# Automated clutter classification of LIDAR data for Radio Frequency propagation modelling

By

JUSTIN NICHOLAS WORSEY

Department of Electrical & Electronic Engineering
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering.

JUNE 2021

Word count: Forty thousand

## Abstract

Many technologies and applications now necessitate an awareness of their geographical surroundings, typically employing an array of sensors to capture the environment. An autonomous vehicle may utilise high definition cameras, LIDAR and RADAR; not only for collision avoidance but also for simultaneous location awareness and mapping (SLAM). Similarly, telecommunication network planning can benefit from the utilisation of RF propagation tools which include representations of target environments sourced from high resolution aerial photography and/or LIDAR point clouds. Mobile LIDAR scanners can capture a three dimensional environment and even colour each point using information from an associated camera. However, the resulting data is potentially very large, permutation invariant and clustered. Manually classifying this data, to maximise its utility in a propagation model, is not easily scalable; being both labour intensive and time consuming. This research defines a pipeline which facilitates the automatic classification of point cloud data and its subsequent consumption into a propagation model. It examines the use of sub-sampling as a mechanism to reduce the number of points to work with in order to reduce computational complexity and/or cater for lower resolution point clouds. It finds that a combination of edge-feature retention and sub-sampling at a resolution of 32 cm produces equivalent results to sub-sampling at 4 cm, whilst reducing the number of points by over 95%. Real-world LIDAR data, mapping 450 metres of an area in Bristol, UK, was captured, classified and converted into a series of simplified meshes before being input into a propagation model for comparison with human labelling and non-labelled (same class for all) data. Results superficially favour hand-labelled data, then automatically classified data followed by the non-labelled data. However, the results raise significant questions about whether the low-resolution of the hand-labelled data is truly representative of the real-world. In particular, how the coarseness of low-resolution can not only propagate rays more easily but also unintentionally block the line-of-sight. One major finding was the importance of labelling objects. This is not only to model the environment correctly, but also to discard undesirable objects which prevent propagation, if not removed.

# Dedication and acknowledgements

To my beautiful and long suffering wife Jules, who wholeheartedly supported my decision to embark upon this journey and whom I will be forever grateful and in her debt. I am not sure I would have been able to do this without her support and encouragement, especially during the dark days. I will always love you Jules. Also my dogs, Arthur and Herbie, who waited patiently for walkies as I read another paper.

I would like to thank my supervisors, Dave Bull, Simon Armour and in the early days, Andy Nix. They each had differing perspectives and provided great support and guidance. My industrial sponsor GCHQ and industrial supervisor Ian Hindmarch. Ian went above and beyond to support my work and for that I am in his debt. The CDT Communications team who provided the necessary programme for me to re-skill, learn and prosper. This includes Suzanne, Mark, Olly and my supervisors (who also ran the programme). Olly provided great comfort as a pastoral tutor during some dark, albeit predictable, days. Irrespective of my future, the CDT programme was invaluable to me; especially the taught component, which provided me with the necessary self-confidence to succeed. I would also like to thank the people at the Vision Labs, including the administration of Bridget and Sarah; but also researchers such as Aaron who were always at hand to give advice.

I would like to thank my son Elliott. I believe we were the first father son duo within the department and he never had a problem with the potential embarrassment of bumping into me during a lecture or at MVB. His patient explanation of complicated (no doubt, simple to him) mathematical concepts provided great insight. Also, my long standing friend Roger, who has shared the journey with me by embarking on the same CDT. Roger and I frequently discussed our PhDs in the wider context of being very mature students (although, I feel Roger has relived some of his younger days). I would like to thank my CDT cohort. We were a mixed bag of people from all walks of life but came together and helped each other. In particular, Adam, whom I am hopeful we shall stay in touch; Dave, for his no nonsense outlook and support, and the two Nigels who were unable to finish the course but, nonetheless, contributed and enriched my experience. A special dedication goes to Di, for choosing me as his best man.

## Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: .................................................... DATE: ..........................................

# Table of Contents

# List of Tables

**Table** **Page**

# LIST OF FIGURES

# Glossary

| | |
|---|---|
| **ALS** | airborne laser scanning. 13 |
| **ANN** | artificial neural network. 21, 22, 47 |
| **BCL** | Bilateral Convolution Layer. 52 |
| **CASI-1500** | Compact Airborne Spectrographic Imager 1500. 15 |
| **CNN** | convolutional neural network. xi, xiii, xiv, 3, 4, 19, 21, 22, 25, 28, 29, 33, 34, 35, 40, 41, 43, 44, 45, 46, 47, 50, 51, 52, 53, 54, 64, 129 |
| **CS-LBP** | Centre Symmetric Local Binary Pattern. 17, 19 |
| **DEM** | Digital Elevation Model. 8, 41, 103, 105 |
| **DoG** | difference-of-Gaussian. 17 |
| **DTM** | Digital Terrain Model. 8, 41, 103, 105 |
| **EM** | Expectation Maximisation. 37, 38 |
| **FDTD** | finite-difference time-domain. 7 |
| **FLOP** | floating point operations. 34 |
| **FoV** | field of view. 93, 98 |
| **GAN** | Generative Adversarial Network. 21, 31 |
| **GO** | geometric optics. 7, 8 |
| **GPS** | Global Positioning System. 14, 15, 97 |
| **GPU** | Graphics Processing Unit. 2, 21, 28, 34, 35, 60, 61, 72 |
| **GUI** | graphical user interface. 8 |

**HVS** human visual system. 17, 22

**IoU** intersection over union. 34

**ITU-RS** International Telecommunications Union-Radiocommunications Sector. 8

**KL** Kullback-Leibler. 24

**kNN** k-Nearest Neighbour. 31

**LBP** Local Binary Pattern. 17, 19

**LIDAR** Laser Imaging, Detection and Ranging. xiii, xiv, xvi, 2, 11, 12, 13, 14, 15, 16, 17, 22, 39, 40, 41, 43, 44, 45, 46, 47, 49, 50, 51, 54, 64, 88, 91, 92, 93, 94, 96, 97, 98, 102, 103, 125, 127, 129, 130, 131

**LOS** free space line-of-site. 5, 6, 106, 107, 109, 112, 116, 117, 118, 121, 122, 126, 131

**MEIS-II** multi-detector electro-optical imaging sensor. 16

**MIMO** multiple-input-multiple-output. 8

**MoM** method of moments. 7

**NaN** not a number. 72

**OS** Ordnance Survey. 12, 96, 105

**OSM** Open Street Maps. 12

**PCA** Principal Component Analysis. 17, 18, 19, 29

**PCL** Point Cloud Library. 100

**R-CNN** 'Regions with CNN features'. 16, 21, 33, 34, 50

**ReLU** Rectified Linear Unit. 22, 26, 27, 28, 64

**RF** Radio Frequency. 1, 2, 5, 6, 7, 9, 12, 19, 41, 49, 89, 91, 92, 107, 125, 127, 129

**RGB** red, green, blue. xiii, xiv, 15, 40, 41, 51

**RGB-D** red, green, blue, depth. 39, 41, 129

**RMSE** root mean square error. 13

**SIFT** Scale-Invariant Feature Transform. 17

**SLAM** Simultaneous localization and mapping. 14, 49, 93, 96, 97

**SLIC** Simple Linear Iterative Clustering Superpixels. 40, 42

**SPE** simple projective environment. 8, 9

**SURF** Speeded-Up Robust Features. 17

**SVM** Support Vector Machines. 31, 33

**TLS** terrestrial laser scanning. 13

**TU** Texture Unit. 17

**UAV** unmanned autonomous vehicle. 12, 14

**UE** user equipment. 1, 91, 106, 107, 109, 116, 117, 126

**UK** United Kingdom. 42

**UoB** University of Bristol. xiii, 8, 13, 14, 15, 42

**UoB** University of Bristol's. 41

**UTD** Uniform Geometrical Theory of Diffraction. 7, 8

**V1** primary visual cortex. 16, 17, 18, 19

**YOLO** You Only Look Once. 21, 34, 35, 50

# Chapter 1

# Introduction

Real-world Radio Frequency (RF) propagation measurements facilitate a 'gold standard' technique for understanding and analysing the physical environment of interest in the context of RF communications. For example, these measurements provide valuable information which enables the optimal placing of base-stations. Whilst aspirational, it is not always feasible to evaluate the domain to this degree; certainly not on any large scale or within tight time constraints. This is because the requirement for specialist equipment operated by experienced personnel can be very expensive, both in terms of the time required and financial resources needed [94][pg.51]; this is before considering the difficulties posed by the terrain itself which extends to include areas affected by ongoing war or natural disasters.

An alternative to physical site surveying is to predict how signals propagate through the environment using computer based models. This includes estimating the time of flight, amplitudes and the arrival angles of the multi-path components which arrive at the receiver [9]. There exists a trade-off between the accuracy of the model employed and its complexity [72], which necessitates suitable computational resources. However, the optimal model need only offer a good approximation, sufficient for general network planning. This is because the environment being modelled is constantly changing. Permanent networks have to be tolerant of the effects of seasonal variation, temporary structures and traffic. Furthermore, networks have to consider pedestrians, who are not only moving within the environment, but also carry the user equipment (UE) the base-stations are trying to serve.

Network planning has become more sophisticated recently. This has been driven by the advent of technologies, such as 5G, which require more base stations. Consequently, there is a greater reliance upon the tools-of-the-trade including RF propagation modelling. Fortunately, these tools have evolved over time to capitalise on better computational resources. For example, the ProPhecy

3D multi-element modelling tool [72] has evolved from defining environments with raster-based images to the definition of complicated structures using polygons. Not only has this provided a more accurate representation of the environment, the model has higher capacity to consider many more physical material types. This has only really been made possible by improvements in processing capacity and associated memory resources. However, due to accuracy considerations, modelled environments are typically hand-labelled and are therefore time consuming, costly and hard to scale up.

At the same time, the fields of image processing and artificial intelligence have rapidly evolved for similar reasons. Large strides were taken in 2012 with AlexNet [56]. It used similar neural network techniques to that of the LeNet 5 network [58], which was introduced in the 90s. The major break through came from the utilisation of multiple Graphics Processing Unit (GPU) cards, allowing it to train with larger datasets which had also become available. This reinvigorated these research fields, allowing ideas such as driver-less vehicles to become a reality. Furthermore, this renewed vigour has given rise to deep learning architectures [112] which go beyond basic imagery and are capable of tackling large point cloud and graph-based datasets.

## 1.1 Thesis objective and contribution

The objective and contributions are given here:

### 1.1.1 Objective 1

- **Objective** To provide RF propagation modelling tools with automatically created environments in which to model propagation. In doing so, it will significantly reduce human interaction, which is both costly and time consuming.

- **Contribution** The thesis will define an automatic clutter classification pipeline. The pipeline will classify LIDAR point cloud data and convert it into the necessary form suitable for consumption within the ProPhecy 3D multi-element modelling tool.

### 1.1.2 Objective 2

- **Objective** To examine the impact of sub-sampling upon point cloud classification. It will do this to show that the pipeline is capable of working with lower resolution data.

- **Contribution** The pipeline will use a state-of-the-art deep learning architecture which has been modified to place emphasis upon retaining edge features from heavily sub-sampled point cloud data. It will be shown, via comprehensive experimentation, that this technique enables the deep learning model to perform at levels it previously could not, including levels where less than 5% of the point cloud points are available.

### 1.1.3 Objective 3

- **Objective** To prove that the solution is capable of replacing hand-labelled environments by either i) producing more accurate results or ii) showing that a fast and efficient scalable solution is an acceptable trade-off.
- **Contribution** The thesis will demonstrate that environments created by the pipeline perform only marginally worse than an equivalent, human-labelled, environment. However, the proposed solution is automatic and considerably quicker. It will also point out that no definitive statement can be made to confirm that the results of the human-labelled environment perform, as expected, in the real-world. i.e. just because the results are better in the model does not mean they are in reality.

### 1.1.4 Objective 4

- **Objective** To provide the propagation model with a simplified environment in order to reduce computational complexity.
- **Contribution** Pipeline generated meshes will be simplified including introducing a novel technique for reducing the complexity of foliage. This approach finds the centroids of clusters of points and creates a mesh wrapper to surround them. The effect of this is to create balloon-like meshes to incorporate branches and leaves.

### 1.1.5 Objective 5

- **Objective** To justify the use of either classified or hand-labelled data.
- **Contribution** It will be shown that it is important to label an environment. Not just so that the propagation model can assign the correct physical property, but also as a mechanism to removed unwanted artefacts.

## 1.2 Overview of thesis

An overview of the remaining chapters of this thesis is given here. An argument for the use of automatic clutter classification over human intensive labelling is presented in chapter 3. Both techniques describe the environments which propagation tools attempt to model. An overview of related sources of data which facilitate classification will be given. Prior work, which has successfully classified clutter, and machine learning techniques to extract features will also be discussed.

A broad overview of the field of deep learning is given in chapter 4. This will start by defining a basic neural network and show how non-linear activation functions are affected by learnt weights. A significant part of this chapter will be dedicated to the CNN. This will include how

AlexNet [56] exploited advances in technology to resurrect interest in neural networks. It will discuss some of the weaknesses of neural networks and techniques to overcome them. It provides a short history to the current state-of-the-art models which are relevant to clutter classification. A series of experimentation will be defined, and the results given, to compare the CNN to a relatively new implementation of capsule networks.

The description of deep learning sets the stage for chapter 5. This chapter moves away from conventional image-based classification tasks and reviews techniques to classify point cloud and other datasets. As point clouds can be very large, it will consider the benefits and challenges of sub-sampling the data. It will first present a series of sub-sampling experimentation using a simple deep learning architecture to characterise the points of a synthetic dataset. Finally, the chapter will move to real-world data and a state-of-the-art model. The front-end, pre-processing component, of model will be modified in order to retain edge features.

Chapter 6 will define an automatic clutter classification pipeline. This includes the use of the state-of-the-art model from the previous chapter and steps taken to process the classified data into meshes suitable for use with the ProPhecy 3D multi-element modelling tool mentioned in chapter 3. A section will review the sources of data which will be fed into the pipeline for subsequent evaluation by the modelling tool. This includes describing the short-comings observed from the manual collection of point cloud data using a hand-held scanner. Thanks to the assistance of the industrial sponsor, Dr Ian Hindmarch, the output from the pipeline was imported into the propagation tool for testing. These results will be discussed and compared with existing ground-truth environments.

# CHAPTER 2

# RF PROPAGATION OVERVIEW

This chapter will give a brief overview of RF propagation and why it is suitable for propagation modelling. It will discuss the strengths and limitations of such modelling and introduce the target propagation modelling tool.

## 2.1 RF propagation

The fundamentals of RF propagation are well-defined mathematically. For example, the received power $P_{R,LOS}$ [1], assuming free space line-of-site (LOS), is given by

$$P_{R,LOS} = 10log_{10}\Big(\frac{P_T G_T G_R \lambda^2}{(4\pi d)^2}\Big) \tag{2.1}$$

where $P_T$ is the transmitted power, $G_T$ and $G_R$ are the antenna gains of the transmitter and receiver respectively, $\lambda$ is the wavelength of the carrier frequency and $d$ is the distance between the transmitter and receiver. The path loss $PL(dB)$ can be found using

$$PL(dB) = 10log_{10}\frac{P_T}{P_R} \tag{2.2}$$

where $P_T$ is the transmitted power and $P_R$ the received power [77][pg.17]. Path loss represents the attenuation between transmitter and receiver.

For typical scenarios, the LOS component is only one element of what is received at the receiver; albeit it is the dominant component, if a LOS exists. This is because the electromagnetic wave emitted is usually dominant according to its antenna gain. Consequently the wave may find multi-paths to the receiver by means of specular reflection, diffraction and diffuse scattering.

Briefly, specular reflection results from the wave encountering an object whose dimensions are large relative to its own wavelength [94][pg.54]. Diffraction is a mechanism which permits the wave to bend around an object giving rise to shadowing, areas which do not have LOS with the transmitter; and diffuse scattering occurs when the dimensions of the encountered object is small relative to the wavelength of the wave. In this instance the wave is re-radiated (or diffused) in several different directions [7]. Reflection and scattering are inversely correlated. Dependent upon the wavelength, a brick wall may appear smooth, rough or somewhere in-between. For example, a 1 GHz wave has a wavelength of 30 cm compared to a wavelength of only 5 mm for a 60 GHz wave. If it is smooth, and the physical characteristics of the material permit it, then the wave will reflect. If it is rough then it will scatter.

As electromagnetic waves travel at the speed of light it is clear that multi-path components will arrive at the receiver at different times, dependent on the route taken. This is known as the time-delay spread. Furthermore, wave energy is attenuated with distance and the arriving waves will constructively or destructively interfere with each other upon arrival, depending on their phase.

Given the mathematical nature of RF propagation it follows that a computer-based model can be constructed to predict it. One solution would be to solve Maxwell's equations in the model, but this is presently too expensive, computationally speaking. As a result, models set out to simplify the physical real-world and therefore, can only ever offer an approximation. This approximation largely depends on the ratio of wavelength to the dimensions of the scatterers [65]. It is also clear that without a precise understanding of the environment, including its clutter, only an approximation could ever be achieved, even if the model was mathematically accurate. This is due to the impact of aforementioned phenomena such as reflection, diffraction and scattering upon crudely classified objects within the environment i.e. clutter needs to be assigned its correct physical characteristics.

In general, RF propagation models tend to reside between empirical and deterministic (geometrical) types [98]. Empirical models are built using statistical characterisation of the received signal [94][pg.55]. They are simpler to build, require less computation but produce coarse results. Where once coverage, in terms of the optimal placement of base-stations in order to achieve the greatest footprint, was the main objective, attention has now moved toward maximising the capacity of the network. This reflects the need to fully utilise the limited bandwidth to meet an ever-growing demand [10]. Consequently, the coarse approximations of the empirical model are not well-suited to this task.

Deterministic models employ geometrical information detailing the location of environmental clutter such as buildings, trees and other artefacts which reside upon the terrain. Consequently, these models are more complicated than their empirical counter-parts, requiring a corresponding increase in computational effort. However, they result in much more accurate predictions. Semi-empirical models are formed by a combination of both empirical and deterministic modelling techniques.

Predominantly deterministic models use geometric optics (GO) based ray tracing techniques. GO de-constructs the wave into infinitesimally small straight-line conduits or rays and assumes energy is radiated through them. However, the theory is limited to direct, reflected or refracted rays. In order to cater for diffracted rays, the Uniform Geometrical Theory of Diffraction (UTD) is also incorporated into the model [94][pg.59]. Other modelling methods including finite-difference time-domain (FDTD) and method of moments (MoM) attempt to address specific limitations in ray tracing [94][pg.61-62].

Models which employ ray tracing tend to use one of two types, brute force or image based. Brute force ray tracing examines a series of rays that are emitted with equal energy from the transmitter. The model will map the route of each ray, taking into consideration all interactions the ray has with its environment, including generating additional rays to represent reflection, refraction [65] and diffraction [94].

Image based ray tracing defines a tree graph of images which are generated in every plane of the source and includes any subsequent reflecting surfaces. In this instance, each image has a straight-line distance to the receiver that is identical to the actual path taken by the ray [65] [94][pg.59]. A check upon the ray's path is essential to ensure that it intersects all the necessary panels to be received at the receiver.

For computational efficiency, rays are pruned according to the number of objects they encounter along their path e.g. a ray which has been reflected from greater than $n$ objects will be discarded because its energy would have been attenuated to the extent it no longer makes a significant contribution to the received power, assuming it even arrives at the receiver. Pruning also assumes that this type of ray would arrive outside the expected range of the time-delay spread given the distance travelled. The pruning thresholds are referred to as the order of diffraction, reflection etc.

Importantly, models must rely upon the correct identification of any environmental clutter and, to a lesser extent, should accommodate dynamic variations such as seasonal changes, pedestrians and vehicles by accepting that the environment is an approximation of the real-world. Poor classification of the clutter will adversely affect the modelled RF propagation and, in turn, its accuracy. This is due to the presentation of incorrect physical characteristics. Avoiding such classification is therefore highly desirable.

### 2.1.1 Strengths and limitations of modelling

It is not feasible to take an exhaustive set of physical propagation measurements which reflect an ever changing real-world environment. Deterministic propagation modelling which utilises ray tracing algorithms offer a substitute that is sufficient for most applications of the data e.g. optimal base station placements. They are cheaper to deploy and allow a multitude of different scenarios to be simulated, including those which consider trees being in-leaf or not [77][pg.52].

Mathematically they are a compromise, in that they predominately utilise GO; but even this approach can be computationally expensive depending on the size and complexity of the target environment and the model's order of interaction thresholds [72].

Even if deterministic models were mathematically accurate, representation of the environment is still only an approximation. For example, the legacy version of ProPhecy crudely defined the environmental clutter to five classifications and whilst the newer version may be able to identify asphalt for example, it is still only a generalisation. Furthermore, due to geometrical processing ambiguities, where rays can be deemed to diffract over the topmost surface, any structure where the sky cannot be seen is not modelled.

### 2.1.2 ProPhecy 3D multi-element modelling tool

The simplest ray tracing propagation models utilise a single antenna with rays projected over a two dimensional plane. A more realistic model would consider a three dimensional environment. Clearly, the complexity of the model increases notably in this instance, especially where the size and detail of the target environment is large. Furthermore, ray tracing can accommodate the addition of multiple elements by modelling each element in isolation, so does not appear to be a limitation from the outset. However, the computational costs of these factors are not insignificant and lead to a trade-off between runtime efficiency and prediction accuracy [72].

Tameh *et al* [109] introduced an outdoor 3D multi-element modelling tool called ProPhecy which, until recently was maintained by the UoB [125]. It employs a rigorous image based ray tracing technique. Additionally, in recognising the data capacity benefits of multiple antenna systems, the advanced modeller was designed to utilise a series of multiple-input-multiple-output (MIMO) optimization techniques. This facilitates efficient computation by avoiding evaluating each element in isolation.

The propagated rays of the model are projected over three dimensional geometry using GO Fresnel reflection coefficients and the UTD with slope diffraction. It includes mechanisms to calculate the effects of diffraction upon the rooftops and corners of buildings, reflection and scattering over both the terrain and buildings. Furthermore, the International Telecommunications Union-Radiocommunications Sector (ITU-RS) foliage loss model and a hybrid scattering model are also incorporated.

The original version of the modeller defined its environment from three dimensional Digital Terrain Model (DTM) and Digital Elevation Model (DEM) raster images used in conjunction with a clutter map. The clutter map gives rudimentary classification of the structures within the environment. These classifications are limited to buildings, foliage, open areas, water and other (tunnels and bridges) [80]. The next generation of the modeller is still in its infancy as it does not currently have a graphical user interface (GUI). However, it offers significant improvements over its raster based predecessor by utilising vector based ray tracing using a simple projective environment (SPE) field model [27].

Whilst there are some limitations using an SPE model, for example it cannot model arches, holes or tunnels [27]; the ability to create more complicated environments present itself. This includes the ability to define suburban dwellings, tower blocks, forts and castles in much greater detail including inclines like slanting rooftops. This is a major improvement considering raster based structures used within the original version were typically defined as simple geometric structures, with a fixed 90°rotation on the terrain i.e. always pointing in line with the compass directions north, south, east and west. Furthermore, the classification of clutter is no longer restricted to only five types and includes labels such as water, stone, road/rail avenue and asphalt [27][pg.16].

## 2.2 Summary

The mathematical nature of RF propagation has conveniently led to the development of propagation modelling tools including the ProPhecy 3D multi-element modelling tool. It has been discussed that, because of computational complexity, these models can only ever offer an approximation. One area which could be improved upon, with negligible impact on computational cost, is the environments with which the models were attempting to model. It was also suggested that whilst human labelling of the environment was accurate, it was time consuming and expensive. Consequently, it was hard to scale up leading the way for an automated solution.

# CHAPTER 3

# Clutter classification

Accurately describing the environment is a desirable component of deterministic propagation modelling. Conveniently, automated clutter classification is now feasible due to advances in machine learning and given the wealth of open source datasets which are readily available for training them. Successes have already been made in road detection, tree species and the recognition of buildings using datasets containing high resolution aerial photography and LIDAR.

This chapter will justify why automatic clutter classification has an important role. This includes permitting time critical evaluation of environments and the ability to scale up. It will overview relevant sources of data which can be classified, discussing issues including inherent spatial relationships of images and size issues from LIDAR. A section on prior work will focus upon the successful classification of relevant objects such as trees, buildings and road. This will include a broader section which discusses how features can be extracted from image-based datasets using machine learning.

## 3.1 Justification for automatic clutter classification

Accepting the limitation that a propagation model is currently only capable of using mathematical approximations of the real-world, one area in which improvements may be made is in the quality of the modelled environment. Whilst some parameters directly affect computation, others such as the correct classification of clutter deliver quality improvements at negligible processing cost. Although describing clutter is an infrequent task, it is nonetheless time consuming for a human operative to accomplish, although it is likely to be more accurate. Furthermore, as the complexity of the labelling increases it can be hard to scale up and potentially, incurs a significant labour cost.

Automated assessment of a three dimensional geographical environment is highly desirable for many reasons. This includes the creation of accurate environments which can be used within RF propagation modelling tools for general network planning, including the rapid deployment of communication aids in areas affected by war or natural disasters. Furthermore, understanding what physical resources are available in these areas can be vital and time sensitive.

The key challenges for any form of adequate classification include the sources of data available, their resolution and variations such as differences in foliage due to season and lighting effects. Limited training data and unreliable ground truth are other obstacles. Reliable automated classification of the clutter is therefore an aspirational goal; especially given the wealth of real-world environment data that is currently available within the public domain which facilitates this. Admittedly, this would not necessarily remove all requirements surrounding human validation. This is because of the potential for errors within the classification to still exist, including misclassification due to adversarial examples.

In the context of propagation modelling, clutter classification has two main benefits. The first is for assigning physical material properties to the objects within the environment so that rays within the tool can propagate correctly. The other is for removing unwanted artefacts which adversely affect the model. This includes the removal of pedestrians and vehicles which are not fixed objects within the environment. However, this does raise questions about the level of accuracy required within the model, and even if this is attainable. The main body of this thesis attempts to answer this.

Clearly, propagation models are not the only applications to benefit from environmental awareness; other applications include real-time detection of the surroundings to enable the successful navigation of unmanned autonomous vehicles (UAVs) [100]; and the documentation of architecture of historical significance [22]. However, environment analysis, other than in the context of propagation modelling, is beyond the scope of this thesis.

## 3.2 Overview of terrain mapping datasets

This section gives a brief overview of the main types of dataset which describe the terrain and can be used for analysis and classification. LIDAR is described first because it is subsequently used as a component of ortho-rectification applied to vertical aerial photography. Multi-spectral imagery is considered last. This is because it can be seen, in the context of terrain mapping, as a superset of the capturing capability of vertical aerial photography; especially because they share the same deployment methods e.g. the device is mounted on the under carriage of a fixed wing aircraft. Other sources of data include crowd-sourced Open Street Maps (OSM) and Ordnance Survey (OS) maps which complement the datasets described in this section.

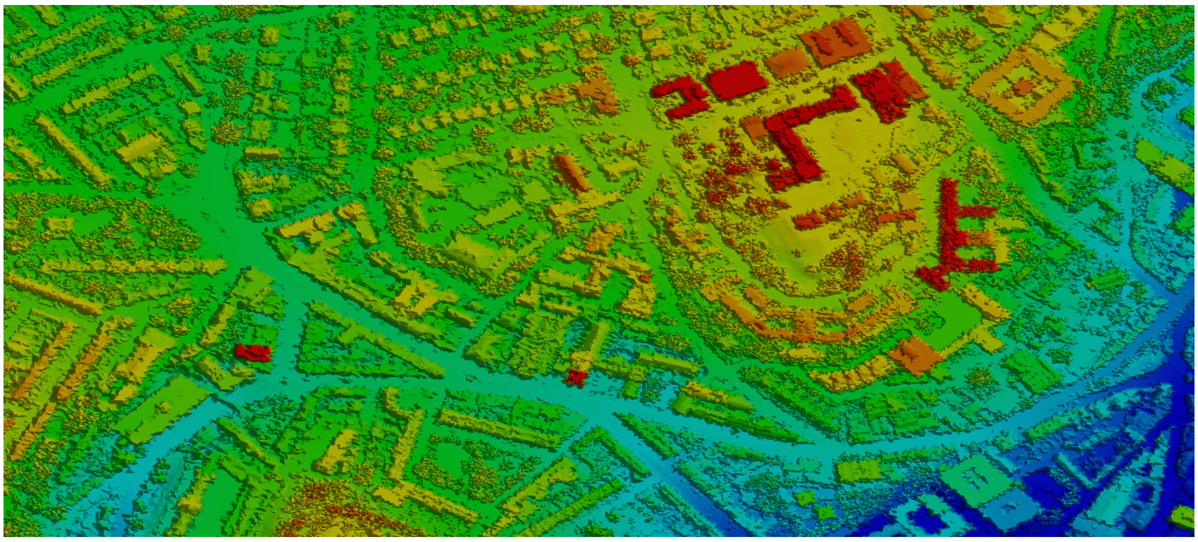


Figure 3.1: LIDAR scan of the area surrounding the UoB showing the intensity values of the point cloud at a resolution of one metre. Source: The Environment Agency.

### 3.2.1 LIDAR

Laser Imaging, Detection and Ranging (LIDAR) measures the reflection and return time from a pulsed laser light. The datasets are obtained by either scanning from the air (airborne laser scanning (ALS)), the ground (terrestrial laser scanning (TLS)) or by the use of a hand-held scanner. Results are referred to as point clouds which comprise a set of 3D coordinates and an associated intensity value. Individual points of a point cloud can be coloured with values captured by an associated camera, but this relies upon accurate calibration of additional sensors such as accelerometers.

Traditionally, LIDAR has been captured and maintained by government entities, such as the UK's Environment Agency, at resolutions around one metre squared. This data was used primarily as a means of surveying the landscape and fits well with the objective of network planning. Figure 3.1 shows the intensity values received from an airborne scan of the University of Bristol (UoB); the dataset was downloaded from the public repository at the Environment Agency with a resolution of approximately one metre and a vertical accuracy of ±15cm RMSE [24]. In this instance, the range of intensity values from red to blue reflect the height (or proximity to the measuring device in the air).

Due to the explosion of research surrounding automated vehicles there is now an abundance of LIDAR data available at street level with considerably better resolution, including datasets with between 1,000 and 2,000 points per square metre. Street level measurements have the advantage that they capture information blocked by a simple top-down view e.g. the full detail below the roof of a petrol station forecourt. Equally, they are unable to capture objects such as the roof tops, which airborne measurements do, meaning height information is compromised.

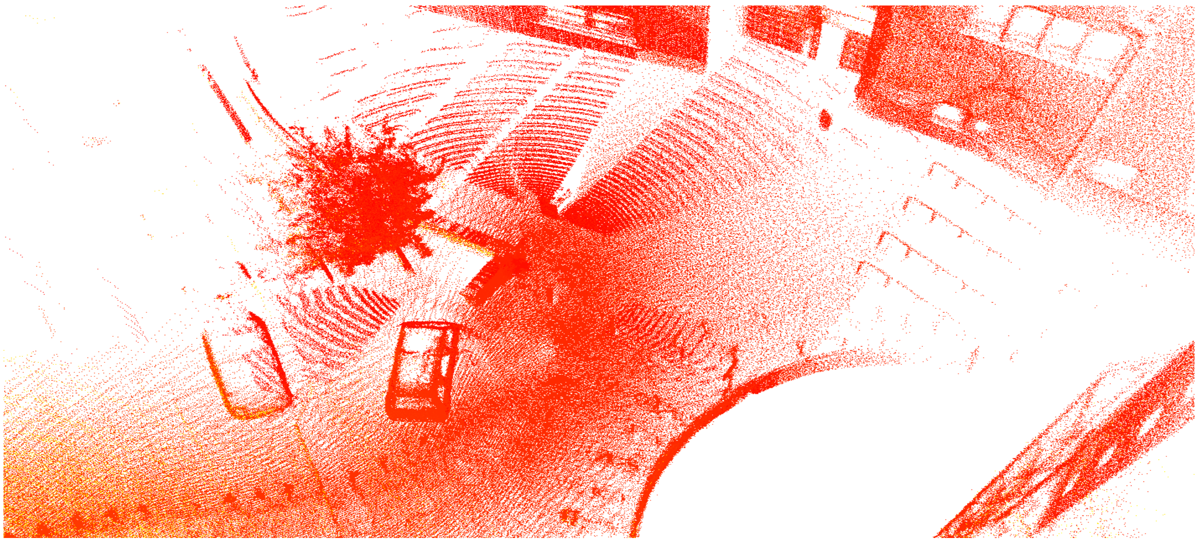


Figure 3.2: Hand-held LIDAR scan of the Merchant Venturers building at the University of Bristol (UoB).

