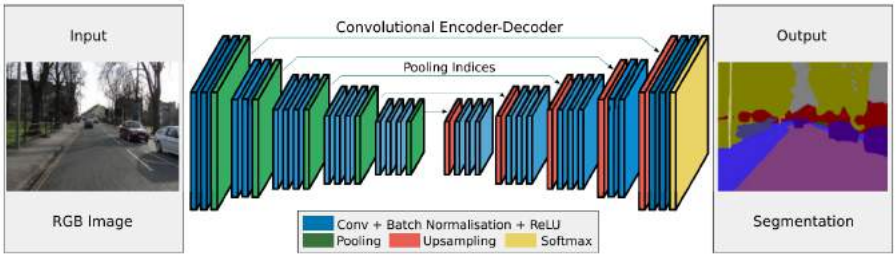


Figure 2.12: SegNet segmentation architecture. Image from [25]

Each encoder produces a set of feature maps followed by a batch normalization and a Rectified Linear Unit (ReLU) layer. Then a pooling layer to decrease the dimension of the convolution layer output is used, while preserving relevant features and removing irrelevant details. The pooling operation will produce translation invariance, which means that the network is still able to detect the class to which the input belongs, even if the inputs were translated. By adding more of these layers, the classification will become more robust, but on the other hand the resolution of the feature maps will decrease.

The decoder will upsample the feature maps turning them into dense feature maps, followed by a batch normalization. Afterwards the decoder output, with high dimensional

features, will send those feature maps to a soft-max classifier which will classify the probability of each pixel belonging into each class. The highest probability will be the respective class chosen.

With this type of encoder-decoder network SegNet, when compared with other state-of-the-art methods, has achieved practical trade-offs in terms of balancing the training time, and memory versus accuracy. SegNet encoder-decoder is more efficient than the typical encoder-decoder, because it only stores the max-polling indices instead of the encoder feature map.

It was introduced by Lin (Tsung-Yi Lin, Piotr Dollár, Ross Girshick) [27] a new method on feature pyramid networks for object detection. This system uses a Feature Pyramid Network (FPN), which is a feature extractor that has the ability of taking a single scale image with an input arbitrary size and outputs feature maps with the same size proportion at multiple levels using a fully convolutional process. This allows this architecture to act like a general way to build feature pyramids in the middle of deep convolutional networks that can be later used in tasks like object detection. To create the pyramid, two processes must be involved, a bottom-up pathway and a top-down pathway. Figure 2.13

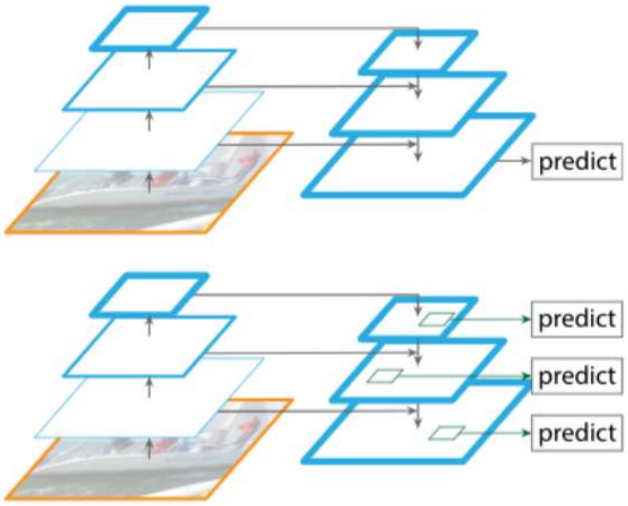


Figure 2.13: Feature Pyramid Network. Image extracted from [27]

The first process is the bottom-up pathway, which is the feedforward computation of the network backbone. This step is in charge of computing a feature hierarchy with feature maps at multiple scales with a scaling step of 2. For each stage it is defined a pyramid level and the output of the final layer of each stage is set as a reference of feature maps. For Residual Neural Networks (ResNets) it is used feature activation outputs on the final residual block of each stage.

The second step of the process is the top-down pathway, which hallucinates higher resolution features by upsampling spatially coarser, but semantically stronger, feature maps from higher pyramid levels. [27] As the article refers, these features are later enhanced with features from the bottom-up pathway via lateral connections and each lateral connection

merges feature maps of the same spatial size from the bottom-up pathway and the top-down pathway. The bottom-up feature map is of lower-level semantics, but its activations are more accurately localized as it was subsampled fewer times.

This architecture presents a clean, practical, and simple architecture that allows building feature pyramids inside Convolutional Neural Networks (CNNs) without the need of computing image pyramids. This architecture shows important improvements, surpassing other multiple strong, heavily engineered competition winners like COCO competition winners, including the 2016 winner G-RMI and the 2015 winner Faster R-CNN+++.

2.4 Comparison between methods

Table 2.1: On the following table a summary of surveys and their contribution to road detection and centerline extraction is presented.

Reference	Contribution
[17]	Improved results for straight road sections, the proposed method can effectively eliminate the influence of negative shadows,trees, and obtain accurate centerlines from VHR aerial imagery and LiDAR data.
[18]	This method uses the cuckoo optimization algorithm in order to get rid of old solutions and a SVM to identify the road region bringing high accuracy.
[19]	Proposed a novel CasNet to extract at the same time the road area and its centerline. This architecture shows a lot of visual and quantitative advantages when compared with other state-of-the-art methods.
[20]	Used a FCN to segment the road area followed by a multiple starting points tracing process and a fusion to produce a fine segmentation and centerline, increasing results in road connectivity and road completeness.
[21]	Suggested a pre-trained ResNet-34 as the encoder network and the vanilla U-Net family as the network decoder. This method allows the system to make fast predictions with a simple architecture, becoming one of the top results on the DEEP-GLOBE - CVPR 2018 road extraction sub-challenge.
[22]	Used a DCNN for road segmentation, and data augmentation for the training dataset. It achieves very good results by overcoming the problem of having complex backgrounds, effectively overcoming the phenomenon of "burr" with high speed and accuracy.
[23]	It's a good solution for centerlines delineation but it cannot process segments with a closed-form.
[24]	Proposed a SSLF to extract road centerlines. It achieves higher accuracy when compared with Miao's method and also gets better results quantitatively and visually while compared with most traditional methods. This method also achieves a higher noise resistance compared with previous unsupervised algorithms.
[25]	Without any post processing, this state-of-the-art method obtains a new best performance on Cityscapes datasets and PASCAL VOC 2012. This article also proposed an advanced version of DeepLabV3 to DeepLabV3+, with improvements on a strong encode-decoder system. Uses atrous convolution to optimize runtime and precision. DeepLabV3+ also uses the adapted Xception model in order to have a faster and more powerful encoder-decoder network.
[26]	SegNet when compared with other state-of-the-art methods has achieved practical trade-offs in terms of balancing the training time, and memory versus accuracy. SegNet encoder-decoder is more efficient than the typical encoder-decoder, because it only stores the max-polling indices instead of the encoder feature map
[27]	The FPN architecture presents a clean, practical, and simple architecture that allows building feature pyramids inside CNNs. This architecture shows important improvements, surpassing other multiple strong, heavily engineered competition winners like COCO competition winners, including the 2016 winner G-RMI and the 2015 winner Faster R-CNN+++.

CHAPTER 3

Proposed System Architecture

This chapter aims at proposing a system that can automatically detect rural roads from aerial images and extract their centerlines.

Recently, most methods of road network extraction primarily focus on two main steps: road region detection and road centerline extraction. [28]

The use of convolutional neural networks to perform the road segmentation, and centerline extraction has increased in the last years. These deep learning methods have achieved notable results when compared with previous methodologies having better results overcoming complex backgrounds, landscapes and having a higher accuracy. [29] [30] And for that particular reason we will chose to focus on deep learning based methods.

3.1 Deep Learning Supporting Concepts

With the increased necessity of having machines that can mimic complex human behaviors, scientists have been looking for ways that machines can mimic natural cognitive functions displayed by humans while maximizing their chances of learning, solving problems, and successfully achieving their goals. [31] Artificial Intelligence (AI) is the science that aims at making machines think, and act like humans. One of the ways that machines use to replicate that behaviour is by using Artificial Neural Networks (ANN). Artificial neural networks are algorithms that can solve problems through the simulation of the human brain, by acquiring knowledge through experience, usually known as Machine Learning (ML): learning, making mistakes, and making discoveries. [32]

Machine Learning is a subfield of AI that focuses on setting computer systems that are able to adapt and learn without the need of explicit instructions or programming, by utilizing algorithms that use historical data as input to infer the future and to make decisions without complete knowledge of all influencing elements. [33]

Machine Learning focuses on learning through three main types of learning: Supervised Learning, Unsupervised Learning, and Reinforcement Learning.

- **Supervised Learning** - In supervised learning, as the name suggests, requires a teacher or supervisor participation in the labeling process. It's given to the computer training data (input and expected output), and a model, designed to teach the system how it should respond to the data. With this information the agent can start reducing the magnitude of the global loss function by correcting its parameters. With multiple iterations, if the algorithm and data are adjustable and consistent, the error between the predicted and expected output should get closer to zero while the overall accuracy of the values should increase. It's important to create a system that has a high level of abstraction in order to work with samples that the algorithm has never "seen" before, otherwise the algorithm will suffer a common error of overlearning (or overfitting), which leads to inaccurate predictions of values. [33] This thesis will focus on this type of learning to develop the road detection algorithm.

- **Unsupervised Learning** - In Unsupervised learning, the computer trains on unlabeled data and has the freedom to find patterns or any meaningful correlations in the datasets without any supervisor. This type of learning can be more unpredictable compared with other learning methods because since it's unsupervised and there is no absolute error measure, values need to be clustered in different methodologies bringing a higher degree of uncertainty. There is also an important variance called Semi-supervised learning that is a mixture of supervised learning, and unsupervised learning, in which the computer is fed labeled data and unlabeled data, and seeks for patterns on its own. [33]

- **Reinforcement Learning** - Reinforcement learning does not need input or output labeling, it learns by interacting with its environment and by receiving feedback from it. The agent pretends to maximize the numerical reward, which encodes the success of performing certain actions over time. Reinforcement learning is especially efficient in dynamic environments where it's not possible to have absolute errors. [33]

When analysing aerial images of rural roads, the sources of influence that can affect our learning algorithm are: different types of road, color, trees, shadows, obstruction of roads by objects, obstruction of roads by natural conditions, angle, and brightness of the sun. All of these factors are not quantities that are directly observed. These factors can be thought of as abstractions that help us comprehend the variability of the data.

This variability of data brings major challenges when it comes to influencing every single piece of data we are able to observe. The grey pixels of the road might be hidden by a black shadow of a tree. The shape of the road might be deformed from satellite view because of road obstruction. The color of the road outline may be very similar to the landscape becoming very hard to differentiate between pixels.

Deep learning will help us overcome this problem and protect the learning algorithm from these high level abstractions. Deep learning is a subfield of machine learning, which itself makes part of a broader family of artificial neural networks. Deep learning focuses on the relationship shared between variables and building complex concepts out of simple concepts.

Computers have difficulty mapping raw sensory input data, for example translate a set of pixels from an image to a specific labeled object. Deep learning solves this challenge by fracturing complex mapping into a series of simple mappings each one corresponding to a layer of the model. First, the image that we perceive is presented in the visible input layer. After that, the model has multiple layers called "hidden layer" as their values are not presented in the data. The multiple hidden layers start extracting abstract features from the input image in order to find which concepts are related. That way the network can explain the relationships in the data observed. This means that deep neural networks learn by adjusting the weights of their connections to transmit input signals through several layers of neurons associated with the right general concepts. This process filters the noise and retains only the most relevant features. As we can see in Figure 3.1, we have an illustration of a deep learning model with three hidden layers. The first hidden layer can identify edges. With the description of the edges, the second layer can identify corners and extended contours. With the description of the second layer with corners and contours, the third layer can detect entire parts of objects. Lastly, with this description the model can recognize and identify objects presented on the image. [34]

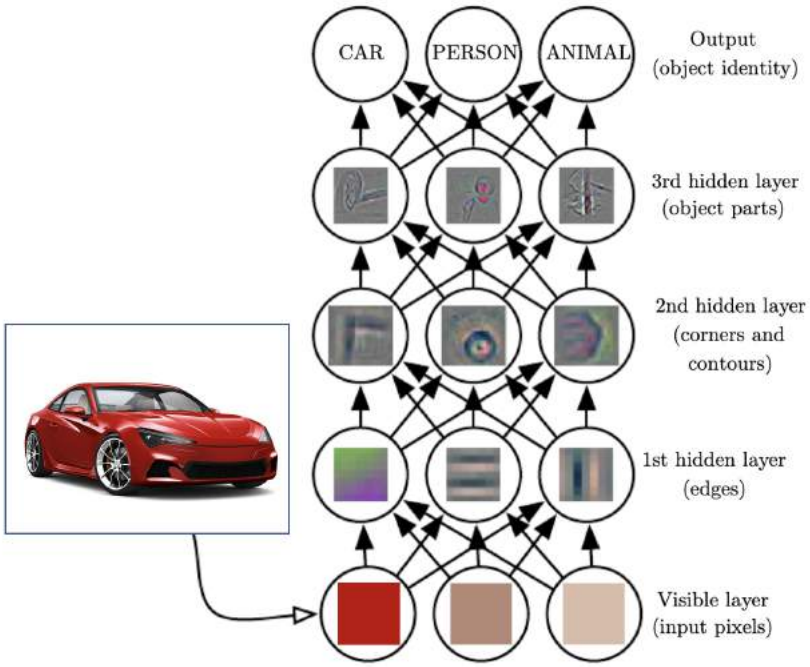


Figure 3.1: Illustration of a deep learning model, picture adapted from Zeiler and Fergus (2014). [34]

3.1.1 Convolutional Neural Networks

3.1.1.1 Feature Extraction

One of the most important classes of deep neural networks in CNNs. A Convolutional Neural Network (CNN) is a kind of neural network normally used for processing images (2D matrix of pixels) as input, and it's able to differentiate patterns and objects by giving importance to learnable network parameters like weights and biases. This means that a CNN can be trained to understand the sophistication and complexity of an image. CNNs, while compared to other classification algorithms, require a lot less pre-processing because with sufficient training they have the capacity to learn these characteristics and filters. CNNs are capable of acquiring Spatial and Temporal dependencies successfully through the application of relevant filters on an image. CNNs are mainly used for image detection and classification. As the name indicates, convolution neural networks make use of a mathematical operation called convolution that shows how the shape of an input I is modified by a kernel K.

Considering x an RGB (Red, Green, Blue) image, and w a feature detector or kernel, and the output of the operation referred as feature map represented by $s(t)$. The operation of convolution can be presented by Equation 3.1:

$$s(t) = \int x(a)w(t-a)da \tag{3.1}$$

Note that the integral can be described as the area under the function $x(a)$, weighted by the function $w(-a)$ shifted by amount t. Another denotation used to represent the convolution operation is with the asterisk, Equation 3.2:

$$s(t) = (x * w)(t) \tag{3.2}$$

Considering t as a discrete variable, and x and w are only defined for discrete values, we can define the discrete convolution in Equation 3.3:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{3.3}$$

Assuming we had a feature detector for edge identification, the result of the convolution would help finding the edges presented on the image. It is also common to only assume points where we store input values, all other values are considered as zero. This reasoning can be defined by the infinite summation as a summation over a finite number of array elements, described as Equation 3.4:

$$s(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m, j-n) \tag{3.4}$$

It's important to note that I is the two-dimensional array, and K is the kernel. We can also take advantage of the commutative property by flipping the kernel and have Equation 3.5:

$$s(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{3.5}$$

Nowadays, multiple neural network libraries use a similar function named as cross-correlation, which is the same as convolution but without flipping the kernel. Equation 3.6 [34] [1]

$$s(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{3.6}$$

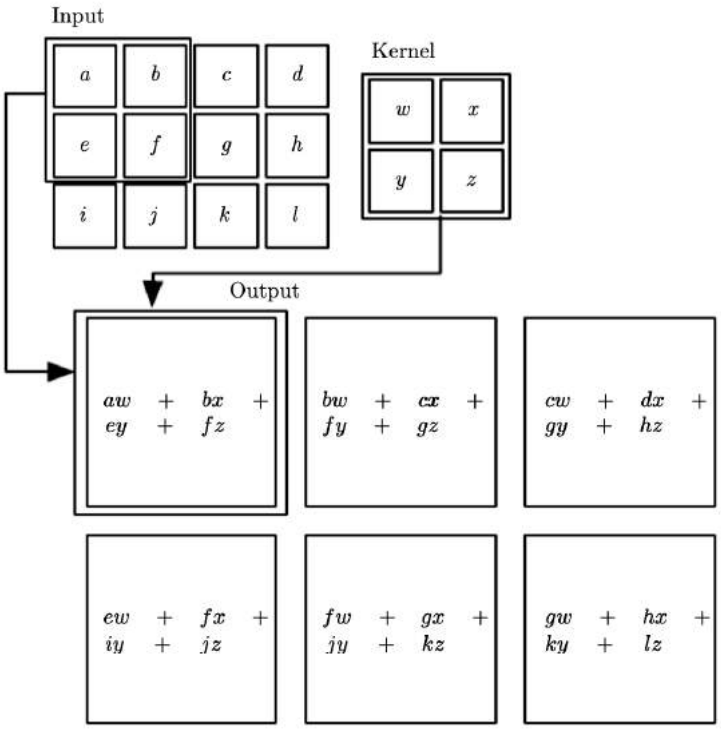


Figure 3.2: Example of the convolution operation with an input image without kernel flipping. The kernel shifts through the input image performing a matrix multiplication between a portion of the image in which the kernel is hovering. The kernel moves from the top left to the right with a defined Stride Value. When it parses the complete image width, it jumps down to the next row, beginning on the left. This action is performed until the whole image is crossed. In this example we only consider positions where the kernel fits entirely within the image, but we can also consider the zero-padding approach were the pixels near the original image margins will be zeros. Image extracted from [34]

[1] Information and image extracted from: [34] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Another key element of CNNs are Pooling layers. Pooling layers are responsible for reducing the spatial size of convolved features. They reduce the amount of calculus performed by the computer and the amount of parameters in the network with a downsampling operation. In Figure 3.3 we can see the two most common pooling functions: Average Pooling. It calculates the average value of the part covered by the kernel and Max Pooling, which calculates the maximum value of the part covered by the kernel. The Max Pooling approach is used more often, as it also works as a noise suppressant.

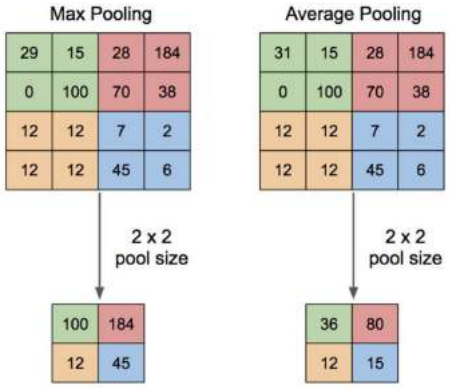


Figure 3.3: Illustration of Max Pooling and Average Pooling operation, Image extracted from: [35].

3.1.1.2 Classification - Fully Connected Layer

With this process we have now allowed the model to understand the features. Nevertheless we still need to classify the final image, and for that, the output will be flatten into a column vector and fed into a feed-forward neural network applying backpropagation on every training iteration, adjusting weights and biases. With multiple epochs the model is now able to classify images and generate output through a probabilistic distribution by using activation functions such as sigmoid , ReLU, or Softmax. [35] In Figure 3.4 we can observe a representation of the CNN architecture. Three stages are defined: feature extraction, classification and probabilistic distribution. Inside the feature extraction we have multiple convolutions extracting feature maps, followed by ReLU Layers. ReLU Layers turn every negative output value into zero, and take positive inputs directly to the output. The process is followed by pooling operations in order to downsample the feature maps. The features maps are then flatten into a vector column and fed into a fully connected layer. Inside the probabilistic distribution the SoftMax function will be used to predict the final output.

3.1.2 Semantic Segmentation

Semantic segmentation aims at linking a certain categorical label to every pixel in an image. [36] This means we can think of it like a classification problem per pixel, where

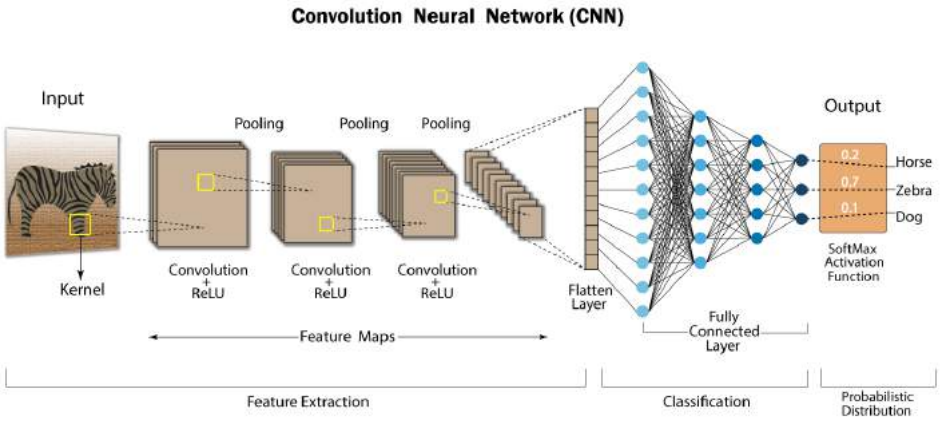


Figure 3.4: Illustration of a CNN designed for object detection. Image extracted from: www.developersbreach.com/convolution-neural-network-deep-learning/

each label corresponds to a different class. There are two types of segmentation: semantic segmentation and instance segmentation. The main difference between them is that semantic segmentation does not make a distinction between different instances of the same object while instance segmentation does. This means that if we have an image with multiple objects, for example cars, semantic segmentation will give the same label to all of the cars pixels, while instance segmentation will give a unique label to every instance of a specific car in the image. A typical architecture in image segmentation consists of having an encoder and a decoder. The encoder will be responsible for extracting features from the image through kernels. The decoder is in charge of producing the final output which is normally a segmentation mask containing the object outline. An example of a segmentation architecture can be observed in Figure 2.12.