Point clouds can be very large with hundreds of millions of points, or even more. Furthermore, they can be potentially unordered. This is true even though there is some Euclidean information captured by the scanning technique. Sophisticated scanners will utilise Global Positioning System (GPS) information and/or Simultaneous localization and mapping (SLAM) to provide spatial information but it cannot be assumed that neighbouring points will be captured in order. To emphasise this point, consider a photograph. All neighbouring pixels are spatial related to their neighbours and are ordered; whereas, a LIDAR scan may capture neighbouring points at the start and end of a walk up and down a road. This means a classifier needs to be permutation invariant and designed to take the size of the point cloud into account.

Point ordering and size are not the only issues a classifier would have to consider. Figure 3.2 presents a two dimensional projection of a hand-held scan which started and ended in the same location, so by definition the point cloud is loosely unordered. It shows an uneven distribution of points, including areas of sparsity, LIDAR shadows cast as objects block the path and patterns which do not reflect the actual environment. These are scan lines which occur because of the multi-beam LIDAR sensor employed by the scanner [91].

### 3.2.2 High resolution vertical aerial photography

A broad definition of aerial photography is the process of capturing one or more images of the ground, usually by mounting a camera to an airborne vehicle such as an aeroplane or UAV. Images captured at an angle are referred to as oblique and those captured pointing directly at the nadir (directly below) are considered vertical images [71].

Since 2006, the Environment Agency have recorded high resolution vertical aerial imagery.

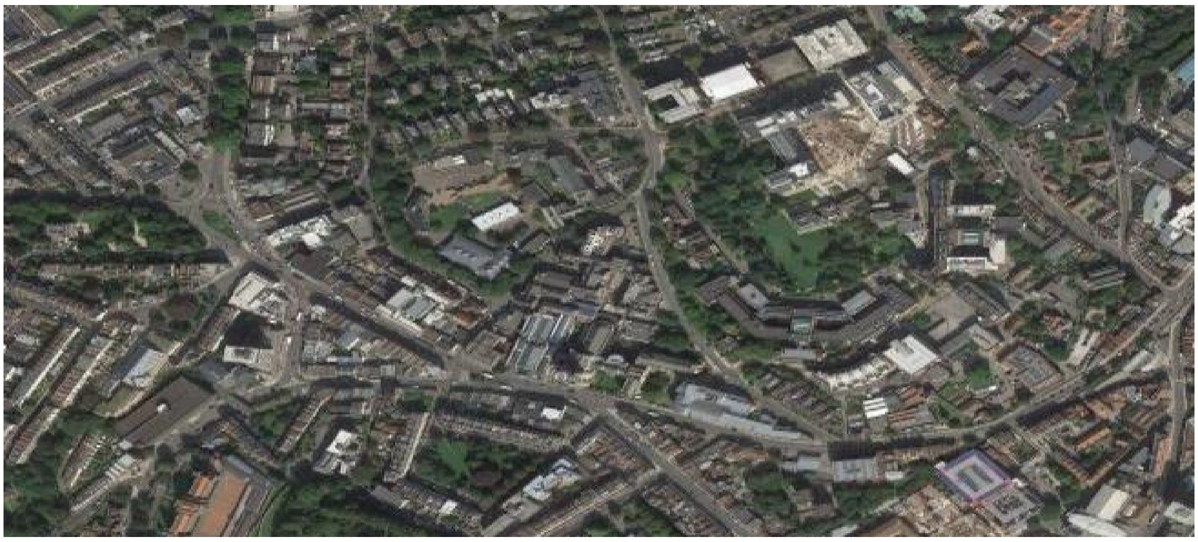


Figure 3.3: True colour RGB vertical aerial imagery of the University of Bristol (UoB). Source: Google maps.

They capture reflected light including RGB and near infra-red at resolutions between 10cm and 50cm [25]. Additionally, they use simultaneous LIDAR and GPS information to ortho-rectify the images to remove systematic sensor and platform induced geometry errors [96]. Figure 3.3 shows true colour RGB vertical aerial imagery of the University of Bristol (UoB); the image used was captured from a screen-shot from Google maps. Whilst Google do not specifically acknowledge the sources of their data they do employ a variety of different capturing techniques e.g. satellite, aerial and street level which are of a sufficiently high quality suitable for research work, including the manual mapping of urban agriculture performed by Taylor and Lovell [110].

### 3.2.3 Multi-spectral

In the context of airborne image capturing, multi-spectral data recording can be considered as an extension of vertical aerial photography. This can be seen by observing that the spectrum recorded by the Environment Agency, within their aerial photography dataset and discussed in section 3.2.2, is multi-spectral. This is because it includes both true colour and near infra-red bands. Multi-spectral devices are capable of acquiring even more of the spectrum, including ultra-violet. For example, the Landsat-8 satellite captures 11 spectral bands with a typical resolution of 30 metres [33]. The RGB bands form the visible true colour component of the data.

Whilst not currently widespread, the Environment Agency deploy a Compact Airborne Spectrographic Imager 1500 (CASI-1500) for cases where the spectral information acquired from regular aerial photography is not adequate enough to map specific variables, such as vegetation type or biomass [23].

An example of usage can be seen in the work of Gougeon [37] who successfully used high spatial aerial multi-spectral images captured by multi-detector electro-optical imaging sensor (MEIS-II), with a resolution of 31 cm per pixel and at an altitude of 439 metres (1440 ft.), for the automatic delineation of individual tree crowns.

## 3.3 Definition of ground truth

Through-out this thesis the definition of ground-truth will be based upon its use as a labelled environment. Each dataset, whether it comprises imagery, LIDAR or meshes will include associated hand-labelled data. In this instance, the main idea is to evaluate how any solution compares with such ground-truth.

## 3.4 Prior work and feature extraction

Automatic texture classification techniques are not new, with some dating back to the early 1990s. They exploited the increase in spatial resolution of satellite imagery at the time [39] [116]. Other machine learning algorithms which use image capturing methods, such as high resolution aerial photography, multi-spectral imaging and stereo-photogrammetry have had success in areas including road detection [69], tree species recognition and their height [26]. Albeit, height measurements captured using stereo-photogrammetry are prone to error [103]. Additionally, LIDAR data has been successfully used to recognise buildings [52], determine terrain models [54] and detect trees; this extends to individual species identification, together with an estimation of crown width [50] [53]. More recently, Kudinov and Hedges provide a proof of concept recognition tool [2] which utilises a 'Regions with CNN features' (R-CNN) based deep learning architecture [40] to detect buildings from LIDAR based upon their roof characteristics.

### 3.4.1 Standard geometric feature extraction from RGB(D) images

An environment can be separated into two groups, man-made and natural textures which can provide the basis for subsequent classification. Man-made features, for example buildings and roads, are typically formed from basic geometric shapes; whereas natural artefacts, including trees and plants, are not. This is most apparent when studying LIDAR data as the points of natural features are mostly scattered and do not exhibit the regularity of a man-made structure [22]. Clearly, this classification does not work in all cases e.g. an environmentally friendly rooftop is a combination of both.

Feature extraction is an important aspect for both natural vision systems and computer-based image processing. It can be characterised by its ability to provide a succinct, potentially dimensionality reduced, description of a scene. From a biological perspective, the primary visual cortex (V1) is responsible for the extraction of low-level features from its receptive fields. These basic features include localised edge orientation, spatial frequency and colour information [20].

Given the success of the human visual system (HVS) it is easy to consider it, as a possible solution, for computer-based image processing systems; commencing with the conversion of pixel intensities of the source image into similar low-level descriptors found within the V1 layer.

Algorithmic approaches that mimic primate vision include Scale-Invariant Feature Transform (SIFT) [59] [60] which utilises difference-of-Gaussian (DoG) to discover local extrema which have the potential to be key locations; and its successor Speeded-Up Robust Features (SURF) [11] which exploits box filters and the integral image [115] to achieve greater computational efficiencies. The probability of a feature's existence is assessed after the keys have been clustered using a Hough transform. SIFT's strengths include its invariance to linear transformations and its ability to tolerate some illumination changes and affine projections [59].

Furthermore, SIFT keys have been used to overlay LIDAR data over aerial imagery [5]. Importantly, this alignment would permit the traversal from one dataset domain type to another, making information that would not normally be available in the present domain accessible e.g. height information from the LIDAR domain can be associated with features extracted from the image domain; an environmentally friendly rooftop planted with vegetation may be viewed differently when height information is provided i.e. it is not on the ground. Conversely, image segmentation information obtained from using techniques such as watershed transformations [43] could be redeployed to segment LIDAR data. It is clear that aligning the differing datasets has the potential to produce sets of feature-rich descriptors for ensuring good machine learning classification. Obviously, alignment can be performed without SIFT keys if both domain types are geo-tagged with the same location information such as British National Grid (OSGB 1936), as used by the Environment Agency.

Local Binary Patterns (LBPs) [73] [79] [101] and its variants [42] provide an efficient mechanism for feature extraction and is based up the development of the Texture Unit (TU) [39] [116]. This technique works by binary encoding the relative pixel intensities from a group of neighbouring pixels against either the centre pixel or, in the case of Centre Symmetric Local Binary Pattern (CS-LBP), a mirrored and flipped pixel relative to the current pixel's position. The result for any one local neighbourhood is a Local Binary Pattern (LBP) code. Ojala et al observed that a small sub-set of rotation invariant LBP codes represented most of the scene [74]. These uniform patterns include V1-like basic features such as bright/dark spots, edges and negative curvature.

Extraction can also be achieved using techniques that utilise a linear superposition such that the image is decomposed into a weighted set of basis functions [4] of the form

$$I(x,y) = \sum_i a_i \theta_i(x,y) \tag{3.1}$$

where $\theta_i$ is the basis function whose weighting is given by the coefficient $a_i$. Principal Component Analysis (PCA) [49] is capable of discovering a dense and efficient set of orthogonal basis functions. These are the eigenvectors derived from the mean-subtracted covariance matrix of a set of training images. The weights $a_i$ are found by projecting the source image, using the dot product, onto each eigenvector. Eigenfaces [99] have successfully matched faces using this technique; although it was observed that the first three major components are particularly sensitive to illumination effects [13].

However, the basis functions derived from PCA do not resemble the features commonly found within the V1 layer and are not spatially localised [28] [4]. To resolve these issues, Olshausen and Field suggest finding a set of, possibly non-orthogonal, sparse codes as basis functions; where sparseness refers to the ability to describe an image by a small subset of basis functions [75] as opposed to having the density of PCA. To achieve sparse coding, Olshausen et al minimise an objective function defined as

$$E = (\text{sum of residuals}) + (\text{penalty}) \tag{3.2}$$

with respect to the coefficient $a_i$ introduced in equation 3.1. The objective function (eq.3.2) comprises two parts; a reconstruction loss term defined as

$$\text{sum of residuals} = \sum_{xy} \Big[ I(x,y) - \sum_i a_i \theta_i(x,y) \Big]^2 \tag{3.3}$$

computed as the square of the difference between the original image and the linear superposition hypothesis, and a penalty or regularisation term defined as

$$penalty = \lambda \Big[ \sum_i S\Big(\frac{a_i}{\sigma}\Big) \Big] \tag{3.4}$$

where the level of sparseness is influenced by the regularisation parameter $\lambda$. The function $S(x)$ is a non-linear function such as $log(1+x^2)$ or $-e^{-x^2}$ [4].

#### 3.4.1.1 Extraction using neural networks

Feed-forward neural networks can be configured as autoencoders to discover basis functions or features. A simple autoencoder architecture comprises three layers; an input layer, hidden or code layer and an output layer. The fundamental principle is to train the weights connecting the hidden layers in order to match output to the input; before examining the internal values of the hidden layer. For an under-complete autoencoder the hidden layer contains fewer dimensions than the input layer and so is forced to identify the best representative features [34][ch.14]. This is due to the inherent bottleneck in the design.

A linear autoencoder of this type is capable of learning an similar set of basis functions to those found using PCA. Unlike PCA however, the basis functions are not guaranteed to be orthogonal and are likely to assume equal variance; although they do span the same space as

PCA [34][ch.14], hence its similarity. Whilst not as efficient as the direct application of PCA, the benefit of utilising an autocoder is in its capacity to generalise. This includes the discovery of basis functions which project onto curved manifolds and not just linear manifolds projected onto by PCA. This can be achieved by sandwiching the code layer between two non-linear layers [46].

Visualisation of the first convolutional layer of a convolutional neural network (CNN) reveals that deep learning networks are also capable of discovering V1 style low-level features [56] [123]. This is achieved by adjusting the weights in the set of related kernels during training. Bringing this full circle, eight $3 \times 3$ kernels combined with a step activation function can be used to discover the aforementioned LBP codes, or four $3 \times 3$ kernels for CS-LBP encoding; although, it is preferable to allow the network to discover the best features without constraint.

## 3.5 Summary

This chapter gave a justification for the use of automatic clutter classification within the context of RF propagation modelling. It was suggested that this would mitigate some of the time and labour expenses surrounding hand-labelling data, although, it accepts that this would not necessarily be as accurate. A description of sources of data which are relevant to propagation modelling was given. This was followed by two forms of prior work. The first, demonstrated that techniques had already successfully classified buildings, roads and trees in various ways. The latter presented an overview of common machine learning techniques which have been used to extract features from image-based sources of data. This was extended to include neural networks which are able to capitalise upon discovered V1-like and other features to become fully fledged classifiers capable of labelling datasets for use within propagation modelling.

# Chapter 4

# Deep learning architectures

In recent times, deep learning has become synonymous with the detection and recognition of objects using computers. Whilst the overall building blocks have been defined over many decades, the field began to excel when researchers were able to exploit technological advances. Advances such as greater memory, parallel processing power combined with access to much larger datasets. This chapter presents an overview of relevant deep learning concepts which can be utilised as part of a pipeline for the automatic classification of clutter. It will start by defining the biologically inspired neuron unit in the context of an artificial neural network (ANN).

The chapter will then overview the powerful convolutional neural network (CNN) which is designed around the fundamental idea that feature detectors can be reused across the entire image to find multiple features. It will begin by illustrating the LeNet5 architecture which predates the aforementioned advances in technology, but forms the basis of all subsequent models. It will discuss problems within deep learning, such as the vanishing/exploding gradient and how to use regularisation in order to avoid over-fitting during training. It will highlight the susceptibility of CNNs to adversarial examples and how this gave rise to Generative Adversarial Networks (GANs), which use two competing networks to learn with each other. Work by Zeiler et al will be shown in order to visualise the expanding field of view seen in each layer of the CNN.

A number of different image processing architectures, which incrementally improve the state-of-the-art, will be described including i) AlexNet, famed for its use of GPUs; ii) ConvNet for increasing the depth of the network whilst being constrained by processing power; iii) Inception for its investigation into the receptive field whilst increasing depth; and iv) ResNet for asking the question of how deep can networks be. Image detection and segmentation of objects within a single input image will also be reviewed using R-CNN and You Only Look Once (YOLO) architectures.

Whilst powerful, CNNs are known to have limitations including spatial invariance which prevents architectures from describing the angle of rotation of an object. Hinton et al [93] attempt to improve upon CNNs with the introduction of the capsule network, which they believe reflect biological structures in the HVS better. This chapter will give an overview of the capsule network, including how it utilises a vector based activation function instead of a traditional scalar function, and how it uses dynamic routing to route data efficiently through the network. It will also explain why equivariance is preferable to invariance and how this requires less training data.

Finally, a series of experimentation, which has been performed to compare the performance of the capsule network against a similarly sized CNN, will be discussed. It will describe the creation of a small, terrain classification, dataset based upon an existing ground truth data source. The experimentation is novel because it uses LIDAR as an additional component of the source input for capsule networks. It is limited in that the networks are relativity small and these would almost certainly be out-classed by any of the state-of-the-art CNNs architectures. This constraint is enforced to permit objective evaluation against the capsule network model proposed by Hinton el al. Whilst these experiments do have numerous short-comings, one minor contribution is to demonstrate concepts of under and over fitting.

## 4.1 Overview of deep learning architectures

### 4.1.1 Basic artificial neural network

The basic building block of an ANN is the neuron. It is illustrated in Figure 4.1 and is crudely based upon a biological neuron. It comprises a number of inputs and a bias node, together with accumulation and activation functions which lead to an output. Each input has an associated weight, which is learnt by the model during training, and is used to scale the input value. Ultimately, the absolute value of the weight reflects the value's importance. This is also true for the bias node whose value is constant. The accumulation function is simply the dot product of all inputs, including the bias node, multiplied by their respective weights. Another way to look at this is to assume that the weights form a discovered feature and the dot product provides a similarity measurement between them and the input values. In reality, complicated models tend to have multiple weights per input, forming feature maps which attempt to capture multiple aspects of the input.

Traditionally, the activation function evaluated the similarity of the input to the learnt feature and fired if it exceeded a threshold e.g. consider a step function which triggers upwards when the value was greater than 100. The purpose of the bias node in the neuron is to centre that threshold to zero. Whilst linear activations can be used in neurons, they cannot solve complicated problems including the XOR function. This is because it is impossible to define a hyper plane between the class boundaries. Most activation nodes use non-linear functions to resolve this including Sigmoid (or Logistic), TanH and variations of Rectified Linear Unit (ReLU). Equations 4.1, 4.2

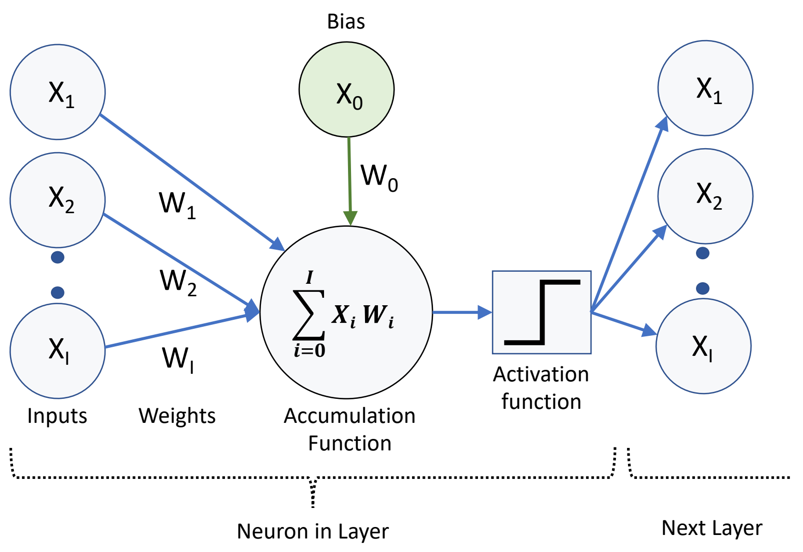


Figure 4.1: A basic representation of a fully connected neuron with a bias node and step activation function.

and 4.3 show the respective activation function and their derivatives. Importantly, activation functions have derivatives making them mathematically suited to learning techniques such as gradient descent. Where an activation function output is undefined, then an agreed output is chosen. This is demonstrated in the Relu function show in equation 4.3 where x becomes zero when the derivative is undefined.

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad f'(x) = f(x)(1-f(x)) \tag{4.1}$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \qquad f'(x) = 1 - f(x)^2 \tag{4.2}$$

$$f(x) = \begin{cases} x & x > 0 \\ 0 & \text{otherwise} \end{cases} \qquad f'(x)\begin{cases} 0 & x < 0 \\ 1 & x > 0 \\ undefined & x = 0 \end{cases} \tag{4.3}$$

Neurons or nodes are placed into layers creating a network; deep learning networks have many layers. Figure 4.2 shows a basic and shallow fully connected model for illustration purposes. It comprises of two input nodes, two nodes in a hidden layer and a single output node. The hidden layer is so called due to not being accessible directly. Each input connects to each hidden neuron and each output from those nodes are used in the final output. A network where every output from a node is connected to every input of a node in the next layer is known as fully connected.

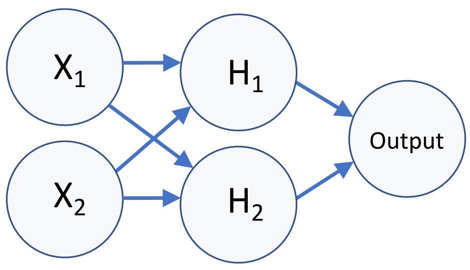


Figure 4.2: A simple fully connected neural network with two inputs, two hidden nodes in a single layer and one output node.

The overall task of a network is to perform function approximation i.e. given a problem they attempt to find function-like behaviour which solves it. Network learning is achieved by running training data through the model and accessing how well it predicts the required response. For supervised learning, this is given in a set of labelled data. The network undergoes a series of iterations until the model is trained, and therefore met its objective, or has exhausted its options under the specific training regime. The training data is of the form

$$\{(x_n, t_n)\}, n = 1 \ldots N \tag{4.4}$$

where $N$ is the total number of examples in the training data, $x_n \in \mathbb{R}^D$ is data and $t_n$ is the associated label. The network trains by evaluating a loss or objective function similar to the aforementioned sparse coding function in equation 3.2, or using probabilistic techniques such as cross entropy. Cross entropy reflects the divergence over the predicted and expected probability distributions. It is defined as

$$H(P,Q) = -\mathbb{E}_{\vec{x} \sim P}[logQ(\vec{x})] \tag{4.5}$$

where $-logQ(\vec{x})$ is the Shannon information content of $\vec{x}$ over the predicted probability distribution $Q$ which is then weighted according to the probability that $\vec{x}$ occurs with the expected probability distribution $P$. If the predicted distribution $Q$ is equal to $P$ then the cross entropy value will be the entropy of $H(P)$. $H(P)$ is defined in a similar way to cross entropy

$$H(P) = -\mathbb{E}_{\vec{x} \sim P}[logP(\vec{x})] \tag{4.6}$$

with the noticeable difference being the function is performed over a single distribution $P$. The entropy $H(P)$ forms the lower bound of the expected information content and is the value the loss function is attempting to minimise towards. Cross entropy is a simplified version of the Kullback-Leibler (KL) divergence function [34][pg.73]. KL incorporates the entropy function $H(P)$

within its equation to base-line the result such that it only shows the actual divergence. It is defined as

$$D_{KL}(P||Q) = \mathbb{E}_{\vec{x}\sim P}\left[log\frac{P(\vec{x})}{Q(x)}\right] \tag{4.7}$$

or equivalently

$$D_{KL}(P||Q) = \mathbb{E}_{\vec{x}\sim P}[logP(\vec{x})] - \mathbb{E}_{\vec{x}\sim P}[logQ(\vec{x})] \tag{4.8}$$

and therefore

$$D_{KL}(P||Q) = H(P,Q) - H(P) \tag{4.9}$$

where $H(P,Q)$ is the cross entropy of the distributions (Eq. 4.5) and $H(P)$ is Shannon entropy of the probability distribution $P$ (Eq. 4.6).

Backward propagation is then used to tweak the weights of the network. Typically, this is performed using a variant of gradient descent optimisation, which takes the derivative of the network and adjusts the weights relative to the others in order to attempt to minimise the computed loss. A learning rate hyper-parameter defines the scale of the derivative applied to the weights; if it is too small then convergence to the global minima could take too long, or it gets stuck in a local minima. If it is too large, then it can continually jump over the global minima. Adaptive learning rates can fine tune the directional jump over time.

Assuming the loss can be minimised sufficiently, the network can be trained to provide reasonable predictions given any appropriate input. Figure 4.3 demonstrates what really happened to the basic network, defined in Figure 4.2, after training. In this instance, the network was trained on data between 0 and 1. It was trained to favour values closer to the centre of the range.

By subsequently inputting small incremental values from 0 to 1 into the trained network, it is possible to see how the learnt weights have affected the activation functions. For example, the tanh function used in the hidden layer nodes morphs into a step-like form (in blue) for the first node and a upwards sloping curve (in orange) for the second. The overall effect is clearly seen when these two curves are merged in the output layer and applied to the sigmoid activation function. The figure superimposes this over the original training data. This shows reasonable activation for input values nearer the centre, as intended. The sigmoid activation function was used on the output to range bound the decision between 0 and 1. A better network and/or more training may have produced an output closer to the training data.

### 4.1.2 Convolutional neural networks (CNNs)

A convolutional neural network is a specific type of neural network architecture which applies learnt features, also known as kernels, to local receptive fields in order to create feature maps.

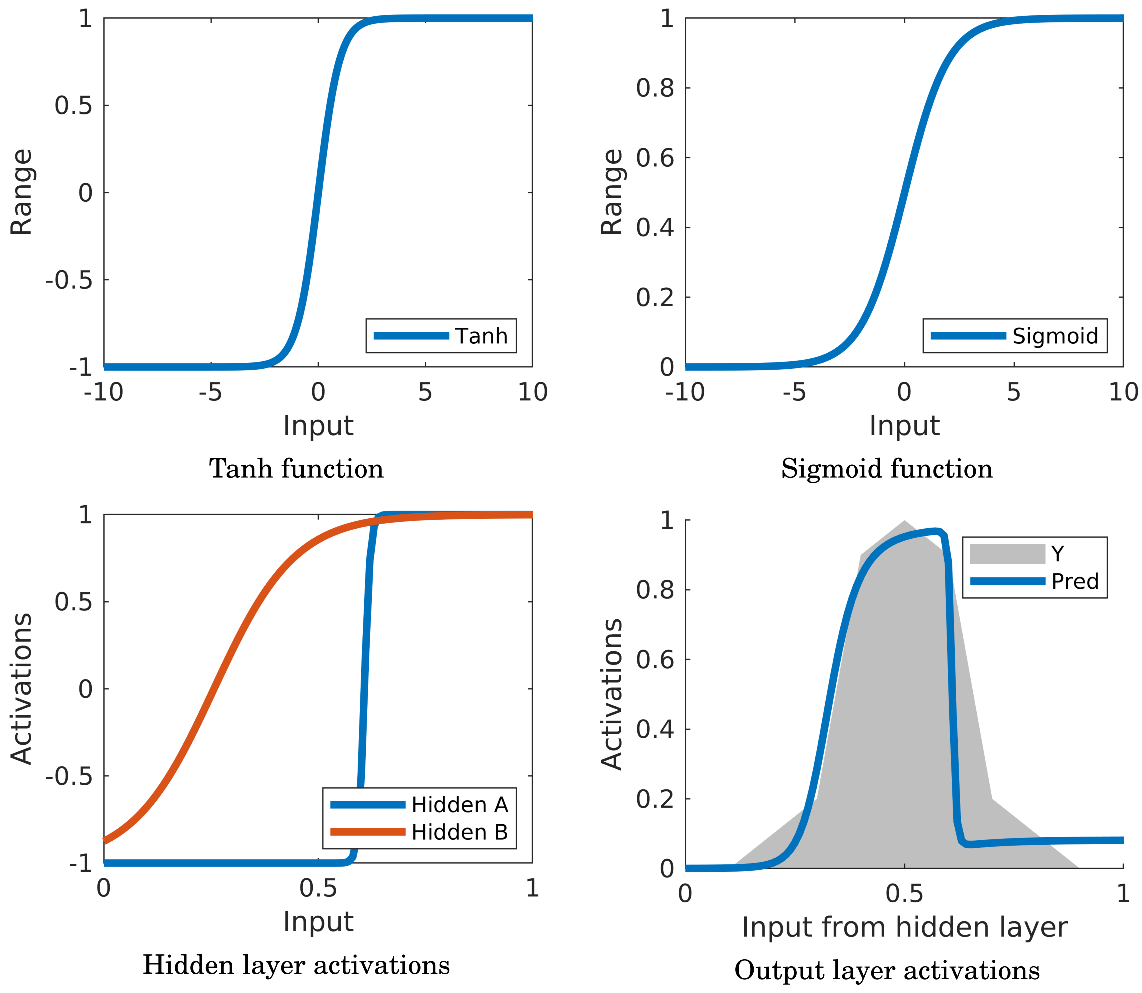


Figure 4.3: Activation functions and how they change depending on the weights and biases of the model. The left hand side shows the hidden layer tanh activations whose output are subsequently merged, via the dot product, before being applied to the output node's sigmoid activation.

The receptive field can be considered as a window which is moved across the entire input of the layer, revealing local neighbourhood features. The amount of movement is determined by the layer's stride in a chosen dimension. One advantage of this approach, over conventional fully connected models, is that the network can apply the same feature detector across the entire input of the layer, significantly reducing the number of parameters to learn.

The kernel is a set of, potentially multiple dimensional, learnt weights which are multiplied with the visual input data accessible through the receptive field window before being added. The dot product computes the input's similarity to the feature. As with the basic network discussed in subsection 4.1.1, the resultant scalar value is applied to a non-linear activation function. A commonly used activation function is the aforementioned ReLU function (see equation 4.3). ReLU ensures that the lower bound is zero whilst applying no upper limit. This means that high values of the receptive field window and a kernel, which are strongly correlated, results in strong neuron activation. Whilst there are other correlations, such as both the receptive field window and kernel being zero, it does not result in a strong activation.

For efficiency in the network, the feature maps can be sub-sampled before being passed up through to the next layer. The primary objectives of the sub-sampling layer are the reduction in dimensionality in order to reduce the model's complexity, the provision of a routing mechanism to subsequent layers and to encourage spatial invariance. Whilst spatial invariance assists the classification of artefacts which have been transformed to a common orientation, it does this at the loss of spatial localisation accuracy [113]. Furthermore, sub-sampling only reduces the number of neurons to route upwards in a brute force manner i.e. it does not intelligently select which information should proceed to an upper layer. Sub-sampling can be achieved using techniques such as max-pooling which selects the most active neuron within a predefined area e.g. a $2 \times 2$ area where max-pooling selects the most active out of four possibilities and results in a 75% reduction of the previous convolutional layer's feature map. A similar reduction could also be achieved by setting the stride value of the previous convolutional layer to the value of 2 for both the $x$ and $y$ axes but without taking the loudest neuron.

As with the basic network in subsection 4.1.1, the weights of the kernels are continually adjusted using a gradient descent technique during training [58]. This is performed to minimise a loss or objective function suitable to the task at hand. One drawback of utilising gradient descent, especially in deep networks, is the phenomena of vanishing and exploding gradients at the lower layers which restricts the network's ability to find optimal weights quickly, if at all [34][pg.282]. Vanishing gradients are when the derivative to apply to the weights becomes so small that converging to a global minimum can take a long time; whereas, exploding gradients have large derivative values giving rise to coarse movements which can step over the optimal solution and never converge to an adequate solution.

Choice of a suitable activation function can mitigate this problem to some degree. Consider the upper and lower bounds of the sigmoid function show in 4.3. The function is designed to squash values between zero and one but can saturate at the bounds when the input values are large, in either direction. The derivative of the sigmoid function is constrained between 0.25 in the center, and 0 at the extremes. This means that the derivative of the function, especially when it is saturated, will be near zero for most changes in input value. This causes a cascading affect, within large sigmoid based networks, where small derivatives are multiplied backwards, as part of the chain rule, towards the lower layers. These multiplications become so small that the effects of applying the derivative to the weights is negligible. A better choice may be the ReLU activation function as it does not squash the values in the upper bound. Its derivative is either 1 for a positive value or 0 for a negative value. However, it does suffer the dead ReLU problem when the derivative of the activation function becomes zero. In this case, no derivative changes can be applied to the lower layers, so no learning occurs. Other variants of ReLU, such a leaky ReLU, resolve this by scaling negative numbers by a small number $\alpha$ instead of setting the output to zero. The derivative value for the negative case is then the scaling factor $\alpha$.

Normalising the original input and using batch normalisation within the layers further remedies the vanishing/exploding gradient problem because it range bounds the input value so that saturation does not occur. Batch normalisation aims to reduce internal covariate shift. This is a phenomenon where the input distributions of hidden layers change depending on previous layers activations [47]. Normalisation within the hidden layers ensures the same distribution across all layers and provides stability across the network e.g. zero mean and unit variance.

The fundamental architecture of a CNN is based on the LeNet 5 network [58]. Modern day CNNs comprise many more convolutional layers, some or no feature map sub-sampling layers together with a number of fully connected layers prior to the final classification layer. This typically yields a probability distribution over the expected classes. This is achieved by applying activation functions such as softmax which is defined as

$$f(\vec{x}_j) = \frac{e^{\vec{x}_j}}{\sum_{k=1}^{K} e^{\vec{x}_k}} \text{ for } j = 1, \cdots, K \tag{4.10}$$

where the function $f(\vec{x}_j)$ yields the exponential value for $x_j$ which has been normalised against the entire set of outputs creating a probability distribution for subsequent ‘one hot’ classification. Figure 4.4 gives an illustration of the architecture of a CNN. It shows the input pixel intensities as an image, a series of convolutional layers combined with adjacent sub-sampling pooling layers and a final fully connected layer preceding the classification output.