3.2 Proposed System Architecture

3.2.1 First stage

In order to perform the road segmentation, first we need to chose a CNN based architecture. In the past years many CNN based architectures have been proposed, for example: The CasNet showed a lot of visual and quantitative advantages when compared with other state-of-the-art methods, but it has a higher level of difficulty associated with the implementation [17], FCNs showed increased results in road connectivity and road completeness [20], DCNNs [37] can overcome the problem of having complex backgrounds, effectively overcoming the phenomenon of "burr" with high speed and accuracy, but can be further improved to reduce the impact of shadows, trees and buildings.[22]

The proposed method is going to use the state-of-the-art DeepLabV3+ architecture in order to make the rural road segmentation. [25] There are many reasons why this

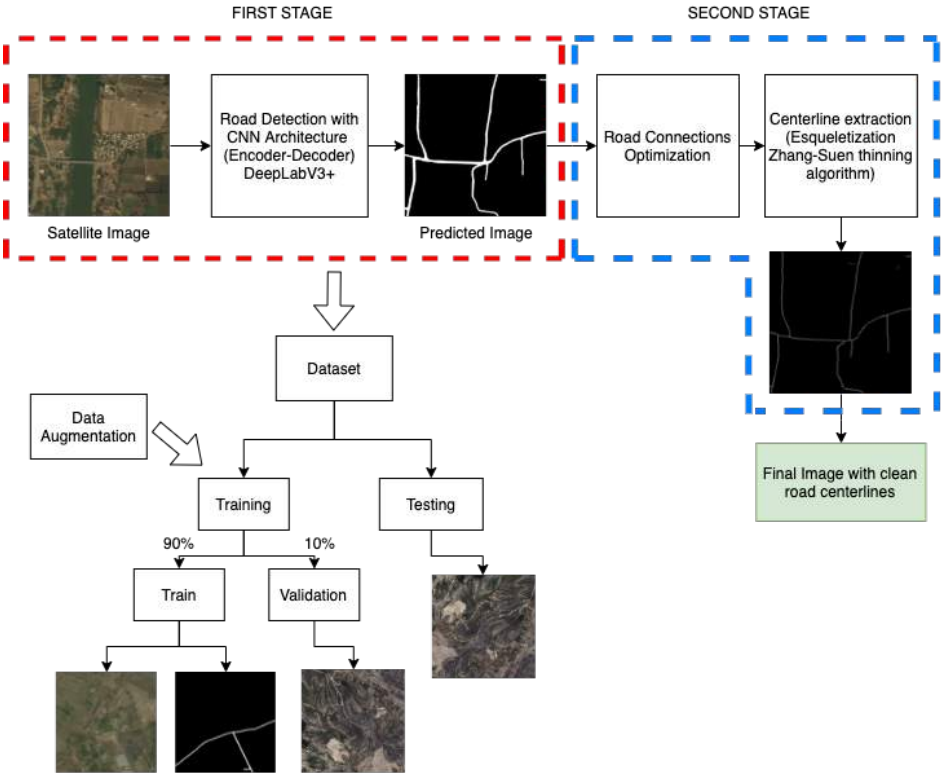


Figure 3.5: Workflow of the proposed method representing the first stage, the road region detection, and the second stage road centerline extraction.
On the first stage, the model that will acquire information about the aerial images will be chosen, with the mission of learning to detect rural roads and predicting a binary mask composed by the roads and background elements. The road region detector consists of training a model based on CNNs in order to perform the semantic segmentation and detect rural roads.
Secondly, an algorithm will be developed in order to optimize the roads connectivity and remove possible objects that might appear on the prediction phase.
And finally the road centerline from the previous segmented roads with an esqueletization method will be extracted. The expected output should be an image with white clean road centerlines with a black background.

architecture was chosen. One of them is this architecture, without any post processing, achieved a new best performance of 89.0% and 82.1% on cityscapes datasets, and the PASCAL VOC 2012 model becoming a new state-of-the-art.

Another reason why this network was chosen was because rural roads do not have a specific type of road material and since we need to detect many types of roads materials like dirt, asphalt, gravel and cement it was chosen an architecture based on deep convolutional neural networks that is not sensitive to road materials. [22]

Since rural aerial images have very complex environments with complicated landscapes, shadows, and trees, another reason why this method has been chosen is because this architecture uses the advantages of the Spatial Pyramid Pooling (SPP) and the Encoder-Decoder method.

- The SPP, [38–41] encode multi-scale contextual information, this means that it gives the network the ability to extract knowledge or apply knowledge to the information and does not require a fixed size input image.
- The Encoder-Decoder architecture has been proved to be very useful in image segmentation. [42–46] The encoder progressively diminishes the feature maps and acquire high semantic information while the decoder progressively recovers the spatial information. The Encoder-Decoder is able to extract sharp object boundaries and it also helps to extract features by using atrous convolution.

3.2.2 Second stage

After the dataset is divided, the network trained, and images are being predicted correctly, we can move on to the second stage, which is the road centerline extraction. On this stage, we want to extract the central position of the rural roads with a single-pixel width.

For that, it will be started by first creating an algorithm to improve the connections of the rural road intersections followed by the removal of noise from the predicted image. After completing this task, we can focus on extracting road centerlines.

There are multiple ways to extract road centerlines. One of the most used methods for road centerline extraction is by using the thinning algorithms like Zhang-Suen [47], and Guo-hall [48].

The goal of the thinning algorithm is to take a binary image, in our case white road, and black background, and draw a one pixel wide skeleton of that image, while maintaining the shape and structure of the road. Thinning algorithms are one of the most practical ways to implement the road extraction, even though it often produces small spurs around the centerline, that can affect the final structure of the road network.

With this architecture it is expected to have a precise and sharp rural road centerline. In the end, the rural road centerline should be a white one pixel-wide road, and the rest of the output image with a black background.

3.3 Synthesis

During the development of this dissertation, a work plan and revision of literature was performed with the mission to ensure that state-of-the-art methods are used in order to reach the initially desired goal of detecting and extracting rural roads centerlines from aerial images.

There are many problems and challenges that might occur while processing aerial images, like occlusion of roads caused by trees, cars, obstacles. A challenge that we will face is that rural roads don't have a specific road material. This means that we will need to be able to detect roads with different types of road materials like dirt, asphalt, gravel and cement. Another challenge that many state-of-the-art methods struggle to implement is to extract road centerlines from very tight curves with low visibility.

To successfully solve many of these problems it is proposed an architecture based on deep convolution neural networks using DeepLabV3+ to make the rural road detection, combined with a road connection optimization in order to connect road intersections and get rid of wrong predicted objects that might appear after the prediction phase. And finally, thinning algorithms like Zhang-Suen and Guo-Hall esqueletization algorithms will be used to perform the rural road extraction.

The next chapter will start by focusing on the theory behind the practical methods applied by making an in-depth study of each module. It will be followed by the implementation, where an environment to develop the model will be chosen. After that, the focus will go towards the network design, where everything about DeepLabV3+ will be explained. Then the dataset model will be split into multiple different sets like: test, train and validation dataset. Afterwards, the training of DeepLabV3+ will be performed so that we can finally start teaching the model on how to adjust the internal network weights and finally make predictions, and extract rode skeletons.

CHAPTER 4

Method Overview

This chapter will be in charge of describing the system throughout the course of this thesis. The necessary steps for the development of the datasets will be presented, followed by an in-depth study of each block of the architecture, and finally it will present the implementation as a unit. The first requisite to build our prototype successfully was to initially select the right development platform in order to conceive and test the system. Since this project demands a high level of processing power due to large amounts of geographical data, Google Colaboratory, or "Colab" was chosen. Colab is a cloud storage service aimed at creating and executing code in Python notebooks directly in the browser, without any pre-installed software. Since it takes a lot of time and resources to train deep learning models on a normal computer, it will be taken advantage of Colab Pro version utilizing (Nvidia K80/T4) as Graphics Processing Unit (GPU) with 12GB/16GB memory to compute the model. The Spyder IDE will also be used to develop supporting scripts and inspect variables while running the code.

4.1 Implementation - Rural Road Detection

4.1.1 Dataset Development

According to the Oxford Dictionary, the definition of a dataset is "a collection of data that is treated as a single unit by a computer". A dataset consists of multiple different elements of data that can be used to train an algorithm with the objective of discovering predictable patterns in the dataset. The predictions results of a deep learning model are directly influenced with the quality of the dataset in which the network will train on. This means that it's crucial to have a consistent and precise dataset in order to increase the overall accuracy of the model. The dataset that will be developed corresponds to the "Mação" municipality in Portugal, Figure 4.2. This district has been severely affected by

forest fires in the past, becoming a necessity to make more informed decisions in the field, in order to mitigate these events. [49]

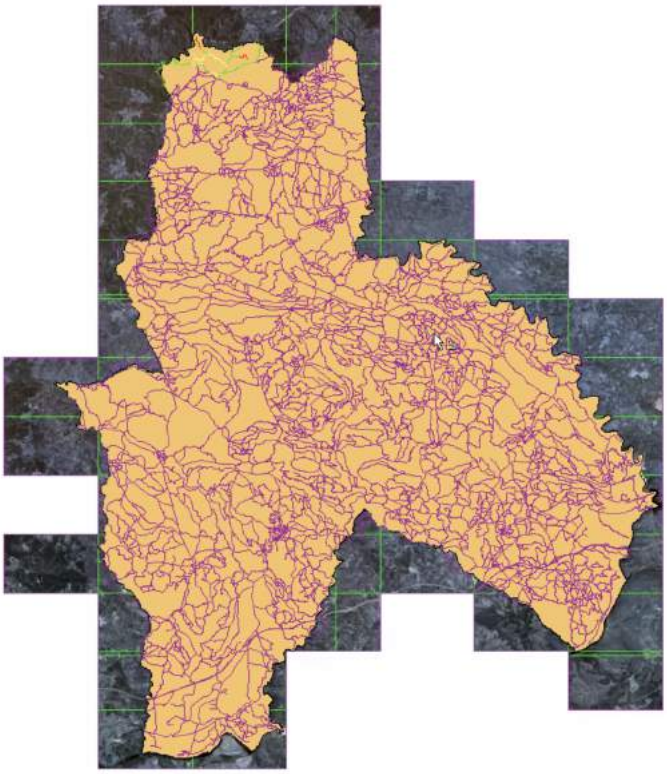


Figure 4.1: Visualization of "Mação" municipality in Portugal, using ArcMap software. Mação is positioned on top of the aerial images covering an aproximate area of $400Km^2$.

To build our dataset, high resolution aerial images provided by Direção Geral do Território (DGT) were used. These images were taken vertically from an aircraft with high accuracy cameras. Aerial images, since they are closer to the Earth's surface, are taken at lower altitude when compared with satellite images. This means that aerial images tend to have more details resulting in higher quality images. Another advantage is that this type of images are much more affordable becoming an excellent choice for the rural road extracting task. DGT aerial images come in Tagged Image File Format (TIFF), with an intensity resolution of 8 bits in RGBI (Red, Green, Blue and Infrared). These same images are cropped in tiles of (16000x10000) pixels, with a spatial resolution of 0.25m. Each image pixel corresponds to 0.25m on the landscape, which means that each tile has a field of view of 4Km by 2.5Km.

As we can see in Figure 3.5, the first step of our proposed architecture consists of creating a dataset to train our model. Thus, a ground truth image associated with each aerial image was first created. The aerial images were imported to photoshop in order to conceive the ground truth masks. These ground truth masks were created by adding a black layer on top of the aerial image and then using a white brushing tool on top of the rural roads to outline the roads tracing network. After manually doing this task

for several images, multiple pairs of aerial images with their corresponding masks were created. The next step was to cut those large aerial images (16000×10000 pixels) into smaller tiles (1024x1024 pixels). In Figure 4.2 can be seen a representation of the cropped tiles process Figure 4.3.

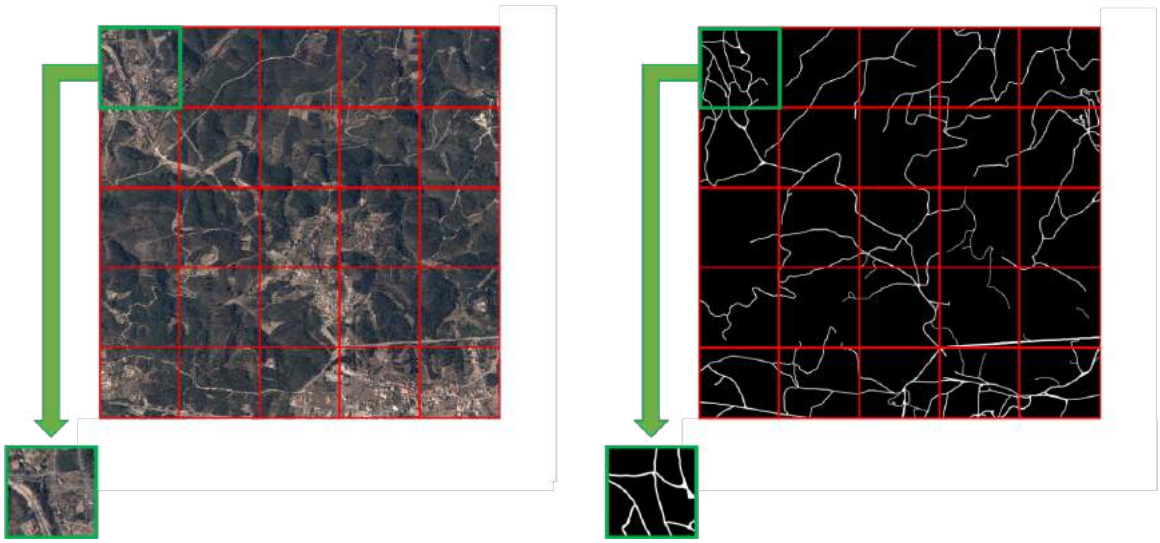
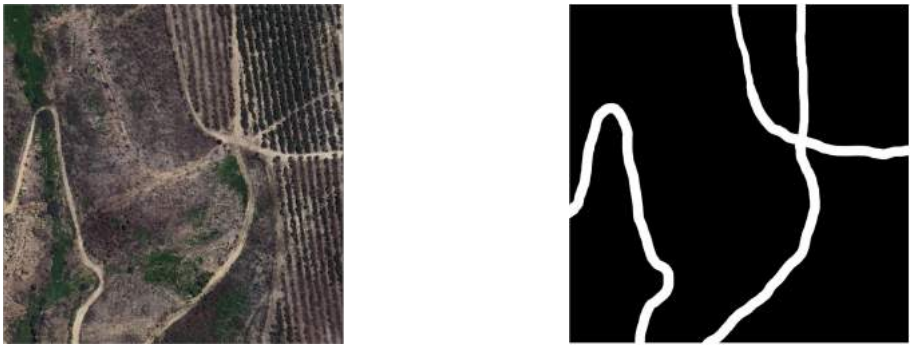


Figure 4.2: Aerial images and corresponding masks cutting process in 1024x1024 pixels tiles, each tile corresponds to an area of landscape of 256 meters by 256 meters.

Since the process of manually outlining the rural roads to create the ground truth masks is very tedious and time-consuming data augmentation was used. Data augmentation is a way to incorporate new data from the data already available. In neural networks the model performance is greatly dependent on the dataset, so in order to build a robust model, the focus should be on providing as much data as possible to the model. We can feed new data into the model just by making a some morphological operations on the image tiles. In Figure 4.4 we can see the process of augmenting aerial images and masks.



(a) (b)

Figure 4.3: Representation of both aerial and mask tiles. In picture (a) it is presented the cropped aerial image tile of 1024x1024 pixels, and in picture (b) it is shown the respective ground truth mask with the same size.

The next step consists of separating the aerial images dataset into three different sets:

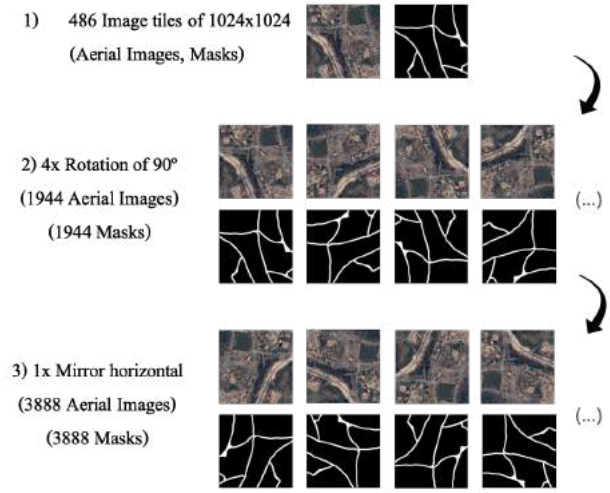


Figure 4.4: Enlargement of the dataset with data augmentation. 486 aerial image tiles were used with their corresponding ground truth masks. Those images were then rotated ninety degrees four times increasing the dataset size to 1944 images. Finally, a mirroring operation was performed doubling the dataset size to 3888. By performing data augmentation, the dataset went from having 486 pairs of aerial images and masks to 3888 pairs.

- **Train dataset** - The training dataset, as the name suggests, is responsible for training and helping the network determine how to adjust ideal parameters and biases comprising a model.
- **Validation dataset** - The validation dataset will make part of the training dataset and it is used to build the model, as it evaluates the performance of the validation. This dataset is used to prevent overfitting in the network by introducing features beyond the training dataset.
- **Testing dataset** - The testing dataset is a set of images that the network has never "seen" before, which respects the same probability distribution as the training dataset. The network tries to describe and predict the final evaluation based on the learned model. For the test dataset, a set of 81 images with their respective ground truth masks was created for testing the model to, later, evaluate the results with multiple metrics.

All of the aerial images and ground truth masks in the train, validation, and testing datasets have their respective paths saved into a CSV file. Those paths are stored inside dataframes for faster access and easier data manipulation.

4.1.2 One-Hot Encoding

A lot of machine algorithms are not able to handle categorical variables (variables that contain label values instead of numerical values). In order to surpass this challenge it is

normally required that these variables are converted into one-hot vectors, also known as dummy variables. These variables have binary values, similar to bits, taking values of one and zero (equivalent to true and false). To encode aerial images, the algorithm needs to be able to distinguish between "Road" and "Background". With that goal in mind, the categorical values begin in 0 and go up to N-1 categories, being the N equal to the number of classes, which in this case, is equal to 2. Firstly, the classes were read with their respective values from a CSV file. Rural roads masks have values of [255,255,255], while the background have only values of [0,0,0], with a shape of (1024x1024x3). Figure 4.5

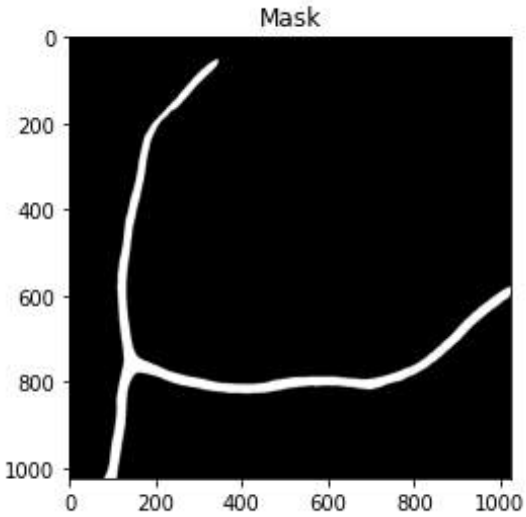


Figure 4.5: Original mask with white and black pixels corresponding to the rural road, and the background labels respectively.

It is then performed one-hot encoding by replacing each mask pixel value with a vector of length equal to number of classes. The classes are the vector [0,1] corresponding to the rural roads, and the vector [1,0] to the background pixels, resulting in an image shape of (1024x1024x2). It is returned a 2D array with the same height and width as the input, but with a depth size of number of classes, in Figure 4.6 one can see the masks that are fed into the proposed model in order to train it.

When necessary, the process of one-hot encoding can also be reversed. This operation takes a one-hot encoded image (with a depth equal to the number of classes), and transforms it into an array with only 1 channel, in which each individual pixel has the value of the classified class key, resulting in an image of shape (1024x1024) with the road value equal to 1, and a background value equal to 0, Figure 4.7

The last step to fully reverse the mask from (1024x1024) to the original mask shape (1024x1024x3), we only need to add the original label values from the CSV file previously read, to the 1 channel class keys array. Figure 4.8

The process of one-hot encoding, and reverse one-hot encoding allows the algorithm to process categorical variables and do a better job at making predictions, making it possible to switch from categorical variables to numerical values and vice-versa.

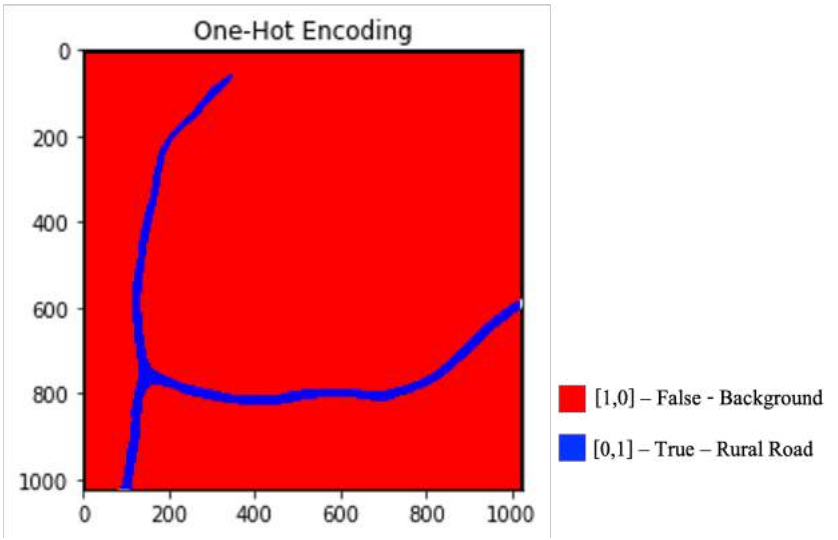


Figure 4.6: Since a one-hot encoded image doesn't have a direct visual representation, we have considered the vector [1,0] as "False" representing the background, and the vector [0,1] as "True" representing the rural road.

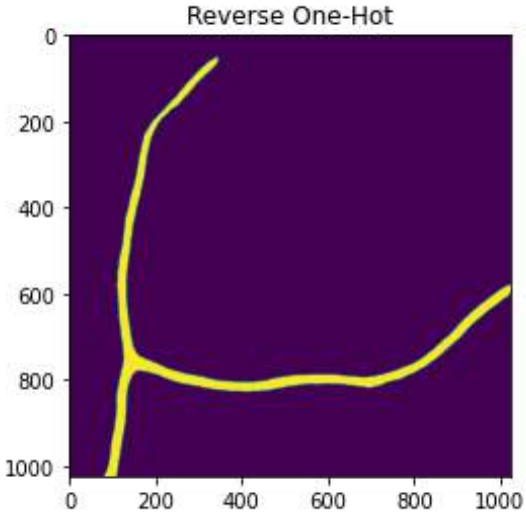


Figure 4.7: Representation of a reverse one-hot encoded image. This image will have a depth of 1, meaning that it can only consider the height and the width channels resulting in a shape of (1024x1024), having rural roads pixels with a value of 1, and background pixels with the value of 0.

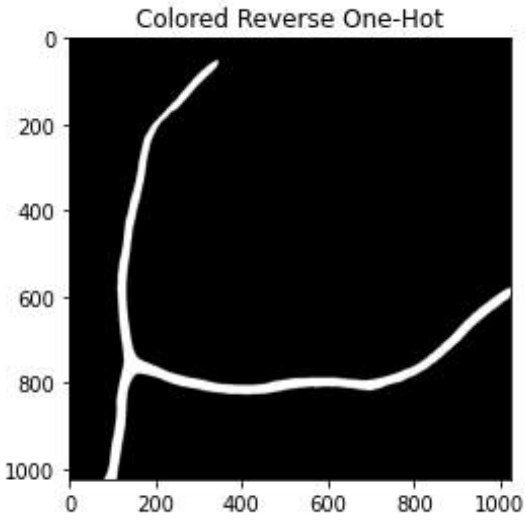


Figure 4.8: Representation of a colored reverse one-hot encoded image. It will be added to the reverse one-hot encoded image the 3 RGB channels with the values of [255,255,255] and [0,0,0], recovering the original mask.

4.1.3 Defining the Dataset

To begin with, a dataset class called *RuralRoadsDataset* with the abstract Python pytorch class *torch.utils.data.Dataset*[1] was created. This class receives as arguments: a dataframe containing images and labels paths, a list with the RGB values of the segmentation mask classes, the augmentation pipeline, and preprocessing operations. An image with a shape of (1024x1024x3), and a one hot encoded mask with a shape of (1024x1024x2) is returned. The *RuralRoadsDataset* class inherits the next methods:

Listing 4.1: Defining RuralRoadsDataset class

```
...

class RuralRoadsDataset(torch.utils.data.Dataset):

    #Where it will be read the csv data
    def __init__(
            self,
            df,
            class_rgb_values=None,
            augmentation=None,
            preprocessing=None,
    ):
        self.image_paths = df['aerial_image_path'].tolist()
        self.mask_paths = df['mask_path'].tolist()

        self.class_rgb_values = class_rgb_values
        self.augmentation = augmentation
        self.preprocessing = preprocessing

    #To support indexing such that dataset[i].
```

[1] https://pytorch.org/docs/stable/data.html

```
    #Reading of images to __getitem__ is more efficient
    def __getitem__(self, i):

        # read images and masks
        image = cv.cvtColor(cv.imread(self.image_paths[i]), cv.COLOR_BGR2RGB)
        mask = cv.cvtColor(cv.imread(self.mask_paths[i]), cv.COLOR_BGR2RGB)

        #otsu binarization, improves image quality, only 0 and 255 pixels.
        ret , mask = cv.threshold(mask,127,255,cv.THRESH_BINARY)
        #Mask to one hot encoding
        mask = one_hot_encode(mask, self.class_rgb_values).astype('float')


        # apply augmentations
        if self.augmentation:
            sample = self.augmentation(image=image, mask=mask)
            image, mask = sample['image'], sample['mask']

        # apply preprocessing
        if self.preprocessing:
            sample = self.preprocessing(image=image, mask=mask)
            image, mask = sample['image'], sample['mask']

        return image, mask

    #len(dataset) returns the size of the dataset.
    def __len__(self):
        return len(self.image_paths)
...
```

- **___init___**- The init method is the construtor of a class, where the csv data will be read (dataframe with images/label paths, class_rgb_values, augmentation, and preprocessing).
- **___len___**- Allows the method len(dataset) to return the size of the dataset.
- **___getitem___**- The getitem method supports indexing such as dataset[i]. Where the images are read becoming more memory efficient as all the images are not stored in the memory at once but read as required. The ___getitem___ will also return the aerial image and the mask, being a tuple pair of (x, y).

The *RuralRoadsDataset* class will be essential to create iterable objects like the training dataset, validation dataset, and testing dataset.

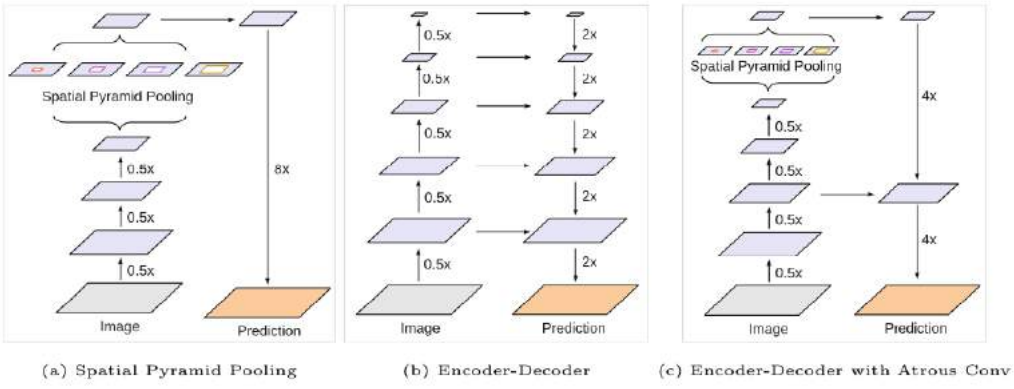
4.1.4 Deep Learning Model

Now that the datasets are ready, a deep learning model needs to be defined to perform the road segmentation. In this thesis our proposed method focuses on using the state-of-the-art deep learning model for semantic image segmentation DeepLabV3+, by Google, to perform the rural rode detection. [25] DeepLabV3+ has strong architecture characteristics to solve this task, such as:

- **Spatial pyramid pooling** - Spatial pyramid pooling has been proven to be a flexible solution for handling different scales, sizes, and aspect ratios. [50, 51] Spatial pyramid pooling encodes multi-scale contextual information by probing the incoming features with filters and applying several parallel atrous convolutions with different rates. This means that atrous convolutions are able to control the resolution of how features are computed, also known as Atrous Spacial Pyramid Pooling (ASPP). In the last years these models have outperformed and demonstrated good results in segmentation becoming benchmarks. In Figure 4.9 (a) a representation of Spatial Pyramid Pooling can be seen.

- **Encoder-Decoder** - The Encoder-decoder architecture have been widely used in semantic segmentation. [52–54] It consists of two main parts. The encoder will progressively reduce the spatial size of the feature maps and gather high semantic information, while the decoder will recover the spatial size and detailed object boundaries. This means that the structure is able to extract sharper object outer limits by progressively recovering spatial information. In Figure 4.9 (b), one can see a representation of an encoder-decoder.

- **Depthwise Separable Convolution** - Depthwise separable convolution has been applied in recent neural network designs. [55] A depthwise separable convolution has the main purpose of dramatically reducing the overall computational costs and number of parameters while keeping an equal or even higher performance. This result is achieved by performing depthwise spatial convolution for each channel independently followed by a pointwise convolution (1x1 convolution).



(a) Spatial Pyramid Pooling (b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv

Figure 4.9: The model DeepLabV3+ combines the advantages of the spatial pyramid polling and the encoder-decoder, resulting into a encoder-decoder with atrous convolution. Image from [25]

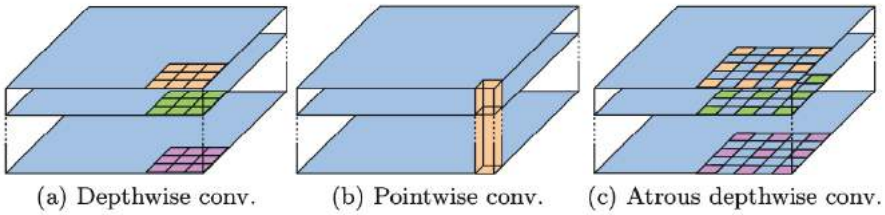


Figure 4.10: Picture (a) represents a depthwise separable convolution, that applies a single filter for each input channel. Picture (b) represents a pointwise convolution which is a convolution from the output of the depthwise convolution. Lastly, picture (c) represents the atrous depthwise convolution with $rate = 2$. Image from [25]

4.1.4.1 Encoder-Decoder with Atrous Convolution

Atrous Convolution - Atrous convolution grants us the ability of exactly control the feature maps resolution inside the model, and adapt the filter's field-of-view with the purpose of capturing information without a specific size. Considering the feature map output as Y, the convolution kerner as w, and the input feature map as x, the atrous rate r, for each location i the convolution operation is given as:

$$Y[i] = \sum_{k} x[i + r.k]w[k] \tag{4.1}$$

If the rate is equal to one, it can be thought of as the standard convolution that we discussed in the previous chapter. When the rate starts increasing, it also increases the field-of-view. [56] This characteristic gives the convolution the ability to change the rate value in order to encode at multiple scales.

4.1.4.2 DeepLabV3+ Encoder

The DeepLabV3+ architecture makes use of the previous DeepLabV3 encoder by using atrous convolution to extract feature maps [51]. ASPP is using four parallel convolution operations, with two different types of convolution: 1x1 convolution and 3x3 convolution, with rates [6, 12, 18]. In order to perform semantic segmentation, an output stride = 16 can be used for the best relationship between speed and accuracy, or an output stride = 8 for an even higher performance, but with more computational costs. The output stride is the relationship between the input image resolution compared to the output resolution. This means that, for an output stride = 16, the output feature map will be 16 times smaller than the input image. DeepLabV3+ improves the ASPP module by using multiple convolutions with different rates while being able to extract features with arbitrary resolution depending on the computer resources available. The feature map presented at the output will have 256 channels and plenty of semantic information. The encoder can be observed in the blue module in Figure 4.11.

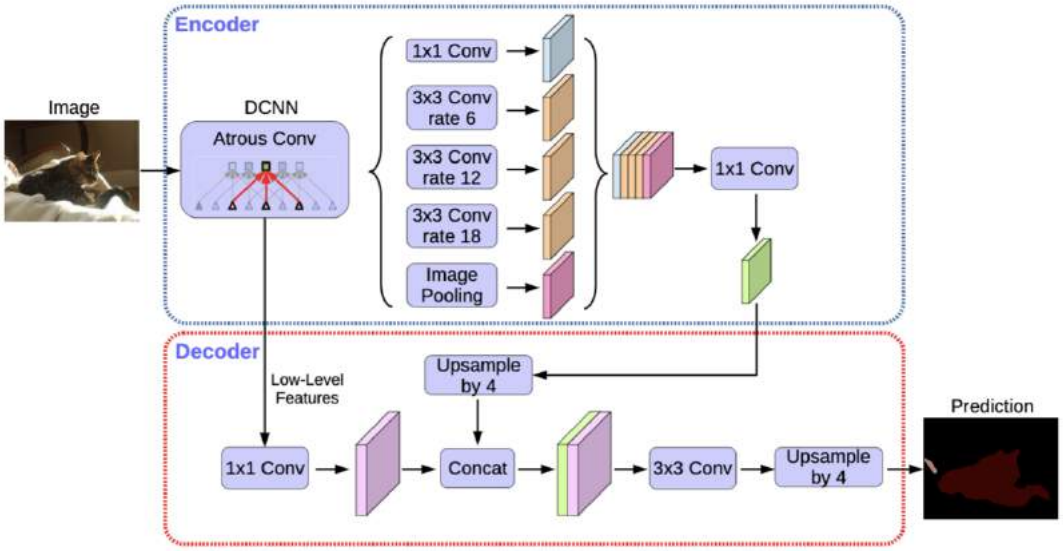


Figure 4.11: DeepLabV3+ network architecture composed of an encoder and a decoder module. The encoder extracts essential information from the image. Image from [25].

4.1.4.3 DeepLabV3+ Decoder

After the encoding procedure, the feature maps are bilinearly upsampled by 4, followed by a concatenation with the respective low level features with the same resolution. A pointwise 1x1 convolution is used to diminish the number of channels and parameters. Another reason to diminish the number of channels is due to the low level features maps, which normally have a higher amount of channels making rich feature maps lose their relevance. Lastly, 3x3 convolutions are applied, with the intention of refining the feature maps, and afterwards, a bilinear upsampling with a factor of 4 is used to bring the feature map to the original size of the input image. The decoder can be observed in the red module in Figure 4.11.

4.1.5 Defining DeepLabV3+

After we pick our deep learning model, we need to be able to teach DeepLabV3+ what rural roads are. To begin with, a Python image segmentation neural network framework based on PyTorch is used. [57] In addition, several DeepLabV3+ model parameters need to be defined, such as:

Listing 4.2: Defining DeepLabV3+ model

```
...
ENCODER = 'resnet50'
ENCODER_WEIGHTS = 'imagenet'
CLASSES = select_classes
ACTIVATION = 'sigmoid'

model = smp.DeepLabV3Plus(
    encoder_name=ENCODER,
    encoder_depth=5,
    encoder_weights=ENCODER_WEIGHTS,
    encoder_output_stride=16,
    decoder_atrous_rates=(12,24,36),
    in_channels=3,
    classes=len(CLASSES),
    activation=ACTIVATION,
    upsampling=4
)
...
```

- **encoder_name** – This parameter represents the feature extractor encoder, commonly known as the network backbone, that extracts features with distinct spatial resolutions from the input image. For the backbone, a Residual Neural Network (ResNet) is used. This type of networks have been awarded the first place in the ILSVRC 2015 classification task, ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. [58] The chosen residual network was ResNet50, which is a convolutional neural network with 50 layers (48 Convolution layers, 1 Max-Pool, and 1 Average Pool layer) Table 4.2. ResNet uses skip connections to add the output from a previous layer to further layers helping mitigate the vanishing gradient problem, providing the benefit of depth and reducing computational resources.

Encoder	Weights	Params
ResNet18	imagenet / ssl / swsl	11M
ResNet34	imagenet	21M
ResNet50	imagenet / ssl / swsl	23M
ResNet101	imagenet	42M
ResNet152	imagenet	58M

Table 4.1: Different types of residual networks. ResNet50 is used because it is an ultra-deep neural network with over 23 million trainable parameters, and in our context, deeper networks like ResNet101, or ResNet152 were not used, as they consume most of the available memory resources inside Google colab.

- **encoder_depth** – The encoder depth corresponds to the number of downsampling operations inside the encoder. It can vary between 3 to 5. In our case, the encoder depth will be considered equal to 5. In each stage the feature map size decreases by half compared to the previous stage. This means that for stage zero, the feature

Layername	Output size	ResNet-50
conv1	112×112	7×7, 64, stride 2
conv2_x	56×56	3×3 max pool, stride 2
		$\begin{pmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{pmatrix} \times 3$
conv3_x	28×28	$\begin{pmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{pmatrix} \times 4$
conv4_x	14×14	$\begin{pmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{pmatrix} \times 6$
conv5_x	7×7	$\begin{pmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{pmatrix} \times 3$
	1×1	Average Pooling, 1000-d fc, softmax
FLOPs		3.8×10^9

Table 4.2: The Resnet50 architecture includes: In the first layer a convolution 7×7 with 64 kernels, and a stride of 2, resulting in one layer. Secondly a 3×3 max pooling layer with a stride of 2. Next it has a convolution with 1×1, 64 kernels, followed by a 3×3, 64 kernels and a 1×1, 256 kernels. This set of three layers are repeated 3 times resulting in a total of 9 layers. The following convolution has a 1×1, 128 kernel followed by a 3×3, 128 and a 1×1, 512 kernel. This convolution is repeated 4 times giving a total of 12 layers. The next convolution contains a 1×1, 256 kernel followed by a 3×3, 256 kernel and a 1×1, 1024 kernel. This convolution is repeated 6 times resulting in a total of 18layers. The last convolution has a 1×1, 512 kernel followed by a 3×3, 512 kernel and a 1×1, 2048 kernel. The convolution is repeated 3 times resulting in 9 layers. Lastly it is performed an average pooling operation, followed by a fully connected layer containing 1000 nodes, and a softmax operation resulting in one layer. All of these layers make a total sum of 50 layers deep.

maps shapes are [(N, C, H, W),] for a depth equal to one, it will be [(N, C, H, W), (N, C, H/2, W/2)] and so forth. N is the batch size, C represents the number of channels, H is the height of input plane in pixels, and W is the width in pixels.

- **encoder_weights** – The encoder weights represent the multiplication factor of the kernels in the convolutional layers. For each encoder in Table 4.1 the respective pre-trained weights are also presented. "Imagenet" [59] is used as the pre-trained encoder weights. Using a pre-trained encoder vastly speeds up the training time of the model.

- **encoder_output_stride** – The encoder output stride represents the relationship between the input image resolution compared to the output resolution of the last encoder features. An encoder_output_stride = 16 is used, for the best relationship between speed and accuracy.

- **decoder_atrous_rates** – The decoder atrous rates are the dilatation rates for the ASPP unit. Decoder_atrous_rates = (12, 24, 36) is used.

- **in_channels** – This parameter represents the number of input channels. Since we are using RGB images, the value will be equal to 3.

- **classes** – The classes parameter represents how many classes the output has. Rural road detection only implies 2 classes, the road class and the background class.

- **activation** – The activation function is the function that is used next to the last convolutional layer, in order to generate the output based on the inputs. Some examples of activation functions are "sigmoid", "tanh", "logsoftmax", "softmax", and "identity". Since our problem only has 2 classes (road, background), it is recommended to use a two-class logistic regression by using the sigmoid activation function. Equation 4.2

$$\phi(z) = \frac{1}{1 + e^{-z}} \tag{4.2}$$

- **upsampling** – The upsampling parameter will be the factor that keeps the same input to output racio. This factor is equal to 4, since after the encoding module the feature sizes are decreased by 16.

4.1.6 Defining Dataloaders

With the aim of improving the dataset management, and make the machine learning pipeline process easier, it was defined dataloaders. As seen previously, the dataset object will be in charge of of storing the data. The dataloaders will handle, and iterate through the dataset object being able to batch the data, shuffle it, and load the data in parallel by using multiple workers processing information at the same time. The main reason why it uses dataloaders instead of just simply operating with the dataset directly is that if used directly through a for loop, a lot of features will be lost while iterating over the data.

In order to create dataloaders the iterator *torch.utils.data.DataLoader*.[2] Two dataloaders were defined, one for the train loader, and another for the validation loader. The dataloaders parameters that are used are:

- **dataset** - This parameter refers to the dataset that is iterated by the dataloader, which can be the train, validation, or testing dataset.

[2]Information extracted from:
https://pytorch.org/tutorials/recipes/recipes/custom_dataset_transforms_loader.html

- **batch_size** - The batch size defines the number of samples utilized in one iteration that will be propagated through the network. Usually the bigger the batch size the faster the model will train, but also consuming more Random Access Memory (RAM). This parameter is especially important when we are not able to fit the whole dataset in the Colab GPU memory.

- **shuffle**- If the shuffle is set to "True", the internal RandomSampler is used, which will exchange the indices of all samples, otherwise, by default, this value is set to "False".

- **num_workers** - The number of workers parameter will be in charge of how many sub-processes are used to load data. Since our machine is not using all of the resources and cores at the same time, we can add multiple workers with batches queued in memory to speed up the model training.