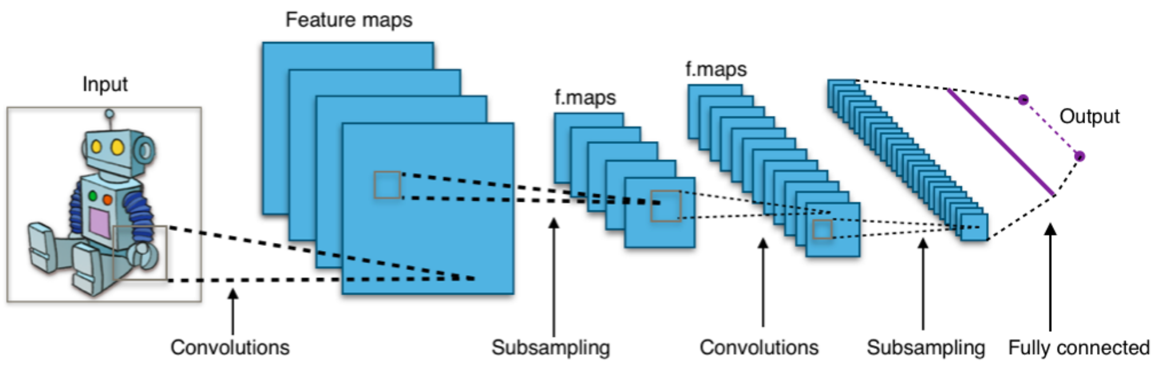


Figure 4.4: An example of the architecture of a CNN showing the convolutional, sub-sampling and fully connected layers. [8].

Krizhevsky et al [56] achieved significant improvement against the then state-of-the-art with their CNN model called AlexNet. This was achieved by the utilisation of newly available large datasets such as ImageNet [92] which helped training, together with multiple GPUs capable of processing the model’s 650,000 neurons. The model was distributed over five convolutional layers with three fully connected layers at the top. Unlike LeNet5, the convolutional layers of

AlexNet did not always have an associated sub-sampling layer. Whilst the dataset was already large, additional training data was created on the basis that more training samples reduced over-fitting. The additional data was generated by utilising data augmentation techniques including label-preserving transformations and the modification of pixel intensities using the components discovered from PCA. Additionally, it was found that over-lapping the pooling areas prevented further over-fitting. However, a substantial reduction in over-fitting was observed when they utilised a technique called Dropout. Dropout is a regularisation technique that is achieved by stochastically disabling neurons and their connections during optimisation [102] removing the model's ability to get fixated upon specific features. The affected neurons are subsequently re-enabled for testing making sure the resulting scale-up in output magnitude is compensated for.

#### 4.1.2.1 Visualisation

Visualisation of the network at different layers can help understand the model and, in turn, accommodate subsequent modification to improve its performance. Zeiler et al [123] employed a deconvolutional network [124] to tease out the features of a given layer. Deconvolutional networks set out to reverse the process performed by a convolutional network, working backwards from the layer in question to the source image. Their work reveals that ascending the CNN increases the field of view and with it features start to emerge which resemble real-world objects e.g. the outline of a dog's head. Figure 4.5 illustrates the detected features from a five layer convolutional network where the top half of the image is the source and the bottom half represents the detected features. The image is a subset of the work taken from Zeiler et al [123].

It was found that lower level features are constructed relatively swiftly during the iterative training process whilst the model needs to be almost fully converged to discover more detailed features. Experiments with scaling, translation and rotation show that low-level features are significantly affected by these transformations; whereas, the more detailed features appear invariant to scaling and translation. These findings reinforce the notion that the sub-sampling layers facilitate spatial invariance and that its impact increases up through subsequent layers of the network. Their experiments also demonstrate that the model is not robust to the effects of rotation unless the initial input is, itself, rotationally symmetrical. The implication is that the CNN requires either a training dataset which includes rotated images or the model's input includes dynamically created transformations in order to be resilient.

#### 4.1.2.2 Adversarial examples

Whilst visualisation via deconvolutional networks gives the impression of similarity between biological and deep learning interpretations of visual information, it has been shown that this is not the case. Adversarial examples created by the addition of specific minor perturbations to the source image, which are almost indistinguishable to the human eye, can significantly alter

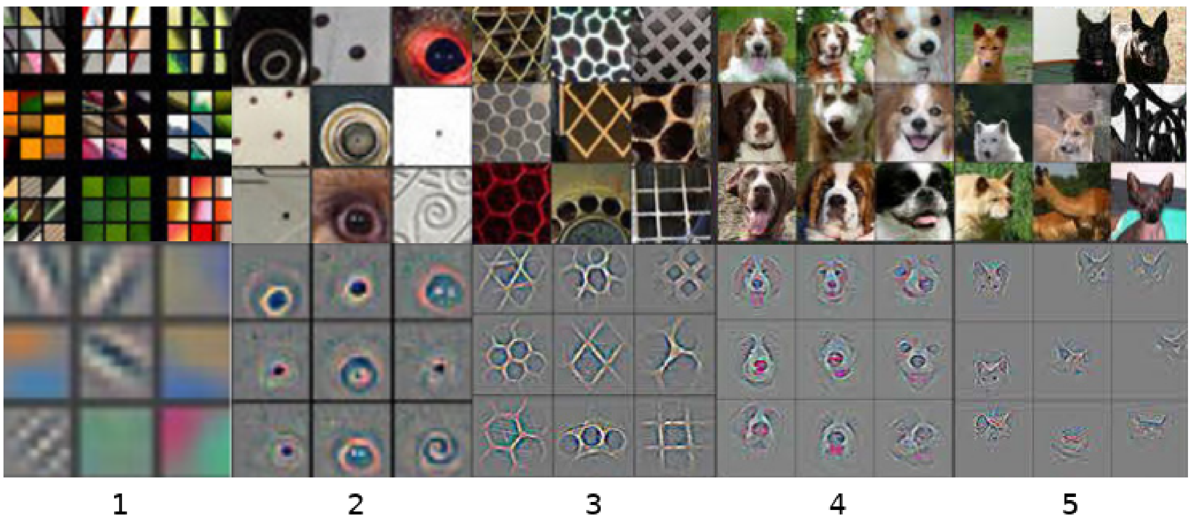


Figure 4.5: Visualisation of each layer of a five layer convolutional network. Layer 1 shows V1 features detected whereas layers 2 through 5 show the top nine activations for the respective layers. The top part of the image is the source, the bottom half are the detected features. Image is cut and paste from Zeiler et al [123]

the classification by finding blind spots in classification space [108]. Figure 4.6 taken from [108] illustrates the subtle differences of an adversarial example over the original. In this instance a school bus, which is correctly classified, is subsequently incorrectly classified as an ostrich after having perturbation included. Krizhevsky's AlexNet model [56] was the target network trained using the ImageNet dataset [92].

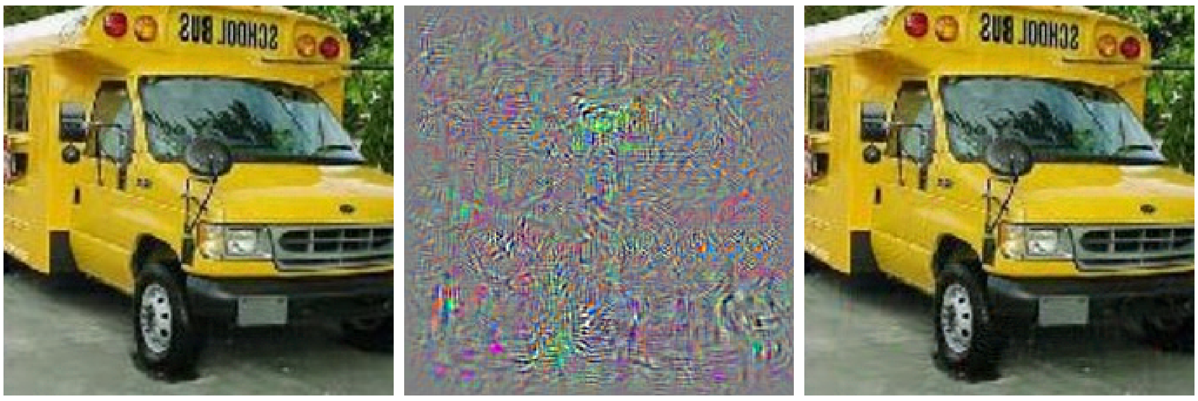


Figure 4.6: An adversarial example showing the correctly classified school bus on the left, the perturbation imposed on the original image and the resultant adversarial example. The adversarial example, shown on the right, is classified as an ostrich [108]. The model used was AlexNet [56]

Goodfellow et al suggests that the underlying linear nature of neural networks explains the susceptibility of the model to adversarial examples [36]. They state that even though the models themselves are non-linear they are designed with linearity in mind, in order to aid optimisation. Their explanation for the occurrence of misclassification is rooted on the idea that infinitesimal adjustments, that are outside of the scope of the the data's normal resolution, can have large impacts when they are added together. They explicitly highlight the vulnerability of utilising the aforementioned softmax regression technique in that respect. An easy way to visualise resolution issues is to consider a set of ten histogram bins that represent a range from 0 to 999 meaning each bin covers a range of 100 values. If $x$ has a value of 99 then it gets placed into the first bin. However, if it is 100 then it shifts to the next bin.

This phenomenon is not limited to deep learning and extends to include several machine learning algorithms such as Support Vector Machines (SVM), decision trees and k-Nearest Neighbour (kNN) [76]. Furthermore, adversarial examples created for a particular model and training set can transfer across to another data domain and/or model providing the classification task is the same. This permits an attacker to infer information using a substitute model without any knowledge of the intended target other than its output labels. The inherent linearity of each model, imperfections within the algorithms themselves and an inadequate lack of a representation of the entire input space play a fundamental role in determining a model's fragility to adversarial attack.

However, data augmentation can be achieved by the generation of adversarial examples and is also found to act as a regularisation technique [108] [36]. The natural extension to adversarial training is the creation of GANs where a model comprises both a discriminative and a generative element. In this instance, the generative network acts as an adversary whose role is to present and fool the discriminator with invented data in a similar fashion to a counterfeiter providing fake products [35]. In essence, the two components are competing players in a minimax game where the intention is to incrementally improve both of them, enabling the discriminator to become proficient at spotting what is real or not, and the generator the ability to create near-real data. This has a large number of applications including the creation of artwork or fake presidential speeches.

### 4.1.3 Deeper convolutional networks

Simonyan et al introduce ConvNet/VGG to demonstrate that a network, based upon similar architecture to AlexNet, could improve accuracy by adding more layers to create a deeper network [97]. In order to maintain computational efficiency they constrain the footprint of their convolutional layers to $3 \times 3$ but show that two adjacent layers, which do not have a pooling layer between them, has a receptive field equivalent to a $5 \times 5$ layer, and three adjacent layers is equivalent to a $7 \times 7$ receptive field. The notable difference, being the addition of activation functions at each layer permitting more discriminative behaviour; and a significant reduction in parameters to learn [97].

Most convolutional network architects focus on choosing their model's receptive field windows e.g $3 \times 3$, $5 \times 5$ convolutional layers. Inception (or GoogLeNet) utilises modules which include a selection of receptive windows and max-pooling functionality whose results are concatenated into the module's output [107]. The architecture utilises $1 \times 1$ kernel filters; not only as an receptive field in itself, but as a mechanism to reduce the dimensionality of the data, permitting the model to have more layers due to the associated reduction of computational expensive parameters. Inception has 22 parameterised layers.

A $1 \times 1$ filter is the equivalent to a pooling technique which affects the number of channels/features discovered instead of reducing the width and height of the data. By placing $1 \times 1$ filters before $3 \times 3$ and $5 \times 5$ filters, the model could yield the same result with fewer parameters to learn. Consider a scenario where a $5 \times 5$ filter is required to output 64 features maps from an input of 192. If a less expensive $1 \times 1$ filter first reduces the feature map dimensionality by half then the $5 \times 5$ filter has less to do. The $1 \times 1$ filter is also used to reduce the dimensionality of any max-pooling within the module.

The architecture also includes probe points within the network. These are similar to the final softmax classification stage and are attached to the some of the hidden layers. This permits the system to assess how well the network is doing at different depths.

Conventional wisdom suggests that deeper networks should have more accuracy, a theory which He et al tested [41]. Inherent in their design was initial input and intermittent normalisation in the form of batch normalisation, which mitigated the vanishing/exploding gradient problem, aggravated further by deeper networks. They found that too many layers significantly worsened the network's accuracy during training. Upon investigation they conclusively ruled out over-fitting and instead discovered a degradation problem creeping in after 18 layers. They resolved this problem by creating Residual Networks which utilise ResNet blocks.

Figure 4.7 illustrates the fundamental design of a ResNet block. In this example, a skip connection from the previous layer is piecewise added to the last convolutional layer in the ResNet, prior to the layer's activation function. The skip connection is not limited to the number of convolutional layers it can jump. Where the network performs dimensionality reduction via a pooling layer, then the skip connection is multiplied against a set of weights to change its dimensionality accordingly. Residual blocks fundamentally change what the affected layers are trying to achieve. Without a skip connection the adjacent convolutional layers are trying to approximate a function H(x). With the skip connection, they are trying to discover the difference (residual) between H(x) and x as shown in equation 4.11.

$$F(x) = H(x) - x \tag{4.11}$$

In simple terms residual blocks are trying to determine what adjustment can be made to x to correct it. Importantly, where no adjustment can be learnt, the residual block's weights tend towards zero giving rise to an equivalence of an identity function. This means that only the value

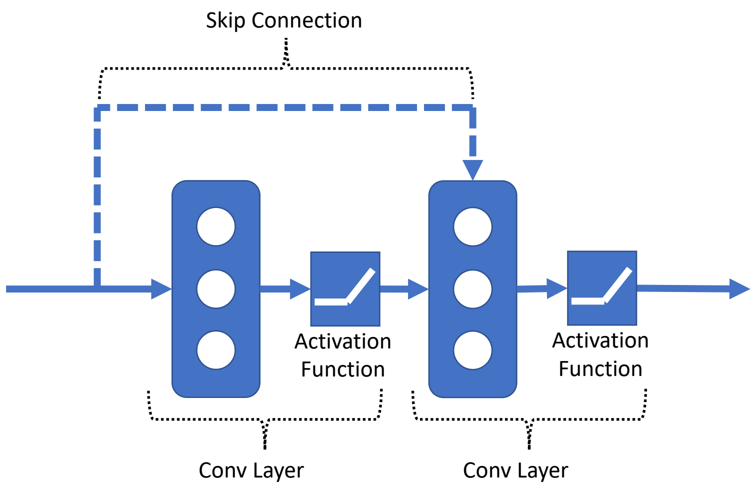


Figure 4.7: An illustration of a two convolutional layer residual network block.

of x in the skip connection is carried forward. This, not only suggests that information can bypass layers that can no longer be trained, but also that the depth of the network is dynamic and is able to fit the training problem accordingly i.e. if the last twenty layers of a network are all the identity function then they have no impact. Residual networks give rise to very deep networks, comprising several hundreds of layers.

#### 4.1.3.1 Object detection and segmentation

Early CNNs focused on giving a single classification to the input. To detect objects, within an environment, would require an iterative sliding window across the entire input, in a similar way to how the Viola Jones algorithm works [114]. This may be computationally acceptable for Haar features and the integral image but inefficient in the context of neural networks. The sliding window approach doesn't consider how the same objects may appear larger or smaller depending on where it is in the image. This would necessitate multiple scans at different window sizes.

Girshick et al solved this problem by using an algorithm called selective search to discover around 2,000 regions of interest of varying sizes and incorporating it into their 'Regions with CNN features' (R-CNN) architecture [32]. In essence, this acts as a filter and vastly reduces the amount of computation required in the neural network. These regions have an affine transformation applied in order to fit the input constraint of the CNN, prior to detection. The actual classification of the model is performed using an SVM. The CNN also learns four offset parameters during training which allows adjustments to pinpoint the exact location of the object relative to the region. This allows the model to place bounding boxes around detected objects.

Whilst R-CNN is an improvement on the sliding window concept it suffers from two major drawbacks. The first being that it can only feed regions of interest into the CNN one at a time, making it slow. To resolve this, Girshick et al developed Fast R-CNN [31] which uses an additional convolutional section to consume the entire image and a set of region proposals. It first creates a feature map before using the region proposals to extract a vector of the same dimensionality for all regions. Each feature vector is then fed into a fully connected network for classification. In essence, the convolutional part of the architecture is performed at the same time for all regions, whereas previously each region would be done separately.

The second drawback was the use of the selective search algorithm. It could not learn, meaning it could continue to present regions of interest that were not good. Furthermore, it was considered relatively slow compared with GPU harnessing CNNs. Faster R-CNN [87] solves this by using a CNN network, known as regional proposal network, to find regions and then using them in the same way as Fast R-CNN to extract fixed-length feature vectors. Mask R-CNN [40] builds upon Faster R-CNN by encouraging the network to predict a mask for the detected object. It follows that good image segmentation can be achieved if you have a good mask of the objects you have detected.

The family of architectures appropriately called YOLO [84] simultaneously discover class and bounding boxes of objects within a global context; whereas, the R-CNN family works with local regions. YOLO does this by dividing the image into a grid prior to presenting it to the CNN. Each grid cell predicts a number of bounding boxes, together with a series of associated confidence values. The cell is considered responsible for an object whose centre falls within it. Confidence is defined by

$$B_c = Pr(Object) \times IoU_{groundtruth} \tag{4.12}$$

where $Pr(Object)$ is the probability of the object being within the bounding box $B$ and confidence is upper bounded by the intersection over union (IoU), the amount of overlap of the object and the bounding box taken from the ground-truth of the training sample. Each cell also predicts a set of probabilities that the object belongs to a particular class. In principle the algorithm is fast because it is dealing with the image simultaneously; but was found to suffer high localised error rates when compared with Fast R-CNN and had low recall [85]. Recall is defined as the number of positive samples found against the total positive samples where high recall implies a high number of positive classifications have been made.

In order to address these issues, YOLOv2 [85] employs batch normalisation, trains its classifier with higher resolution data and moves away from predicting the bounding boxes directly and instead employs anchor boxes. Anchor boxes are predefined bounding boxes based on the context of the training data e.g. based upon every width and height of the bounding boxes contained within the training data. These are chosen by finding the top bounding boxes which best fit the objects. Emphasis was also given to increasing the speed of the model and the underlying architecture was changed from Inception to their own DarkNet, which required fewer floating point operations (FLOP). DarkNet is a deep network comprising, predominately, of a series of $3 \times 3$ convolutional layers. YOLOv3 [86] improves further by modifying the DarkNet to include residual networks and contains other minor improvements; whilst YOLOv4 focuses upon transforming the algorithm, from a theoretical proposition, to a real-time production model by increasing its speed, even when using a single GPU [16]. It achieves this by reconstructing the overall YOLO architecture by selecting the best modules for each role e.g. determining a suitable object detector for speed whilst maintaining best accuracy, experimenting with activation functions, using data augmentation and regularisation techniques.

### 4.1.4 Capsule based architectures

Capsule networks are billed as an evolution of CNNs and were first introduced in the context of autoencoders by Hinton [45]. Sabour, Hinton et al [93] develop the idea further in an effort to address perceived inadequacies within the ubiquitous convolutional network; namely the aforementioned loss of spatial information (invariance) as it progresses through the network, which causes an inability to extrapolate and therefore requiring more training images to compensate i.e. a CNN cannot perform rotations so the model must be trained with an additional set of rotated images. A fundamental difference with the capsule idea is its focus on equivariance. Equivariance permits a set of features to be rotated within the capsule by a learnt rotation parameter i.e. an eye and a nose can be rotated in conjunction with one another to meet an upper layer's expected set of features. The routing of features through the network is also addressed. Sabour et al consider the use of max-pooling within a CNN as inefficient because it simply routes the most active neurons/features to the next layer, which it is then obligated to process, irrespective of the relevancy of the information. Capsule networks resolve this by using a dynamic routing algorithm which routes features of child capsules by agreement with the next layer's parental capsules. The effect is to route only relevant information to the upper layers e.g. an eye detected in a child capsule is routed to a parent capsule that detects faces and not to the capsule of a house.

The model utilises the notion of capsules as an attempt to replicate the biological equivalent of columns. In reality, capsules can be seen as multiple convolutional sub-layers embedded within each layer. Architecturally speaking, the first layer of a capsule network is a regular convolutional layer, exactly the same as that of the CNN. The second layer, known as the primary capsule layer, groups dominant local features from the first layer into capsules. It captures characteristics such as pose (position, size, orientation), deformation, velocity, albedo, hue and texture. Unlike CNNs whose features are treated in isolation and with equality, each capsule stores a vector of closely related features [93].

Sabour et al's initial work [93] cannot use a traditional scalar activation function to represent the capsule's existence and instead opts for employing a squashing function that can work with vectors. This transforms the magnitude of the capsule's feature vector into a value between zero and one, which is then used to indicate the probability that some or all of the features within the capsule exist. Whilst the magnitude is squashed, the vector's orientation is maintained. This is done in order to retain the proportionality of the associated features which have been detected by the capsule.

The squashed activity vector of the capsule is then routed to one or more parental capsules present in the next layer. Initially this output is sent to all parents in order to allow them to evaluate the amount of agreement the child capsule possesses. Hinton's basis for using agreeableness is the belief that coincidences in higher dimensions are improbable. This leads to the notion that features that relate strongly to the current view point of the parent capsule are probably relevant.

Agreement is a reflection of how closely related the feature vector is to a cluster of already existing vectors within the parent and is computed using the dot product

$$a_{ij} = \vec{v}_j \cdot \vec{\hat{u}}_{j|i} \tag{4.13}$$

where $a_{ij}$ is the agreement between the child capsule $i$ and its potential parent $j$. $\vec{v}_j$ is the squashed activity vector for the parent capsule and $\vec{\hat{u}}_{j|i}$ is the child capsule's prediction vector defined as

$$\vec{\hat{u}}_{j|i} = \vec{W}_{ij}\vec{u}_i \tag{4.14}$$

which multiplies the child's output $u_i$ with a learnt weight matrix. The squashed activity vector is defined as

$$\vec{v}_j = \frac{||\vec{s}_j||^2}{1 + ||\vec{s}_j||^2} \frac{\vec{s}_j}{||\vec{s}_j||} \tag{4.15}$$

$$\vec{s}_j = \sum_i c_{ij}\vec{\hat{u}}_{j|i} \tag{4.16}$$

where $\vec{s}_j$ is the summation of the predicted vectors multiplied by a coupling coefficient whose role is to facilitate demand for more from child capsules which agree with the parent and, conversely, less from those that are not in agreement. Dynamic routing is an iterative process which utilises these coupling coefficients in order to create an efficient parse tree over the network. However, the iterative process is computationally expensive during training and suggests that only large networks will gain from the benefits of pruning.

Figure 4.8 illustrates the capsule network architecture. This example has been designed to evaluate the MNIST handwritten digits dataset [57] [93]. A single convolutional layer is first utilised to transform the pixel intensities of the image into a set of 256 feature maps. The feature maps are then passed to the primary capsule layer in order to capture vectors of localised features. Dynamic routing is then applied between the primary capsules and ten digit capsules, each digit capsule representing their respective classification digits. A ‘one hot’ output layer then predicts the classification based upon the probability distribution. Note that the number of dimensions used to describe the local area increases from eight for the parent capsules to sixteen for the digit capsules. This is because it takes more parameters to describe more complicated regions of interest within the capsule.

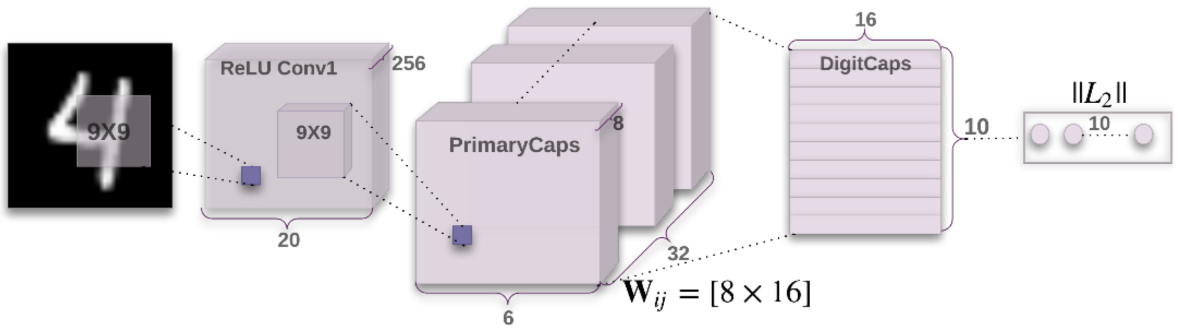


Figure 4.8: An example of the architecture of a shallow capsule network [93].

During training, an attempt is made to recreate the original image using the most likely digit capsule. Reconstruction is achieved by employing a three layer fully connected network. This interprets the capsule’s instantiation parameters, present within its feature vector, into a visual representation. The objective is to minimise the sum of squared euclidean distances between the decoded digit capsule image and the pixel intensities of the source image. The reconstruction loss computed is then fed back into the main network as part of the model’s overall loss function.

Figure 4.9 illustrates the major point behind instantiation parameters and reinforces the inverted graphics nature that capsule networks are trying to achieve. It does this by demonstrating the effect of tweaking six such parameters individually in steps of 0.05 within the range -0.25 to 0.25. For example, the first row can be seen to affect the scale and thickness of the digit by visually showing an increase in both. It should be noted that all instantiation parameters are learnt and cannot be guaranteed to represent the same set of characteristics across different training sessions.

Subsequent work by Hinton et al [44] replaces the squashed activity vector with a pose matrix and a sigmoid activation function which represents the capsule’s activation probability. Replacing the squashed vector with the logistic function allows for the use of an objective function in the routing algorithm and greater efficiencies. Dynamic routing is achieved by the utilisation of Expectation Maximisation (EM) which iteratively determines the mean and variance for a set of Gaussians contained within the parent capsule. Each Gaussian represents a different characteristic from the pose matrix. The data points within the Gaussians are from a single dimension of the vectorised version of the product between each active child capsule’s pose matrix and an associated transformation matrix. The use of Gaussians allows for distinctions such as

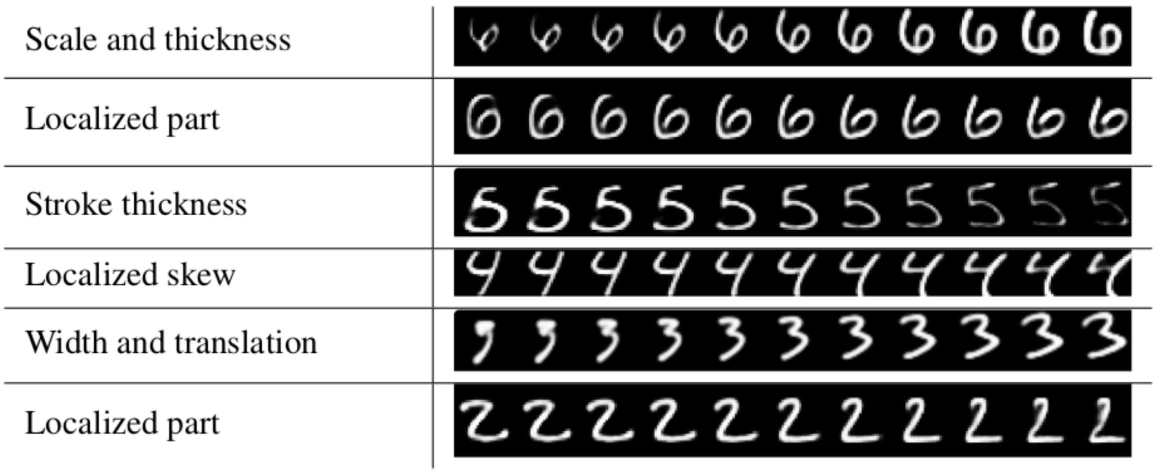


Figure 4.9: A demonstration of tweaking six instantiation parameters in steps of 0.05 over a range between -0.25 and 0.25. Examples include effects on skew, thickness, width and translation. Image taken from [93].

between good and very good agreement between the capsules; whereas, the previous method, of cosine angle distance measuring between two pose vectors, saturated at 1 and was therefore less sensitive at the extremes. Ultimately, EM provides a good mechanism for the parent capsule to explain its data. Furthermore, their later work does not attempt to reconstruct the image to measure its loss. It is likely that this is because the datasets possess more complicated samples than the original MNIST digits.

For dynamic routing, the mean for each characteristic of the pose matrix is computed by

$$\forall h : \mu_j^h = \frac{\sum_i R_{ij} V_{ij}^h}{\sum_i R_{ij}} \tag{4.17}$$

where $h$ represents one of the dimensions from within the vectorised pose matrix computation. $R_{ij}$ is computed for each parental capsule $j$ during the E-step procedure of the EM routing algorithm and is the product of the activation probability $a_j$ computed from a logistic function and its likelihood normalised, as a sum-to-one probability distribution across all capsules of the layer. It is initialised as a proportion of the number of capsules within the layer because the M-step is executed prior to the E-step on the basis that the activation probabilities are already established in the lower child capsules. The weighting of the activation probability $a_i$ for each child capsule $i$ is then applied to $R_{ij}$ accordingly during the M-step. The sum-to-one probability distribution calculation is defined as

$$\forall j \in \Omega_{L+1} : R_{ij} = \frac{\vec{a}_j \vec{p}_j}{\sum_{k \in \Omega_{L+1}} \vec{a}_k \vec{p}_k} \tag{4.18}$$

where the Gaussian probability density function $\vec{p}_j$ is

$$\forall j \in \Omega_{L+1} : \vec{p}_j = \frac{1}{\sqrt{\prod_h^H 2\pi(\sigma_j^h)^2}} exp\Big(-\sum_h^H \frac{(V_{ij}^h - \vec{\mu}_j^h)^2}{2(\sigma_j^h)^2}\Big) \tag{4.19}$$

Similarly the variance is computed by

$$\forall h : (\sigma_j^h)^2 = \frac{\sum_i R_{ij}(V_{ij}^h - \mu_j^h)^2}{\sum_i R_{ij}} \tag{4.20}$$

The logistic function that yields the activation probability $a_j$ for the capsule is defined as

$$a_j = logistic\big(\lambda\big(\beta_a - \sum_h cost^h\big)\big) \tag{4.21}$$

where $\lambda$ is an inverse temperature parameter that increases per iteration of the algorithm and $\beta_a$ is the cost associated with computing the mean and variance of an active capsule. The logistic or sigmoid function is defined in section 4.1.1, equation 4.1. The cost function

$$cost^h = \big(\beta_u + log\big(\sigma_j^h\big)\big)\sum_i R_{ij} \tag{4.22}$$

is the default cost associated with assessing the child capsules where $\beta_u$ is the cost of describing the poses of the child capsules to the parent. Both $\beta_a$ and $\beta_u$ are learnt.

## 4.2 Comparison between CNN and capsule networks

The performance of two deep learning architectures, a convolutional and a capsule network have been evaluated. The motivation was to assess, in the context of terrain classification, claims that capsule networks require less training data because of the inherent equivariance within the architecture; and that they are designed to handle highly segmented data [93] by their understanding of the part-whole relationship e.g. two eyes, a nose and mouth all belong to a face, but you do not need all of these attributes to detect one. Experimentation involved using vertical aerial photography, orthogonally projected LIDAR for height information, and a combination of both. It was novel in that it used LIDAR as a source input for the capsule network permitting red, green, blue, depth (RGB-D) analysis. To facilitate this, a relatively small dataset of approximately 80,000 samples was created using a segmentation algorithm. It focused upon a $4.5km^2$ area of Bristol, UK which had accompanying ground truth classification data labelled manually.

Both evaluated models possessed only a few layers and are therefore relatively shallow by today's standards. It is important to be clear that the models compared are near replicas of previously published networks adopted from the work of Sabour et al's dynamic routing

| **Capsules** | | |
|---|---|---|
| **Layer** | **Capsules** | **Features per capsule** |
| Conv1 | | (256, Stride=1 with VALID padding) |
| Primary | 64 | 8 |
| Digit | 5 or 10 | 16 |

Table 4.1: Capsule network architectural overview

| **CNN** | | | |
|---|---|---|---|
| **Layer** | **Features** | **Kernel** | 5 |
| Conv1 | 256 | **Stride** | 1 |
| Conv2 | 256 | **Padding** | SAME |
| Conv3 | 128 | **Activation** | ReLU |
| FullyConnected1 | 328 | **Softmax** | 5 or 10 |
| FullyConnected2 | 192 | **Loss** | Cross Entropy |

Table 4.2: An overview of the architecture employed for the CNN model

paper [93], hence their design. This is done in order to make an objective comparison of how capsule networks behave with different sources of data without stepping too far away from the original model or datasets. Experimentation extended to include the CIFAR-10 dataset [55] which was used by Sabour et al; this was used as a benchmark.