- **drop_last**- The drop last parameter will be set to "True" when the number of samples in the dataset is not entirely divisible by the batch_size, ignoring the last batch.

To create the train dataloader, some parameters are used, such as the training dataset, a batch size equal to 4, shuffle set to "True", number of workers equal to 4, and drop last equal to "True". For the validation dataloader the validation dataset is used, as well as, a batch size equal to 4, shuffle set to "False", number of workers equal to 4, and drop last equal to "True".

4.1.7 Defining Hyperparameters

The following steps consist of defining hyperparameters. Hyperparameters are settings that can be adjusted before training a ML model. They have a high level of importance, being strongly directed with the training time, infrastructure prerequisites, cost of resources, model accuracy and convergence. The hyperparameters used will serve as controllers of the learning process as described below:

- **epochs**- The number of epochs corresponds to the number of complete passes by the algorithm in the whole training dataset. 3 training epochs are set throughout the dataset.

- **device** - The device can either utilize the Compute Unified Device Architecture (CUDA), or the Central Processing Unit (CPU). The device that is used to process the algorithm is the Application Programming Interface (API) package of CUDA. This package adds support to CUDA tensors, by applying the same function as CPU tensors, but utilising CPUs for computation.

- **loss** - A loss function is used in order to measure how well a prediction was made, comparing the predicted value with the ground truth value. Therefore, we shift the learning problem into an optimization problem with the goal of minimizing the loss function. Thus, it influences how the internal weights are tuned, while executing backpropagation, becoming a direct influence on the model's performance. In this case the DiceLoss function is used, which simply is one minus the dice coefficient.

- **metrics**- The metrics measure and monitor the performance of the model during the training and testing phase. Was initially used IoU, and later state-of-the-art metrics were adopted.

- **optimizer** - Adam is used for the optimizer, which is an algorithm for first-order gradient-based optimization of stochastic objective functions. [60]. This method requires less memory, is computationally efficient, and it is appropriate for problems that have considerable amounts of parameters and data. The hyperparameters also have an intuitive understanding, requiring almost no adjustment. The optimizer is set with a learning rate of 8×10^{-5}.

Listing 4.3: Defining training and validation epochs

```
...
train_epoch = smp.utils.train.TrainEpoch(
    model,
    loss=loss,
    metrics=metrics,
    optimizer=optimizer,
    device=device,
    verbose=True,
)

valid_epoch = smp.utils.train.ValidEpoch(
    model,
    loss=loss,
    metrics=metrics,
    device=device,
    verbose=True,
)
...
```

4.1.8 Training DeepLabV3+

Deep learning models need to learn the mapping relationships between inputs and outputs in order to make predictions. This process includes discovering a series of weights that are a good fit to solve a specific problem, which in, our case, is to teach the model to distinguish between rural roads, and background pixels from the aerial images. After creating the training epochs, the model can start to be trained. The training process consists of simply looping through the data iterator and feed the inputs to the network and optimize it. After

iterating through the number of epochs, the model is saved as a .pth model. This process takes an approximate time of 3-4hours per training, while using Google Colab GPUs.

4.1.9 Prediction on Test Data with DeepLabV3+

After training the DeepLabV3+ model and completing all epochs, we can now start making predictions. Similar to the training dataset and the validation dataset instances, a new instance is created, called test dataset, with the class *RuralRoadsDataset*. This instace is passed down to the test dataloader, in order to be used with by DeepLabV3+ and later make the evaluation of the model.

The process to predict images consists of iterating through each position of the test dataset object. Each position of this object has an aerial image and the respective ground truth associated. The aerial images have a shape of (3, 1024, 1024), corresponding to (Channel, Height, Width), having the channel initially 3 RGB components. The aerial images will be then unsqueezed. This process adds a dimension of 1 in the 0th position resulting in a shape of (1, 3, 1024, 1024). These tensors are sent into the DeepLabV3+ model. The model makes a prediction returning a mask with a shape of (1, 2, 1024, 1024), remembering that the 2 represents the two possible one hot encoded vectors [0,1] or [1,0]. The prediction mask is then squeezed to remove the 0th position resulting in a shape of (2, 1024, 1024). The following step consists of converting the prediction mask from Channel, Height, Width (C,H,W) format into Height, Width, Channel (H,W,C) format, by transposing the predicted mask to (1024, 1024, 2). The last step consists of performing a reverse one-hot encoding, turning the predicted mask into a one depth channel (1024x1024) and then adding 3 RGB channels with the values [255,255,255] to rural roads pixels, and [0,0,0] to background pixels. In Table 4.3 one can see a comparison between the aerial images, ground truth and predicted masks by the model.

Nr.	(a)	(b)	(c)
1			
2			
3			
4			
5			

Table 4.3: Rural road detection network for the Mação district. Column (a) represents the aerial image tiles; in column (b) the ground truth masks, and column (c) represents the predicted mask by the model.

4.2 Implementation - Rural Road Centerline Extraction

After the rural roads are predicted by the model, the next step is to extract the rural roads centerlines. Roads centerlines are vectors with the information related to the exact geographic center of the road. The data stored in roads centerlines simplify its use in multiples GIS applications. Road centerlines are normally used to gather specific information about the road like: road name, type of pavement material, road diameter, emergency dispatch, travelling path, velocity limits and number of road ways. This type of information is extremely useful not only in the daily use of car navigation systems but also in the forest fire context, as they provide the infrastructure allowing firefighters to make better and more informed decisions in the field.

The rural road centerline extraction is performed with the use of thinning algorithms. These algorithms play a valuable role in depressing image noise and improving the minutiae extraction algorithm, thus improving the performance of the system. [61] The thinning

algorithm permits to decrease the image complexity by reducing thick image objects into thin lines of one pixel wide skeletons that go through the middle of the object. A lot of thinning algorithms have been developed in the past years. [48, 62–65] There are two main types of thinning algorithms classifications, iterative boundary removal algorithms and non-iterative distance transformation algorithms, Figure 4.12. The iterative type deletes pixels on the object bounders until there are no more pixels to be removed resulting in a one pixel width object skeleton. The non-iterative type is not suitable for generic applications as they are not robust enough, particularly for different thicknesses and patterns with complex stroke directions. Inside the iterative methods there are also two sub fields, the sequential and parallel. In sequential algorithms the pixels are inspected for deletion in a established sequence in every iteration. The deletion of the pixel P in the n^{th} iteration depends on the previous operations performed in the last $(n-1)^{th}$ iterations. In a parallel algorithm the pixels are inspected individually in a parallel manner in every iteration, which means that the deletion of pixels in the n^{th} iteration exclusively depends on the result of the n^{th} iteration. [63]

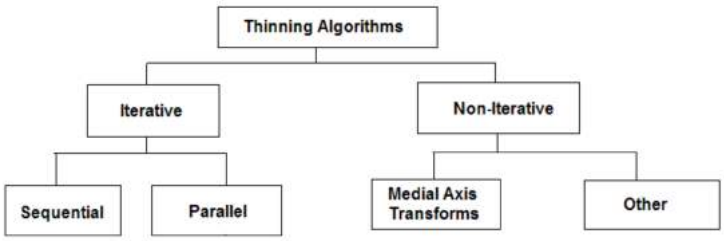


Figure 4.12: Different thinning algorithms classifications. Image from [61]

4.2.1 Zhang and Suen Thinning Algorithm

One of the approaches that are used to make the rural road centerline extraction are the Zhang and Suen Thinning algorithm. This iterative parallel algorithm has been commonly used in pre/post processing, and image recognition tasks. This operation is usually used in binary images where the goal is to erode the thickness of foreground pixels and produce a new binary image with the same objects skeletonized. This algorithm is one of the most used thinning algorithms. [66] It is also known as 2-pass algorithm as it resides on two sub-iterations: on the first sub-iteration, the goal is to delete the South-East boundary pixels, and the North-West corner pixels, while the second one focuses on deleting the North-West boundary pixels and the South-East corner pixels that are the opposite orientations, Figure 4.13. A flowchart of the Zhang and Suen thinning algorithm is shown in Figure 4.14.

All of the pixels on the outline of the image are removed in order to only extract pixels that belong to the skeleton. [67] In the fist sub-iteration, the contour point $P1$ is deleted if the next conditions are satisfied:

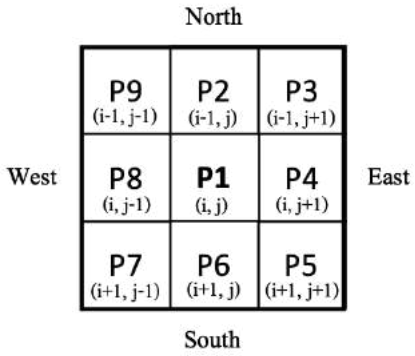


Figure 4.13: Zhang Suen 3 x 3 neighborhood.

1. $2 \leq B(P1) \leq 6$
2. $A(P1) = 1$
3. $P_2 \times P_4 \times P_6 = 0$
4. $P_4 \times P_6 \times P_8 = 0$

Towards the second sub-iteration, the conditions 3 and 4 change. The contour point $P1$ is deleted if the next conditions are satisfied:

1. $2 \leq B(P1) \leq 6$
2. $A(P1) = 1$
3. $P_2 \times P_4 \times P_8 = 0$
4. $P_2 \times P_6 \times P_8 = 0$

$A(P1)$ is defined as the number of white-black, "0-1" patterns in the respective order, in clock-wise traversal of $P2$ to $P9$ neighbors. $B(P1)$ is defined as the number of non-zero neighbours of $P1$, represented by Equation 4.3:

$$B(P1) = \sum_{i=2}^{9} P_i \tag{4.3}$$

4.2.2 Guo Hall Thinning Algorithm

The Guo Hall thinning algorithm is another example of parallel thinning. [48] The pixels on the contour are inspected for removal in an iterative 3×3 neighborhood process. Every iteration is splitted into two subcycles, one removing the north and east pixels, and the

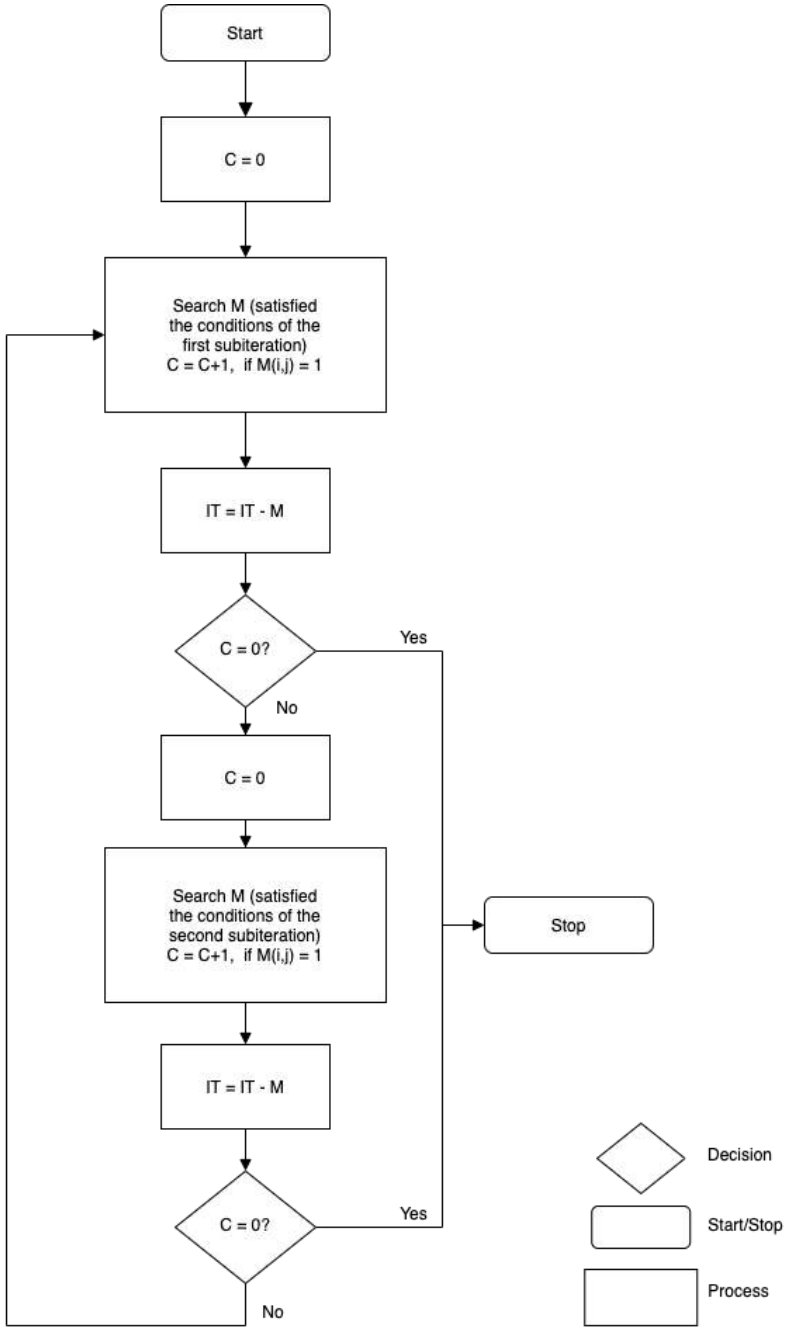


Figure 4.14: Flowchart of the Zhang and Suen thinning algorithm. Firstly, the image is stored in the matrix IT, and is defined a counter C set to 0. The output is stored in the matrix IT. For memory optimization purposes two matrices, IT and M, are used in the process.

other for the south and west pixels. [68] With the goal of preserving end points and removing excessive pixels. A new operator named N(P), defined as Equation 4.4, was proposed:

$$N(P) = min[N_a(P), N_b(P)] \tag{4.4}$$

$$N_a(P) = (P1 \vee P2) + (P3 \vee P4) + (P5 \vee P6) + (P7 \vee P8) \tag{4.5}$$

$$N_b(P) = (P2 \vee P3) + (P4 \vee P5) + (P6 \vee P7) + (P8 \vee P1) \tag{4.6}$$

As shown in Figure 4.15, the operators P2, P4, P6, P8 are the pixel P side neighbours, while P1, P3, P5, P7 are the diagonal neighbours. $N_a(P)$, Equation 4.5, and $N_b(P)$, Equation 4.6, break the ordered set of the pixel P neighbours into four pairs of adjacent pixels and count the number of pairs containing one or two 1s. [61] The symbols $\wedge$ and $\vee$ represent the logical operand AND and OR, respectively.

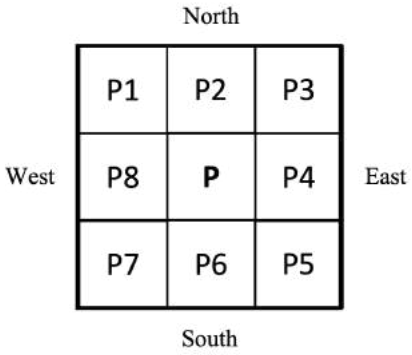


Figure 4.15: Neighborhood definitions for pixel P [48]

If the following conditions are satisfied, the algorithm is going to delete pixels. The thinning operation will end if there are no more additional deletions to be made.

1. C(P) = 1, where C(P) = number of distinct eight connected components of 1s in the 3x3 neighborhood of the pixel P.
2. N(P) = 2 or N(P) = 3
3. Apply one of the next conditions:

   $(P2 \vee P3 \vee NOT(P5)) \vee P4 = 0$ (for odd numbered iteration) or

   $(P6 \vee P7 \vee NOT(P1)) \wedge P8 = 0$ (for even iterations)

The first condition allows the preservation of local connectivity when P is deleted, and also prevents deletion of pixels in the middle of medial curves. The utilization of C(P) permits that a few of the 1s in the middle of two-width diagonal lines are deleted, which in [47, 69] were preserved. The variable N(P) adds an end-point check substituting

B(P) utilized in [47, 69]. If $B(P) = 1$, P is an end-point and $N(P) = 1$. Although when $B(P) = 2$, P can or cannot be an endpoint. This means that N(P) permits end-points to be preserved while deleting several redundant pixels in the middle of the curve. [48]

4.3 Final Architecture

The final architecture is shown in Figure 4.16. The previous modules are now grouped up in a system that gathers the ability to take aerial rural road images, processing them inside the DeepLabV3+ model, making predictions of the road and background pixels, and creating a predicted mask, completing the road detection process. After this step, an optimization algorithm that helps increase the road connectivity on intersections, also deletes small pixels and artefacts cleaning the final image. And, finally it uses thinning algorithms in order to extract the roads centerlines. The final architecture implementation can be consulted in Appendix A, inside the Google Drive Link, through Google Colab application.

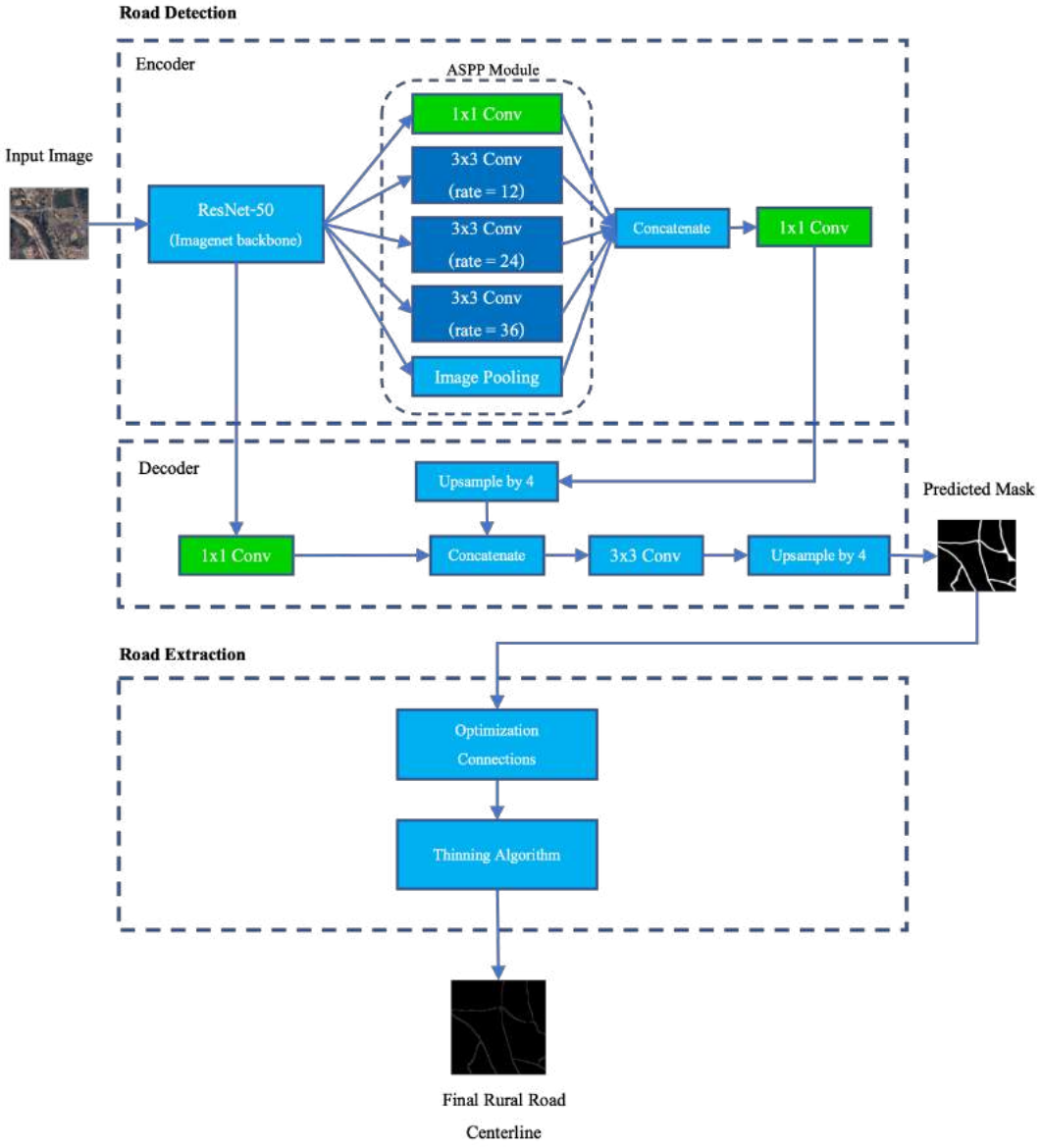


Figure 4.16: Final rural road centerline extraction system composed of the road detection and road extraction module.

CHAPTER 5

Validation and Results

The current chapter describes the results and validation of the previously proposed model architecture. Firstly, all of the metrics that will be taken into the study of road detection and centerline extraction are defined. For the road detection, a qualitative and quantitative comparison between the DeepLabV3+, Unet, FPN model is made, and the strengths and weaknesses of each model are stated. Secondly, a qualitative optimization for road interceptions is made, as well as the removal of small white objects that introduces noise into the predicted image. Last but not least, the road centerline with thinning algorithms like Zhang-Suen and Guo-Hall are extracted, making a comparison between results by using multiple buffers with different road width sizes.

5.1 Evaluation Metrics

To evaluate the road detection model, multiple metrics are defined. A study of the confusion matrix is made, which provides an in depth analysis to truly judge the model, and characterize the whole performance of the algorithm. The confusion matrix rows presents the class instances while the columns represents the predicted class instances. There are 4 important terms in this type of evaluation:

- **True Positives (TP)**- The cases in which the model predicted YES, and the output was YES. Represents the road length of pixels correctly extracted.
- **True Negatives (TN)**- The cases in which the model predicted NO, and the output was NO. Represents the background pixels correctly extracted (not necessary in our case).
- **False Positives (FP)**- The cases in which the model predicted YES, and the output was NO. Represents the road pixels that were incorrectly extracted.