### 4.2.1 Methodology

Three varieties of experimentation were performed using i) vertical aerial photography, ii) orthogonally projected LIDAR data and iii) a combination of both. Figure 4.10 gives an overview of the end-to-end process used for experimentation. The design takes input data in the form of both RGB imagery and projected LIDAR data. It employs Simple Linear Iterative Clustering Superpixels (SLIC) in order to find regions of interest, which are then labelled and added to a dataset. SLIC was used for its simplicity but a better approach may have been to consider utilising CNN based architectures as introduced in Secion 4.1.3.1. The dataset is then used to train models based upon CNN and capsule network architectures.

The capsule network architecture used for these experiments consists of a single convolutional layer which is used to transform the image from the pixel domain to 256 feature maps, a primary capsule layer with 32 capsules formed from grouping the features from the convolutional layer (8 features per capsule) and a classification capsule layer which was reduced from ten to five classifications (buildings, trees, open land, water and other) when tested with the Bristol dataset. The architecture is defined within Sabour's paper with the actual implementation based upon code samples taken from Géron [30] and an outline is given in Table 4.1. The network was further modified, prior to experimentation, to incorporate 64 primary capsules to cater for the extra

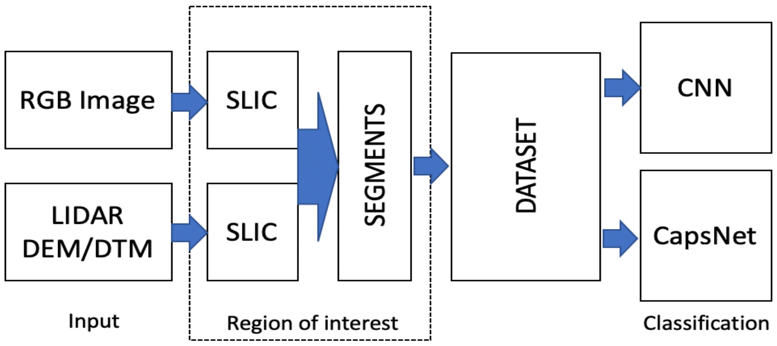


Figure 4.10: Block diagram showing the process to find regions of interest from both RGB and LIDAR sources of data before being added to a dataset suitable for classification by the two architectures.

dimensions of the RGB-D source data [93]. A second capsule network was created which added an extra convolutional layer prior to the primary capsule layer. The objective of this was to test the impact of better defined features upon the capsules and is based on the experimentation of Xi et al [120].

A standard, albeit shallow, CNN was used as the comparison to the capsule network models. Its architecture is a close replica of the network employed by Sabour et al [93] as a baseline for their experimentation of the MNIST dataset [57]. An overview of its architecture is given in Table 4.2 showing 3 convolution layers followed by 2 fully connected layers prior to a softmax classification. The model was modified to cater for the five classifications of the Bristol dataset (CIFAR had ten) and also includes options to use i) Dropout, ii) L2 norm regularisation or iii) bypass regularisation completely.

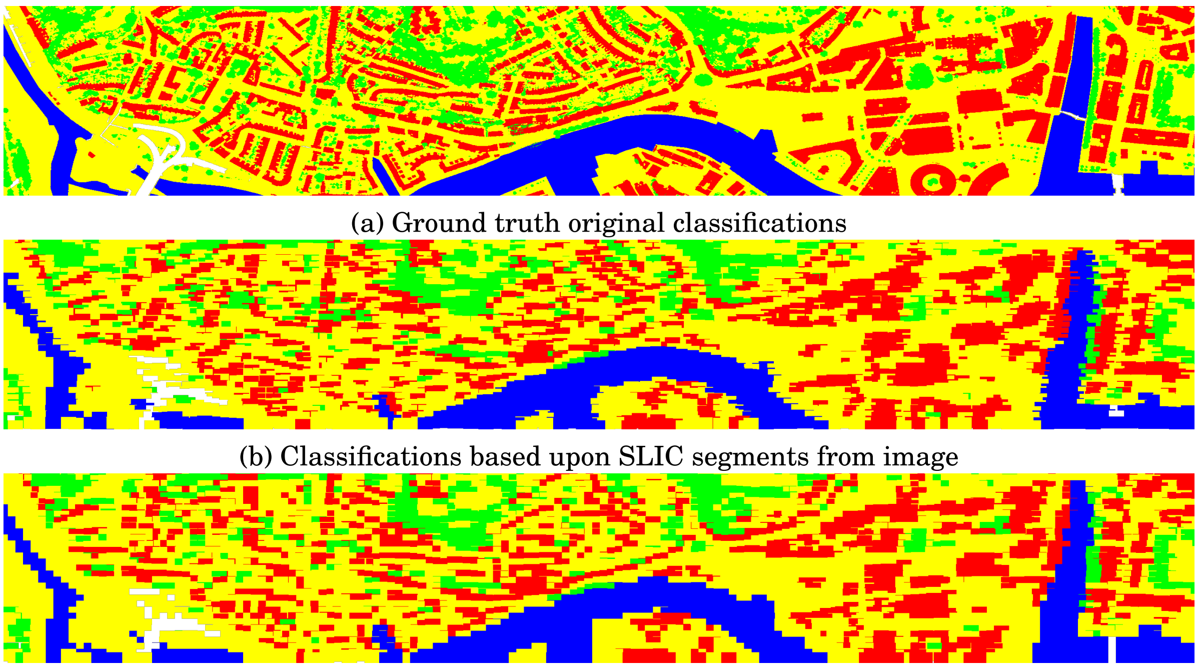
In total five models were evaluated i) the original capsule network as described by Sabour er al, ii) the capsule network with an additional convolutional layer to create a hybrid, iii) a CNN with no regularisation, iv) a CNN with Dropout and v) a CNN with L2 regularisation. As a benchmark, these models were also trained using the CIFAR-10 dataset which did not include depth information. Hold-out validation was used for all experiments to assess how well the models were fitting.

#### 4.2.1.1 Ground truth dataset preparation

A dataset was created using a one metre resolution vertical aerial photograph together with the corresponding LIDAR. It was labelled using ground truth data taken from the University of Bristol's (UoB) Prophecy deterministic RF propagation modelling tool [80] which included hand-labelled clutter classification, also in $1m^2$ segments. The data from the varying sources was aligned using the British National Grid (OSGB 1936) spatial reference and height was computed as the difference between the DTM and DEM data files provided from the LI-

DAR. The target location forms an approximate $4.5km^2$ rectangular area focused upon the UoB ($51^\circ 27' 30.3N, 2^\circ 36' 13.5W$) within the United Kingdom (UK) extending to incorporate well known local locations such as Bristol Zoo, Queen Square and Cotham School.

SLIC [6] was utilised to provide segmentation and applied independently to both the aerial photograph and height components creating two, potentially overlapping, lists of approximately 25,000 segments each which were subsequently merged. For every segment, the image and height map were clipped as $28 \times 28$ metre (pixel) samples in order to meet the input requirements of the models. The resultant sample was labelled using the mode of the associated clutter classifications which was provided as ground truth. To bias the classification towards the centre point of the sample, a filter was employed to capture only classifications from a centred $14 \times 14$ metre (pixel) overlay, with the remaining classifications discarded as background. Figure 4.2.1.1 illustrates the classifications of the SLIC generated samples versus the original ground truth classifications, showing imperfect but significant overlap.


(a) Ground truth original classifications

(b) Classifications based upon SLIC segments from image

(c) Classifications based upon SLIC segments from height data

Figure 4.11: Mode classifications (Key: Red=Buildings, Green=Foliage, Yellow=Open Areas, Blue=Water, White=Other)

Table 4.3 gives the classification breakdown of the circa 86,000 samples created for the Bristol dataset by the SLIC process previously defined. The table includes the approximate percentile split across the training, validation and test partitions. The training partition was formed by taking two out of every three segments, leaving the remaining third segment to form the test partition. The validation set represents a random sampling of 10% of the training data. Data augmentation (vertical and horizontal flipping) was performed to address the disproportional

| **Class** | Building | Foliage | Open Areas | Water | Other | % |
|---|---|---|---|---|---|---|
| **Training** | 16980 | 15778 | 15766 | 3014 | 212 | 60% |
| **Validation** | 1743 | 1828 | 2167 | 0 | 12 | 7% |
| **Test** | 9409 | 8855 | 8993 | 1531 | 112 | 33% |
| **Total** | 28132 | 26461 | 26926 | 4545 | 336 | |

Table 4.3: A breakdown of the distribution of the samples within the dataset post data augmentation.

classification of the 'open areas' class which appears to be a catch-all in the ground-truth. As table 4.3 demonstrates, the augmentation effectively balanced three classes but was unable to significantly improve the under-represented classes e.g. water and other. The number of samples in the Bristol based dataset was also limited. This was done in order for it to be comparable in size to the CIFAR-10 dataset.

### 4.2.2 Results and discussion

The results from both types of architecture are presented here. The parameters of the architectures are given in table 4.2. In total, five models were evaluated which included the original capsule network, a series of similar sized CNNs which focused upon regularisation techniques and a hybrid capsule network which had an additional convolutional layer. These models were tested using four sources of data, three from the Bristol dataset (image-only, height-only, image and height combined) and CIFAR-10 (image-only). Figure 4.12 shows the accuracy (%) of the training (left hand side) and testing (right hand side) of the models, having consumed the aforementioned data sources. Table 4.2.2 gives example accuracy (%) values during testing including the first and last epoch of the series.

The accuracy during training appears to demonstrate reasonable convergence for most models, with accuracy reported above 98% by epoch 200. This is with the exception of two specific cases of the CNN which utilised L2 regularisation (shown in bold in Table 4.2.2). This model was unable to remove under fitting issue when using image-only data for both the Bristol and CIFAR-10 datasets. Consequently, this led to poor performance during the subsequent testing for the L2 regularisation model which experienced significantly less accuracy than its peers. An explanation for the behaviour of the L2 regularisation results can be found in the choice of hyper-parameter scale used i.e the regularisation parameter selected was too large and had the effect of causing under-fitting in the model by over-regularisation.

The testing phase results can be broken into two main data source types, those with LIDAR and those without. In the case of the image-only experimentation, it is apparent that, relative to the other models (excluding the aforementioned L2 model), the original capsule network does not perform well, with accuracy values of 51.92% and 42.89% for both the Bristol dataset and CIFAR-10 respectively. This compares to equivalent test results of 72.33% and 53.03% when evaluating

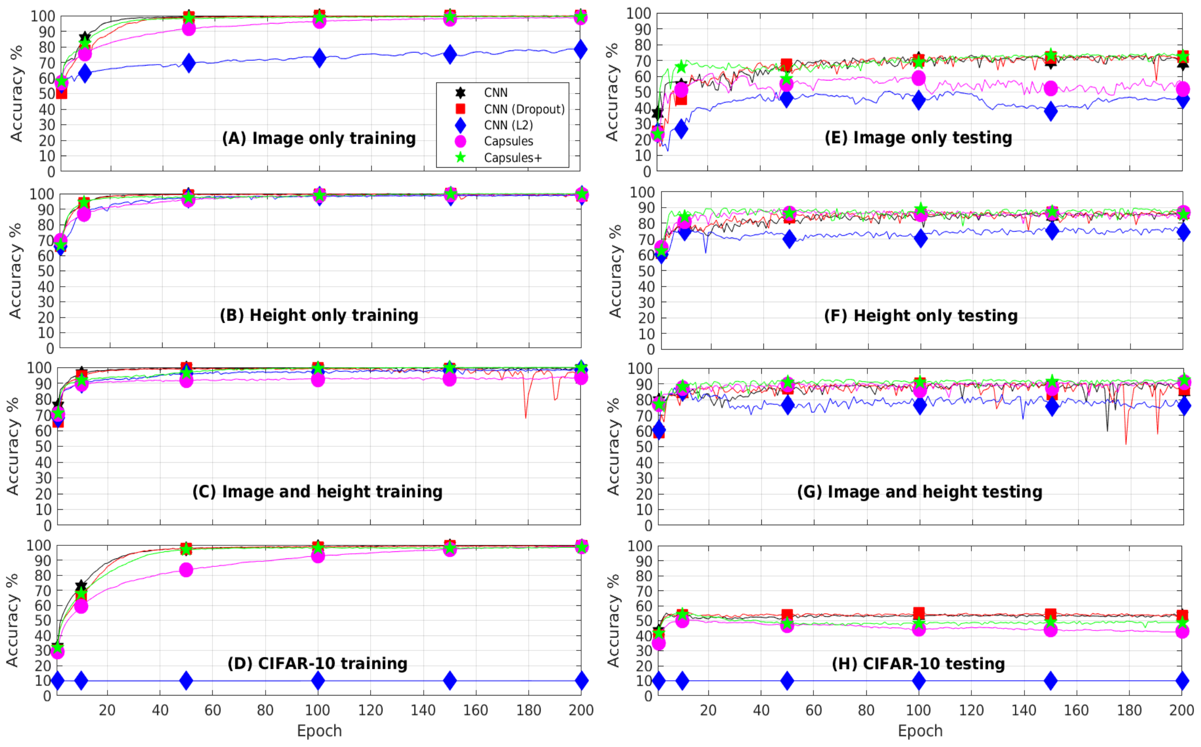


Figure 4.12: Illustration of accuracy (%) during training (figures A to D) and testing (figures E to H) when evaluating the Bristol (image-only, height-only and both image and height) and CIFAR-10 (image-only) datasets. Models include CNN (No regularisation, Dropout and L2 Norm) and two capsule networks.

the CNN model which employed Dropout. Unlike the L2 model, this cannot be explained by the model's under fitting during training and suggests that the capsule network is over fitting and requires more input. The CIFAR-10 result is especially surprising given the earlier claim of accuracy of 89.4% by Sabour et al; although the latter used data augmentation to increase the number of samples and it used an ensemble of seven networks. The modified capsule network performed better than the original capsule network and achieved similar results to the best CNN model (Dropout) with an error loss of 72.50% (Bristol) and 48.45% (CIFAR). The modified capsule is different for two reasons i) it has an additional CNN layer, ii) by implication of the first point, it is slightly deeper.

The results when evaluating with the addition of LIDAR data show significant improvement over the experiments which solely used imagery. Again, the CNN with L2 regularisation performed worst, achieving circa 75% accuracy, with some evidence of over fitting in the test phase. The additional height information appeared to prevent the under fitting problem in training, highlighting the model's sensitivity to the L2 scale hyper-parameter. All other models achieved an accuracy of circa 86% when using height-only data; with the capsule networks able to exploit the combination of image and height data to outperform the CNN models with an accuracy of 90.74% (original capsule) and 92.23% (modified capsule), versus circa 85% for the other CNNs

| | **Training** | | | **Testing** | | |
|---|---|---|---|---|---|---|
| **Epoch** | 1 | 100 | 200 | 1 | 100 | 200 |
| **Image only** | | | | | | |
| CNN | 56.69% | 99.62% | 99.86% | 36.49% | 71.15% | 68.11% |
| CNN (dropout) | 50.27% | 99.55% | 99.84% | 24.87% | 72.02% | 72.33% |
| CNN (L2) | **56.32%** | **72.65%** | **78.58%** | **24.24%** | **44.94%** | **45.75%** |
| Capsules | 56.93% | 96.47% | 98.75% | 23.51% | 58.93% | 51.92% |
| Capsules+ | 57.68% | 98.92% | 99.38% | 23.34% | 68.41% | 72.50% |
| **Height only** | | | | | | |
| CNN | 68.10% | 99.77% | 99.84% | 61.06% | 85.52% | 85.42% |
| CNN (dropout) | 66.09% | 99.48% | 99.03% | 61.01% | 86.87% | 86.55% |
| CNN (L2) | 65.93% | 98.48% | 98.70% | 60.25% | 70.64% | 74.55% |
| Capsules | 69.56% | 98.84% | 99.51% | 64.61% | 85.69% | 86.72% |
| Capsules+ | 66.60% | 98.61% | 99.79% | 62.67% | 88.88% | 85.34% |
| **Image & height** | | | | | | |
| CNN | 75.39% | 99.52% | 99.46% | 79.24% | 88.60% | 84.72% |
| CNN (dropout) | 65.96% | 99.15% | 96.62% | 59.44% | 90.05% | 86.58% |
| CNN (L2) | 68.30% | 97.11% | 98.56% | 60.63% | 76.70% | 75.88% |
| Capsules | 70.69% | 92.29% | 93.94% | 77.02% | 85.70% | 90.74% |
| Capsules+ | 70.82% | 99.48% | 99.82% | 76.89% | 91.12% | 92.23% |
| **CIFAR-10** | | | | | | |
| CNN | 32.15% | 98.93% | 99.50% | 42.90% | 53.15% | 53.34% |
| CNN (dropout) | 30.08% | 98.53% | 99.38% | 40.43% | 54.99% | 53.03% |
| CNN (L2) | **10.03%** | **9.81%** | **9.81%** | **10.00%** | **10.00%** | **10.00%** |
| Capsules | 28.89% | 92.69% | 98.82% | 35.06% | 44.27% | 42.89% |
| Capsules+ | 32.04% | 98.13% | 98.28% | 41.62% | 48.11% | 48.45% |

Table 4.4: Percentage accuracy taken as three epoch snapshots during a) training and b) testing when evaluating the Bristol (image-only, height-only and both image and height) and CIFAR-10 (image-only) datasets. Models include CNN (No regularisation, Dropout and L2 Norm) and two capsule networks. The motivation for using these models is given in section 4.2

excluding L2 regularisation.

The results suggest that features extracted from sources which included LIDAR data played an important role in the overall success of both types of architectures, implying it can more easily be discriminated against. It is easy to see that classifications may lend themselves to this outcome with water forming a stable base height, open areas e.g. grasslands at ground-level and elevation above ground-level for buildings and trees. Furthermore, there is a distinction between man-made and natural artefacts within the environment where natural features are mostly scattered and do not exhibit the regularity of a man-made structure [22]. This texture difference would certainly assist in providing a suitable class boundary between buildings and trees. The fusion of both image and height data provided a modest benefit, especially with the capsule network. This is primarily due to the additional available features based on colour and textural design, which the capsule networks exploited best. Overall, it was encouraging to see that LIDAR data, in isolation, could produce reasonable results, although this may become more challenging as the number of classifications increases, especially if there are similar looking artefacts which can easily be singled out by image data, yet not by LIDAR e.g. a red postbox or black bin of similar height.

The most striking observation was the general poor performance of the image-only sources of input during testing, especially when the original capsule network classified the CIFAR-10 dataset 42.89%, even though the model had been trained reasonably well. This suggested it was over-fitting significantly. Reasons for over fitting include lack of training data and/or the need for regularisation. Regularisation in a capsule network is performed using a reconstruction loss, embedded as part of the architecture. This leaves a question over its performance with less training data; a primary claim of the capsule network, which formed the basis for experimentation.

The architectural design of the models was not arbitrary and were inspired by Sabour et al's dynamic routing paper [93]. This made the CIFAR-10 capsule network results even more surprising, given their claims of accuracy around 89.4%. There are three fundamental differences between the implementation used in this paper and the dynamic routing paper which may explain the discrepancy; namely their use of a none-of-the-above class, additional data augmentation on the CIFAR-10 dataset to increase the number of training samples and the use of a seven network ensemble. Another way to look at such an ensemble is to assume it has some similarity to the idea of Dropout, because it is creating differently weighted nodes, which in itself is another form of regularisation. As stated earlier, the over fitting seen in these results could be the result of a lack of such regularisation suggesting that the reconstruction loss utilised in isolation is not an effective strategy. This may also explain the removal of reconstruction in Hinton's subsequent expectation maximization [44] version of capsule networks; although, this may be more related to the increased complexity of the datasets subsequently used.

It was interesting to see that when an additional convolutional layer was added, its performance behaved more in line with the CNN based models. In all likelihood, the additional layer produced fewer, but more detailed, features which could be better represented by capsules.

Whilst it was not specifically tested for, it was observed that the capsule network models were significantly slower than their CNN counter-parts during training, taking circa 20 times longer, whilst providing only negligible speed improvements during the test phase. One of the fundamental principles behind capsule networks was the idea that pruning the network, by providing dynamic routing, would improve performance. In this instance, the depth of the model was not significant enough to capitalise on this.

## 4.3 Summary

Deep learning forms an important building block in the pursuit of good image recognition, detection and segmentation. This chapter gave an overview of some of the main concepts within the field. It started with the fundamental and biologically inspired neuron presented within the context of an ANN. An overview of Convolutional neural networks (CNNs) was then given, as they form the backbone of many state-of-the-art architectures.

Notions such as the vanishing gradient problem, regularisation and susceptibility to adversarial examples were also discussed. Along with the CNNs inherent reliance upon spatial invariance which restricts the architecture to view the same, but rotated, object as different. Capsule networks were discussed as a potential improvement preferring equivariance over invariance, requiring less training data. The new architecture showcased an overhaul of the routing mechanism which feeds data through the network. It was suggested that this technique is the equivalent of an efficiently pruned tree.

A series of experimentation was then described. These experiments were performed to test claims made in favour of capsule networks over CNNs and were tailored for terrain classification. Capsule networks did not perform as well as expected when evaluated against image-only data, especially when compared with the dropout CNN which outperformed the original capsule network by 40% and 9% for for the Bristol and CIFAR-10 datasets, respectively. Only the modified capsule network matched the dropout CNN, but that was because it had an additional convolutional layer added.

When using LIDAR based data, the models, excluding the L2 regularisation model, achieved good accuracy over 85%. This included both capsule networks, which achieved accuracies of 90.74% (original capsule) and 92.23% (modified capsule). It suggests that the capsule network representation is better able to capture the discriminatory features associated with height.

However, there were significant short-comings in the design of these tests. Constrained by keeping the proposed architectures of the original capsule network paper, the models were shallow and easily superceded by state-of-the-art models. The shallowness meant that the capsule network was not able to capitalise on efficiencies created by its pruning mechanism. Furthermore, the resolution of the data was low quality and the number of unique classes within the training data was too small, meaning that the probability of accidentally selecting the correct classification was relatively high. To compound issues, the wrong hyper-parameter value was selected for L2 regularisation which ultimately led to under fitting during training. However, the evaluation did prove fruitful in some respects. Namely, it provided a good insight into under and over fitting, showed the impact of over-regularisation. They also demonstrated how vertical LIDAR data could be effective in classification due to good height discrimination. Subsequent experiments, detailed in later chapters, learn from the positives and negatives discovered here.

Chapter 5

# Direct point cloud classification and characterisation

Many technologies and applications now necessitate an awareness of their geographical surroundings, typically employing an array of sensors to capture the environment. An autonomous vehicle may utilise high definition cameras, LIDAR and radar; not only for collision avoidance but also for SLAM. Similarly, telecommunication network planning can benefit from the utilisation of RF propagation tools, which utilise these sources of data in order to represent the target deployment environment to a reasonable degree. Mobile LIDAR scanners utilise SLAM in order to map the environment in three dimensions and can even colour the point cloud using an associated camera. The resulting data from such devices is potentially very large, permutation invariant and clustered.

This chapter gives an overview of architectures which are designed to work with LIDAR data. Some architectures achieve this by projecting the point cloud onto a suitable medium and then subsequently use that transformation in a conventional image processing fashion. In others, the data is encoded into another form or the architecture is capable of working directly with the raw data points.

Two sets of experiments have been performed to characterise the importance of the points in the point cloud. This is performed via sub-sampling until the point the model, under examination, breaks down. The experiments also served to determine the trade-off between accuracy and speed of classification. The motivation for these experiments came from the observation that an early direct point cloud architecture, called PointNet, discovers a set of salient points which define the underlying structure of the objects within the dataset [81]; making the architecture robust to data loss. The creators of the architecture referred to them as critical points.

These critical points appear to closely resemble the effect of a spatial filter. If true, then the algorithm itself could potentially become a sub-sampler to reduce the number of input point for other architectures. Due to the manner in which the algorithm could be probed to see what critical points were found, PointNet became the focal point of the first set of experiments. These evaluated how different sub-sampling techniques affected the model. Synthetic data from the ModelNet40 dataset [127] was utilised. This was chosen to avoid any unnecessary complications e.g. the effects of noisy real-world data.

Results from the first set of experiments show that reducing the number of input points did significantly reduce the computational time by up to 90%, whilst still maintaining an accuracy which is within 2% of the original input (87.3%). Crucially, it was found that the original 2,048 points could be reduced to 32 points whilst retaining over 80% mean validation accuracy for the ModelNet40 dataset; although the structure of the dataset may have helped.

The latter section moves to a state-of-the-art CNN architecture using the NPM3D [91] real-world dataset captured in France. A benchmark is established before evaluating the model's robustness to sub-sampling. Subsequent experimentation used an edge extraction technique to determine how the model performed with just edge features. Both sub-sampling and edge only experimentation are shown to exhibit classification problems. Too much sub-sampling was found to render the model useless, with a sudden drop from 83.61% to 6.52% at one stage; too much edge definition appeared to bias the classification in favour of the predominant classes such as building and roads. A combination of sub-sampling and edge feature retention was found to work best, achieving accuracy results of circa 80%, versus 87.34% for the original; whilst using a fraction of the available data points.

## 5.1 From point cloud projection to direct analysis

Technological advances in processing power and memory have led to a relativity recent resurgence in the application of CNNs architectures for object recognition, detection and image segmentation. This includes the creation of architectures such as Mask R-CNN [40], YOLO [84] and their subsequent iterative improvements, which provide excellent state-of-the-art results. This advancement has not easily been transferred to other forms of data, such as LIDAR, whose fundamental properties differ from conventional RGB(D) images. One of the main reasons for this is the fundamental design of the local neighbourhood-based convolutional filters whose strength comes from their ability to exploit the spatial relationship inherent in the homogeneous structure of image data.

Raw LIDAR is typically captured as a point cloud and, whilst it still includes spatial information, the structure of the stored information does not possess the uniformity of image-based data. LIDAR point clouds are three-dimensional, the points can be unordered (meaning that any solution needs to be permutation invariant) and the distribution of points is clustered which, conversely, gives rise to problems with sparsity. Object definition is also not necessarily consistent. For example, a rotating LIDAR scanner will capture more points closer to its origin than at its furthest range. It follows that an object closer to the origin will be better defined simply because it has more measurements. Although, one could argue that this is similar to perspective applied to objects further away in a standard image. Finally, each point has an associated intensity or reflectance value which is unlike the RGB information captured from an image; although each point can be coloured with values captured by an associated camera (although from experience sometimes not wise to assume to system can reliably perform this).

Multi-view CNNs [105] project the point cloud into a series of images which represent different viewing angles. The images possess single channel reflectivity/intensity values, due to the type of data captured by LIDAR. They are processed, as an ensemble, by a regular CNN infrastructure prior to bringing the results together to determine the final classification. In a similar vein, Squeezeseg [119] projects the point cloud onto a two-dimensional spherical surface, as does Rangenet++ [67]. The motivation behind spherical projection is to reflect the aforementioned rotating LIDAR capturing technology which distributes points according to proximity to the measurement device. These models tend to suit their problem domain well. Multi-view CNN is good at recognising objects from different perspectives; whereas, spherical projection models provide real-time object detection crucial for autonomous vehicles as snapshots.

Projections have a number of drawbacks for LIDAR. A Velodyne scanner whose range has been capped at 20 metres in order to achieve a high point density will capture between 1,000 and 2,000 points per square metre [89]. If this was projected onto an area of $100 \times 100$ (representing centimetres) then it would populate no more than 20% of the plane. If the point cloud was extended to include multiple capture locations, as would be required to survey a road, then it is feasible for points behind solid objects to be projected onto the same plane due to the aforementioned sparsity. Retaining the coordinates, which indicate where the device was at the point of capture, would mitigate this but would force any subsequent evaluation to follow the original capture path, treating every capture location independently and losing any freedom to traverse the entire three-dimensional space. Furthermore, projecting onto a smaller plane would result in a loss of information as points would map to the same location.

ShapeNet 3D [127], VoxNet [64] and successors [82] pre-process the point cloud by transforming it into voxels prior to classification. Whilst this does retain the overall three-dimensional shape of the data, information can be lost due to its discretization effect [104] and the overall footprint of the data to be processed can remain large. The former can be resolved if the volumetric frame has sufficient resolution, but this does little to deal with uneven distributions of the points, including any issues surrounding sparsity. Occupancy grids, employed by VoxNet, generate a probabilistic estimation of where to look for objects and to some extent mitigate the latter issue of size by refining the areas of interest. Other issues include the potential trade-off between the size of the receptive field, used by the CNN, and the availability of resources and computation required. Consider that a $30 \times 30 \times 30$ representation of the data requires the same number of weights as a $165 \times 165$ two-dimensional single channel image [127]. One further problem, which applies to all solutions which retain three geometric dimensions, is how to select the appropriate view-point given the overwhelming freedom to choose any angle.

OctNet attempts to address the sizeable foot-print problem of volumetric approaches by the tackling the sparse nature of three-dimensional geometric environments [88]. It does so by binary encoding the voxels into a series of octree structures at different levels of resolution in order to capture the salient details of the data. Octree-based CNNs (O-CNN) [117] takes this further by encoding the normals which lie upon the surface of the object, ignoring any internal points. Both use a three-dimensional convolutional filter over their octree representations. Again, these techniques require either i) pre-processing the point cloud either into voxels or ii) information not necessarily available within the source point cloud e.g. the need to estimate normals in the case of O-CNN.

All the above solutions assume the point cloud data is Euclidean, which is not guaranteed e.g social network data such as from Facebook can be represented as a point cloud and is not Euclidean. They also utilise the extrinsic properties of the data and are therefore sensitive to variants in deformation and pose [17]. Extrinsic utilisation can be viewed as applying a filter some distance away from the source data/image, like a lens; if a feature in the data is deformed then it will not be seen by the filter, which remains intact. Intrinsic utilisation places the filter directly on to the data, allowing the filter to adjust to deformation.

Bruna et al [19] create a spectral analytical based solution which uses the Laplacian-Beltrami operator to find an orthogonal set of basis functions. These functions can be seen as a generalisation of the Fourier series extended to graphs [63]. However, the spectral filters are basis dependent and cannot be applied to different manifolds. Furthermore, without a smoothing spectral transform filter, localisation cannot be guaranteed [21] making it difficult for transfer learning i.e. trained on one shape and used on another. Spectral approaches tend to be computationally expensive because they need to compute the Fourier transform and its inverse in order to traverse from the spatial to frequency domain. This is in addition to performing eigen-decomposition to find the basis functions.

Geodesic CNNs [62] creates intrinsic geodesic patches which are applied to an underlying Riemannian manifold operating within the spatial domain and does not suffer the domain transfer problems associated with spectral analysis. However, this technique is restricted to work with triangular meshes and the patches must be constrained to be small relative to the shape [17]. Anisotropic CNNs [17] create anisotropic heat kernel patches to resolve these issues.

Other similar architectures also adapt their convolutional filters to permit alternative representations of the data. SPLATNet [104] uses a convolutional layer which is a generalisation of bilateral filtering called Bilateral Convolution Layer (BCL). The architecture projects the points onto a permutohedron lattice which is then convolved using a specifically designed filter; but the method does not scale well due to its computational demands [67].

Graph embedding architectures such as [51] and [118] fit an underlying graph structure to the local neighbourhood of points and use an appropriate filter such as edge convolution. These can easily be generalised into other non-geometric spaces such as social media networks. MONet [70] constructs its patches using a parametric approach. It is a generalisation of the aforementioned intrinsic approaches because the parameters selected can form the fixed patches of the other techniques.

One example architecture which has been designed specifically to classify raw point cloud data is PointNet [81]. It is an extreme graph-embedding classification technique which treats each point in the cloud as single node with no edges. It uses a symmetric function (max-pooling) in order to be permutation invariant and finds a global signature as a representation of the entire input data. It employs the use of Spatial Transformer Networks [48] in order to be invariant to transformations performed on the point cloud. Unlike other graph-embedding methods, PointNet discards all spatial relationships between nodes. Whilst it can perform segmentation by merging the global signatures with the local features of each node, it does not handle complicated scenes well [83] [67]. PointNet++ [83] addresses this issue by hierarchically applying PointNet to local clusters of points. Both approaches work well when the points are aligned to the canonical coordinate framework but suffer with real-world environments [95]. 3D Point Capsule Networks adapts PointNet to work within the framework of capsule networks [126].