- **False Negatives (FN)**- The cases in which the model predicted NO, and the output was YES. Represents the road length that was not extracted.

		Prediction	
		Positive	Negative
Ground Truth	Positive	TP	FN
	Negative	FP	TN

Table 5.1: Confusion matrix table.

Three benchmark metrics proposed by Wiedemann [70, 71] were used to assess the quantitative performance in the road detection, and the centerline extraction process. The first metric is Completeness (COM) Equation 5.1, the second one is Correctness (COR) Equation 5.2, and lastly Quality (Q) Equation 5.3. Another additional metric known as F1-Score (F1) Equation 5.4 is defined, used in [72].

$$COM = \frac{\text{Length of matched reference}}{\text{Length of reference}} \approx \frac{TP}{TP+FN} \in [0;1] \tag{5.1}$$

$$COR = \frac{\text{Length of matched extraction}}{\text{Length of extraction}} \approx \frac{TP}{TP+FP} \in [0;1] \tag{5.2}$$

$$Q = \frac{\text{Length of matched extraction}}{\text{Length of extracted data} + \text{Length of unmatched reference}} \approx \frac{TP}{TP+FN+FP} \in [0;1] \tag{5.3}$$

$$F1 = \frac{2 \times COM \times COR}{COM + COR} \approx \frac{2TP}{2TP+FN+FP} \in [0;1] \tag{5.4}$$

The COM metric represents the proportion of area matched in the reference area characterizing the integrity of the road extraction. The COR metric represents the proportion of the correct degree of road area extracted, the Q metric represents how good the final result is, by considering the correctness and completeness. The additional F1 metric is widely used to measure the harmonic average between the completeness and correction, it is used as a way to measure the accuracy of our binary classification system. The goal is to tune all of these four metrics and achieve optimal values as close as possible to 1.

For road centerline extraction, the approach needs to be slightly changed. As the human operator is manually extracting the roads, discrepancies between the manually labeled centerline and the real centerline. This means that it is not suitable to make the comparison between the centerline extracted, and the ground truth centerline with a

single-pixel width, as one small shift of pixels can influence the whole performance of the model. As a way to solve this issue, a method is used to compare distinct road centerline algorithms, known as "buffer method". This method compares the matching extracted data with the reference data where every fraction of the network is within a given buffer width ρ. Figure 5.1

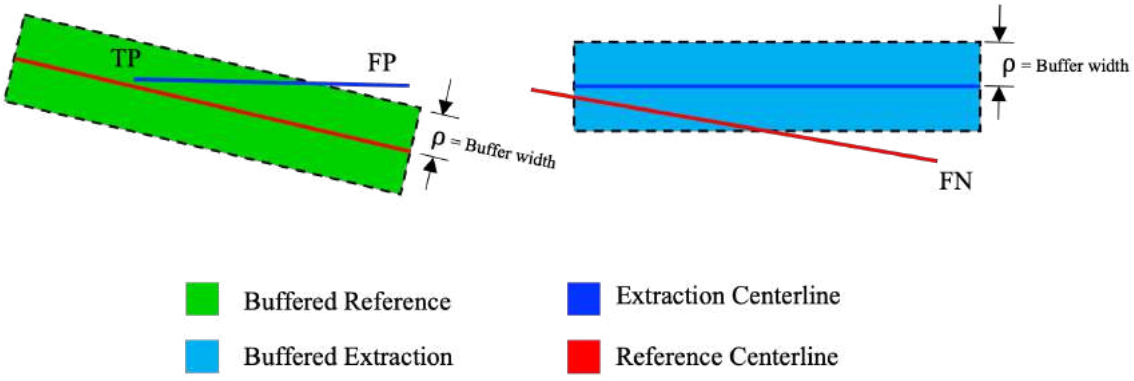


Figure 5.1: Road centerline extraction using the buffer method [48]

To find the TP, and FP pixels, a dilation of the reference centerline with a certain buffer width ρ is made. The next step consists of making the intersection between the dilated reference data and the extracted centerline data, resulting in the matched extracted data (TP) and unmatched extracted data (FP). To find the FN pixels, a dilation of the extracted centerline with a buffer width ρ is performed, followed by an intersection with the reference data, resulting in the matched reference data and unmatched reference data (FN). After making this procedure, it is possible to calculate all of the metrics previously defined. [73]

5.2 Comparison between methods

To verify the performance, the proposed architecture is compared with other state-of-the-art approaches in either road detection and road centerline extraction. Since our dataset does not belong to a benchmark dataset and is completely unique, the output results will be biased to our implementation, so it was decided to make a comparison with the following methodologies.

For road detection a comparison between 3 state-of-the-art methods is made, using deep neural networks. The first one is the DeepLabV3+, which makes part of our road detection algorithm, to create the rural road segmentation. [25] The second one is an architecture called Unet [74], this model achieved a new benchmark performance becoming a new state-of-the-art method in biomedical image segmentation. This model also won the Grand Challenge for Computer-Automated Detection of Caries in Bitewing Radiography at ISBI 2015, and won the Cell Tracking Challenge at ISBI 2015. The third method that is used

as a comparison is the FPN [27], by Facebook AI Research (FAIR). This method achieves state-of-the-art single-model results on the COCO detection benchmark, outperforming all existing single-model entries, including COCO 2016 challenge winners. This method has become a very accurate and practical solution to multi-scale object detection.

For road extraction a comparison between two thinning morphological methods was made. The first one is the Zhang-Suen thinning algorithm [47], and the second one is the Guo-Hall thinning algorithm [48].

All of the road detection experiments were declared with equal initialization and optimization parameters in the training procedure. For all of the tests the same learning rates were used, as well as, number of epochs set to 3, network backbone (resnet50), pre-trained weights (imagenet), with a batch size set to 4, number of classes equal to 2, and the same activation function (sigmoid). The models parameters initialization can be consulted in Appendix A. The models were trained using Google Colab Pro GPUs inside a macOS operating system with a 2.6GHz 6-core Intel Core i7 processor.

5.3 Road Detection Evaluation

To evaluate the proposed architecture on road detection a qualitative and quantitative comparisons are performed between the DeepLabV3+, Unet, and the FPN model. These results are respectively presented on Table 5.2 and Table 5.3. In Table 5.2 5 different images randomly chosen from the test dataset distributed by rows are shown.

In the first column (a) the aerial images are presented. In column (b) there are ground truth masks manually labeled, in column (c) the images predicted by using the DeepLabV3+ achitecture are shown. In column (d) there are the prediction results by using the Unet architecture, and finally in column (e) the prediction results of the FPN architecture are shown.

By using DeepLabV3+, we can detect roads on complex rural environments and the spatial pyramid pooling, combined, with the encoder-decoder, allows the model to extract sharp features around the road edges. This model is able to detect roads even with shadows, trees, and other objects in the middle of the road being a robust option on rural road detection. The DeepLabV3+ model had some difficulties connecting different types of roads (i.e asphalt, dirt, cement, grass) leaving incomplete connections on the rural roads.

The Unet model had more complete connections compared to DeepLabV3+, although it had a lower correction score. As can be observed, this model introduced some white objects that shouldn't be on the final predicted image. In Image 4 and Image 5, the Unet model had some difficulties extracting complete roads on environments with road occlusions created by trees and shadows, creating incomplete road branches.

Lastly, the FPN model creates complete roads with sharp edges overcoming complex backgrounds even with multiple trees covering the aerial sight. The FPN model also introduces some white objects from incorrectly predicted labeling that should be later removed.

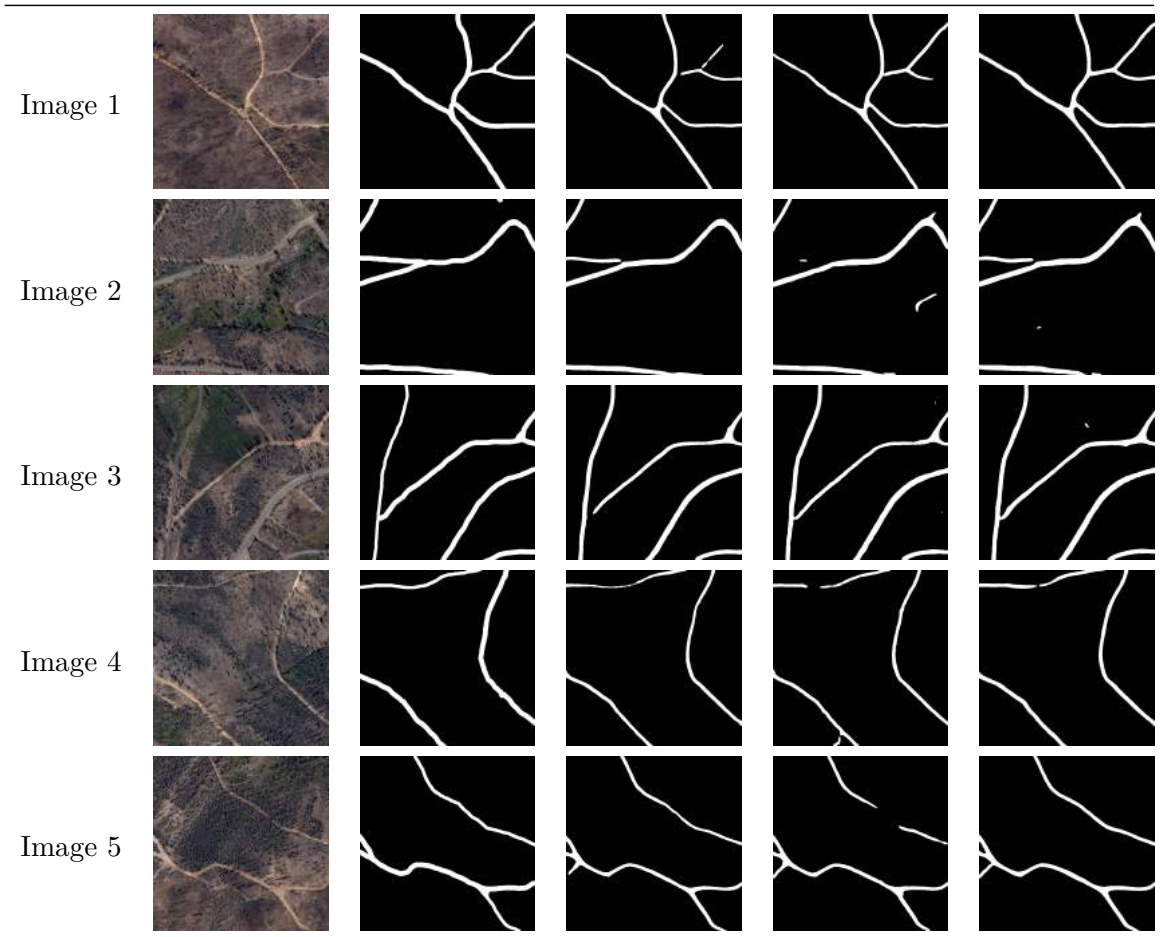


Table 5.2: Comparison of rural road detection networks for the Mação district using DeepLabV3+, Unet and FPN models.

Comparing the 3 models on the road detection process, Table 5.3, the DeepLabV3+ model achieves the highest average correctness score of the 3 models. The Unet model, compared with DeepLabV3+ and FPN, does not out perform any of the highest scores. The FPN model achieves the highest scores on average completeness, quality and F1 score. From this experience, DeepLabV3+ performes better if the priority is a high degree of correctness, and the FPN performes better if our goal is to have a higher degree of completeness, quality and F1 score. In Figure 5.2, Figure 5.3, and Figure 5.4 a graphical representation of the metrics for the average results of the test dataset can be seen, Image 1, and Image 2 respectively.

	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.6368	**0.9573**	0.6192	0.7648	0.8380	**0.9353**	**0.7921**	**0.8840**	0.6267	**0.8567**	0.5640	0.7239
Unet	0.6767	0.9464	0.6517	0.7891	0.7519	0.8733	0.6780	0.8081	0.6404	0.8247	0.5564	0.7210
FPN	**0.7739**	0.9234	**0.7272**	**0.8420**	**0.8518**	0.9021	0.7798	0.8763	**0.7066**	0.8067	**0.5975**	**0.7533**

Table 5.3: Rural road detection quantitative results, where the best values are shown in bold. It should be noted that the last column represents the average performance of the images presented in the whole dataset.

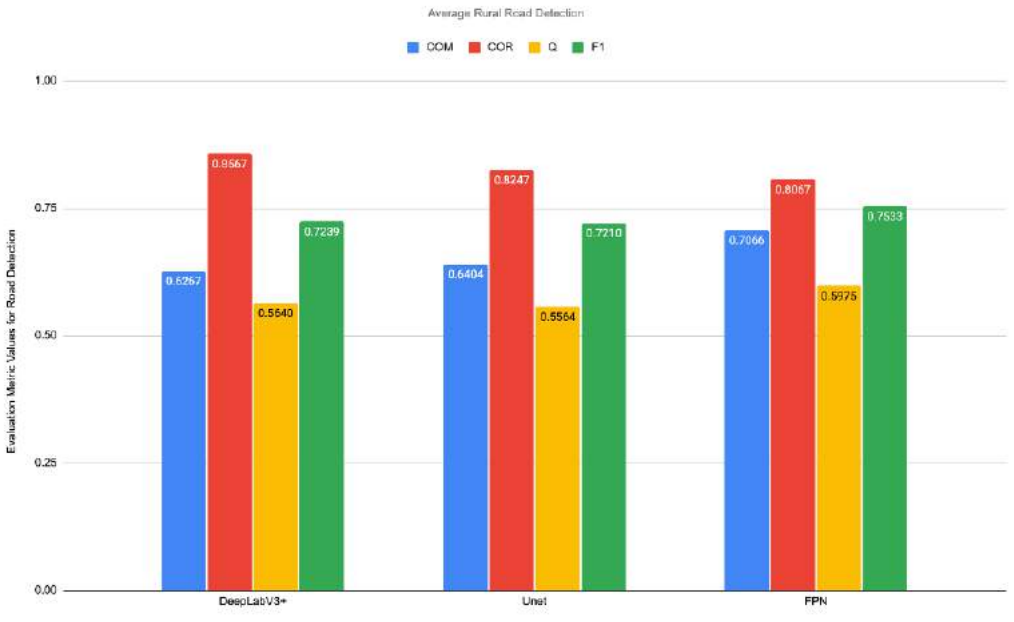


Figure 5.2: Rural road detection average metrics evaluation for the test dataset. On this experiment the three models DeepLabV3+, Unet, and FPN Network were measured with different metrics. The blue color represents the Completeness, the red color represents the Correctness, the yellow color represents the Quality, and finally the green color represents the F1 Score metric. All of these metrics have values between 0 and 1

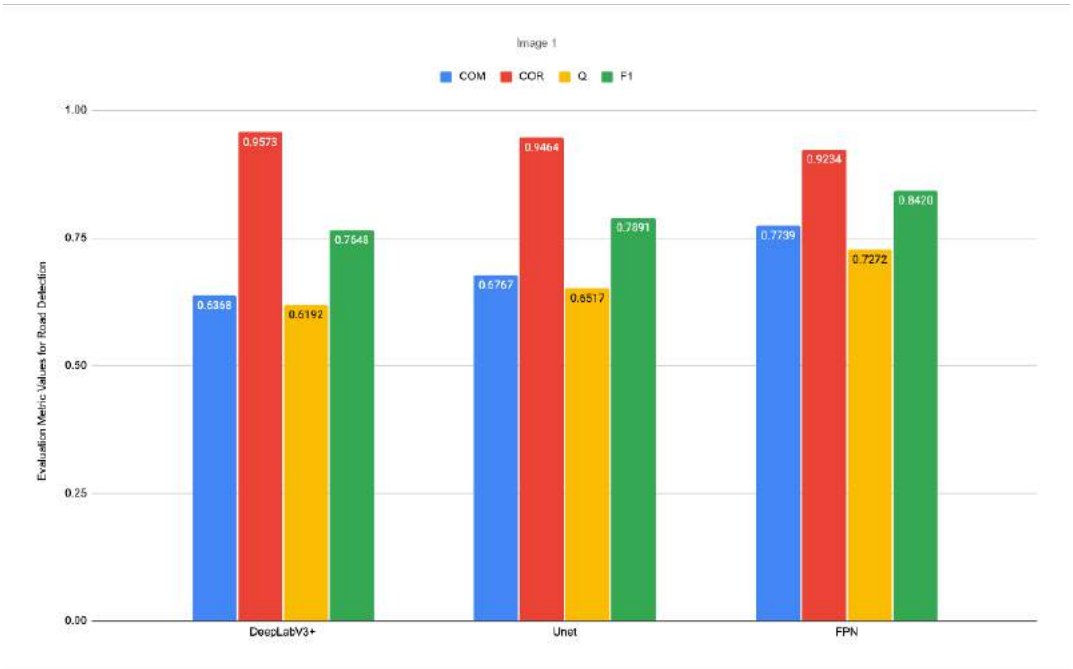


Figure 5.3: Rural road detection metrics for Image 1

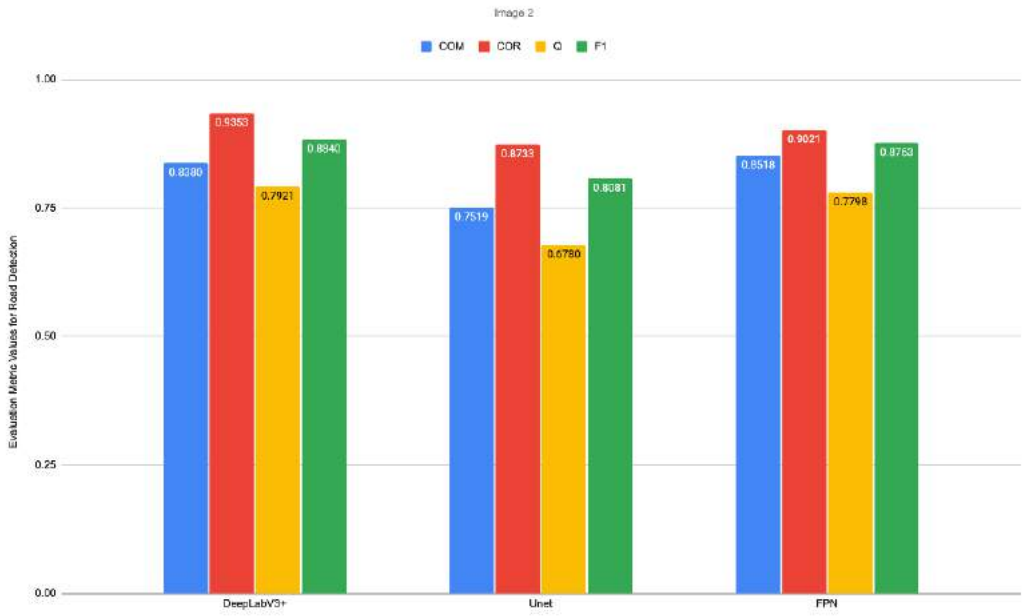


Figure 5.4: Rural road detection metrics for Image 2

(a) Aerial image. (b) Rural road detection predicted image.

Figure 5.5: On this first example (a) we can see that DeepLabV3+ is robust model against strong shadows and small bushes in the middle of the road. In (b) the model had no problems overcoming and predicting this type of complex landscape.

(a) Aerial image. (b) Rural road detection predicted image.

Figure 5.6: In this case the aerial image (a) had thin rural roads with a high density of shadows and small bushes in the middle on the road. In (b) the DeepLabV3+ also had no problems making relatively accurate predictions.

(a) Aerial image. (b) Rural road detection predicted image.

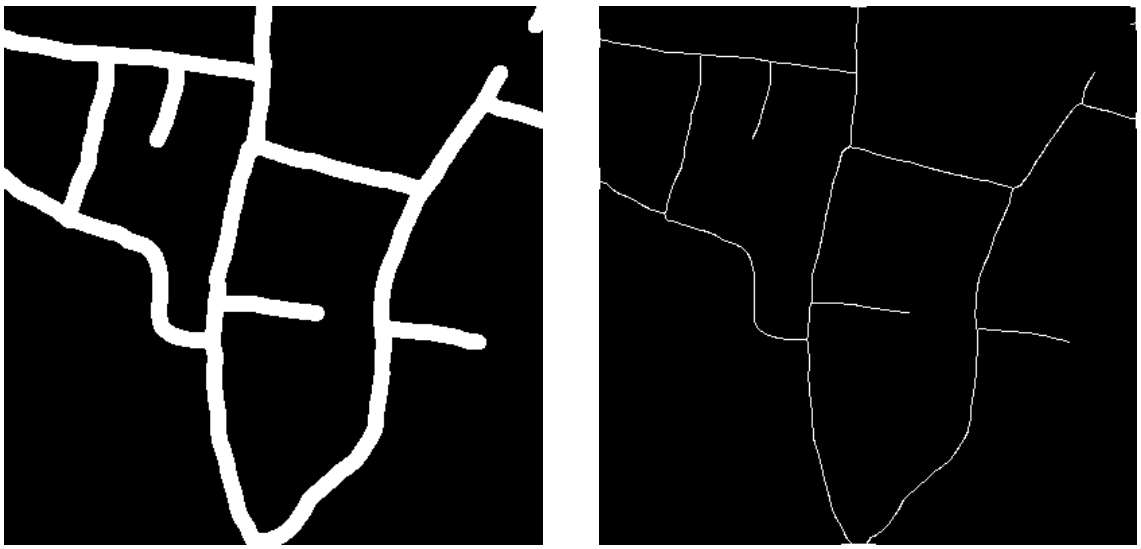
Figure 5.7: The aerial image in (a) had two total occlusions from the sky view but once again the model showed a very high robustness even with the road completely obstructed with trees and vegetation. The only part the road misses a connection is near top of the green rectangle where the road change types from asphalt road, to dirt road. This problem will be further improved.

(a) Aerial image. (b) Rural road detection predicted image.

Figure 5.8: In this last example we can see a case where the model couldn't detect the rural roads properly as it was an environment with a high complexity with a lot of road textures, shadows, with patterns not particularly well defined. These cases can be difficult even for a human operator to say with certainty if it's an actual rural road path or not.

5.4 Road Connections Optimization

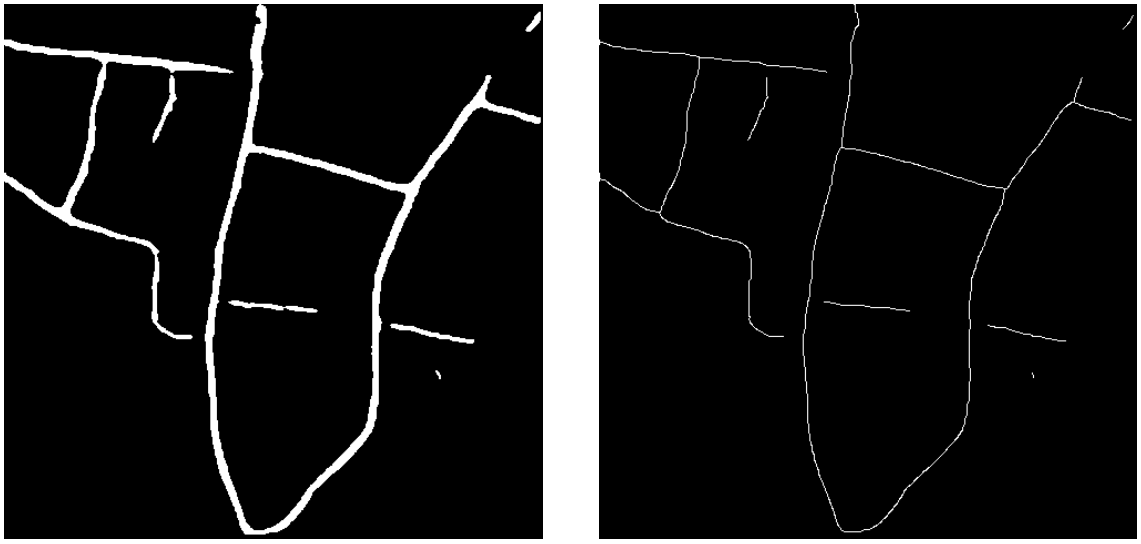
After detecting the roads with the proposed model, we now need to create the road centerline skeleton. It was noticed that the model had some difficulties intersecting roads with different types of materials, (ie. asphalt road, and dirt road). In Figure 5.9 we can see an example of the road detection reference and the respective skeleton, compared to the predicted road with their respective skeleton Figure 5.10 (a). On this comparison it's clear that this is a problem that must be endorsed in order to have a clear road skeleton like Figure 5.9. (b).



(a) Ground truth rural road mask. (b) Ground truth extracted centerline.

Figure 5.9: Ground truth rural road mask, and the respective extracted centerline with Zhang-Suen thinning algorithm.

The optimization approach proposes to connect different types of roads and remove small white objects wrongly predicted, consisting of a four stage process that can be seen in Figure 5.11. In the first stage, we make a prediction and detect the rural road, where we can observe the red circles spots in which the road is not connected (a). The second stage consists of making 5 iterations on a dilation with a 5×5 kernel window. What this dilation does is, it connects roads of different types that are on a close neighbourhood resulting in (b). The following step consists of making a skeleton of the previous dilated image. As it can be seen in (c), the connections have been established creating clean intersections between different types of roads that weren't previously set. Although it can still be seen a small object that must be removed. To address that issue the stage four serves as a small objects remover, it uses the exhaustive connected components methods to label all objects in the mask and if the objects are smaller than a minimum size of 140 pixels, the object is removed. The value of the minimum size was obtained by testing multiple values to an approximate size that removes most of unnecessary pixels without damaging or removing any important road fragments. This allows us to obtain a clean predicted

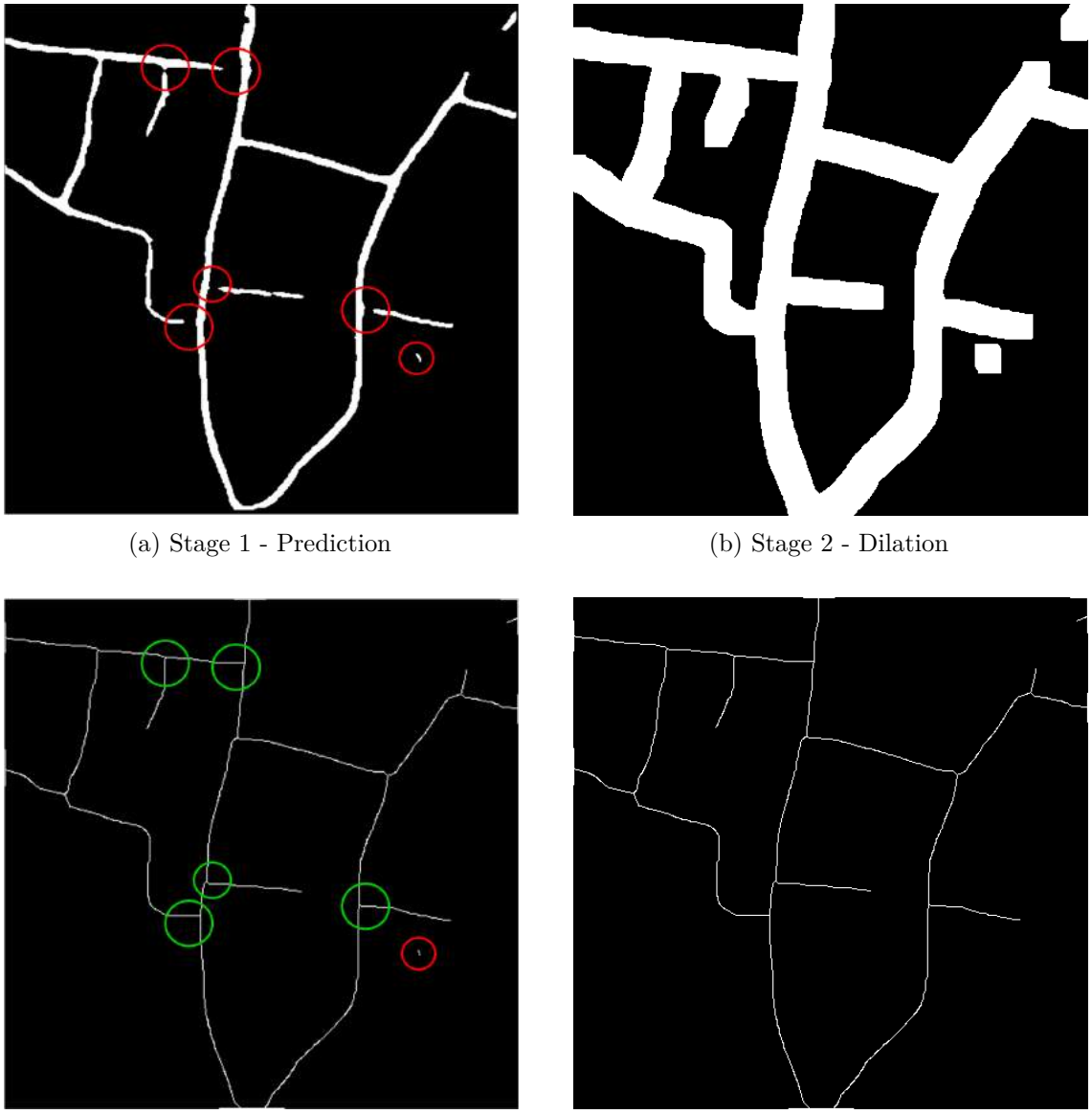


(a) Predicted rural road mask.

(b) Predicted extracted centerline.

Figure 5.10: Predicted rural road, and respective extracted centerline with Zhang-Suen thinning algorithm. Some of the connection problems between different types of roads can be seen. Incorrect predictions that generate small white objects are also removed from the predicted image with the optimization process.

rural road centerline observed in (d), which gives a better visual performance, closer to the reference ground truth centerline presented in Figure 5.9 (b). This algorithm increases the completeness of DeepLabV3+ but, on the other hand, decreases slightly the correctness and f1 score. This is due to multiple dilations that can add extra pixels completing missing connections but since the algorithm does not know exactly the position of those pixels, the correctness slightly decreases. The optimization process will be utilized before two kinds of thinning algorithms, it will be first used the Zhang-Suen thinning algorithm and later the Guo-Hall thinning algorithm.



(a) Stage 1 - Prediction

(b) Stage 2 - Dilation

(c) Stage 3 - Thinned image

(d) Stage 4 - Thinned image optimized

Figure 5.11: The 4 stages of the optimization process.

5.5 Road Extraction with Zhang-Suen Algorithm

In Table 5.4 we are presented with the visual performance of the rural road extraction using the Zhang-Suen Thinning algorithm for different models. The results of the road extraction method are shown in 255, and 0 for the background. It can be seen from Table 5.4 that DeepLabV3+ produces smooth roads relatively correct, although it missed some of the thinner roads on Image 1 and the road connection on Image 3, it shows very satisfactory results. The Unet completed the thinner lines and connections that DeepLabV3+ couldn't, on Image 1, but introduced some FPs (white lines that are on the exterior of the reference map) on Image 2. It is noticeable that Unet completed the connections on Image 3, but performed poorly on image 4 and 5 leaving FNs (pixels

wrongly identified as background) and incomplete road branches on places with high forest density where it covers the road. The FPN shows complete rural roads with only some extra FPs. These FPs are difficult to identify even with a human operator extracting them by hand because sometimes it is not exactly perceptible if there is actually a road path or if it's just the complexity of the background. A quantitative comparison in the tables Table 5.5, Table 5.6, Table 5.7 and Table 5.8 was made. It was chosen values of $\rho = 2$, $\rho = 4$, $\rho = 6$, and $\rho = 8$ pixels, (corresponding to 0.5m, 1m, 1.5m, and 2m on the field). Since this thesis focuses on extracting rural roads instead of the typical urban roads, the ρ values were slightly increased to adjust to the bigger error margin and terrain complexity that exists on these environments. On these conditions it can be observed that on the average dataset test DeepLabV3+ outperformed Unet, and the FPN model on Correctness, Quality and F1 Score for the three different ρ values. The Unet did not outperformed any of the models on the proposed metrics. The FPN model achieved the highest completeness score of the three models. From this experience we can clearly observe that DeepLabV3+ had the most correct roads and FPN had the most complete roads.

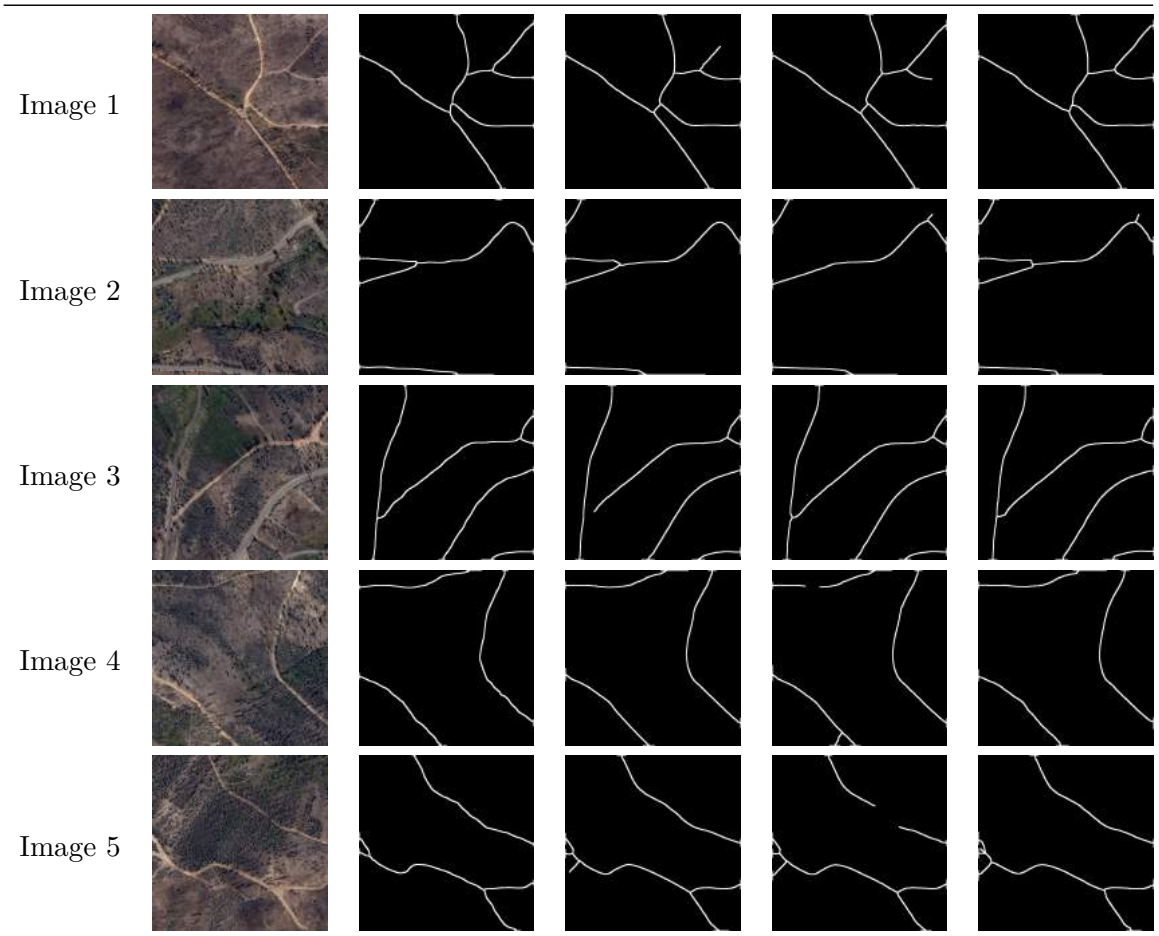


Table 5.4: Rural road extraction network for the Mação district using the Zhang-Suen thinning algorithm. The road centerline was dilated 3 pixels in the images presented in order to be easier to see the roads without effecting the final result.

$\rho = 2 = 0.5$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	**0.5539**	**0.5933**	**0.4015**	**0.5729**	0.6583	0.6403	0.4806	0.6492	0.4758	**0.4850**	**0.3184**	**0.4804**
Unet	0.5377	0.5646	0.3801	0.5508	0.5918	0.6199	0.4342	0.6055	0.4681	0.4634	0.3052	0.4657
FPN	0.5067	0.5071	0.3395	0.5069	**0.6761**	**0.6597**	**0.5013**	**0.6678**	**0.4894**	0.4646	0.3164	0.4767

Table 5.5: Rural road extraction quantitative results for a $\rho = 2$.

$\rho = 4 = 1$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.7599	0.8045	0.6415	0.7816	**0.8376**	0.8188	0.7066	0.8281	0.6937	**0.7076**	**0.5441**	**0.7006**
Unet	0.8026	**0.8274**	0.6875	0.8148	0.7263	0.7632	0.5927	0.7443	0.6840	0.6758	0.5178	0.6799
FPN	**0.8268**	0.8153	**0.6964**	**0.8210**	0.8366	**0.8240**	**0.7098**	**0.8303**	**0.7109**	0.6756	0.5359	0.6928

Table 5.6: Rural road extraction quantitative results for a $\rho = 4$.

$\rho = 6 = 1.5$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.8480	0.8983	0.7738	0.8725	**0.9227**	**0.9063**	**0.8423**	**0.9144**	0.8063	**0.8234**	**0.6940**	**0.8148**
Unet	0.9181	**0.9427**	0.8696	0.9302	0.7898	0.8357	0.6834	0.8121	0.8013	0.7908	0.6621	0.7960
FPN	**0.9592**	0.9425	**0.9062**	**0.9508**	0.9009	0.8904	0.8110	0.8957	**0.8187**	0.7796	0.6695	0.7987

Table 5.7: Rural road extraction quantitative results for a $\rho = 6$.

$\rho = 8 = 2$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.9211	0.9749	0.8998	0.9473	**0.9368**	**0.9240**	**0.8698**	**0.9304**	0.8489	**0.8683**	**0.7588**	**0.8585**
Unet	0.9605	**0.9826**	0.9444	0.9714	0.8031	0.8505	0.7037	0.8261	0.8524	0.8434	0.7363	0.8479
FPN	**0.9938**	0.9805	**0.9746**	**0.9871**	0.9203	0.9137	0.8468	0.9170	**0.8709**	0.8308	0.7430	0.8504

Table 5.8: Rural road extraction quantitative results for a $\rho = 8$.

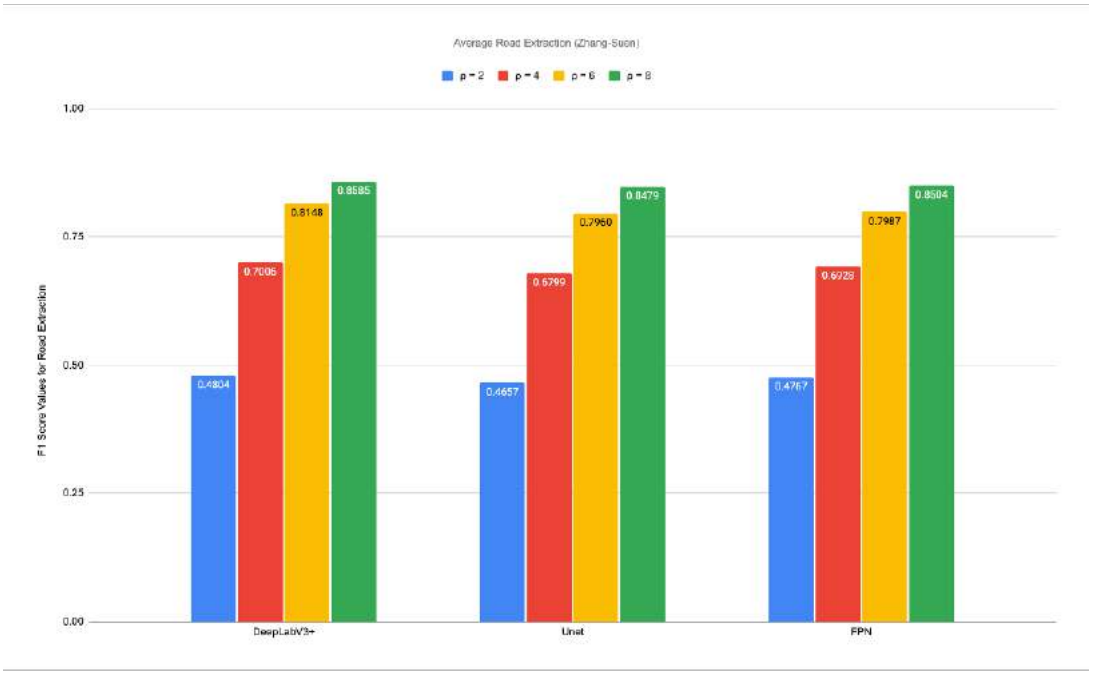


Figure 5.12: Rural road extraction average metrics for the whole test dataset. On this experiment was measured the Zhang-Suen metrics for the three models DeepLabV3+, Unet, and FPN Network for different ρ values. The blue color represents $\rho = 2$ (0.5m in the field), the red color represents $\rho = 4$ (1m in the field), the yellow color represents $\rho = 6$ (1.5m in the field), and finally the green color represents $\rho = 8$ (2m in the field). On the vertical axis it is represented the F1 Score metric values.

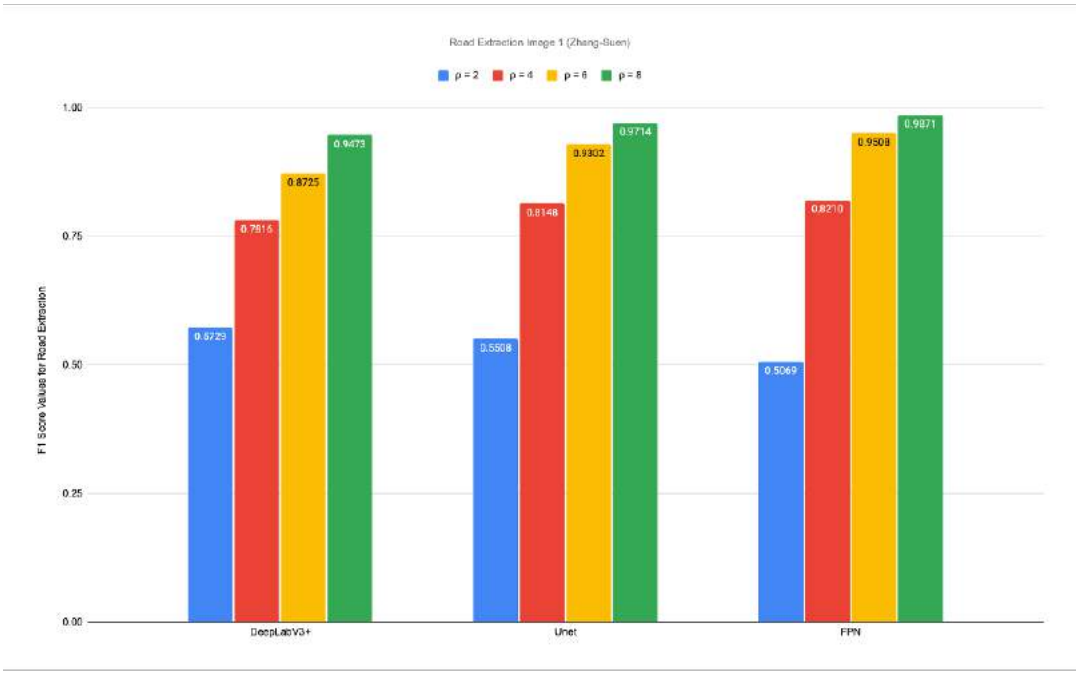


Figure 5.13: Rural road extraction metrics for Image 1.