One of the key elements of the PointNet family of algorithms is their ability to extract features which represent critical points. These points define the important structural information of the object [81]. Superfluous points are simply discarded and, to some extent, PointNet acts as if it was applying a spatial filter prior to classification. One major benefit is that the global signature used for classification is unaffected by losing non-critical points making PointNet robust to missing data [81].

3D modified Fisher Vectors [14] uses a grid of spheres and determines the Gaussian information about where each point resides on a grid relative to each sphere. It defines a Fisher vector of Gaussian information as a global signature and uses a CNN for classification.

KPConv [112] redefines the pixel-based convolutional layer of a CNN into a representation which utilises the proximity of local, radius-based, spherical neighbourhoods of points against the points of a pre-defined set of three dimensional kernel point features. Euclidean distance between the input points and kernel points is used to provide a simple correlation which is defined by

$$h(y_i, \tilde{x}_k) = max(0, 1 - \frac{||y_i - \tilde{x}_k||}{\sigma}) \tag{5.1}$$

where $y_i$ is a neighbouring point to the current focal point $x$, which has been centred using

$$y_i = x_i - x \tag{5.2}$$

and $\tilde{x}_k$ is a location point in the kernel feature with $\sigma$ as the amount of its influence. Influence can be seen as adjusting the gradient of the drop off, from being close to the kernel point or not e.g. a low value of $\sigma$ creates a steep drop meaning that a neighbouring point has to be close to be influenced; conversely, a high value gives the kernel point greater influence. Points which are closer to the kernel point yield higher correlations values. Each neighbouring point is evaluated against all the points within the kernel and the weights are calculated using

$$g(y_i) = \sum_{k<K} h(y_i, \tilde{x}_k) W_k \tag{5.3}$$

which is similar to the basic neuron similarity stage described in section 4.1.1, which uses a linear kernel as opposed to this correlation kernel. The final stage of the kernel point convolutional layer is to perform the convolution against the feature set for all of the points within the receptive field. The major difference between pixel-based and kernel point based convolutions is that kernel point convolution finds a similar representation to pixel intensity using proximity. Kernel point features are pre-defined and constrained to have a number of points; whereas, CNNs typically use adjacent pixels in the receptive field as their definition. Deformation of the kernel points features can be achieved within the network by learning subtle shifts. This incorporates both distance and repulsion regularisation techniques to constrain the points within a certain boundary, whilst preventing them from collapsing in on each other. Deformation, over an attempt to learn kernel point definitions, is preferred in order to limit the number of points. In some ways, this is similar to using anchor boxes discussed in section 4.1.3.1, where computational effort is preserved by not trying to find every solution.

## 5.2 Characterisation of point cloud data using a synthetic dataset

Point cloud data which captures a three-dimensional environment is not guaranteed to be uniformly distributed. This is especially true with data captured using a rotational LIDAR scanner as the density of the points correlates strongly to the proximity of the measuring device. This is because the distance travelled at the circumference is greater than for any inner circle assuming a constant velocity. Consequently, there is less time for sampling at the periphery. Furthermore, point clouds can contain millions of points of data.

Applying a sub-sampling filter, such as Poisson disk, to the point cloud should preserve its integrity whilst removing the higher frequency components nearer the origin of the capturing device and reducing the number of points. This filter works by imposing a minimum radius between points. Any points within that distance are discarded. Therefore, it is important to consider the chosen radius in order to control the amount of points filtered.

Whilst there are many approaches which can be employed for point cloud classification, the authors of PointNet [81] specifically highlight its ability to extract and work with critical points, those which contain the structural information of the objects within the cloud. Whilst PointNet is no longer state-of-the-art, its design makes it easy to view the critical points discovered [106]. This is achieved by noting the final activations which were carried forward to form the global signature and mapping these to the input; made easier due to the choice of using max-pooling as the symmetric function.

The behaviour of PointNet appears similar in nature to a spatial/low-pass filter, one which is capable of preserving important features. Furthermore, both the application of a spatial filter and the extraction of critical points greatly reduce the number of points to work with during the classification stage. If the inner workings of PointNet are similar to a spatial filter, then it is tempting to consider whether this could be incorporated in the early stages of a state-of-the-art architecture to provide the underlying structure with considerably less points. However, as previously stated, PointNet cannot handle complicated scenes demanded by these networks.

By using sub-sampled data as an input into PointNet, it is possible to see whether the sub-sampling techniques preserve the underlying structure sufficiently. If true, sub-sampling would maintain structural integrity, reduce the number of points required and constrain the distribution of the remaining points i.e. removing high frequency components to give a better distribution. The first step would be to feed the discovered critical points of a trained PointNet network into an untrained version and assess the difference. Little difference between results would imply the model is efficient and capable of minimising the number of points required to describe the object. Subsequent experimentation would assess varying degrees of sub-sampling to assess the model and at which point does it fail. The overall objective being to characterise sub-sampling techniques by determining the trade-off between accuracy and computational complexity, and therefore find the best sub-sampling technique from those employed.

### 5.2.1 Experimentation set-up and methodology

For classification training, the PointNet model can be viewed in two parts. The first performs a process similar to an intelligent spatial filter by discovering the salient points critical to the structural integrity of the objects being trained. The second builds a global signature from these points in order to predict the classification of the object.

Based on the hypothesis that the first part of the model is a feature preserving sub-sampler a series of tests were performed on the synthetic dataset using the PointNet model:

- the original point cloud data was used to create a benchmark: the full 2,048 points were employed as the source for each object input to PointNet [**original**].

- the extraction of the critical points from the above benchmark were fed back into PointNet to evaluate the efficiency of the model [**critical**].

- Poisson disk sampling was iteratively applied to the dataset, using an increased radius, to filter a reducing range of points which were then processed with PointNet [**poisson**].

- Voxel grid sampling was iteratively applied to the dataset with an increasing voxel size which resulted in filtering a reducing range of points. These centroid points were then processed with PointNet [**grid**].
- Random sampling was iteratively applied to the dataset taking fewer points to match the proportions of the other sub-samplers. These were then processed with PointNet [**random**].

The sub-sampling tests set out to question i) whether the model could still find the critical points, which preserved the integrity sufficiently enough, for good classification, ii) what computational and resource benefits could be found, iii) how accuracy changed and iv) whether the design of the synthetic dataset made it robust to missing data i.e. is synthetic data too artificial to get meaningful results.

Whilst there are many filtering techniques to consider [38] three sub-sampling algorithms were chosen: i) Poisson disk, ii) Voxel grid and iii) Random. Poisson disk sampling was selected because it closely resembles the distribution observed within the retina [121] [68]. It employs parallel random dart throwing and ensures that no two points are within a radius $r$ from each other [18]. Voxel grid sampling is similar, in that it can filter by area by setting the voxel size parameter accordingly. It differs by retaining the centroid of all the points within the voxel cube, capturing the barycentric property of the local data i.e. the centre of mass. Whilst this point distortion may feel counter-intuitive, a benefit of this approach is it can encode edge features, albeit, in a subtle manner. For example, an evenly distributed flat surface would find the centroid in a central position of the voxel (accepting a flat structure is predominantly two dimensional, so not the literal centre of the x,y,z planes); whereas, an edge may pull away from the centre and reflect the three dimensional pull of the neighbouring points. Finally, random sampling was included in order to give an insight into the effectiveness of the other two sub-samplers e.g. does the rigid distribution of Poisson and deforming nature of Voxel sampling give rise to the loss of important features such as those found at the edges.

To maintain focus upon its ability to find the critical points of an object, the ModelNet40 synthetic dataset [127] was used, ensuring scene simplicity. Whilst this negates the need for a sub-sampler to remove high frequency components due to the distribution of points in real-world data, it is assumed any sub-sampling would have addressed it, regardless of the type of data sourced i.e. the output of good sub-sampled data would look similar to the data which doesn't have any high frequency components to start with. ModelNet40 dataset [127] comprises forty synthetic point-based classification categories with 9,768 training and 2,468 test samples. All experiments were trained and evaluated over 100 epochs.

Each experiment was performed 100 times and all sub-sampling experiments were repeated with a decreasing number of points to determine where, if at all, the classifier was unable to sufficiently discriminate between the classes. In total, the number of points was decreased 13 times, meaning each type of sub-sampling experiment was performed 1,300 times; whereas, the [**original**] and [**critical**] experiments were each performed 100 times.

The points for each object within the dataset were sub-sampled prior to being input into the model. The only exception being when the model was completely trained, and had had the critical points of all objects extracted to become a new dataset. All sub-sampling experimentation was primed to filter to the approximate the number of critical points extracted from the [**original**] experiment.

The experiments were performed on a single GPU node on the BlueCrystal phase 4 supercomputer hosted at the University of Bristol, UK. Each GPU node is a Tesla P100-PCIE-16GB capable 21 teraFLOPS of 16-bit floating-point (FP16) performance [3].

### 5.2.2 Results and discussion

Five experiments were performed to test how pre-processing a point cloud using a spatial filter would affect the accuracy and computational performance of PointNet. The first experiment [**original**] was a benchmark and used an unfiltered point cloud as an input. This experiment consumed the full 2,048 points per object. Of these points, less than 10% were found to be, what PointNet would consider as, critical. Across all ModelNet40 objects, the [**original**] experiment had an average validation accuracy of 87.3%. All results are shown in Table 5.1 and illustrated in Figure 5.1. This benchmark result is encouraging because it is comparable with the accuracy result of 89.2% achieved by Qi et al [81] who evaluate PointNet using 1,024 points per ModelNet40 object.

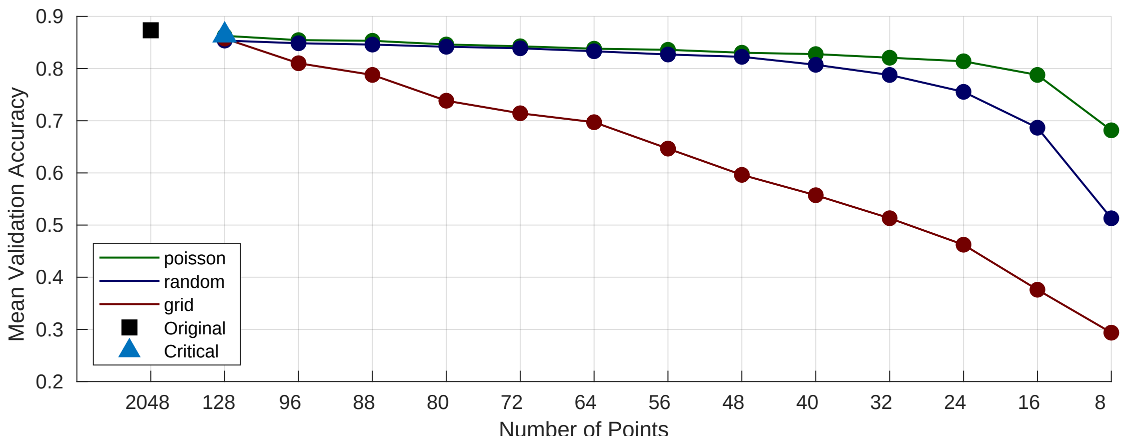


Figure 5.1: The mean validation accuracy results for each different type of experiment including the reduction of sub-sampled points.

A breakdown of the number of critical points found per class from the [**original**] experiment, relative to the number extracted by the initial Poisson disk sampling experiment, can be seen in Figure 5.2. On average the number of critical points found across all object classes was 122 points and this was used as the initial target number of points to filter to, aligned to 128 points

($2^7$) for base 2 convenience. The figure also shows an average of 124 Poisson points per object class, where the radius parameter was selected to be as close to the number of critical points.

Any sub-sampling which resulted in the number of points being higher than the required input points for an object was capped and, likewise, any below the required amount made up to the required number of points with zeroes. Whilst this is not ideal, it constrained the input to the same size, a requirement of the model. Figure 5.3 shows the plots of four sample objects showing a) the original 2,048 dense point cloud and b) the point cloud after Poisson spatial filtering was applied. The radius selected reflects the desired number of points required in order to approximate the actual number of critical points found by the [**original**] experiment.

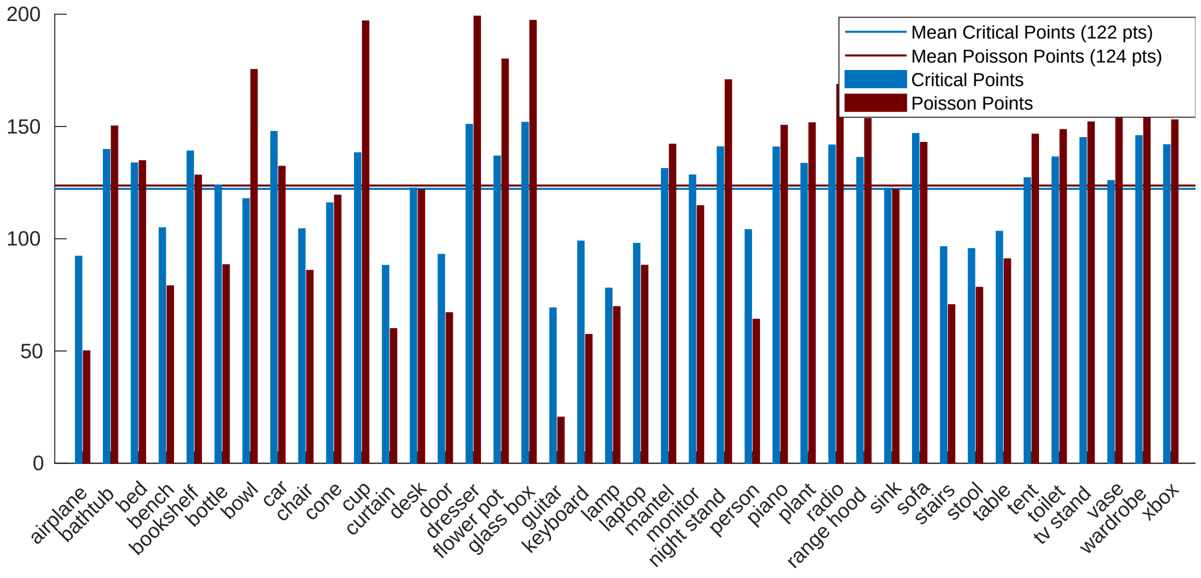


Figure 5.2: A chart showing the relationship between the number of critical points extracted from the unmodified original experiment and Poisson disk sampling points on a class-by-class basis.

The [**critical**] experiment was used to observe the efficiency of the PointNet model. The hypothesis being that if the importance of the critical points was high then the model should return the same set of points. Whilst reasonable, the average validation accuracy for this test was

Table 5.1: Breakdown of the final epoch for each type of experiment showing mean validation accuracy as a percentage.

| | **Number of Points** | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **2048** | **128** | **96** | **88** | **80** | **72** | **64** | **56** | **48** | **40** | **32** | **24** | **16** | **8** |
| **original** | 87.3 | | | | | | | | | | | | | |
| **critical** | - | 86.3 | | | | | | | | | | | | |
| **poisson** | - | 86.2 | 85.4 | 85.3 | 84.6 | 84.2 | 83.8 | 83.6 | 83.0 | 82.7 | 82.0 | 81.3 | 78.7 | 68.1 |
| **grid** | - | 85.7 | 81.0 | 78.7 | 73.8 | 71.4 | 69.7 | 64.6 | 59.6 | 55.7 | 51.3 | 46.2 | 37.6 | 29.3 |
| **random** | - | 85.3 | 84.8 | 84.6 | 84.1 | 83.8 | 83.3 | 82.7 | 82.2 | 80.7 | 78.7 | 75.5 | 68.6 | 51.3 |

down slightly at 86.3% with a small reduction in the number of critical points returned. Table 5.2 gives the number of critical points that were found by PointNet for each initial experiment. With exception of the [**original**] experiment, the table shows the output assuming 128 points was provided as the source number of points; each set was derived by their respective experimentation type. The experiment which fed the critical points, found by the benchmark, back into PointNet was also constrained to 128 points; meaning that, on average, 6 points were zero filled per object. The table shows that, in the case of the [**critical**] experiment, the number of critical points discovered by PointNet had dropped from 92 to 88 on average for the aeroplane class with the other example classes losing a single point. This suggests that the additional zeroes used to make the input 128 points had negligible effect on the model and/or the data.

The latter concern is evaluated by the sub-sampling experimentation which reduced the number of points fed into the model and is address in subsection 5.2.2.1. Initially 128 points were used to reflect the aforementioned critical points discovered from the [**original**] experiment. Figure 5.4 illustrates the probability density functions, estimated by the initial sub-sampling experiments plus the [**original**] and [**critical**] experiments. The number of points in the [**critical**] experiment is fixed because it is the output from the [**original**] experiment. What is noticeable is that the mean validation accuracy for each type of experiment is not significantly different and the standard deviations are small. Table 5.1 gives the results which show they are no more than two percentage points away from each other. Whilst the results favour the hypothesis that a reduction in input points should be possible, they also demonstrate that there are no discernible differences made by the choice of sub-sampling technique when using this number of input points. This leads to the question, at which point does the number of points adversely affect the model's

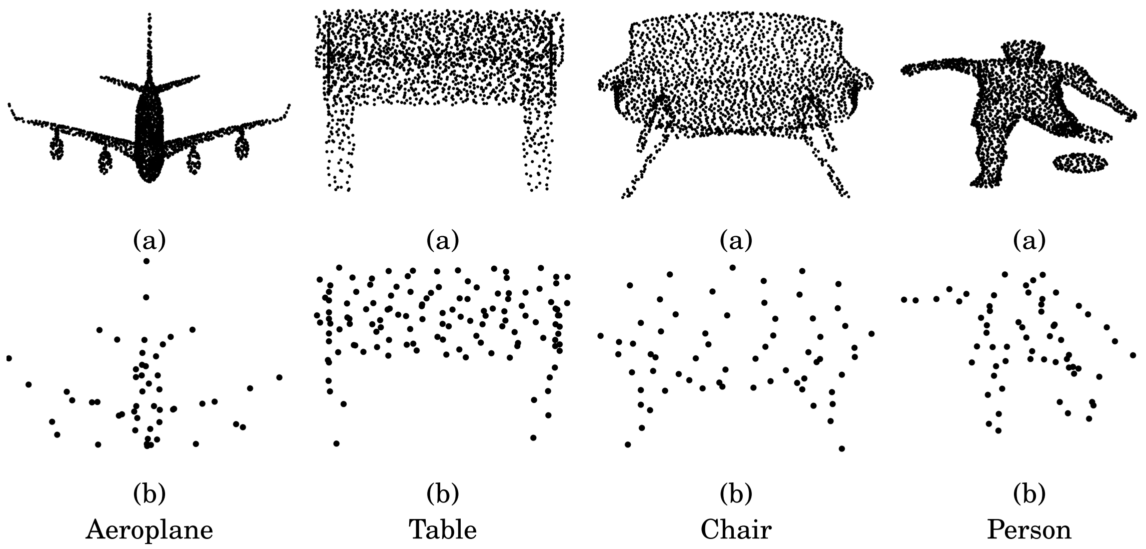


Figure 5.3: Plots of four point cloud sample objects. The top row (a) represents the dense point cloud and the bottom row (b) the remaining points after the Poisson spatial filter has been applied.

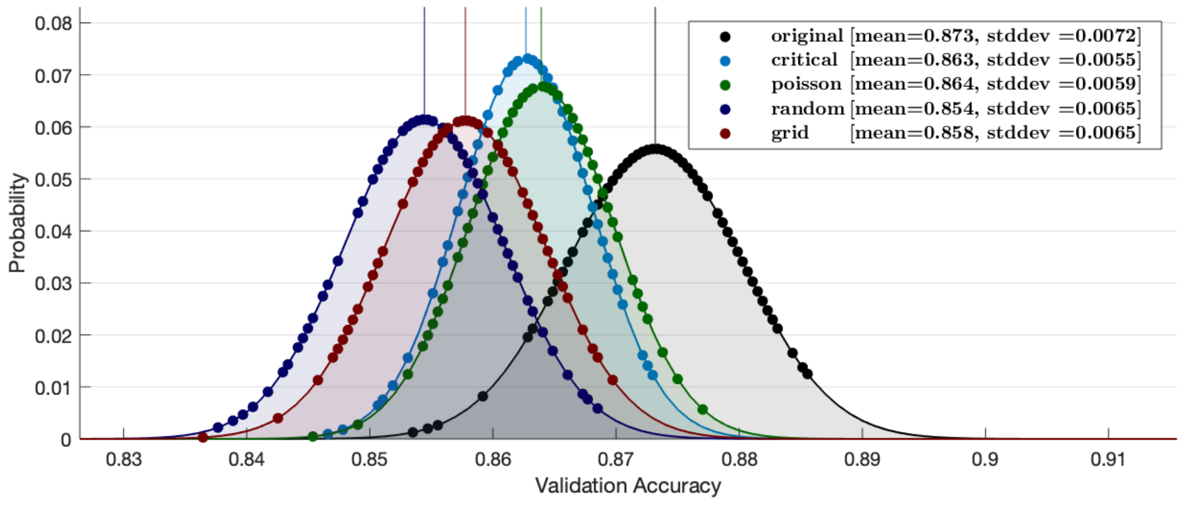


Figure 5.4: A series of probability density functions showing the outcome of the initial 100 tests for each type of experiment.

ability to correctly discriminate between the classes and what that says about the classification boundaries of the ModelNet40 dataset.

The execution time of each experiment was also captured by wall-clock measurement available in Tensorflow i.e. the difference between the start and end time as taken by observing a clock on the wall. Wall-clock times do not accurately state the amount of processing time consumed by each experiment. The mix of GPU and CPU processing would make this an overly complicated measurement. To make the wall-clock measurement meaningful, a single node was reserved from the BlueCrystal supercomputer and used exclusively for each experiment. Figure 5.5 shows the total execution time of each experiment. The accompanying table 5.3 shows a considerable reduction in time taken to train and evaluate both the [**original**] and [**critical**] experiments, with an improvement in lowering the computational resource required such that training time reduced from 196 minutes to 19 minutes.

Across all the initial sub-sampling experiments, the number of critical points found by the model dropped only marginally. This can be seen in Table 5.2. Furthermore, there was a noticeable improvement in processing time. The choice of implementation used for Poisson disk sampling

Table 5.2: The number of critical points found on average by PointNet for the final classification of four sample classes.

| **Class** | **Original** | **Critical** | **Poisson** | **Grid** | **Random** |
|---|---|---|---|---|---|
| Aeroplane | 92 | 88 | 78 | 80 | 76 |
| Table | 103 | 102 | 92 | 93 | 92 |
| Chair | 104 | 102 | 93 | 90 | 92 |
| Person | 104 | 103 | 91 | 89 | 94 |

did add an average of circa 60 minutes to the otherwise impressive results. This, in itself, is still a considerable improvement over the [**original**] experiment. This could have be improved by using a better implementation of the algorithm, one which capitalised upon the parallel processing capability of the GPU node.

Whilst Figure 5.3 visualises the points of four sample objects for a) the original points and b) after the initial application of the Poisson disk filter, the corresponding plots of the critical points found after the initial set of experimentation can be seen in Figure 5.6. To the human eye, one can just about make out what the objects are; although one needs to bear in mind that these are two-dimensional projections of a three-dimensional point cloud.

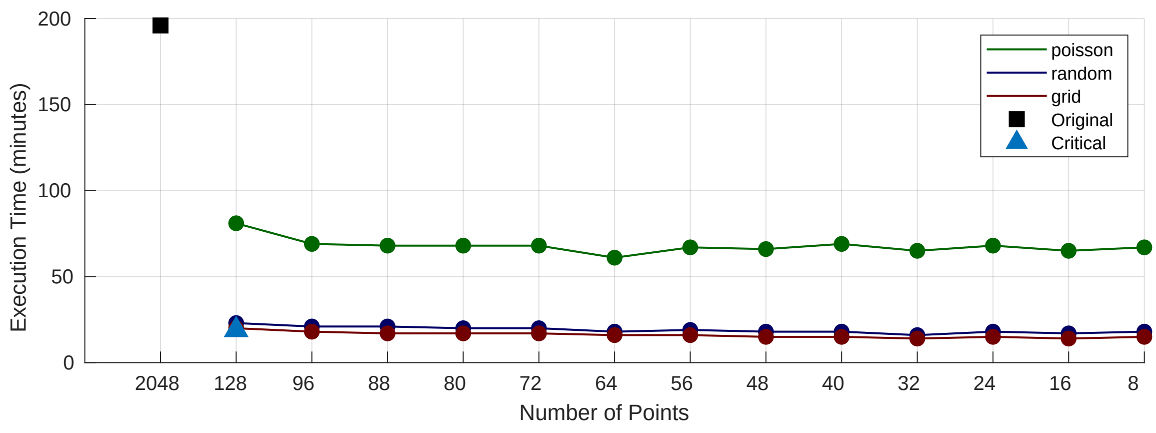


Figure 5.5: Comparison of the execution time for the experiments performed.

#### 5.2.2.1 Iteratively reducing the number of input points

Given the clear lack of distinguishable results using 128 points, a final set of experiments concentrated on characterising the behaviour of the model by reducing the number of sample points until the point of failure. Figure 5.1, with the corresponding table 5.1, presents the results of this experimentation. Noticeably, it is Voxel grid sub-sampling which shows an almost linear degradation in classification as the number of input points is decreased; whereas, both Poisson

Table 5.3: Execution time (in minutes) for the experiments performed.

| | Number of Points | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **2048** | **128** | **96** | **88** | **80** | **72** | **64** | **56** | **48** | **40** | **32** | **24** | **16** | **8** |
| **original** | 196 | | | | | | | | | | | | | |
| **critical** | - | 19 | | | | | | | | | | | | |
| **poisson** | - | 81 | 69 | 68 | 68 | 68 | 61 | 67 | 66 | 69 | 65 | 68 | 65 | 67 |
| **grid** | - | 23 | 21 | 21 | 20 | 20 | 18 | 19 | 18 | 18 | 16 | 18 | 17 | 18 |
| **random** | - | 20 | 18 | 17 | 17 | 17 | 16 | 16 | 15 | 15 | 14 | 15 | 14 | 15 |

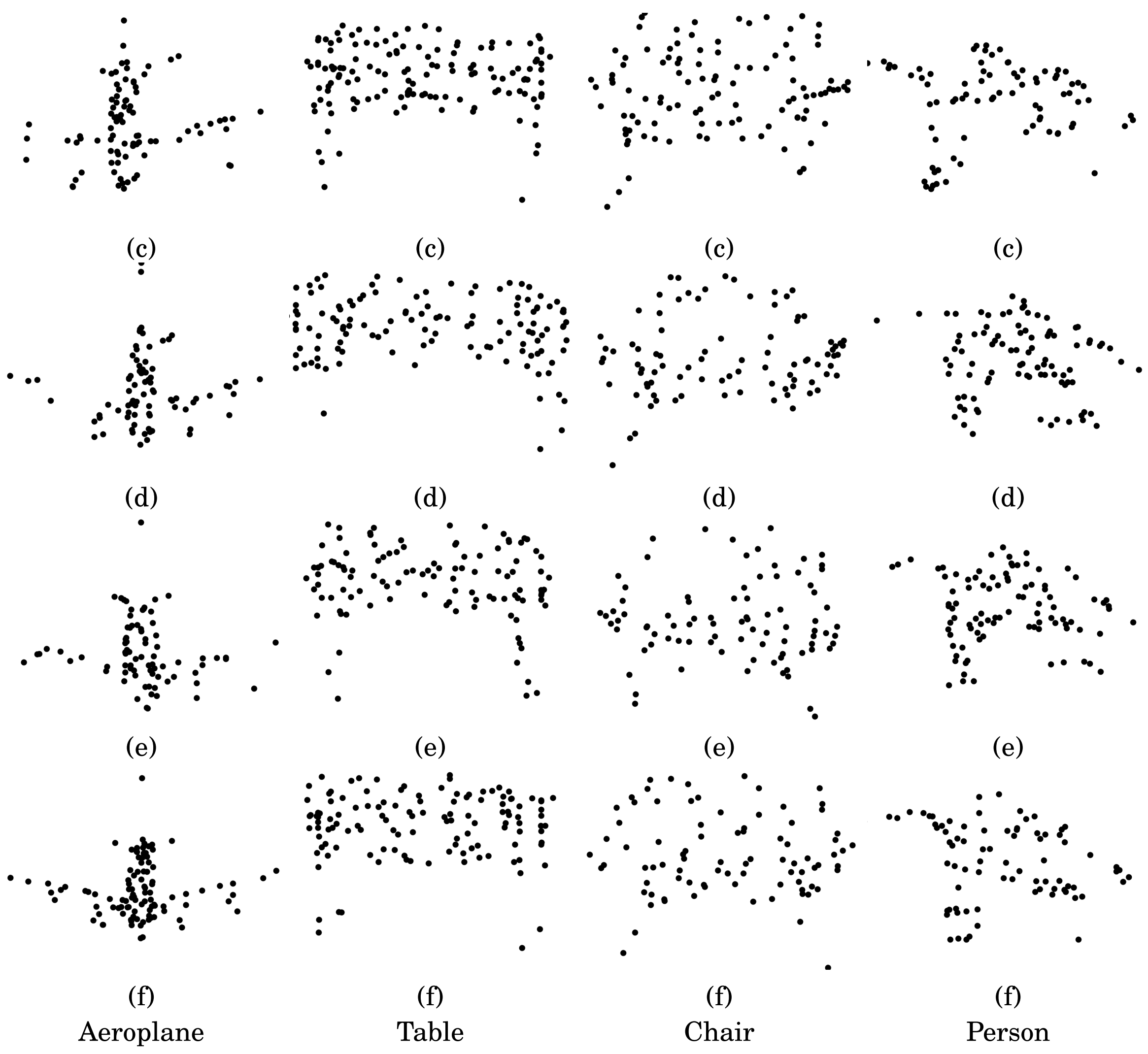


Figure 5.6: Plots of critical points from four example point cloud objects. The top row (c) represents the critical points found by PointNet without modification (experiment [**original**]) followed by (d), (e) and (f) which represent the critical points found after the spatial filters had been applied for experiments [**poisson**],[**grid**] and [**random**] respectively.

and Random sub-sampling maintain a good level of mean validation accuracy until the number of points has been reduced to below a threshold of 32.

#### 5.2.2.2 Discussion

An investigation into whether the deep learning architecture PointNet [81] is characteristically similar to a spatial filter has been performed. PointNet discovers and isolates important or critical points from within large amounts of point cloud data which represent the underlying infrastructure; and, in doing so removes superfluous data. Its behaviour is that of an intelligent sub-sampler. If this hypothesis holds then it would be reasonable to expect that, by providing

PointNet with only meaningful source points, then accuracy could be maintained whilst reducing the number of input points. The reduction in input points should also yield a corresponding rise in processing performance given that there will be fewer parameters to process.

A benchmark experiment was first performed which resulted in a validation accuracy close to that achieved by Qi et al [81]. This is an important and meaningful result because it demonstrates that the model used for evaluation was broadly in-line to that of PointNet's designers. Four further types of experiments were performed. Three of these involved sub-sampling the point cloud data using a variety of techniques. These were performed in order to reduce the input into the model. A significant experiment simply fed, back into PointNet, the critical points it found from the original experiment. This was performed to observe the efficiency of the model.

It is clear that an improvement in processing time could be achieved by reducing the number of input points without significantly affecting the mean validation accuracy of the model. Whilst the experiment which fed the critical points into the model disappointed by not returning them intact, the number of critical points discovered was not significantly different and this was reflected in the results. All initial experiments were within 2% of the original experiment and this extended to randomly sampling the points. Subsequent experimentation set out to discover the minimum set of points required to provide a reasonable level of accuracy. The number of input points from the dataset was reduced to 8 out of 2,048. Only when the number of points fell below 32 points was there a noticeable change in the classifier, although Voxel grid sub-sampling degraded in an almost linear fashion from the start (Table 5.1).

The fundamental difference between Voxel grid and Poission disk sampling may explain this surprising result. Poisson disk filters to leave a set of points which are constrained to be a minimum distance from each other i.e. by a radius. Whilst it removes points which violate the constraint, it does not adjust the remaining points. Therefore, the object's structure remains largely intact, albeit decimated. On the other hand, Voxel grid sampling finds the barycentre of all the points within its minimum distance restriction, defined by cube size. This will certainly deform the object, especially as the cube size increases. It suggests that it is this deformation that blurs the classification boundaries of the objects.

Crucially, PointNet appears to be capable of a reasonable level of validation accuracy (slightly above 80%) using a small subset of points (32 from 2,048 points). This does tend to imply that the class boundaries within the dataset are too well defined; although achieving higher rates of accuracy may not prove so simple and deformation, as suggested in the Voxel grid sub-sampling case, may be problematic due to the limited number of training samples i.e. fewer than 10,000.