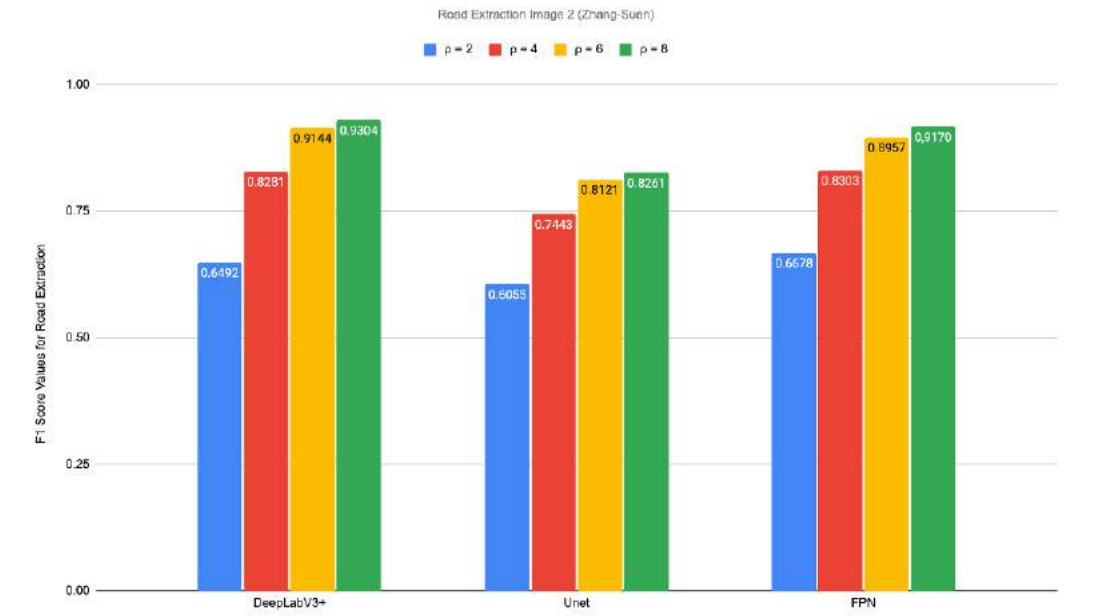


Figure 5.14: Rural road extraction metrics for Image 2.

5.6 Road Extraction with Guo-Hall Algorithm

In this next experiment the Guo-Hall thinning algorithm is tested to extract the rural road centerlines. In Figure 5.15 can be observed a visual comparison between the Zhang-Suen and Guo-Hall thinning algorithms. The Guo-Hall compared with the Zhang-Suen algorithm produces more curved lines. This effect is specially visible in road interceptions. In this experience the goal is to see if any of these algorithms produces slightly better results than the other, or if the final results are similar.

(a) Road extraction using Zhang-Suen. (b) Road extraction using Guo-Hall.

Figure 5.15: Visual comparison between road extraction algorithms.

On Table 5.9 the results of the road extraction using the Guo-Hall thinning algorithm are shown. On this test the results were very similar to the Zhang-Suen thinning algorithm not showing any big discrepancies between different ρ values in Table 5.10, Table 5.11, Table 5.12, and Table 5.13. Both algorithms achieved very similar results having a difference of them of less than 1% between them, achieving both satisfactory results on rural road extraction tasks. Once again time DeepLabV3+ achieved superior results on Correctness, Quality, and F1 Score. Unet did not outperform any of the three models, and FPN achieved again the top Completeness score.

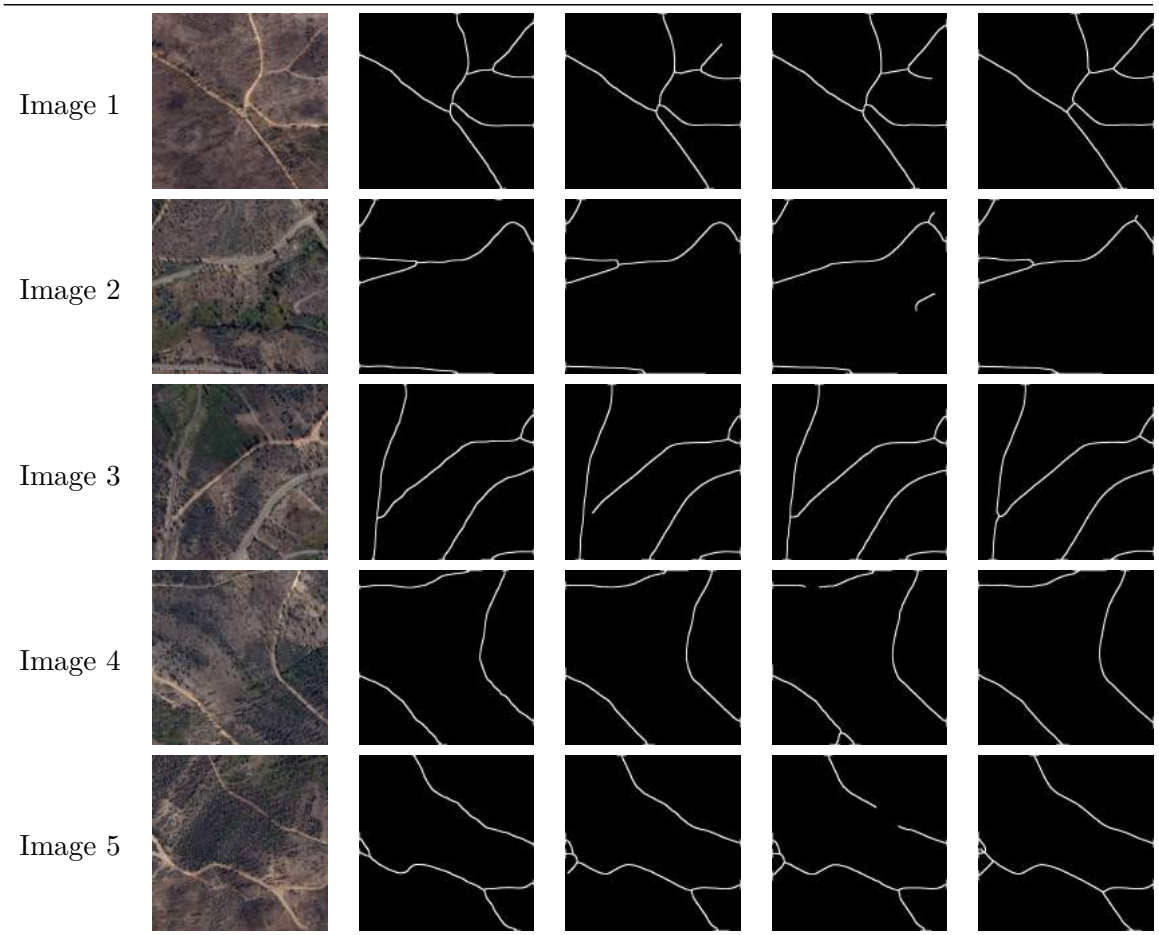


Table 5.9: Rural road extraction network for the Mação district using the Guo Hall algorithm to perform the skeletonization.

$\rho = 2 = 0.5$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	**0.5336**	**0.5587**	**0.3754**	**0.5459**	0.6205	0.5969	0.4373	0.6085	0.4760	**0.4801**	**0.3167**	**0.4780**
Unet	0.5118	0.5274	0.3509	0.5195	0.5546	**0.6307**	0.4187	0.5902	0.4661	0.4621	0.3034	0.4641
FPN	0.4882	0.4777	0.3183	0.4829	**0.6280**	0.6167	**0.4517**	**0.6223**	**0.4925**	0.4639	0.3163	0.4778

Table 5.10: Rural road extraction quantitative results for a $\rho = 2$.

$\rho = 4 = 1$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.7506	0.7857	0.6231	0.7678	**0.8253**	0.8004	0.6845	0.8127	0.6947	**0.7047**	**0.5428**	**0.6997**
Unet	0.7799	**0.8017**	0.6538	0.7907	0.7129	0.8098	0.6106	0.7583	0.6831	0.6785	0.5171	0.6808
FPN	**0.8170**	0.7984	**0.6772**	**0.8076**	0.8245	**0.8130**	**0.6930**	**0.8187**	**0.7135**	0.6748	0.5351	0.6936

Table 5.11: Rural road extraction quantitative results for a $\rho = 4$.

$\rho = 6 = 1.5$m	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.8475	0.8905	0.7675	0.8685	**0.9179**	**0.8930**	**0.8269**	**0.9053**	0.8037	**0.8176**	**0.6876**	**0.8106**
Unet	0.9220	**0.9498**	0.8792	0.9357	0.7763	0.8830	0.7039	0.8262	0.8020	0.7980	0.6662	0.8000
FPN	**0.9691**	0.9476	**0.9198**	**0.9582**	0.9002	0.8833	0.8045	0.8917	**0.8201**	0.7796	0.6697	0.7993

Table 5.12: Rural road extraction quantitative results for a $\rho = 6$.

$\rho = 8 = 2m$	Image 1				Image 2				Avg. (Test set)			
Architecture	COM	COR	Q	F1	COM	COR	Q	F1	COM	COR	Q	F1
DeepLabV3+	0.9379	0.9808	0.921	0.9589	**0.9409**	**0.9191**	**0.869**	**0.9299**	0.8492	**0.8659**	**0.7568**	**0.8575**
Unet	0.9586	**0.9882**	0.9477	0.9732	0.7921	0.9062	0.7321	0.8453	0.8520	0.8518	0.7417	0.8519
FPN	**0.9959**	0.9813	**0.9773**	**0.9885**	0.9228	0.9113	0.8467	0,9170	**0.8719**	0.8309	0.7432	0.8509

Table 5.13: Rural road extraction quantitative results for a $\rho = 8$.

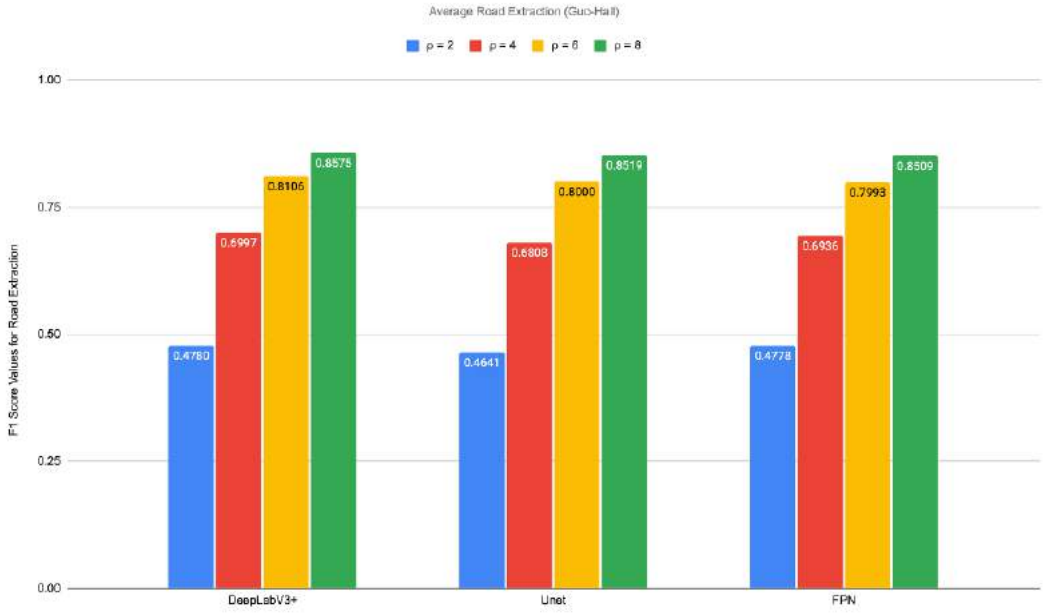


Figure 5.16: Rural road extraction average metrics for the whole test dataset.

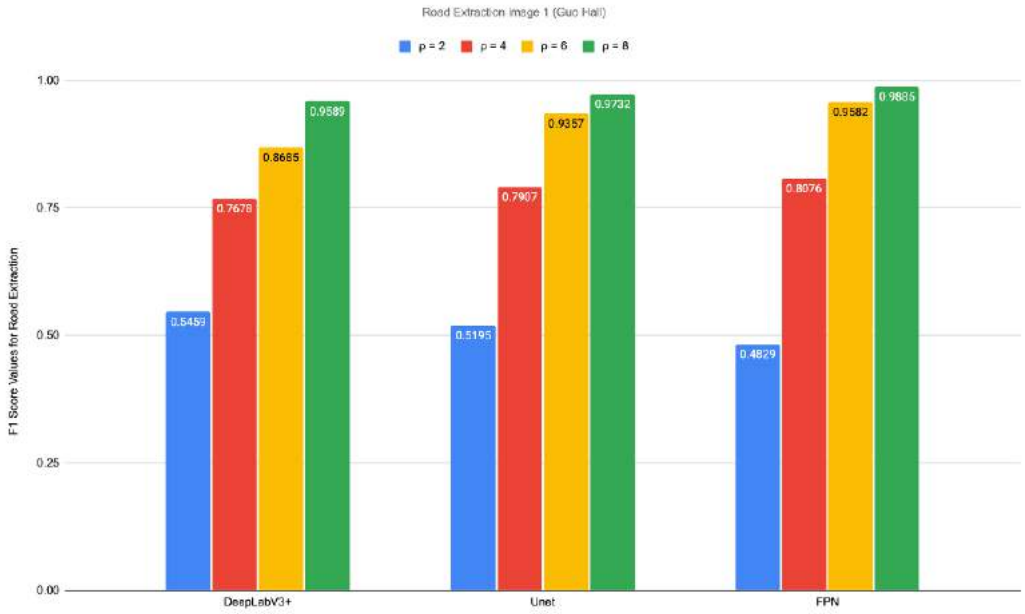


Figure 5.17: Rural road extraction metrics for Image 1.

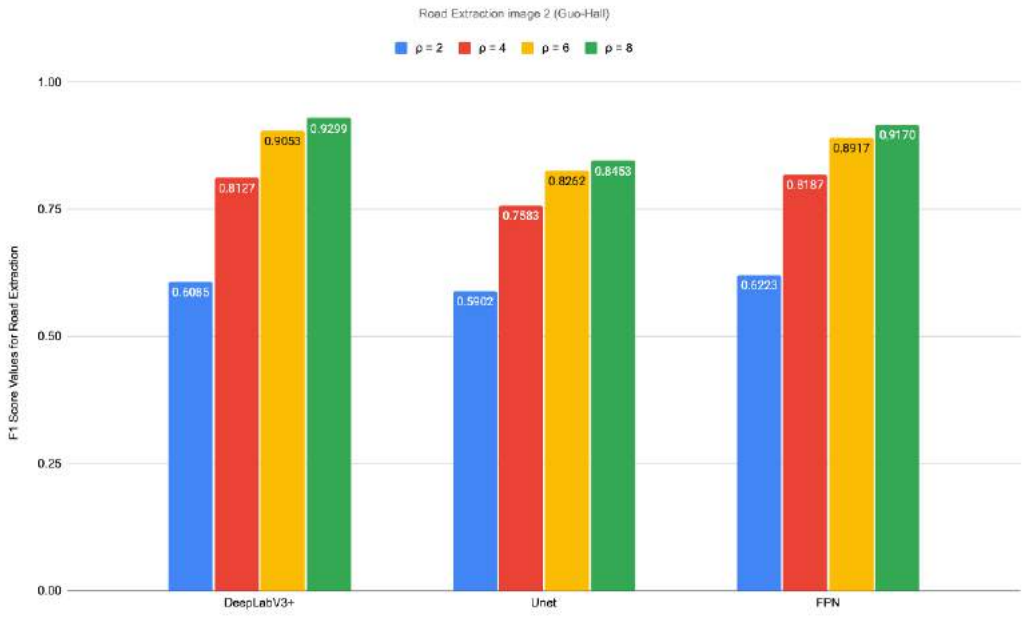


Figure 5.18: Rural road extraction metrics for Image 2.

5.7 Global Results Analysis

In the previous chapter the dataset was created from scratch This means that we are not using a benchmark dataset and our results will be biased to the Mação municipality compared with other algorithms. According to the authors' knowledge, it was the first time that such recent deep learning methods were used to extract rural roads centerlines. Bearing this in mind, the results can't be directly compared to other state-of-the-art methods but we can see where the proposed method stands.

In literature, scientists usually use bench mark datasets of urban roads, which are very distinct from rural roads. Urban roads usually have more well-defined roads, patterns, and most of the times the roads are made of asphalt being easier to train the model to those specific features. Considering this situation, rural roads have a lot of environment unpredictability caused by natural conditions and background landscape, making it harder to detect clean and precise rural road centerlines. Some of the natural challenges that rural roads have compared to urban ones are: the low visibility in places with high density of trees, the presence of big shadows obstructing the road, and having roads with several types of material making the deep learning training process more difficult.

Using a buffer width of $\rho = 8$ (corresponding to a width of 2 meters on the field), and using the Zhang-Suen thinning algorithm to extract the rural roads we have achieved the following results: Completeness= 0.8489, Correctness= 0.8683, Quality= 0.7588, and F1score= 0.8585. Comparing the Quality metric with [19] our proposed method it could not beat their proposed method score, but it achieved superior results on road centerline extraction in relation to their Huang-C and Miao-C method for $\rho = 1$ and $\rho = 2$, it should also be noticed that in this article each ρ pixel incrementation corresponds to 1.2m per pixel, which means that the buffer width for $\rho = 2$ translates to a size of 2.4m in reality, which is a bigger buffer width when compared to our proposed method. For $\rho = 1$, Huang-C Quality = 0.6471, Miao-C Quality = 0.6218, and for $\rho = 2$, Huang-C Quality = 0.7027, Miao-C Quality = 0.6735.

Our proposed method achieved a higher Quality score when compared with [17] proposed method for Guangzhou dataset, and again achieving higher scores when compared with Huang, and Miao's method. Their proposed method quality score = 0.7522, their Huang's method quality = 0.6890, their Miao's method quality = 0.7169.

The experiments weren't tested with the same set of variables and data, but overall, the achieved results are very promising as we are facing more challenging problems keeping relevant Quality scores.

CHAPTER 6

Conclusions and Future Work

6.1 Conclusions and Discussion

During the last decades, Portugal has suffered from extreme forest fires that can be substantial mitigated with the use of recent technology. With the goal of helping preserve forests, reduce the danger to society, and help firefighters have a higher efficiency and increased situational awareness in the field, a system was proposed that can automatically detect and extract rural road centerlines from aerial images. The goal was to create a process that becomes faster and more precise than the previous handmade tedious work done by mapping operators.

The proposed method focuses on using and comparing recent deep learning methodologies like DeepLabV3+, Unet, and FPN model to detect rural roads. Later morphological algorithms are used to qualitatively optimize road connections between different types of roads, and lastly, thinning algorithms like Zhang-Suen and Guo-Hall are used to extract the rural road centerlines.

On the presented architecture there are 2 key points that the end user should consider when choosing the deep learning model to perform the road detection. The two key points are the importance degree that will be given to the metric Completeness and Correctness.

For the rural road centerline extraction, the top performance was achieved by using the proposed method with DeepLabV3+ as the road detector, combined with a buffer width of $\rho = 8$ (corresponding to a width of 2 meters on the field), and using the Zhang-Suen thinning algorithm to extract the rural roads. This experience brought the best metrics scores achieving a $Completeness = 0.8489$, $Correctness = 0.8683$, $Quality = 0.7588$, and $F1score = 0.8585$.

Although the proposed method was the best performing one on the road centerline extraction task, it was noticed that the FPN model achieve the highest F1 score for the

road detection and the highest completeness score on all of the road centerline extraction tests. If the priority for the end-user is to only detect rural roads or extract rural road centerlines with a higher degree of completeness, it should also be considered the use of the FPN model.

It can be concluded that the proposed method provides a very practical and viable solution for accurate road detection as well as centerline extraction from aerial images. The proposed method surpasses strong shadows by trees, small bushes and vegetation in the middle of the road, Figure 5.5. With the optimization, it is also possible to connect different types of roads, Figure 5.11, as well as overcome total road occlusions with a considerable size, Figure 5.7.

6.2 Future Work

The proposed method generated very good results detecting rural roads as well as extracting their centerlines. To give continuation and extend the developed work, future improvements should focus on the following topics:

- Validation of the results on the terrain. This task will confirm that the methods developed on this thesis are accurate and verified in the field for the creation of future rural road networks.
- Adding the rural roads centerline data into a geographical information system in a shape file format, followed by the incorporation of the data into real-time mapping applications.
- Increasing the size and quality of the training dataset, becoming more precise focusing on what pixels exactly belong to the background, and what pixels belong to the rural roads. The more samples the datasets have, the more scenarios and complex backgrounds the models can learn from and become better at making predictions.
- Testing other deep learning models to perform the rural road segmentation like the Pyramid Scene Parsing Network (PSPNET)[75], Pyramid Attention Network (PAN)[76], and Multi-Scale Attention Network (MAnet)[77].
- Creating an algorithm specifically to detect incomplete road intersections and dead end roads.
- Finally, it would be relevant to start utilizing RGB images with the Infrared component, so the deep learning models can also learn about the reflectance of green zones with vegetation and trees.

Bibliography

[1] R. P. A. e Ação Climática. *National Forestry Accounting Plan Portugal 2021-2025*. URL: https://apambiente.pt (cit. on p. 1).

[2] P.-R. Fires and B. Area-Mainland. In: (2021). URL: https://www.pordata.pt/en/Portugal/Rural+fires+and+burnt+area+-+Mainland-1192 (cit. on p. 1).

[3] M. Lourenço et al. "An Integrated Decision Support System for Improving Wildfire Suppression Management". In: *ISPRS International Journal of Geo-Information* 10.8 (2021). ISSN: 2220-9964. DOI: 10.3390/ijgi10080497. URL: https://www.mdpi.com/2220-9964/10/8/497 (cit. on p. 1).

[4] S. Marques et al. "Characterization of wildfires in Portugal". In: *European Journal of Forest Research* 130.5 (Jan. 2011), pp. 775–784. DOI: 10.1007/s10342-010-0470-4. URL: https://doi.org/10.1007/s10342-010-0470-4 (cit. on p. 1).

[5] G. Cheng et al. "Accurate urban road centerline extraction from VHR imagery via multiscale segmentation and tensor voting". In: *Neurocomputing* 205 (2016), pp. 407–420. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2016.04.026. URL: http://www.sciencedirect.com/science/article/pii/S0925231216302855 (cit. on pp. 2, 5).

[6] F. Xiao, L. Tong, and S. Luo. "A Method for Road Network Extraction from High-Resolution SAR Imagery Using Direction Grouping and Curve Fitting". In: *Remote Sensing* 11.23 (2019). ISSN: 2072-4292. DOI: 10.3390/rs11232733. URL: https://www.mdpi.com/2072-4292/11/23/2733 (cit. on p. 5).

[7] W. Shi, Z. Miao, and J. Debayle. "An Integrated Method for Urban Main-Road Centerline Extraction From Optical Remotely Sensed Imagery". In: *IEEE Transactions on Geoscience and Remote Sensing* 52.6 (2014), pp. 3359–3372. DOI: 10.1109/TGRS.2013.2272593 (cit. on p. 5).

[8] M. Gure et al. "Use of satellite images for forest fires in area determination and monitoring". In: *2009 4th International Conference on Recent Advances in Space Technologies*. 2009, pp. 27–32. DOI: 10.1109/RAST.2009.5158210 (cit. on p. 5).

[9] N. Aeronautics and S. Administration. "Landsat 9 brochure final 508 compliant". In: (2020) (cit. on p. 8).

[10] USGS. "Landsat Continuing to Improve Everyday Life". In: (2019). URL: https://landsat.gsfc.nasa.gov/sites/landsat/files/2019/02/Case_Studies_Book2018_Landsat_Final_12x9web.pdf (cit. on p. 8).

[11] M. Gure et al. "Use of satellite images for forest fires in area determination and monitoring". In: *2009 4th International Conference on Recent Advances in Space Technologies*. 2009, pp. 27–32. DOI: 10.1109/RAST.2009.5158210 (cit. on p. 9).

[12] K. Kaku. "Satellite remote sensing for disaster management support: A holistic and staged approach based on case studies in Sentinel Asia". In: *International Journal of Disaster Risk Reduction* 33 (2019), pp. 417–432. ISSN: 2212-4209. DOI: https://doi.org/10.1016/j.ijdrr.2018.09.015. URL: https://www.sciencedirect.com/science/article/pii/S2212420918304801 (cit. on p. 9).

[13] G. Demisse et al. "Using Satellite Images for Drought Monitoring: A Knowledge Discovery Approach". In: *Journal of Strategic Innovation and Sustainability* 7 (July 2011), pp. 135–152 (cit. on p. 9).

[14] M. R. Hussain. *"An Overview of Geographic Information System (GIS)"*. July 2016. DOI: 10.13140/RG.2.1.3569.5603 (cit. on p. 9).

[15] E. UK and Ireland. URL: https://www.esriuk.com/en-gb/what-is-gis/overview (visited on 08/16/2021) (cit. on p. 9).

[16] M. Luaces et al. "A Generic Architecture for Geographic Information Systems". In: (Jan. 2004) (cit. on pp. 10, 11).

[17] Z. Zhang et al. "Road Centerline Extraction from Very-High-Resolution Aerial Image and LiDAR Data Based on Road Connectivity". In: *Remote Sensing* 10.8 (Aug. 2018), p. 1284. DOI: 10.3390/rs10081284. URL: https://doi.org/10.3390/rs10081284 (cit. on pp. 11, 12, 21, 28, 76).

[18] J. D. Jayaseeli and D. Malathi. "An Efficient Automated Road Region Extraction from High Resolution Satellite Images using Improved Cuckoo Search with Multi-Level Thresholding Schema". In: *Procedia Computer Science* 167 (2020), pp. 1161–1170. DOI: 10.1016/j.procs.2020.03.418. URL: https://doi.org/10.1016/j.procs.2020.03.418 (cit. on pp. 12, 21).

[19] G. Cheng et al. "Automatic Road Detection and Centerline Extraction via Cascaded End-to-End Convolutional Neural Network". In: *IEEE Transactions on Geoscience and Remote Sensing* 55.6 (2017), pp. 3322–3337 (cit. on pp. 13, 21, 76).

[20] Y. Wei, K. Zhang, and S. Ji. "Simultaneous Road Surface and Centerline Extraction From Large-Scale Remote Sensing Images Using CNN-Based Segmentation and Tracing". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.12 (Dec. 2020), pp. 8919–8931. DOI: 10.1109/tgrs.2020.2991733. URL: https://doi.org/10.1109/tgrs.2020.2991733 (cit. on pp. 14, 21, 28).

[21] A. Buslaev et al. "Fully Convolutional Network for Automatic Road Extraction from Satellite Imagery". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2018, pp. 197–1973. DOI: 10.1109/CVPRW.2018.00035 (cit. on pp. 15, 21).

[22] S. Sun et al. "Road Centerlines Extraction from High Resolution Remote Sensing Image". In: *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*. 2019, pp. 3931–3934. DOI: 10.1109/IGARSS.2019.8898382 (cit. on pp. 15, 16, 21, 28, 29).

[23] Z. Miao et al. "A Method for Accurate Road Centerline Extraction From a Classified Image". In: *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of* PP (Dec. 2014), pp. 1–1. DOI: 10.1109/JSTARS.2014.2309613 (cit. on pp. 16, 21).

[24] Q. Guo and Z. Wang. "A Self-Supervised Learning Framework for Road Centerline Extraction From High-Resolution Remote Sensing Images". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13 (2020), pp. 4451–4461. DOI: 10.1109/JSTARS.2020.3014242 (cit. on pp. 17, 21).

[25] L.-C. Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation." In: *ECCV (7)*. Ed. by V. Ferrari et al. Vol. 11211. Lecture Notes in Computer Science. Springer, 2018, pp. 833–851. ISBN: 978-3-030-01234-2. URL: http://dblp.uni-trier.de/db/conf/eccv/eccv2018-7.html#ChenZPSA18 (cit. on pp. 18, 21, 28, 39–42, 58).

[26] V. Badrinarayanan, A. Kendall, and R. Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495. DOI: 10.1109/TPAMI.2016.2644615 (cit. on pp. 18, 21).

[27] T.-Y. Lin et al. *Feature Pyramid Networks for Object Detection*. 2017. arXiv: 1612.03144 [cs.CV] (cit. on pp. 19, 21, 59).

[28] S. Sun et al. "Road Centerlines Extraction from High Resolution Remote Sensing Image". In: *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*. 2019, pp. 3931–3934. DOI: 10.1109/IGARSS.2019.8898382 (cit. on p. 22).

[29] C. Henry, S. Azimi, and N. Merkle. "Road Segmentation in SAR Satellite Images With Deep Fully Convolutional Neural Networks". In: *IEEE Geoscience and Remote Sensing Letters* (Feb. 2018). DOI: 10.1109/LGRS.2018.2864342 (cit. on p. 22).

[30] W. XIA et al. "Road Extraction from High Resolution Image with Deep Convolution Network – A Case Study of GF-2 Image". In: vol. 2. Mar. 2018, p. 5138. DOI: 10.3390/ecrs-2-05138 (cit. on p. 22).

[31] S. J. Russell and P. Norvig. *Artificial Intelligence: a modern approach*. 3rd ed. Pearson, 2009 (cit. on p. 22).

[32] J. H. Fetzer. "What is Artificial Intelligence?" In: *Artificial Intelligence: Its Scope and Limits*. Springer Netherlands, 1990, pp. 3–27. DOI: 10.1007/978-94-009-1900-6_1. URL: https://doi.org/10.1007/978-94-009-1900-6_1 (cit. on p. 22).

[33] G. Bonaccorso. *Machine Learning Algorithms: A Reference Guide to Popular Algorithms for Data Science and Machine Learning*. Packt Publishing, 2017. ISBN: 1785889621 (cit. on pp. 22, 23).

[34] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016 (cit. on pp. 24, 26).

[35] M. Yani, M. B. I. S Si., and M. C. S. S.T. "Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail". In: *Journal of Physics: Conference Series* 1201 (May 2019), p. 012052. DOI: 10.1088/1742-6596/1201/1/012052. URL: https://doi.org/10.1088/1742-6596/1201/1/012052 (cit. on p. 27).

[36] P. Wang et al. "Understanding Convolution for Semantic Segmentation". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Mar. 2018. DOI: 10.1109/wacv.2018.00163. URL: https://doi.org/10.1109/wacv.2018.00163 (cit. on p. 27).

[37] H. Noh, S. Hong, and B. Han. "Learning Deconvolution Network for Semantic Segmentation". In: *ArXiv* (May 2015). DOI: 10.1109/ICCV.2015.178 (cit. on p. 28).

[38] H. Zhao et al. "Pyramid Scene Parsing Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6230–6239. DOI: 10.1109/CVPR.2017.660 (cit. on p. 30).

[39] L.-C. Chen et al. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs." In: *CoRR* abs/1606.00915 (2016). URL: http://dblp.uni-trier.de/db/journals/corr/corr1606.html#ChenPK0Y16 (cit. on p. 30).

[40] K. He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: Springer International Publishing, 2014, pp. 346–361. DOI: 10.1007/978-3-319-10578-9_23. URL: https://doi.org/10.1007/978-3-319-10578-9_23 (cit. on p. 30).

[41] K. He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: Springer International Publishing, 2014, pp. 346–361. DOI: 10.1007/978-3-319-10578-9_23. URL: https://doi.org/10.1007/978-3-319-10578-9_23 (cit. on p. 30).

[42] T. Li, M. Comer, and J. Zerubia. "Feature Extraction and Tracking of CNN Segmentations for Improved Road Detection from Satellite Imagery". In: *2019 IEEE International Conference on Image Processing (ICIP)*. 2019, pp. 2641–2645. DOI: 10.1109/ICIP.2019.8803355 (cit. on p. 30).

[43] X. Lu et al. "Multi-Scale and Multi-Task Deep Learning Framework for Automatic Road Extraction". In: *IEEE Transactions on Geoscience and Remote Sensing* 57.11 (2019), pp. 9362–9377. DOI: 10.1109/TGRS.2019.2926397 (cit. on p. 30).

[44] E. Shelhamer, J. Long, and T. Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4 (Apr. 2017), pp. 640–651. DOI: 10.1109/tpami.2016.2572683. URL: https://doi.org/10.1109/tpami.2016.2572683 (cit. on p. 30).

[45] H. Noh, S. Hong, and B. Han. "Learning Deconvolution Network for Semantic Segmentation". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, Dec. 2015. DOI: 10.1109/iccv.2015.178. URL: https://doi.org/10.1109/iccv.2015.178 (cit. on p. 30).

[46] Y. Xing, L. Zhong, and X. Zhong. "An Encoder-Decoder Network Based FCN Architecture for Semantic Segmentation". In: *Wireless Communications and Mobile Computing* 2020 (July 2020), pp. 1–9. DOI: 10.1155/2020/8861886. URL: https://doi.org/10.1155/2020/8861886 (cit. on p. 30).

[47] T. Y. Zhang and C. Y. Suen. "A Fast Parallel Algorithm for Thinning Digital Patterns". In: *Commun. ACM* 27.3 (Mar. 1984), pp. 236–239. ISSN: 0001-0782. DOI: 10.1145/357994.358023. URL: https://doi.org/10.1145/357994.358023 (cit. on pp. 30, 53, 54, 59).

[48] Z. Guo and R. W. Hall. "Parallel Thinning with Two-Subiteration Algorithms". In: *Commun. ACM* 32.3 (Mar. 1989), pp. 359–373. ISSN: 0001-0782. DOI: 10.1145/62065.62074. URL: https://doi.org/10.1145/62065.62074 (cit. on pp. 30, 50, 51, 53, 54, 58, 59).

[49] H. M. Fernandez et al. "An Assessment of Forest Fires and CO2 Gross Primary Production from 1991 to 2019 in Mação (Portugal)". In: *Sustainability* 13.11 (2021). ISSN: 2071-1050. DOI: 10.3390/su13115816. URL: https://www.mdpi.com/2071-1050/13/11/5816 (cit. on p. 33).

[50] K. He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916. DOI: 10.1109/TPAMI.2015.2389824 (cit. on p. 40).

[51] L.-C. Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: 1706.05587 [cs.CV] (cit. on pp. 40, 41).

[52] J. Long, E. Shelhamer, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3431–3440. DOI: 10.1109/CVPR.2015.7298965 (cit. on p. 40).

[53] V. Badrinarayanan, A. Kendall, and R. Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495. DOI: 10.1109/TPAMI.2016.2644615 (cit. on p. 40).

[54] G. Lin et al. *RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation*. 2016. arXiv: 1611.06612 [cs.CV] (cit. on p. 40).

[55] F. Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017). DOI: 10.1109/cvpr.2017.195. URL: http://dx.doi.org/10.1109/CVPR.2017.195 (cit. on p. 40).

[56] L.-C. Chen et al. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP (June 2016). DOI: 10.1109/TPAMI.2017.2699184 (cit. on p. 41).

[57] P. Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020 (cit. on p. 42).

[58] K. He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (cit. on p. 43).

[59] J. Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848 (cit. on p. 44).

[60] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG] (cit. on p. 47).

[61] N. Khanyile, J.-R. Tapamo, and E. Dube. "A Comparative Study of Fingerprint Thinning algorithms." In: Jan. 2011 (cit. on pp. 49, 50, 53).

[62] N. Han, C. La, and P. Rhee. "An efficient fully parallel thinning algorithm". In: *Proceedings of the Fourth International Conference on Document Analysis and Recognition*. Vol. 1. 1997, 137–141 vol.1. DOI: 10.1109/ICDAR.1997.619829 (cit. on p. 50).

[63] R. Gupta and R. Kaur. "Skeletonization Algorithm for Numeral Patterns". In: *International Journal of Signal Processing, Image Processing and Pattern Recognition* 1 (Dec. 2008) (cit. on p. 50).

[64] L. Ji et al. "Binary Fingerprint Image Thinning Using Template-Based PCNNs". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37.5 (2007), pp. 1407–1413. DOI: 10.1109/TSMCB.2007.903369 (cit. on p. 50).

[65] C. Arcelli and G. Sanniti di Baja. "A thinning algorithm based on prominence detection". In: *Pattern Recognition* 13.3 (1981), pp. 225–235. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(81)90099-6. URL: https://www.sciencedirect.com/science/article/pii/0031320381900996 (cit. on p. 50).

[66] P. Subashini. "'Optimal Thinning Algorithm for detection of FCD in MRI Images'". In: *International Journal of Scientific and Engineering Research (IJSER)* 2 (Jan. 2011) (cit. on p. 50).

[67] L. Ben Boudaoud, A. Sider, and A. Tari. "A new thinning algorithm for binary images". In: *2015 3rd International Conference on Control, Engineering Information Technology (CEIT)*. 2015, pp. 1–6. DOI: 10.1109/CEIT.2015.7233099 (cit. on p. 50).

[68] H. Jain and A. P. Kumar. *A Sequential Thinning Algorithm For Multi-Dimensional Binary Patterns*. 2017. arXiv: 1710.03025 [cs.CV] (cit. on p. 53).

[69] R. W. Hall. "Fast Parallel Thinning Algorithms: Parallel Speed and Connectivity Preservation". In: *Commun. ACM* 32.1 (Jan. 1989), pp. 124–131. ISSN: 0001-0782. DOI: 10.1145/63238.63248. URL: https://doi.org/10.1145/63238.63248 (cit. on pp. 53, 54).

[70] C. Heipke et al. "Evaluation of Automatic Road Extraction". In: *Inter. Arch. Photogramm. Remote Sens.* 32 (Oct. 1997) (cit. on p. 57).

[71] B. Wessel and C. Wiedemann. "Analysis of automatic road extraction results from airborne SAR imagery". In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 37 (Jan. 2003) (cit. on p. 57).

[72] Z. Shao et al. "MRENet: Simultaneous Extraction of Road Surface and Road Centerline in Complex Urban Scenes from Very High-Resolution Images". In: *Remote Sensing* 13.2 (2021). ISSN: 2072-4292. DOI: 10.3390/rs13020239. URL: https://www.mdpi.com/2072-4292/13/2/239 (cit. on p. 57).

[73] B. Wessel and C. Wiedemann. "Analysis of automatic road extraction results from airborne SAR imagery". In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 37 (Jan. 2003) (cit. on p. 58).

[74] O. Ronneberger, P. Fischer, and T. Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV] (cit. on p. 58).

[75] H. Zhao et al. *Pyramid Scene Parsing Network*. 2017. arXiv: 1612.01105 [cs.CV] (cit. on p. 78).

[76] H. Li et al. *Pyramid Attention Network for Semantic Segmentation*. 2018. arXiv: 1805.10180 [cs.CV] (cit. on p. 78).

[77] T. Fan et al. "MA-Net: A Multi-Scale Attention Network for Liver and Tumor Segmentation". In: *IEEE Access* 8 (2020), pp. 179656–179665. DOI: 10.1109/ACCESS.2020.3025372 (cit. on p. 78).

APPENDIX

Deep Learning Models Initiation

This Appendix shows the initialization parameters for the three deep learning models in study (DeepLabV3+, Unet, FPN) in Listing A.1, Listing A.2, Listing A.3.

For more advanced parameters initialization can be consulted the documentation at: https://smp.readthedocs.io/en/latest/models.html#id9

Listing A.1: Initial parameters of the DeepLabV3+ model

```
...
ENCODER = 'resnet50' #Classification model that will be used as an encoder and network backbone.
ENCODER_WEIGHTS = 'imagenet' #Database with pre-trained weights
CLASSES = select_classes #Avaliable classes: background and rural road
ACTIVATION = 'sigmoid' #Activation function to apply after the final convolution layer

#Create rural road detection model
model = smp.DeepLabV3Plus(
    encoder_name=ENCODER,
    encoder_depth=5,
    encoder_weights=ENCODER_WEIGHTS,
    encoder_output_stride=16,
    decoder_atrous_rates=(12,24,36),
    in_channels=3,
    classes=len(CLASSES),
    activation=ACTIVATION,
    upsampling=4
)

...
```

Listing A.2: Initial parameters of the UNET model

```
...
ENCODER = 'resnet50' #Classification model that will be used as an encoder and network backbone.
ENCODER_WEIGHTS = 'imagenet' #Database with pre-trained weights
CLASSES = get_classes #Avaliable classes: background and rural road
ACTIVATION = 'sigmoid' #Activation function to apply after the final convolution layer

#Create rural road detection model
model = smp.Unet(
    encoder_name=ENCODER,
    encoder_depth=5,
    encoder_weights=ENCODER_WEIGHTS,
    decoder_use_batchnorm=True,
    decoder_channels=(256, 128, 64, 32, 16),
    in_channels=3,
    classes=len(CLASSES),
    activation=ACTIVATION,
)

...
```

Listing A.3: Initial parameters of the FPN model

```
...
ENCODER = 'resnet50' #Classification model that will be used as an encoder and network backbone.
ENCODER_WEIGHTS = 'imagenet' #Database with pre-trained weights
CLASSES = get_classes #Avaliable classes: background and rural road
ACTIVATION = 'sigmoid' #Activation function to apply after the final convolution layer

#Create rural road detection model
model = smp.FPN(
    encoder_name=ENCODER,
    encoder_depth=5,
    encoder_weights=ENCODER_WEIGHTS,
    in_channels=3,
    classes=len(CLASSES),
    activation=ACTIVATION,
    upsampling=4,
)

...
```

The full implementation of the rural road detector and extractor can be consulted in the following links inside Google Colab: (DeepLabV3+) , (Unet) , (FPN Network)