The experiments suggest that sub-sampling point cloud datasets can reduce work load. However, the choice of dataset is important to the ratio of sub-sampling and which type of sub-sampling is employed. The selection of the ModelNet40 dataset reveals a number of inconvenient issues, other than sample size. First, the uniformity of the distribution of points which describe an object make it robust to data loss. This can be seen in Figure 5.3 which shows the full 2,048 points describing example objects. Removal of data points which are near to another would still leave the object structure fully intact. From Figure 5.3 it is also possible to see that the objects themselves are too diverse and not just in shape. This permits easy discrimination, without much intelligence required by the model. Take the aeroplane versus the table as an example. Whilst the PointNet architecture could have discovered the underlying structure and visual shape features which describe the object, such as the wing of a plane; it could have just as easily realised that the distance between the extrema points is larger in the aeroplane than the table and based its decision of the scale of the object. A better choice of deep learning architecture would have removed the constraint of using a simplistic dataset.

Whilst section 5.2.1 alluded to the idea that Voxel grid sub-sampling may indirectly preserve the edge features, one significant omission from these experiments is the failure to directly consider their preservation. This, along with the choice of deep learning architecture and dataset, is addressed in section 5.3.

## 5.3 Experimentation with real-world data

Applying sub-sampling to point cloud data has a number of potential benefits. It reduces the number of points within the dataset, making it computationally more efficient as the model has less to deal with. It can also address density issues, such as the one identified in section 5.1, where the use of a rotational LIDAR scanner disproportionally captures points near the origin. However, the results from the synthetic dataset in section 5.2 are inconclusive. This is because of the choice of model and dataset evaluated, which has raised questions about the ease of discrimination between class boundaries, to the actual definition of the features discovered e.g. is it the wing of a plane that defines a dominant feature or its scale. Furthermore, the dataset evaluated was synthetic and not subject to the aforementioned density issues, loss of data or noise. In reality, the dataset selected was largely constrained by the choice of model used to evaluate it.

In order to make a more objective evaluation of the benefits of sub-sampling, a model needs to be capable of handling real-world data which reflects a complicated environment. KPConv [112] is a state-of-the-art architecture which has been evaluated. It is built using a modified ResNet [41] backbone (an overview is given in section 4.1.3) which is able to utilise the receptive field convolutional layer approach over points in a point cloud. Consequently, it exploits the wealth of research into conventional CNNs utilising techniques including deep networks, leaky ReLU activation functions and batch normalisation. The model also ranks third in a benchmarking suite [90] which challenges models to correctly identify features from the real-world NPM3D dataset [91]. The dataset comprises of ten hand-labelled classifications, captured over 2,000 metres, using a rotational Velodyne HDL-32E LIDAR scanner which was mounted on a van and driven down urban streets in Paris and Lille, France. The classifications include large structures such as buildings, road surfaces and foliage; and street furniture including lamp posts, bins and poles. The NPM3D dataset has a 1 cm resolution and includes 143 million points.

The KPConv model together with the real-world NPM3D dataset make a convenient base to explore further sub-sampling.

### 5.3.1 Experimentation set-up and methodology

Fortunately, the authors of KPConv [112] provide an open source version of their architecture [111] which expedited the process of obtaining a benchmark based upon their experimentation. Modifications had to be made to accommodate the choice of Tensorflow version, available compiler and the provision of additional code for the automatic selection of GPU node.

After recreating the benchmark, which had originally sub-sampled the NPM3D dataset at a 4 cm resolution, the following experimentation was performed on the real-world NPM3D dataset using the KPConv model:

- A series of grid sampled experiments were performed with a range from 8 cm to 32 cm, in 4 cm incremental steps [**sub-sampled**] using the inherit grid sub-sampling functionality of the model.
- Edge features were extracted and evaluated using a range of flatness thresholds and neighbourhood constraints [**edges-only**].
- Grid sub-sampled environments with retained edge features were evaluated with sub-sampling from 8 cm to 32 cm. The edges themselves, were sub-sampled at half the value of the grid sub-sampling in order to prevent biasing in favour of them. Duplicate points were discarded [**retained edges**].

The edge features were found by computing the covariance matrix of a set of k-nearest neighbours, before applying a threshold value to filter the smallest eigenvalue of the matrix [12]. The threshold value is used as a means of determining flatness. For experimentation, edge-based datasets were created using a range from 5 to 25. In order to illustrate the behaviour of the threshold, a completely flat surface in a two-dimensional example would yield its smallest eigenvalue as being near 0. Therefore, values nearer 5 are flatter than those at 25. The number of k-nearest neighbours was also adjusted, its range was 5, 10, 25, 50, 75, 100, 125 and 150 neighbours. Each k-nearest neighbour value was used in conjunction with the flatness threshold to create a 162 new edge-based datasets.

Due to time considerations, only thresholds in increments of 5 were considered for the [**retained edges**] experimentation. This constrained this set to 280 experiments from 1176.

The hyper-parameters for using KPConv are given in Table 5.4 which includes training using a full 1,000 epochs with no premature stopping. This was done because of evaluation of the test data had to be performed using a restrictive third party as the data included no labels. Consequently, the validation data was used for testing.

### 5.3.2 Results and discussion

#### 5.3.2.1 Benchmark

The benchmark was created using existing code from the authors of KPConv [112] [111], with slight modification to incorporate the build environment. It sub-sampled the NPM3D dataset to 4 cm resolution, from 1 cm using grid sub-sampling meaning that all points with a 4 cm cube are averaged to find the centre of mass. Figure 5.7 gives the confusion matrix which has been generated from the results of a validation test. It includes both the row and column summaries which demonstrate the level of false positives and negatives. False positives are seen on a row by row basis, and show where a point was incorrectly assigned to the true class. Conversely, false negatives are seen on a column by column basis and indicate where the point was assigned a different class, referenced by the row. The true positive values are given on the diagonal of the confusion matrix. Accuracy for each class is computed as

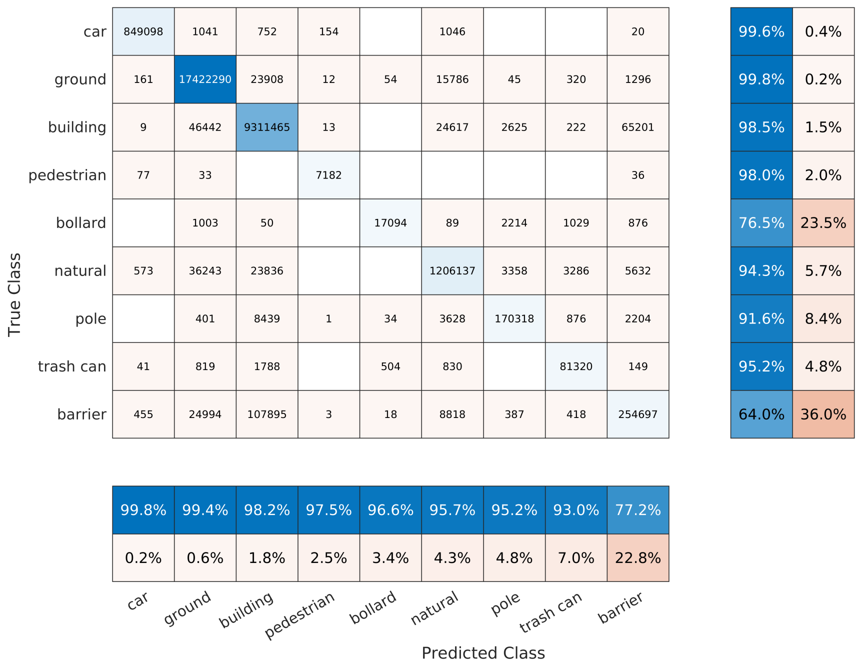
$$Accuracy_{class} = \frac{TruePositive_{class}}{\sum Row_{class} + \sum Column_{class} - TruePositive_{class}} \tag{5.4}$$

which defines the ratio of true positives for a single class over the sum of all results including both false positive and negative values. As both row and column summations include the true positive value it is counted twice. Consequently, the additional true positive count is removed. The mean accuracy is then computed over all class accuracies.

The confusion matrix of the benchmark given in Figure 5.7 shows that ground, buildings and natural are the predominant classes, those with the highest points. Other, less represented classes, perform equally well with the exception of barriers and bollards. The former tend to become confused with buildings and vice versa resulting in significant false positive and negative

Table 5.4: KPConv Hyper-parameters used during training

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| num_kernel_points | 15 | first_subsampling_dl | 0.04m |
| in_radius | 2.0m | density_parameter | 5 |
| KP_influence | linear | KP_extent | 1.0m |
| convolution_mode | sum | offsets_loss | fitting |
| offsets_decay | 0.1 | use_batch_norm | True |
| batch_norm_momentum | 0.98 | max_epoch | 1000 |
| epoch_steps | 1000 | learning_rate | 1e-2 |
| momentum | 0.98 | grad_clip_norm | 100.0 |
| batch_num | 8 | | |



| True Class \ Predicted Class | car | ground | building | pedestrian | bollard | natural | pole | trash can | barrier | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| car | 849098 | 1041 | 752 | 154 | | 1046 | | | 20 | 99.6% | 0.4% |
| ground | 161 | 17422290 | 23908 | 12 | 54 | 15786 | 45 | 320 | 1296 | 99.8% | 0.2% |
| building | 9 | 46442 | 9311465 | 13 | | 24617 | 2625 | 222 | 65201 | 98.5% | 1.5% |
| pedestrian | 77 | 33 | | 7182 | | | | | 36 | 98.0% | 2.0% |
| bollard | | 1003 | 50 | | 17094 | 89 | 2214 | 1029 | 876 | 76.5% | 23.5% |
| natural | 573 | 36243 | 23836 | | | 1206137 | 3358 | 3286 | 5632 | 94.3% | 5.7% |
| pole | | 401 | 8439 | 1 | 34 | 3628 | 170318 | 876 | 2204 | 91.6% | 8.4% |
| trash can | 41 | 819 | 1788 | | 504 | 830 | | 81320 | 149 | 95.2% | 4.8% |
| barrier | 455 | 24994 | 107895 | 3 | 18 | 8818 | 387 | 418 | 254697 | 64.0% | 36.0% |
| | 99.8% | 99.4% | 98.2% | 97.5% | 96.6% | 95.7% | 95.2% | 93.0% | 77.2% | | |
| | 0.2% | 0.6% | 1.8% | 2.5% | 3.4% | 4.3% | 4.8% | 7.0% | 22.8% | | |

Figure 5.7: Confusion matrix for original 4 cm sub-sampled benchmark, ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column).

Table 5.5: Accuracy for the original 4 cm sub-sampled benchmark and all subsequent grid sampled experiments

| | Accuracy | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **Benchmark (4 cm)** | 87.34 | 99.13 | 96.82 | 87.55 | 74.44 | 88.78 | 53.84 | 95.62 | 99.49 | 90.42 |
| **Sub-sampled (8 cm)** | 86.46 | 99.09 | 96.95 | 85.49 | 73.59 | 86.28 | 54.81 | 92.76 | 99.36 | 89.81 |
| **Sub-sampled (12cm)** | 85.07 | 99.00 | 96.92 | 83.27 | 73.01 | 84.85 | 55.02 | 84.62 | 99.21 | 89.71 |
| **Sub-sampled (16cm)** | 83.44 | 98.88 | 97.14 | 83.60 | 70.70 | 81.16 | 58.57 | 76.16 | 99.00 | 85.78 |
| **Sub-sampled (20cm)** | 83.61 | 98.76 | 97.12 | 87.03 | 68.43 | 78.19 | 58.97 | 80.41 | 97.92 | 85.68 |
| **Sub-sampled (24 cm)** | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **Sub-sampled (28 cm)** | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **Sub-sampled (32 cm)** | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **Investigation (32 cm)** | **25.12** | **97.21** | **80.46** | **0.28** | **0.00** | **0.00** | **0.00** | **0.00** | **0.00** | **48.14** |

errors, 36% and 22.8% respectively. Whilst this is not an ideal situation, it is easy to see how a barrier may be mistaken for a wall. Figure 5.8 gives a simple example of the model's confusion between buildings and barriers which is evident in the confusion matrix. It is easy to see why this poses a problem. Potentially, a barrier is a single wall. The data captured in this example shows incomplete houses which could be misclassified. It is also clear that the model has not discriminated against the apex of the house so is possibly looking at localised features.

Bollards, in the main, classify correctly. This is surprising given the small sample size. However, other objects appear to be misclassified as bollards such that there are 23.5% false

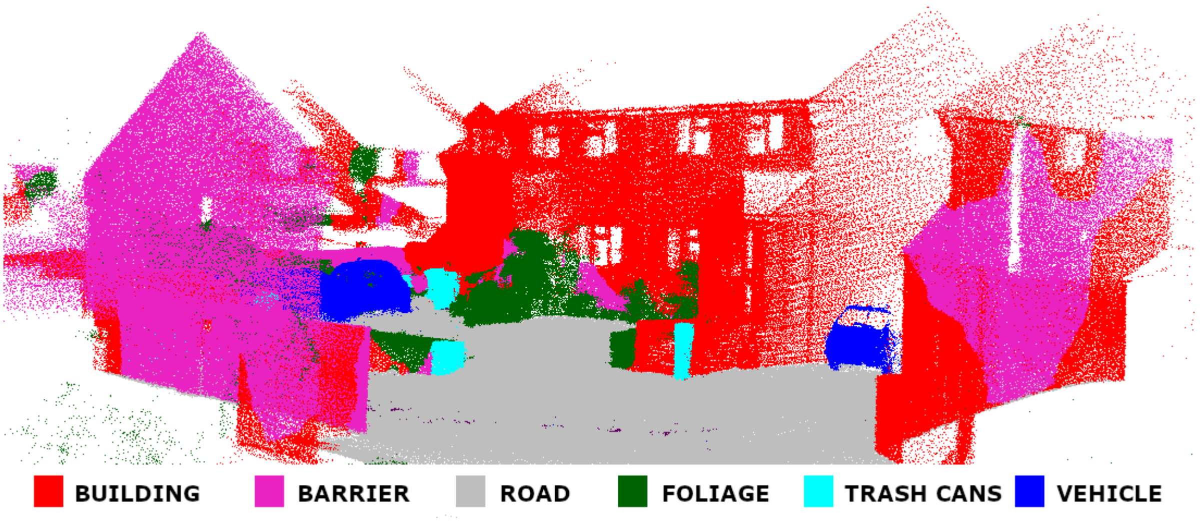


Figure 5.8: A visual illustration of the model's confusion between buildings and barriers. The illustration is just a test sample and does not reside in a dataset.

positives.

Table 5.5 includes the mean and individual accuracy results for the benchmark and shows barriers and bollards with the least accuracy, 53.84% and 74.44% respectively. The mean accuracy for the validation test was 87.34%. For the avoidance of doubt, the actual test data does not give the class information and therefore, it is easier to work with the validation set. The test set mean accuracy for this, as submitted to the benchmarking suite [90] was 82.0%. The benchmarking suite enforced a rule to ensure that test data feedback could only be retrieved once every five days, making it impractical to use the test data and leaving the validation set as the only viable option. The difference between the two means can be explained by the familiarity of the scene data. Whilst the test data is representative of the classification space, it was not physically located

Table 5.6: The number of points used in both the training and validation (V) tests for the original 4 cm sub-sampled benchmark and all subsequent grid sampled experiments

| | **Number of points** | | | | |
|---|---|---|---|---|---|
| | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **Total** |
| **Sub-sampled 4 cm** | 15,396,495 | 9,526,678 | 9,261,095 | 14,771,263 | 48,955,531 |
| **Sub-sampled 8 cm** | 6,380,267 | 3,598,964 | 8,229,693 | 5,796,469 | 24,005,393 |
| **Sub-sampled 12cm** | 3,434,775 | 1,859,975 | 4,531,244 | 2,989,250 | 12,815,244 |
| **Sub-sampled 16cm** | 2,154,589 | 1,140,014 | 2,875,875 | 1,812,326 | 7,982,804 |
| **Sub-sampled 20cm** | 1,478,894 | 762,564 | 1,994,947 | 1,216,189 | 5,452,594 |
| **Sub-sampled 24 cm** | 1,080,720 | 552,263 | 1,460,692 | 872,413 | 3,966,088 |
| **Sub-sampled 28 cm** | 823,679 | 415,225 | 1,119,194 | 653,095 | 3,011,193 |
| **Sub-sampled 32 cm** | 648,639 | 322,453 | 877752 | 506015 | 2,354,859 |

near the location of the training data; whereas, the validation data is adjacent to the training data i.e. a continuation of the same road as one captured in the training set.

A breakdown of the number of points within the point cloud, including both training and validation sets is given in table 5.6. It shows that the validation set comprised 30% of the total available points for the benchmark trial.

#### 5.3.2.2 Sub-sampling from the benchmark

Using the same set-up as that used to create the benchmark, a series of further sub-sampling was performed, in increments of 4 cm. Grid sub-sampling was used which meant that the centroid of all the points within an increasing cube size was found. The results of this sub-sampling is shown in Figure 5.9. The model is capable of sub-sampling up to 20cm before it breaks down. The accuracy data for this is given in table 5.5 showing a negligible change in accuracy from 8 cm to 20cm before a catastrophic drop where the model has classified everything as the ground. At the point of breakdown, the model was been trained on approximately three million points, dropping further as the sub-sampling increased.

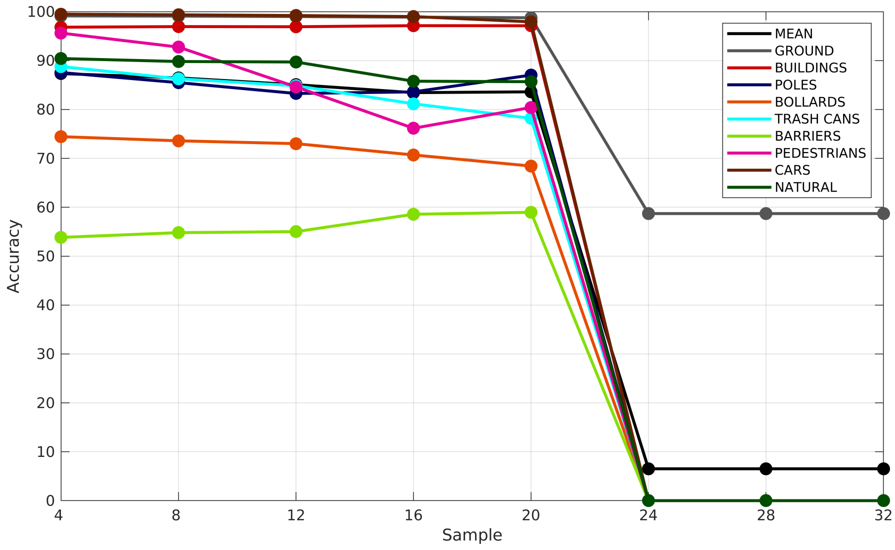
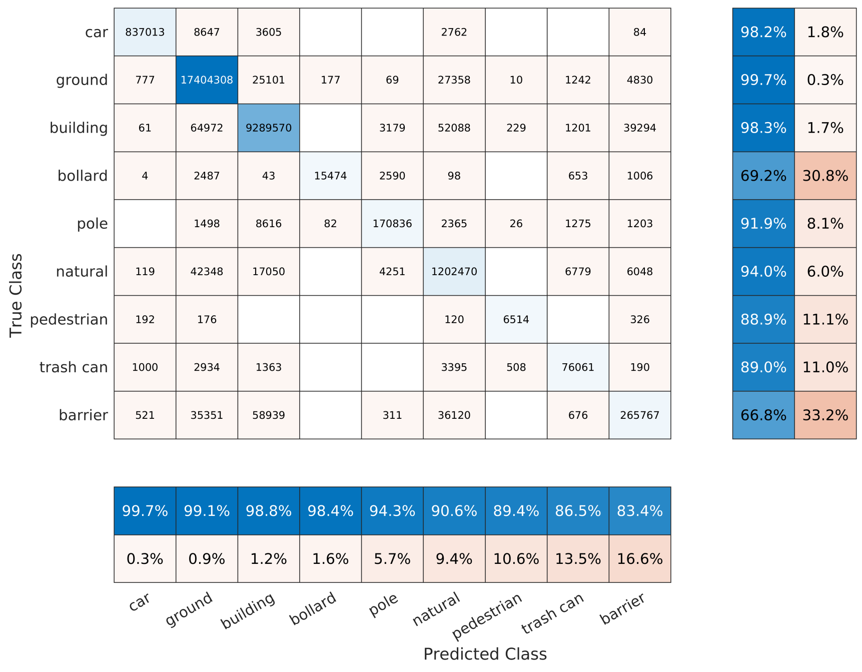


Figure 5.9: Plot of the benchmark and subsequent, incrementally stepped by 4 cm, sub-sampled results

The confusion matrices for both the final good classification (20cm) and the most sub-sampled experiment (32 cm, although this could be any after 20cm) are shown in figures 5.10 and 5.11. As with the benchmark, the 20cm confusion matrix shows large false positive results for the barrier and bollards. The difference being with the improvement in false negatives where buildings are mistaken for barriers. At 32 cm, everything looks the same to the model. Two dimensional projections are shown in Figure 5.12 which demonstrate the visual difference of a street-scene based on the sub-sampling amount; with 4 cm sub-sampling on the left hand side, and 32 cm on the right. A pedestrian has been circled, in black, to demonstrate the significant challenges facing the model. At 32 cm, the data representing the width of a human is no more than a couple of points. Blue cars and green trees may look clustered near the rear of the images, but this is just an effect of the projection. Cars nearer the viewer give a better representation of the spatial presence of an object. Lamp posts are now almost invisible to the naked eye from this view-point.



| True Class | car | ground | building | bollard | pole | natural | pedestrian | trash can | barrier | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| car | 837013 | 8647 | 3605 | | | 2762 | | | 84 | 98.2% | 1.8% |
| ground | 777 | 17404308 | 25101 | 177 | 69 | 27358 | 10 | 1242 | 4830 | 99.7% | 0.3% |
| building | 61 | 64972 | 9289570 | | 3179 | 52088 | 229 | 1201 | 39294 | 98.3% | 1.7% |
| bollard | 4 | 2487 | 43 | 15474 | 2590 | 98 | | 653 | 1006 | 69.2% | 30.8% |
| pole | | 1498 | 8616 | 82 | 170836 | 2365 | 26 | 1275 | 1203 | 91.9% | 8.1% |
| natural | 119 | 42348 | 17050 | | 4251 | 1202470 | | 6779 | 6048 | 94.0% | 6.0% |
| pedestrian | 192 | 176 | | | | 120 | 6514 | | 326 | 88.9% | 11.1% |
| trash can | 1000 | 2934 | 1363 | | | 3395 | 508 | 76061 | 190 | 89.0% | 11.0% |
| barrier | 521 | 35351 | 58939 | | 311 | 36120 | | 676 | 265767 | 66.8% | 33.2% |
| | 99.7% | 99.1% | 98.8% | 98.4% | 94.3% | 90.6% | 89.4% | 86.5% | 83.4% | | |
| | 0.3% | 0.9% | 1.2% | 1.6% | 5.7% | 9.4% | 10.6% | 13.5% | 16.6% | | |

Figure 5.10: Confusion matrix for original 20cm sub-sampled benchmark ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

The impact of sub-sampling at 32 cm needs little explanation for relatively small-scale objects such as poles, bollards and pedestrians. These objects can plausibly become noise to the model. However, there are at least three classifications (buildings, ground and nature) that defy the complete collapse of the model at greater levels of sub-sampling. These also happen to be over-represented within the environment. Walls of buildings and the ground are orientated almost perpendicular to one another; and, foliage (nature) looks like unstructured noise. This clear distinction between man-made and natural object classifications was mentioned earlier in section 3.4, which highlighted geometric structures as a tell-tale sign of man-made objects versus the randomness of nature.

There are two explanations for why these obvious distinctions eluded the classifier. The first concerns the behaviour of the grid sub-sampling technique. Finding the centroids in an ever-increasing space has the effect of smoothing out the randomness of foliage, making it resemble a geometric structure. Performing grid sub-sampling at 4 cm yields a potential 25 points per metre

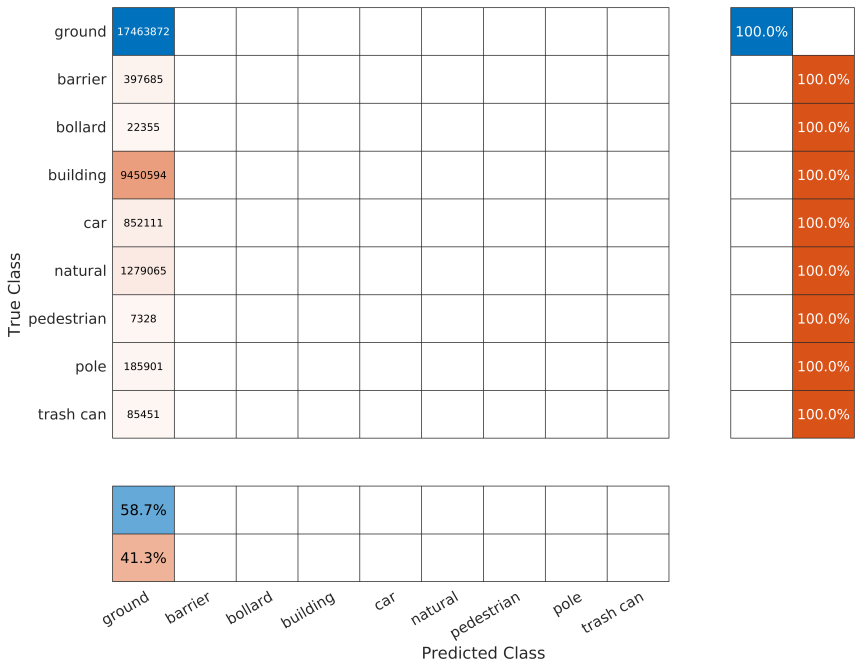
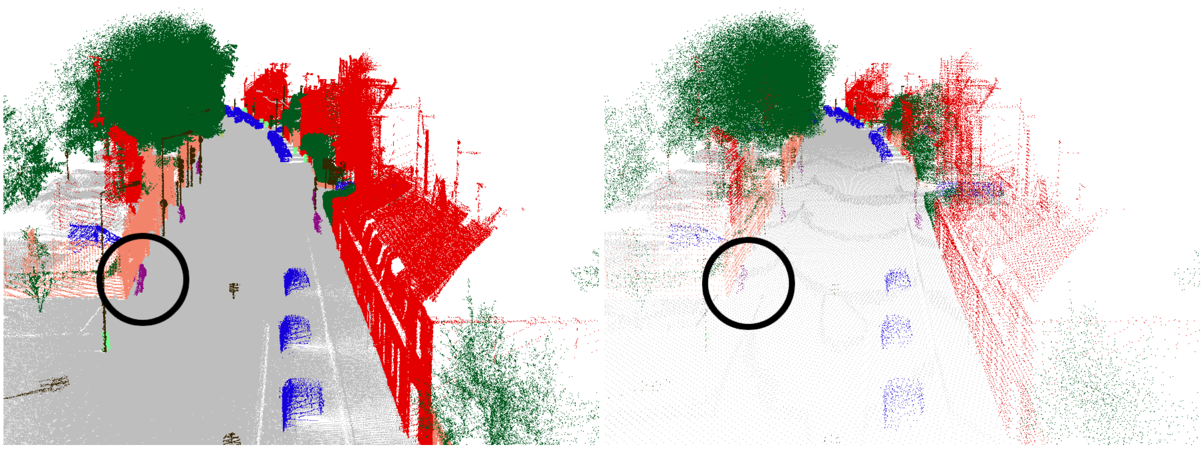


Figure 5.11: Confusion matrix for original 32 cm sub-sampled benchmark showing a breakdown of the model at this level of sub-sampling. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

in a single dimension. However, increasing the sub-sampling to 32 cm reduces this to an eighth, circa 3 points per metre, resulting in considerable smoothing.



4 cm grid sub-sampled (benchmark) 32 cm grid sub-sampled

Figure 5.12: A projection of points showing the street-scene with 4 cm sub-sampling on the left, and 32 cm sub-sampling on the right hand side. The black circle highlights where the pedestrian is.

The second explanation concerns the failure of the model to learn anything from data which has been sub-sampled too much. Two similar examples are given in Figure 5.13 which show the effective receptive field of a given point. For the valid 4 cm sub-sampling, the image shows points similat to a heatmap ranging from red (important) to blue (not important). For the 32 cm sub-sampling, the image shows only dark red points and indicate a complete failure to find any points which has an influence. The effective receptive field describes the amount of influence the neighbouring points have on the feature of interest (the black dot in the illustrations). The left hand side shows a point sampled from the benchmark. The mapping of influence is from blue, where the neighbouring point has no influence, to bright red. In this example many points have influence. Conversely, the right hand side shows the 32 cm sub-sampled experiment. The dark red signifies the influence values were not a number (NaN) indicating that the model was unable to compute the gradients necessary i.e. the complete failure of the model. This finding was also found in the log files which show loss and accuracy switching to NaN after a few epochs.

The intuition into the collapse of the model is that there are not enough points for the kernel point convolutional operator to work with. To rule out other issues first, a series of hyper-parameter tweaking tests were performed, and a move from GPU to CPU. Tweaking included increasing the input radius from 2 metres up to 8 metres in incremental steps, increasing/decreasing the batch number and turning off batch normalisation. This was done even though it appeared counter-intuitive, given its role in preventing vanishing gradients; although, ResNet architectures shouldn't have this problem. None of these tweaks affected the outcome, all failed. The actual solution was to double the hyper-parameter which increases the radius of the field of influence, confirming the intuition. This hyper-parameter was originally constrained to only sample a one metre radius; this is irrespective of the initial radius of the input which, as

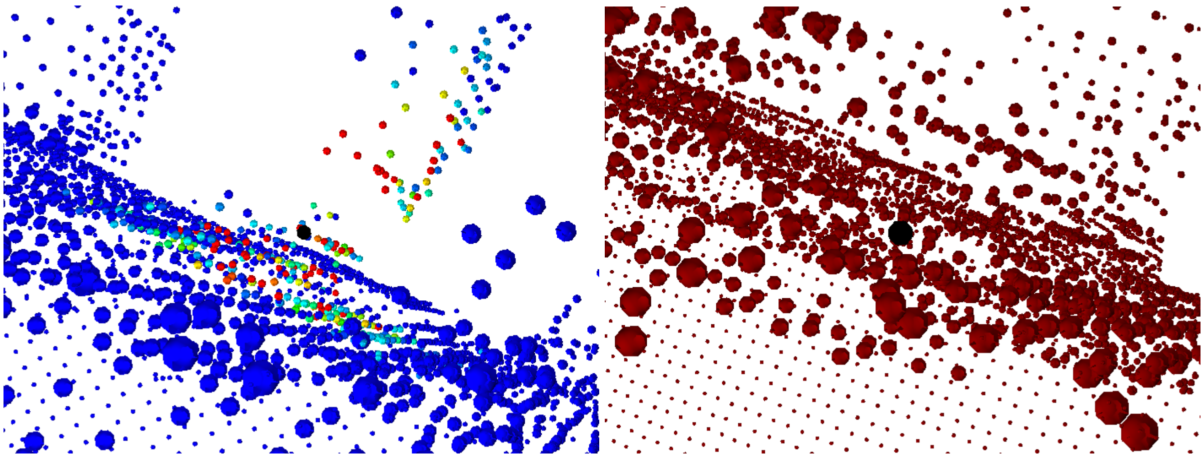

4 cm grid sub-sampled (benchmark) 32 cm grid sub-sampled

Figure 5.13: The effective receptive field of a kernel point with 4 cm sub-sampling on the left shown as a heatmap with bright red (important) to blue (not important), and 32 cm sub-sampling on the right hand side, with dark red as unable to compute.

Table 5.7: Sample validation accuracy results for the edge-only set of experiments

| K | Threshold | mean | ground | buildings | poles | bollards | trash_cans | barriers | pedestrians | cars | natural |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 15 | 60.75 | 97.34 | 90.94 | 64.00 | 44.43 | 57.89 | 31.98 | 10.20 | 87.88 | 62.10 |
| 25 | 15 | 42.34 | 85.37 | 79.04 | 36.34 | 4.29 | 30.24 | 15.02 | 28.85 | 54.45 | 47.44 |
| 125 | 15 | 18.74 | 31.19 | 52.72 | 9.37 | 0.00 | 6.74 | 5.90 | 6.73 | 16.66 | 39.36 |

stated above, had no effect. The result of the successful test is shown in bold in table 5.6 giving a mean accuracy of 25.12 % with ground, buildings and natural as 97.21%, 80.46% and 48.14% respectively. The result does provide confidence that the model can discriminate between ground and buildings based on orientation, when the correct hyper-parameters are chosen.

The field of influence hyper-parameter was only changed in order to prove a hypothesis; for the remainder of the experimentation the original value was used. This was to demonstrate the effectiveness of retaining edge features.

#### 5.3.2.3 Edge features

Sub-sampling at high thresholds has the potential to lose key edge features as the point cloud gets decimated. Based upon that, a series of experimentation was performed to see how well the model worked when presented with only the edge points. The edge points were found using a fast edge detection technique [12] which evaluates the eigenvalues of the covariance matrix created from a set of neighbouring points. Edges are filtered by comparing a flatness threshold value against the smallest eigenvalue. Those over the threshold were considered edge points. The algorithm requires two bits of information i) the number of neighbouring points and, ii) the flatness threshold. Experiments were performed which covered 5, 10, 25, 50, 75, 100, 125 and 150 neighbouring points; and a threshold between 5 and 25, where a threshold closer to zero is perceived to be flatter. All experiments sub-sampled the edges using a resolution of 4 cm.

Figures 5.15 and 5.16 give the mean accuracy results, on a class by class basis, for the edge-only results. The figures show the results averaged by flatness threshold range and k-nearest neighbours respectively. Background patches in the figure show, with limited success, the minimum and maximum accuracy results.

The threshold results, shown in Figure 5.15, decline in an almost linear fashion as the threshold increases. Figure 5.17 shows the points retained as a percentage of the full point cloud and prior to grid sub-sampling to the benchmark resolution of 4 cm. The heatmap demonstrates there is a steady decline in retained edge points due to the increasingly difficult threshold criteria set for them.

The k-nearest neighbour results, shown in Figure 5.16, also reflect the decline in edges, shown in the heatmap. In this instance, the lower neighbourhoods perform the best because they retain more points at lower thresholds and this positively affects the averages plotted. However, it can be observed that the neighbourhood, which comprises of only five points, outperforms the others, even though it does not retain as many points. Table 5.7 gives sample accuracy results for the

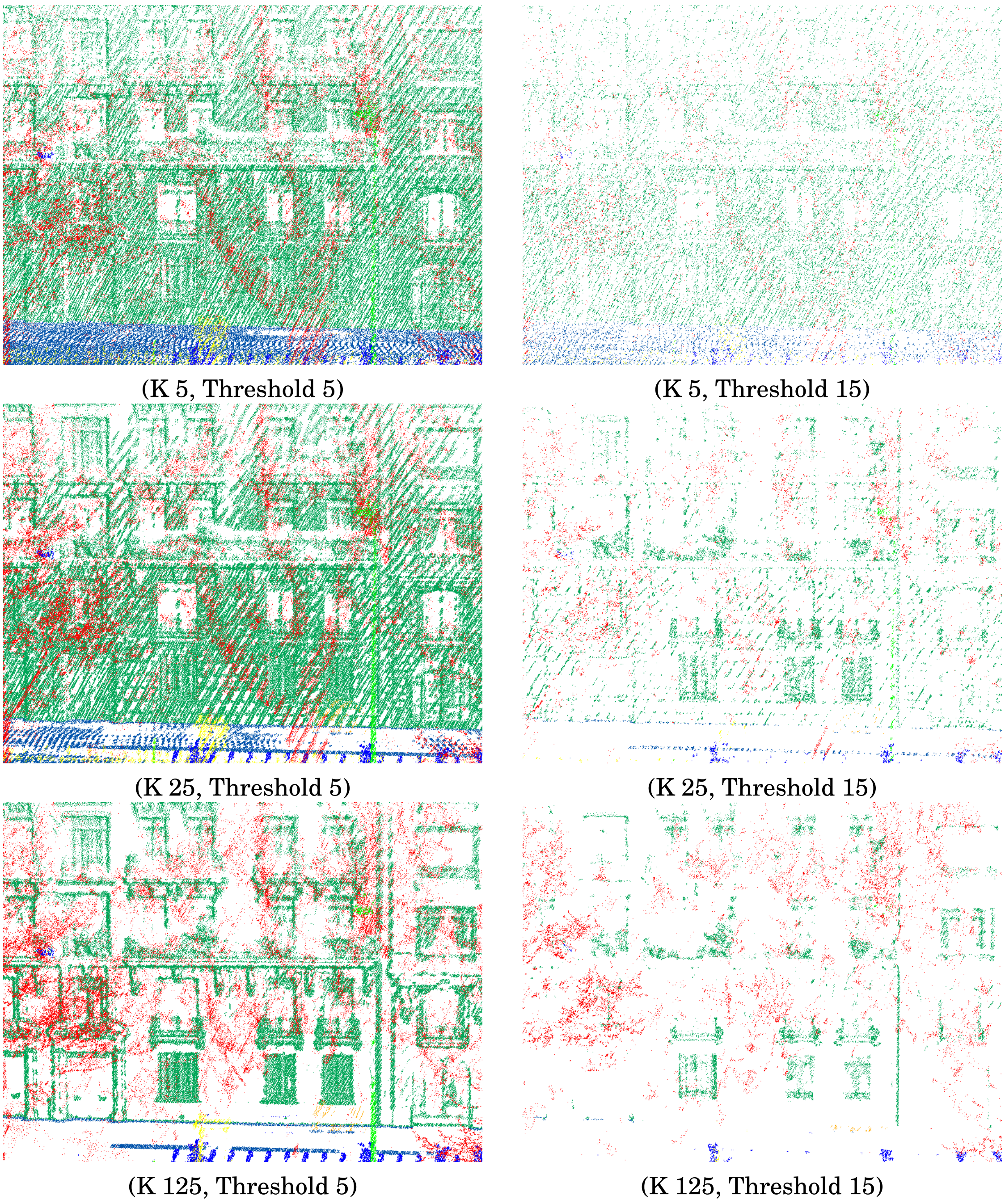


Figure 5.14: Illustration of edge points retained using k-nearest neighbours of 5, 25 and 125, with thresholds of 5 and 15.

k-nearest neighbours at 5, 25 and 125; with the full edge-only validation accuracy results are given in Appendix A. The results show that whilst the small five point neighbourhood does not produce the best result from the start, it manages to maintain better results longer than the others. Hence, why this neighbourhood outperforms the others, on average.

Figure 5.14 provides a plausible explanation. It shows example projections of the edges

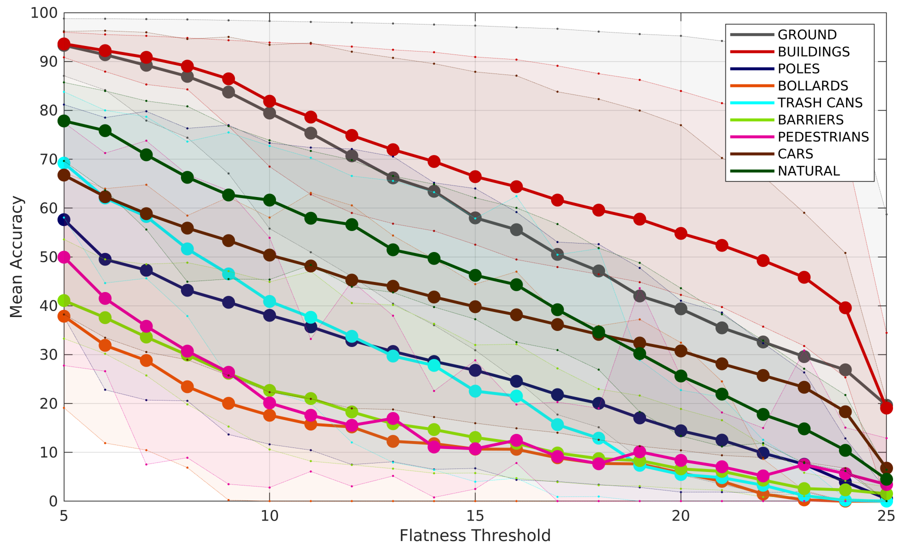


Figure 5.15: Edge-only experimentation giving mean accuracy per classification, averaged by flatness threshold. A threshold of 5 represents a reasonably flat surface; whereas, a value of 25 is a more defined edge.

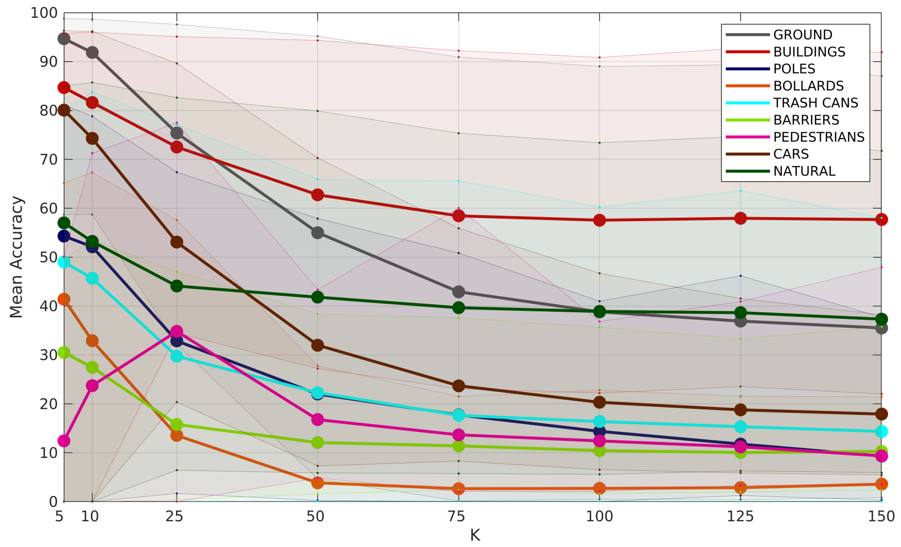
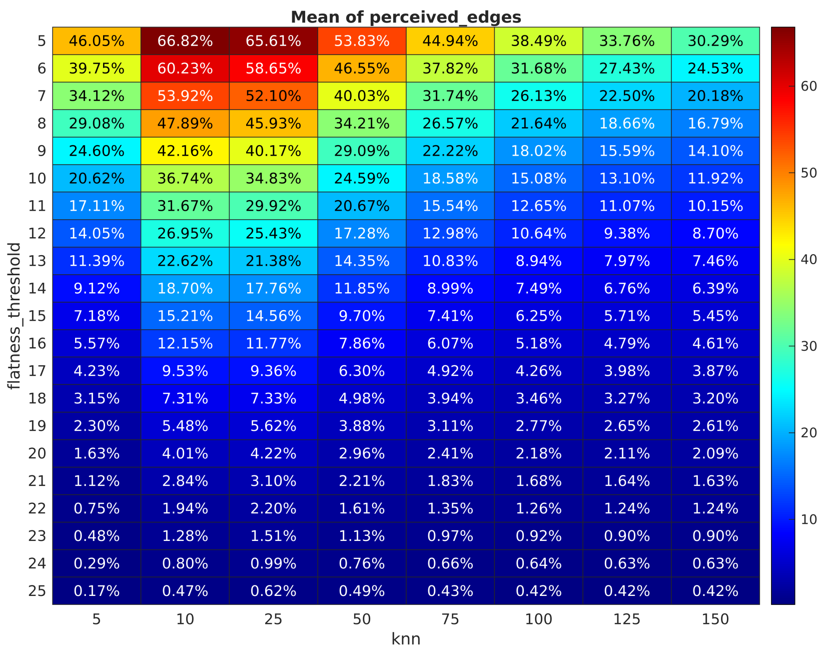


Figure 5.16: Edge-only experimentation giving mean accuracy per classification, averaged by number of neighbours.

retained for the k-nearest neighbours at 5, 25 and 125 points; with thresholds of 5 and 15. Whilst the larger neighbourhoods appear to define the edges better, the neighbourhood of five is much noisier. This is evident to see when comparing all the neighbourhoods with the threshold set at 15. In essence, this smaller neighbourhood is very sensitive to change, making it easier for the flatness threshold to be crossed. The small neighbourhood appears to be acting like a general sub-sampler and is not really detecting good edges. Given the results, the conclusion is that



| flatness_threshold \ knn | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 46.05% | 66.82% | 65.61% | 53.83% | 44.94% | 38.49% | 33.76% | 30.29% |
| 6 | 39.75% | 60.23% | 58.65% | 46.55% | 37.82% | 31.68% | 27.43% | 24.53% |
| 7 | 34.12% | 53.92% | 52.10% | 40.03% | 31.74% | 26.13% | 22.50% | 20.18% |
| 8 | 29.08% | 47.89% | 45.93% | 34.21% | 26.57% | 21.64% | 18.66% | 16.79% |
| 9 | 24.60% | 42.16% | 40.17% | 29.09% | 22.22% | 18.02% | 15.59% | 14.10% |
| 10 | 20.62% | 36.74% | 34.83% | 24.59% | 18.58% | 15.08% | 13.10% | 11.92% |
| 11 | 17.11% | 31.67% | 29.92% | 20.67% | 15.54% | 12.65% | 11.07% | 10.15% |
| 12 | 14.05% | 26.95% | 25.43% | 17.28% | 12.98% | 10.64% | 9.38% | 8.70% |
| 13 | 11.39% | 22.62% | 21.38% | 14.35% | 10.83% | 8.94% | 7.97% | 7.46% |
| 14 | 9.12% | 18.70% | 17.76% | 11.85% | 8.99% | 7.49% | 6.76% | 6.39% |
| 15 | 7.18% | 15.21% | 14.56% | 9.70% | 7.41% | 6.25% | 5.71% | 5.45% |
| 16 | 5.57% | 12.15% | 11.77% | 7.86% | 6.07% | 5.18% | 4.79% | 4.61% |
| 17 | 4.23% | 9.53% | 9.36% | 6.30% | 4.92% | 4.26% | 3.98% | 3.87% |
| 18 | 3.15% | 7.31% | 7.33% | 4.98% | 3.94% | 3.46% | 3.27% | 3.20% |
| 19 | 2.30% | 5.48% | 5.62% | 3.88% | 3.11% | 2.77% | 2.65% | 2.61% |
| 20 | 1.63% | 4.01% | 4.22% | 2.96% | 2.41% | 2.18% | 2.11% | 2.09% |
| 21 | 1.12% | 2.84% | 3.10% | 2.21% | 1.83% | 1.68% | 1.64% | 1.63% |
| 22 | 0.75% | 1.94% | 2.20% | 1.61% | 1.35% | 1.26% | 1.24% | 1.24% |
| 23 | 0.48% | 1.28% | 1.51% | 1.13% | 0.97% | 0.92% | 0.90% | 0.90% |
| 24 | 0.29% | 0.80% | 0.99% | 0.76% | 0.66% | 0.64% | 0.63% | 0.63% |
| 25 | 0.17% | 0.47% | 0.62% | 0.49% | 0.43% | 0.42% | 0.42% | 0.42% |

Figure 5.17: A heatmap showing the edge points retained, as a percentage of the total number of points within the entire point cloud

the model is able to capitalise better on the wider distribution of points available to it. This is in contrast with the larger neighbourhoods, who retain real edge points at the expense of flat surfaces leaving sparse areas. This is then compounded further by employing grid sub-sampling to place the data in similar conditions to the benchmark i.e. edge points get averaged; whereas, a more even distribution retains greater points over the entire environment. This can be seen in the first heatmap shown in Figure 5.18 where the percentage of points retained after sub-sampling increases for the small neighbourhood, whilst the others decrease.

The confusion matrices for the k-nearest neighbours of 5, 25 and 125 with a flatness threshold of 15 are given in figures 5.19, 5.20 and 5.21 respectively. They tend to support the hypothesis that the small neighbourhood is acting like a better distributed sub-sampler; whereas, the others are better at finding edges. The mean validation result for the small neighbourhood, in Table 5.7, is 60.75%. The results show reasonable accuracy on the larger items, especially the ground (97.34%), buildings (90.94%) and cars (87.88%). An initial impression of the mediocre performance of classifying natural objects such as foliage, which has a validation accuracy of only 62.10%, is one of surprise. The corresponding confusion matrix, shown in Figure 5.19, reveals there are many false negatives, where the foliage has been labelled incorrectly as either ground or buildings.

Importantly, this should be viewed in context of only having access to 7.18% of the original

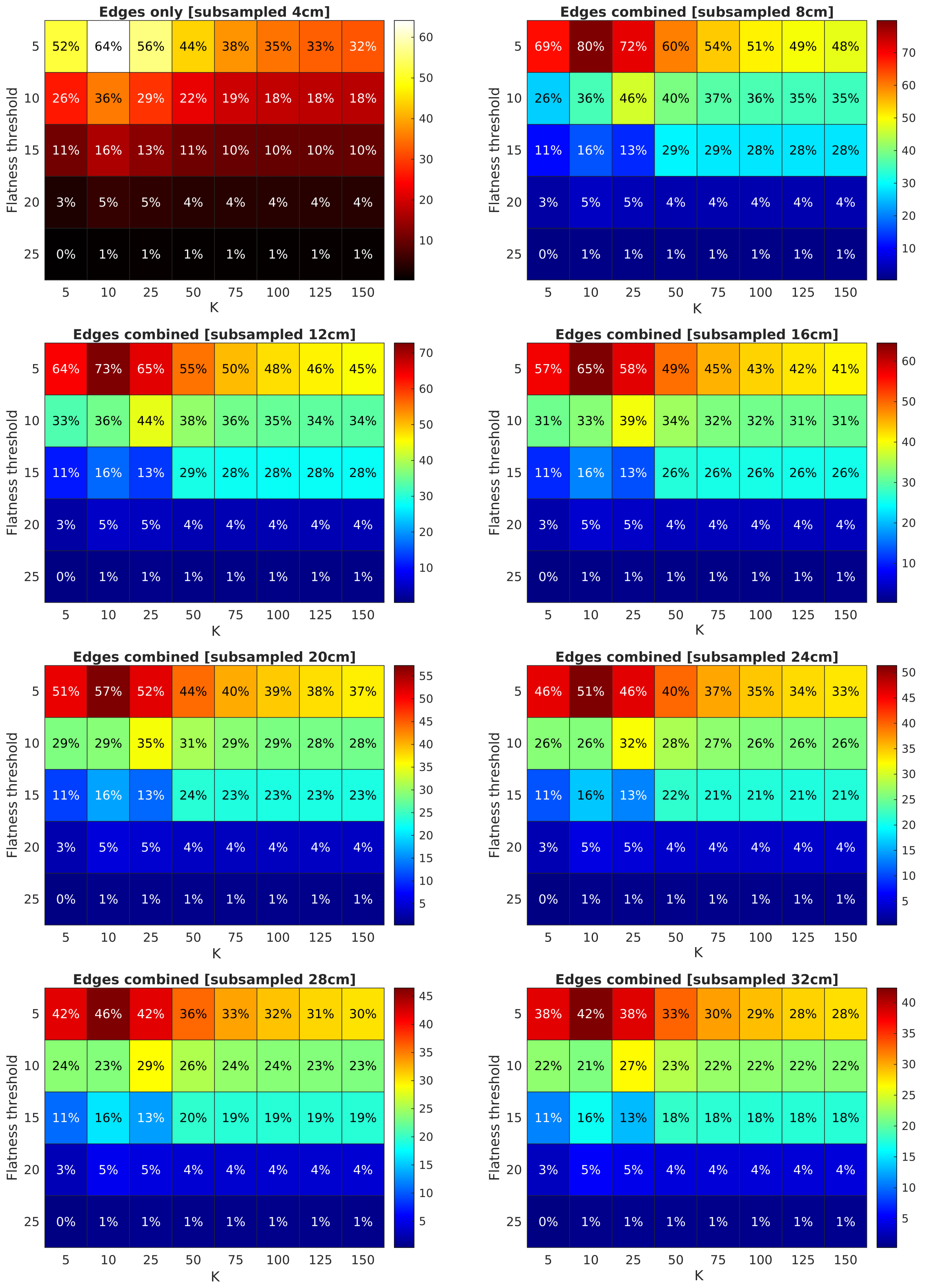
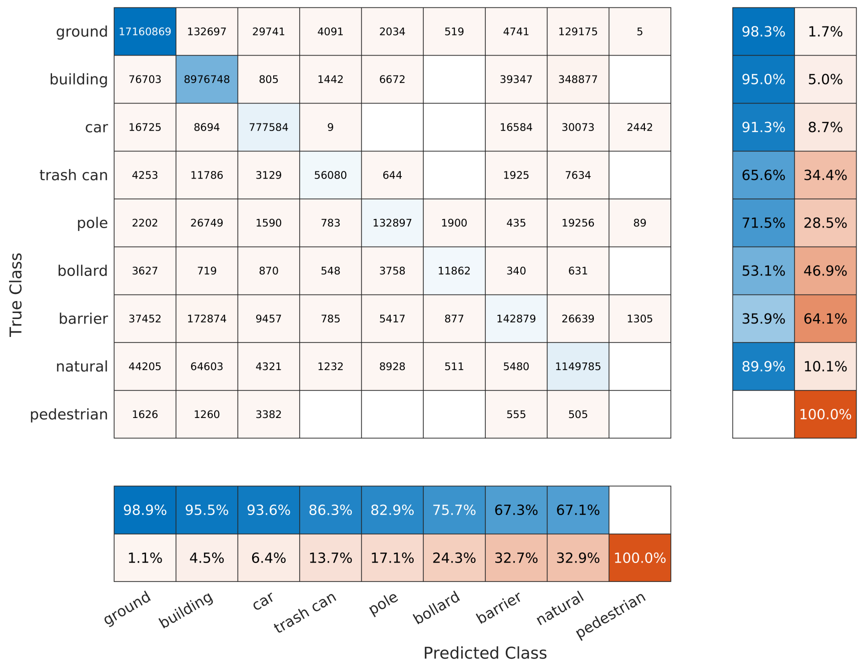


Figure 5.18: Ratio of edge points in the appropriate dataset created after combining edge points with grid sub-sampling where edges are also edges sub-sampled at half the rate e.g. 8 cm sub-sampling has edges sub-sampled at 4 cm. The top left heatmap is edge-only, sub-sampled at 4 cm



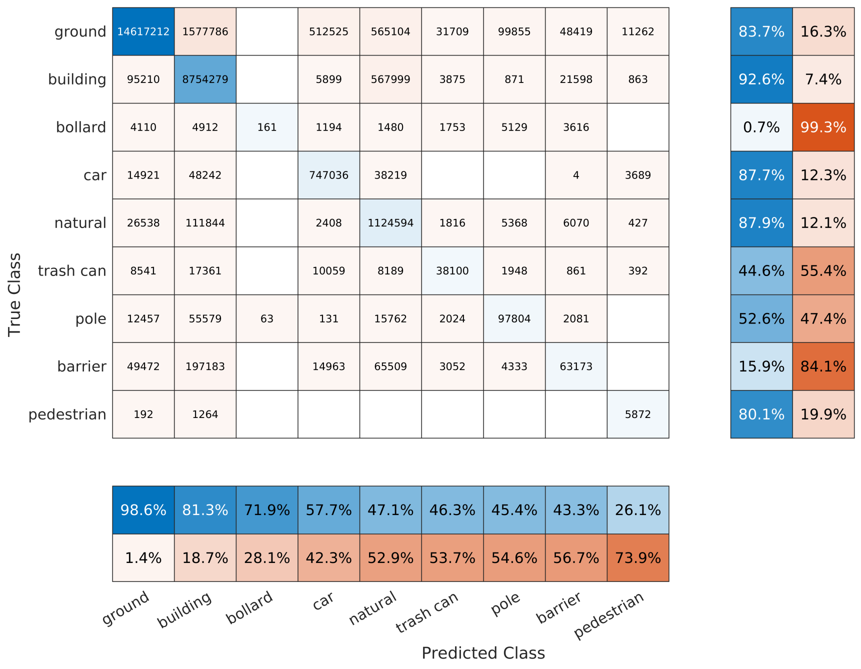
| True Class | ground | building | car | trash can | pole | bollard | barrier | natural | pedestrian | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ground | 17160869 | 132697 | 29741 | 4091 | 2034 | 519 | 4741 | 129175 | 5 | 98.3% | 1.7% |
| building | 76703 | 8976748 | 805 | 1442 | 6672 | | 39347 | 348877 | | 95.0% | 5.0% |
| car | 16725 | 8694 | 777584 | 9 | | | 16584 | 30073 | 2442 | 91.3% | 8.7% |
| trash can | 4253 | 11786 | 3129 | 56080 | 644 | | 1925 | 7634 | | 65.6% | 34.4% |
| pole | 2202 | 26749 | 1590 | 783 | 132897 | 1900 | 435 | 19256 | 89 | 71.5% | 28.5% |
| bollard | 3627 | 719 | 870 | 548 | 3758 | 11862 | 340 | 631 | | 53.1% | 46.9% |
| barrier | 37452 | 172874 | 9457 | 785 | 5417 | 877 | 142879 | 26639 | 1305 | 35.9% | 64.1% |
| natural | 44205 | 64603 | 4321 | 1232 | 8928 | 511 | 5480 | 1149785 | | 89.9% | 10.1% |
| pedestrian | 1626 | 1260 | 3382 | | | | 555 | 505 | | | 100.0% |
| | 98.9% | 95.5% | 93.6% | 86.3% | 82.9% | 75.7% | 67.3% | 67.1% | | | |
| | 1.1% | 4.5% | 6.4% | 13.7% | 17.1% | 24.3% | 32.7% | 32.9% | 100.0% | | |

Figure 5.19: Confusion matrix for a 4 cm sub-sampled edge-only dataset (K 5, Threshold 15) ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

points, as given in the heatmap shown Figure 5.17. It has already been established that the small neighbourhood is just sub-sampling without due care for the edges. Given this, it is reasonable to refer back to section 5.3.2.2 which suggested that natural objects would be smoothed out by applying grid sub-sampling. This smoothing would almost certainly result in natural objects being misclassified, confirming what is shown in the confusion matrix.

Another observation is the very poor classification of pedestrians within the small neighbourhood, at 10.20%. This can be explained since they are under-represented in the dataset. Referring back to Figure 5.12 in section 5.3.2.2 shows how too much sub-sampling converts objects such as pedestrians into background noise. The results of the k-nearest neighbourhood set at 25, which initially has access to twice as many points as the small neighbourhood (shown in Figure 5.17), but is subsequently sub-sampled to have an equivalent amount (shown in Figure 5.18), suggest that edge retention does benefit classes such as pedestrians, at an accuracy of 28.85%. This result is shown in Table 5.7. However, the pedestrian class is the only classification of note. The other, edge sensitive classifications performed considerably worse than the small neighbourhood and, therefore, refute this idea.

Looking at the differences between the two neighbourhoods in Figure 5.14 reveal gaps in the environment where flat surfaces should be. This is more apparent when looking at the larger 125 point neighbourhood which, when sub-sampled, yielded an similar number of available

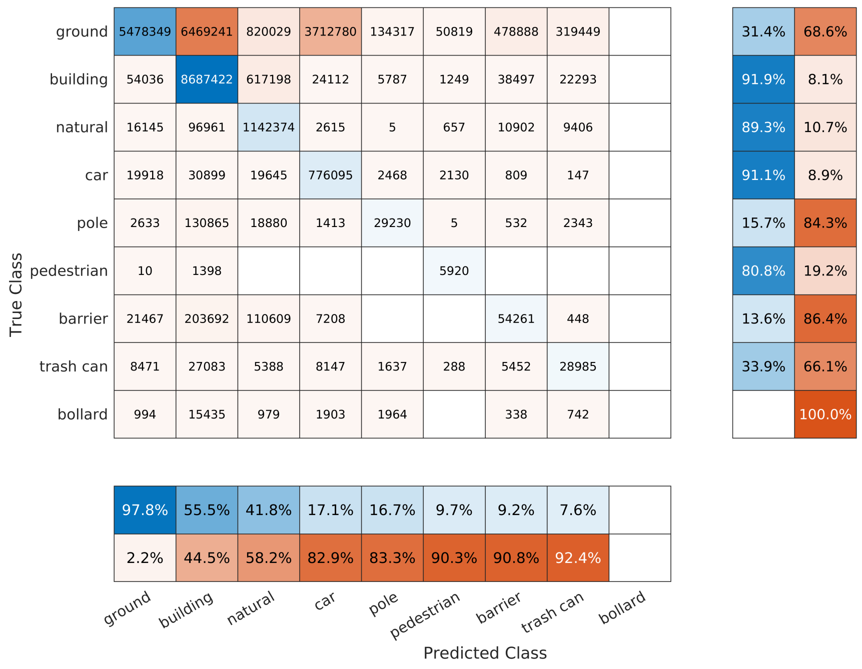


| | ground | building | bollard | car | natural | trash can | pole | barrier | pedestrian | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ground | 14617212 | 1577786 | | 512525 | 565104 | 31709 | 99855 | 48419 | 11262 | 83.7% | 16.3% |
| building | 95210 | 8754279 | | 5899 | 567999 | 3875 | 871 | 21598 | 863 | 92.6% | 7.4% |
| bollard | 4110 | 4912 | 161 | 1194 | 1480 | 1753 | 5129 | 3616 | | 0.7% | 99.3% |
| car | 14921 | 48242 | | 747036 | 38219 | | | 4 | 3689 | 87.7% | 12.3% |
| natural | 26538 | 111844 | | 2408 | 1124594 | 1816 | 5368 | 6070 | 427 | 87.9% | 12.1% |
| trash can | 8541 | 17361 | | 10059 | 8189 | 38100 | 1948 | 861 | 392 | 44.6% | 55.4% |
| pole | 12457 | 55579 | 63 | 131 | 15762 | 2024 | 97804 | 2081 | | 52.6% | 47.4% |
| barrier | 49472 | 197183 | | 14963 | 65509 | 3052 | 4333 | 63173 | | 15.9% | 84.1% |
| pedestrian | 192 | 1264 | | | | | | | 5872 | 80.1% | 19.9% |
| | 98.6% | 81.3% | 71.9% | 57.7% | 47.1% | 46.3% | 45.4% | 43.3% | 26.1% | | |
| | 1.4% | 18.7% | 28.1% | 42.3% | 52.9% | 53.7% | 54.6% | 56.7% | 73.9% | | |

Figure 5.20: Confusion matrix for a 4 cm sub-sampled edge-only dataset (K 25, Threshold 15) ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

points to that of the small neighbourhood. However, taken in isolation, this does not explain the results. For example, the 25 point neighbourhood has a similar, albeit lower, set of results to small neighbourhood with ground, buildings and natural (foliage) accuracies of 85.37%, 79.04% and 47.44% respectively. The worsening foliage result is also similar to what was seen with the small neighbourhood. It is the result of misclassification into the ground and building classes, and can seen in the confusion matrix shown in 5.20.

What is apparent is that the more defined the edge points and the greater the gaps in the environment, the worse the results become. Again, Figure 5.14 shows that, the larger the neighbourhood, the better the definition of the edges at the expense of flat surfaces. This can also be seen in Figure 5.22 which gives heatmap accuracy results for all the edge-only experiments on a class by class basis, including the mean validation accuracy. The only reasonable explanation for what is being observed is that the very definition of the classes is being defined by an increasing reliance upon edges alone, which is squeezing out the smaller classes in preference the dominant classes of buildings, ground and natural. The associated confusion matrices do show some convergence of false negatives towards the proportionally better represented classes. This is especially evident in the confusion matrix representing the results from the neighbourhood with 125 points, which is shown in Figure 5.21.



| True Class \ Predicted Class | ground | building | natural | car | pole | pedestrian | barrier | trash can | bollard | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ground | 5478349 | 6469241 | 820029 | 3712780 | 134317 | 50819 | 478888 | 319449 | | 31.4% | 68.6% |
| building | 54036 | 8687422 | 617198 | 24112 | 5787 | 1249 | 38497 | 22293 | | 91.9% | 8.1% |
| natural | 16145 | 96961 | 1142374 | 2615 | 5 | 657 | 10902 | 9406 | | 89.3% | 10.7% |
| car | 19918 | 30899 | 19645 | 776095 | 2468 | 2130 | 809 | 147 | | 91.1% | 8.9% |
| pole | 2633 | 130865 | 18880 | 1413 | 29230 | 5 | 532 | 2343 | | 15.7% | 84.3% |
| pedestrian | 10 | 1398 | | | | 5920 | | | | 80.8% | 19.2% |
| barrier | 21467 | 203692 | 110609 | 7208 | | | 54261 | 448 | | 13.6% | 86.4% |
| trash can | 8471 | 27083 | 5388 | 8147 | 1637 | 288 | 5452 | 28985 | | 33.9% | 66.1% |
| bollard | 994 | 15435 | 979 | 1903 | 1964 | | 338 | 742 | | | 100.0% |
| | 97.8% | 55.5% | 41.8% | 17.1% | 16.7% | 9.7% | 9.2% | 7.6% | | | |
| | 2.2% | 44.5% | 58.2% | 82.9% | 83.3% | 90.3% | 90.8% | 92.4% | | | |

Figure 5.21: Confusion matrix for a 4 cm sub-sampled edge-only dataset (K 125, Threshold 15) ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

#### 5.3.2.4 Edge features combined with grid sub-sampled data

The previous experimentation with the real-world data reveals two major issues; not only can a model be rendered useless by too much sub-sampling, but focusing too much on specific features, such as edges, can be problematic. Perhaps unsurprisingly, there is also a correlation between the lack of points available to the model and its performance. A series of experiments have been performed to test whether there is a technique which permits sub-sampling, beyond the breaking point of the model, by retaining some edge points.

In total 280 experiments were performed over a sub-sampling range from 8 cm up to 32 cm, in incremental 4 cm steps. The sub-sampling was performed in conjunction with retaining edge points with a neighbourhood range of 5, 10, 25, 50, 75, 100, 125 and 150. Flatness thresholds for the edge points were set to 5, 10, 15, 20 and 25; where the lower value is conceptually flatter than the higher thresholds. The results for the dataset sub-sampled at 32 cm are described here. For this level of sub-sampling, the retained edges points were sub-sampled at 16 cm. This was performed in order to try and lessen the affect of a dominant set of edge points whilst still retaining a reasonable representation.

The threshold results, shown in Figure 5.23 show an almost consistent level of accuracy, tailing off slightly at the extreme. The results for these can be seen in table B.1 in appendix B;

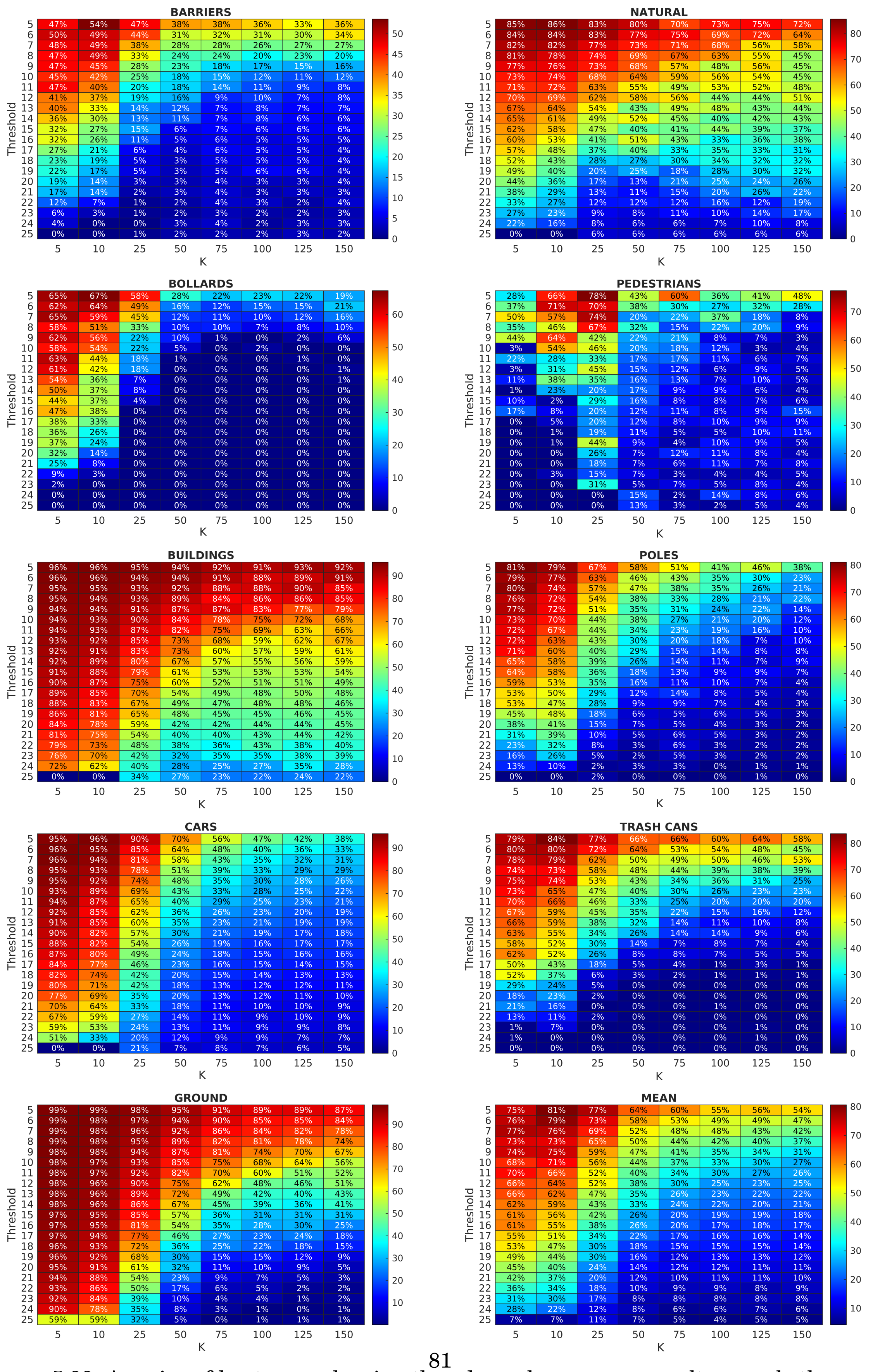


Figure 5.22: A series of heatmaps showing the edge-only accuracy results, over both nearest neighbour and threshold ranges, for each classification. This also includes the mean accuracy.

with the full edge retained validation accuracy results for all sub-sampled levels also given in appendix B. In isolation, it was shown that either dataset performed significantly worse in their experiments. This was a result of either sub-sampling to the point where the model completely failed, or as a result of increasing the flatness threshold (see figures 5.9 and 5.15) and focusing solely upon edges at the expense of everything else.

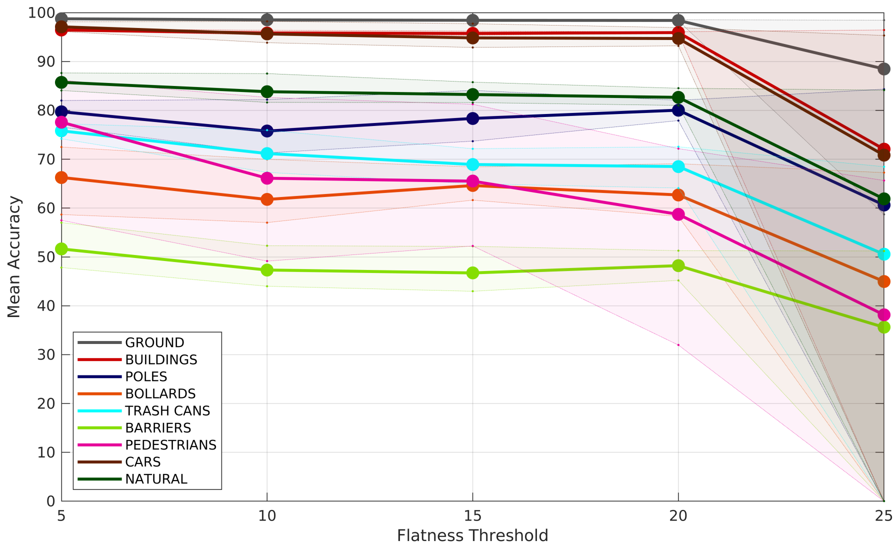


Figure 5.23: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold.

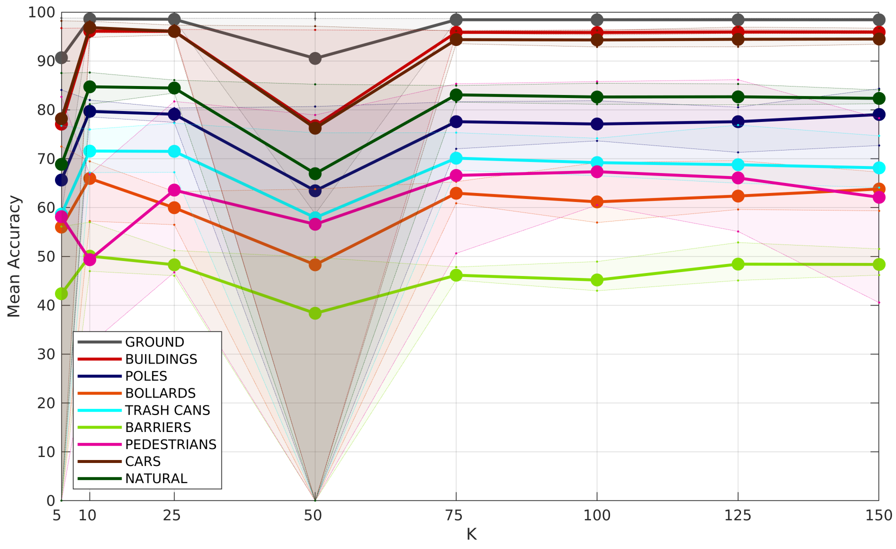


Figure 5.24: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours.

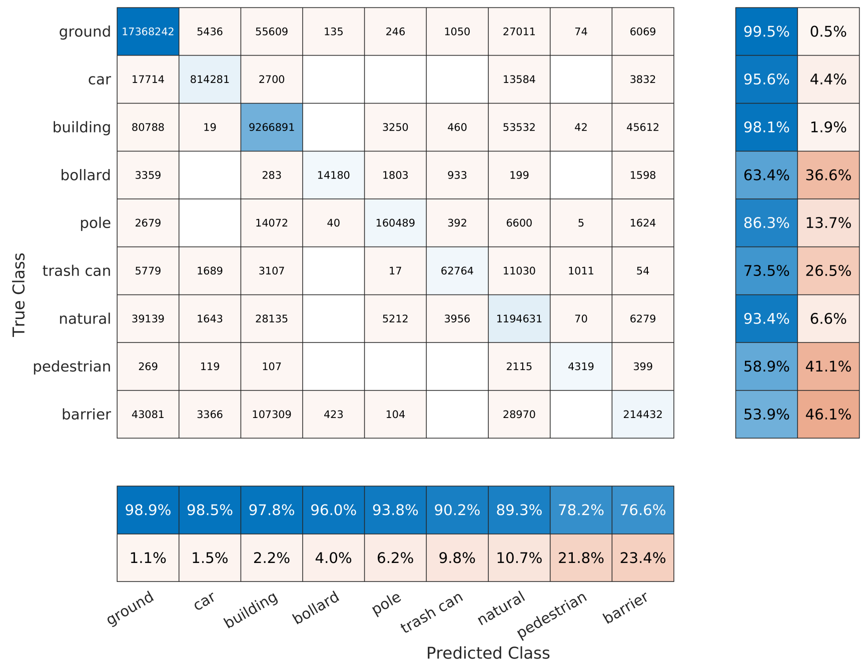
Table 5.18 presents heatmaps which show the percentage of points remaining after sub-sampling. A comparison of the heatmaps for the edge-only percentage points and the 32 cm sub-sampled (which retained edge points sub-sampled at 16 cm) reveals a reallocation of points from the lowest to the middle thresholds. It is this redistribution which helps the accuracy results hold out longer. However, the heatmaps also show there is very limited access to points, circa 1%, when the threshold is set to 0.25. This is despite the edge retaining experiment maintaining a reasonable level of accuracy at this threshold. This suggests that the choice of points retained is critical for the success of the model when presented with harsh sub-sampling. The k-nearest neighbour results, shown in Figure B.2, also demonstrate reasonable and relatively stable mean accuracy across the differing neighbourhoods and reinforce the idea that it is possible to sub-sample at this level. A couple of the highest threshold results did result in the failure of the model and this was attributed to the field of influence problem investigated in section 5.3.2.2. This can be seen in Table B.1 or visually in Figure B.2 where the minimum/maximum background patch fails to zero percentage accuracy.

A confusion matrix for a k-nearest neighbourhood of 75 with a flatness threshold of 0.25 is given in Figure 5.25. Given these neighbourhood and threshold parameters, the model had access to circa 1% of the original point cloud. The corresponding results give an impressive mean accuracy of 75.66%. Problematic classifications include pedestrians, barriers and natural (foliage), which gets confused with buildings and ground due to the smoothing effect of sub-sampling, as suggested in section 5.3.2.2. This is also seen across the board in the corresponding heatmaps given for pedestrians and barriers in Figure 5.26. The confusion between barriers and buildings was already apparent within the benchmark, discussed in section 5.3.2.1 and visualised with a simple example in Figure 5.8.

Figure 5.27 revisits the loss of definition which occurred when sub-sampling at a resolution of 32 cm and show the object has regained some composure as a result of retaining the edges. This was previously shown against the benchmark in Figure 5.12. With retained edges included in the dataset, the pedestrian object shown has regained some composure. Whilst this has resulted in an improvement in accuracy from 10.2% to 50.63%, the pedestrian is also being confused with trash cans. To some extent, this makes sense given the under representation of both classes in the dataset, and relatively close object dimensions. Again, this needs to be framed in the context of having access to circa 1% of the point cloud.

#### 5.3.2.5 Discussion

A series of experimentation was performed to evaluate the success of a state-of-the-art model when presented with significantly sub-sampled real-world data. The first set of experiments focused solely upon finding the limitations of the model. The point-cloud was sub-sampled, using existing grid sub-sampling functionality within the model, from a resolution of 8 cm to 32 cm. This was done in 4 cm incremental steps. The initial results showed reasonable resilience, but a complete failure when the sub-sampling went beyond 20 cm. Further investigation found that



| True Class \ Predicted Class | ground | car | building | bollard | pole | trash can | natural | pedestrian | barrier | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ground | 17368242 | 5436 | 55609 | 135 | 246 | 1050 | 27011 | 74 | 6069 | 99.5% | 0.5% |
| car | 17714 | 814281 | 2700 | | | | 13584 | | 3832 | 95.6% | 4.4% |
| building | 80788 | 19 | 9266891 | | 3250 | 460 | 53532 | 42 | 45612 | 98.1% | 1.9% |
| bollard | 3359 | | 283 | 14180 | 1803 | 933 | 199 | | 1598 | 63.4% | 36.6% |
| pole | 2679 | | 14072 | 40 | 160489 | 392 | 6600 | 5 | 1624 | 86.3% | 13.7% |
| trash can | 5779 | 1689 | 3107 | | 17 | 62764 | 11030 | 1011 | 54 | 73.5% | 26.5% |
| natural | 39139 | 1643 | 28135 | | 5212 | 3956 | 1194631 | 70 | 6279 | 93.4% | 6.6% |
| pedestrian | 269 | 119 | 107 | | | | 2115 | 4319 | 399 | 58.9% | 41.1% |
| barrier | 43081 | 3366 | 107309 | 423 | 104 | | 28970 | | 214432 | 53.9% | 46.1% |
| | 98.9% | 98.5% | 97.8% | 96.0% | 93.8% | 90.2% | 89.3% | 78.2% | 76.6% | | |
| | 1.1% | 1.5% | 2.2% | 4.0% | 6.2% | 9.8% | 10.7% | 21.8% | 23.4% | | |

Figure 5.25: Confusion matrix for a 32 cm sub-sampled dataset with edge retention (K 75, Threshold 25) ordered by positive predictive value. Both row and column summaries are included to show the percentages of true positive outcomes versus false positives (row) and false negatives (column)

the field of influence, or receptive field, was inappropriately set for the levels of sub-sampling the model was subjected to. This meant that the kernel window did not have access to enough points to work with, which resulted in a failure with backward propagation after a few epochs.

When the receptive field was broadened, the model was able to regain composure. However, at a resolution of 32 cm, it was only able to classify objects as either buildings, ground or natural. This limited classification was reflected in the results, with a significant drop in accuracy, when compared to lesser decimation. Nonetheless, this was an important outcome for the integrity of the model. Before the field of influence hyper-parameter was changed, a fundamental problem had emerged from within the results. Whilst it was easy to explain-away the smoothing effects of grid sub-sampling on natural objects, which see them morphing into basic geometric shapes which can be confused with man-made objects; it was not easy to give an explanation to why buildings and ground objects could not be discriminated against. At the limit of sub-sampling, it should only take a small number of points to determine the difference between buildings and the ground i.e. by the orientation of the points, as the two classes are almost perpendicular to each other. By increasing the field of influence, the model was able utilise this distinction. However, a choice was made to restore the hyper-parameter back to the value used by the benchmark. This decision was taken to test the bounds of the model without diverging from the benchmark's initial

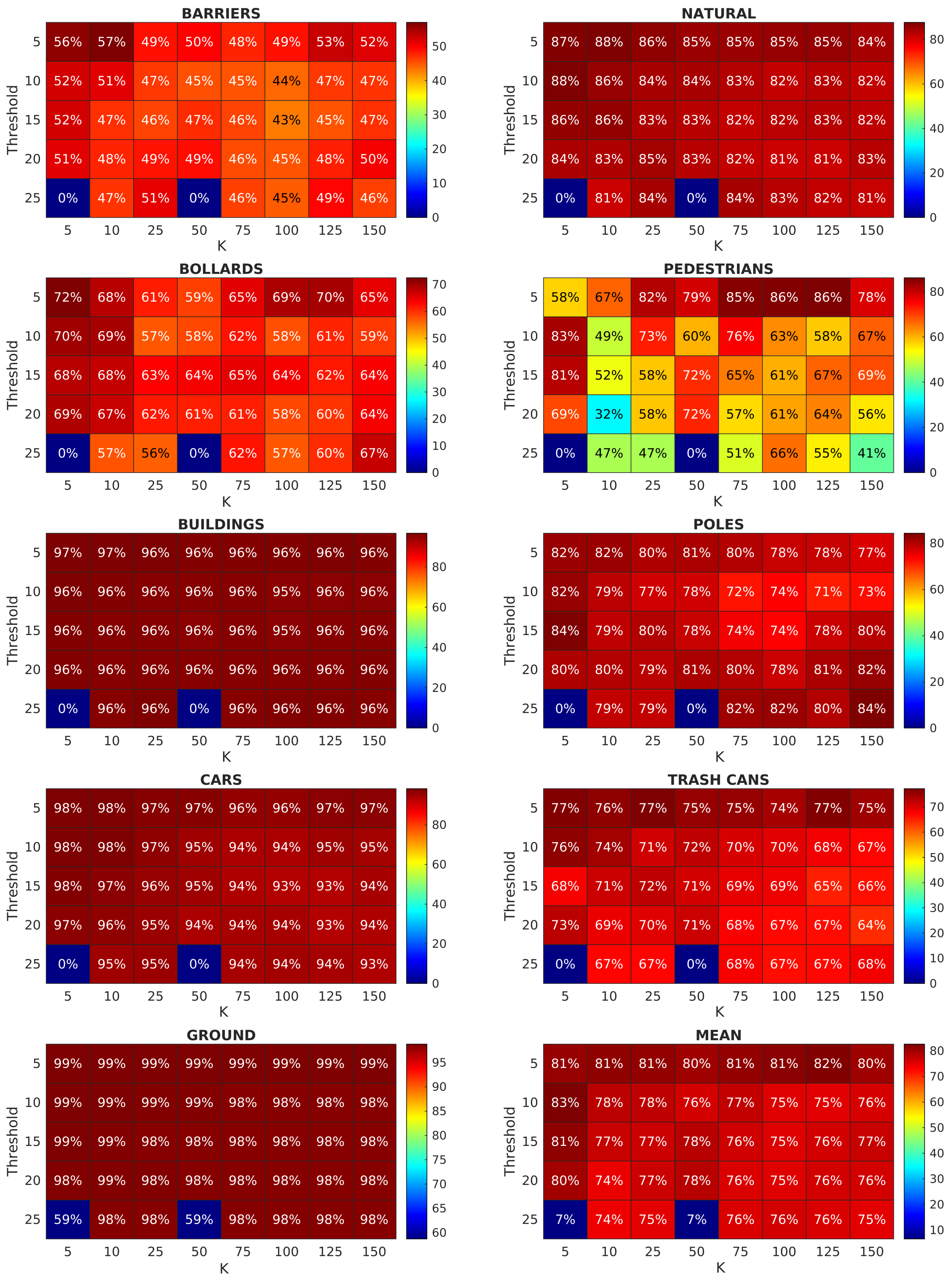


Figure 5.26: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 32 cm) point cloud.

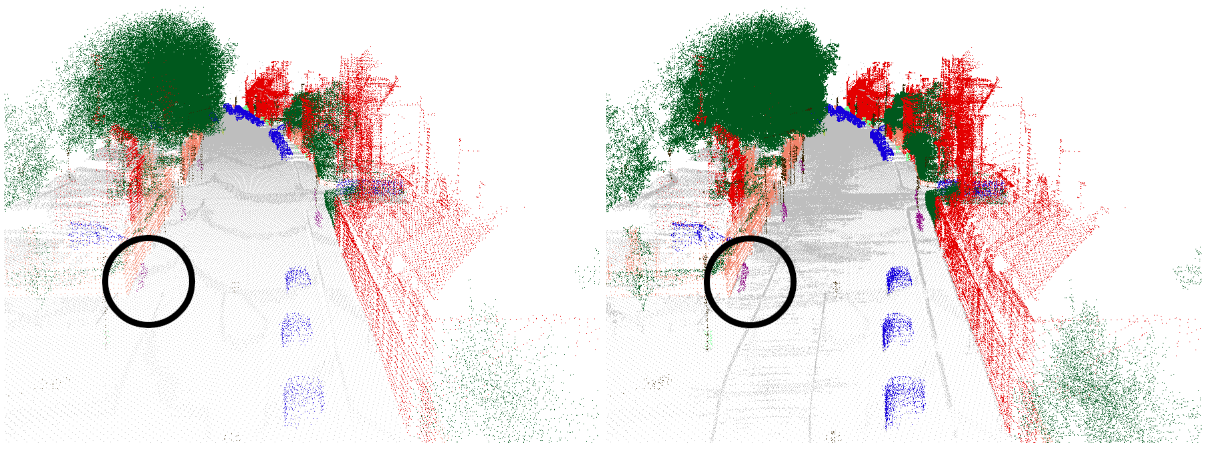


Figure 5.27: A projection of points showing the street-scene with 32 cm sub-sampling on the left, and 32 cm sub-sampling combined with edges on the right hand side.

settings.

The next set of experiments considered how the model performed when given a basic representation of the objects. This was achieved by finding the edge points using a fast edge detection technique which used the eigenvalues of the covariance matrix from the local neighbourhood. A range of flatness thresholds and differing k-nearest neighbouring points gave an insight into how these neighbourhoods elicit edge feature information. It was found that the smallest neighbourhood was too sensitive and did not find reliable edge points, making it no different from a random sub-sampler. This makes sense, as the variance between points was computed on a small sample size and would naturally produce coarse results. A greater sample size of neighbouring points would be more robust to outliers.

The results demonstrated that when the flatness threshold was increased it produced a much better definition of an edge feature. This was to be expected as the algorithm used the threshold against the smallest eigenvalue. A flat surface is defined by having two large eigenvalues representing, for example, the width and height of the plane; whilst, having a small value for its depth. As the threshold is increased, the variance of the smallest eigenvalue would also have to increase in order to be considered an edge. Experimentation revealed that as the definition of edge features improved, there was a corresponding loss of flat surfaces. This left large areas of the environment with little or no points. Given that the objective was to retain only the edge points this is stating the obvious. However, the results of this convergence produced an unexpected outcome, where the model had become increasing reliant upon the edges to classify the predominant, and importantly flat buildings and ground objects. This was done at the expense of the other object classes which would have benefited from these edge features such as trash cans, poles and bollards.

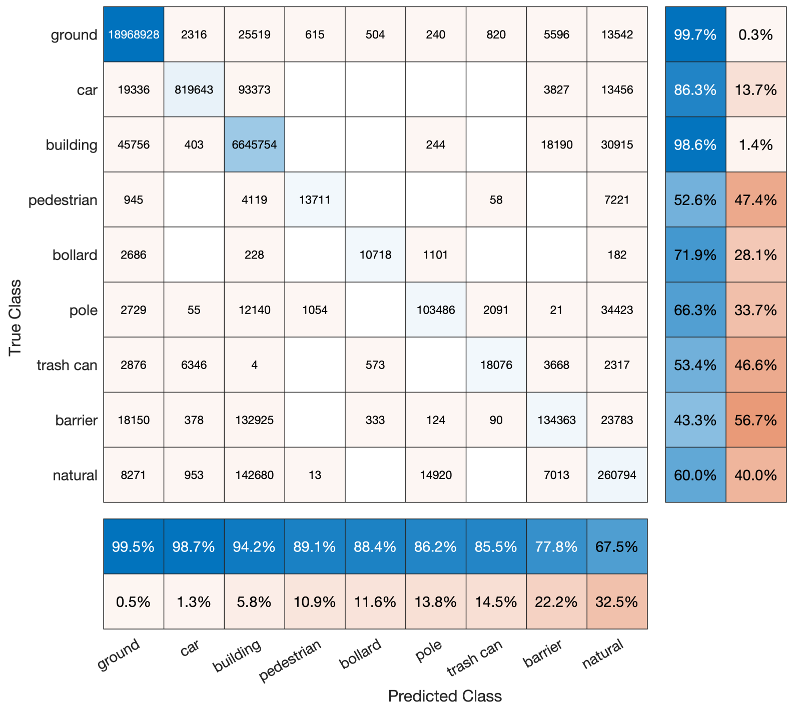
A final set of experimentation was undertaken to determine if a combination of sub-sampling and edge retention could perform better than either in isolation. The sub-sampling component would ensure that flat surfaces got a fair representation. Notably, this could also be achieved using decimation which far exceeded the model's field of influence capability, as flat surfaces do not require many points. The retained edge points were also sub-sampled at a resolution half of the main point cloud. This was performed in order to mitigate any bias which may have existed as a result of clusters of edge points. A benefit to retaining some edge points, was that the model became more resilient to sub-sampling i.e. the edge points prevented the complete collapse of the model. Part of the explanation for this was that the edges were never sub-sampled beyond a resolution of 16 cm, which is within the model's tolerance. In context with experimentation which went beyond a resolution of 20 cm and subsequently failed, it can be assumed that the model is using the edge points as some form of anchor point in which it discovers additional, flat surface information.

The combination of sub-sampling whilst retaining edge points was found to perform reasonably well when compared with the benchmark. Most accuracy results of the combined experiments were just under 80% versus the benchmark of 87.34%. Importantly, these results were achieved with significant reductions in the number of points used; with some experiments using less than 5% of the point cloud used to establish the benchmark. To highlight the above point, the experiment, discussed in 5.3.2.4, which set the k-nearest neighbour to 75 with a flatness threshold of 25 was submitted to the same benchmarking suite [90] on the 29th December 2020, where the original authors of the KPConv model were ranked third. This experiment used less than 5% of the 4 cm sub-sampled dataset which the KPConv authors had used. Furthermore, it had sub-sampled at a resolution of 32 cm, which had been shown in section 5.3.2.2 to break the model when using the same set of hyper-parameters. The confusion matrix and results of that submission are presented in Figure 5.28 and Table 5.8. The results show a mean test accuracy of 69.9% and a ranking of tenth.

Table 5.8: Public submission of a dataset which has been sub-sampled at a resolution of 32 cm with edges retained. Rank shown as 10th overall.

| | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
|---|---|---|---|---|---|---|---|---|---|---|
| Result (%) | 69.90 | 99.20 | 92.90 | 60.00 | 65.70 | 49.00 | 38.60 | 48.60 | 85.40 | 89.70 |
| Rank | 10 | 11 | 12 | 11 | 12 | 7 | 12 | 9 | 10 | 8 |

This result is lower than that presented in Table B.1, which is given as 75.66%. The explanation for this was given in section 5.3.2.1, which stated that validation data would be used for evaluation because the NPM3D test data contained no ground-truth. The only way to use the test data was to submit it to the aforementioned benchmarking suite; which permitted one submission every five days, making it impractical for these experiments. Therefore the 69.9% result reflects real test data.



| | ground | car | building | pedestrian | bollard | pole | trash can | barrier | natural | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ground | 18968928 | 2316 | 25519 | 615 | 504 | 240 | 820 | 5596 | 13542 | 99.7% | 0.3% |
| car | 19336 | 819643 | 93373 | | | | | 3827 | 13456 | 86.3% | 13.7% |
| building | 45756 | 403 | 6645754 | | | 244 | | 18190 | 30915 | 98.6% | 1.4% |
| pedestrian | 945 | | 4119 | 13711 | | | 58 | | 7221 | 52.6% | 47.4% |
| bollard | 2686 | | 228 | | 10718 | 1101 | | | 182 | 71.9% | 28.1% |
| pole | 2729 | 55 | 12140 | 1054 | | 103486 | 2091 | 21 | 34423 | 66.3% | 33.7% |
| trash can | 2876 | 6346 | 4 | | 573 | | 18076 | 3668 | 2317 | 53.4% | 46.6% |
| barrier | 18150 | 378 | 132925 | | 333 | 124 | 90 | 134363 | 23783 | 43.3% | 56.7% |
| natural | 8271 | 953 | 142680 | 13 | | 14920 | | 7013 | 260794 | 60.0% | 40.0% |
| | 99.5% | 98.7% | 94.2% | 89.1% | 88.4% | 86.2% | 85.5% | 77.8% | 67.5% | | |
| | 0.5% | 1.3% | 5.8% | 10.9% | 11.6% | 13.8% | 14.5% | 22.2% | 32.5% | | |

Figure 5.28: Confusion matrix for public submission of a dataset which has been sub-sampled at a resolution of 32 cm with edges retained.

## 5.4 Summary

This chapter has investigated the performance of deep learning architectures used to classify geometric point cloud data captured by LIDAR. An overview of relevant architectures was given before two architectures were selected for experimentation. It was hypothesised and shown that the, albeit dated, PointNet architecture behaved like a spatial filter by its ability to find the underlying structure of the object.

It was further hypothesised that it should be possible to gain an improvement in processing speed, without a significant reduction in classification performance, by discarding superfluous points prior to processing by the network. This hypothesis was found to be broadly true, yielding significant performance improvements whilst retaining accuracy within 2% of the benchmark. A final hypothesis, concerning PointNet's ability to become a sub-sampler for use in a more state-of-the-art model was disproved. This was because PointNet could only operate with relatively simple environments. This limitation led to a poor choice of synthetic dataset which did not reflect real-world conditions; including the uneven distribution of points, inclusion of noise and point omissions.

Further concern was raised about the strength of discrimination between classes, given that the model was able to classify to an accuracy above 80% whilst using as little as 32 points from an original 2,048 point object. This suggested that the objects were too diverse. It was also questionable whether the model defined its classification space by geometric properties such as the scale of the object, rather then the shape of it. An example asked which type of features defined the differences between an aeroplane and a table; the shape of a wing or the scale of the aeroplane.

This led to a series of experimentation using a state-of-the-art model with real-world data. Again, sub-sampling was the main focal point of these experiments but with emphasis on intelligently decided which points to keep. A benchmark was established prior to subjecting the model to using heavily decimated datasets, which were found to take it beyond its capability. Modifying the architecture to retain edge features allowed the model to achieve accuracy results of circa 80% (range 74.1% to 82.55%) for the most decimated point cloud, compared to the benchmark of 87.34%. This was achieved with a fraction of the points used, sometimes as low as 5%.

Whilst all of the results show some form of degradation in their classification accuracy, this has to be put into context of the problem which is being solved. That is, the acceptance that the dynamic nature of the environment, which is the ultimate target, does not necessarily require high levels of accuracy when used by a RF propagation model. Pedestrians, vehicles, new buildings and the effect of seasonal variation of nature all suggest that the output must possess some degree of tolerance. Given these examples, the environment can never be truly captured. Therefore, a slight compromise in accuracy, which sub-sampling inevitably leads to, does not negate its validity for use in this application. One benefit of sub-sampling, is that it led to significant improvements in classification speed because the model had fewer points to process.

The next chapter will use the combination of sub-sampling and edge feature retention as part of a mesh generating pipeline, suitable for use within a propagation model.

# Chapter 6

# Automatic clutter classification pipeline

Deep learning has been shown to produce reasonable classification results of street scenes captured using LIDAR. This chapter will demonstrate that, whilst it may not deliver the highest levels of accuracy, the use of deep learning is a good alternative to traditional hand-labelled classification when applied to RF propagation modelling. This is because the propagation tool must be tolerant of an ever-changing physical environment. To demonstrate this, the chapter will define and evaluate an end-to-end process which is capable of i) performing automatic clutter classification of LIDAR, and ii) the subsequent preparation necessary for the labelled data to be used as an environment within the ProPhecy 3D multi-element modelling tool [125].

The chapter will provide a brief overview of the raw LIDAR data used for testing this process. This includes a small hand collected test sample, the previously evaluated NPM3D dataset [91] (see section 5.3) and a hand collected 460 metre section of Woodland Road, Bristol, UK. It will give the motivation behind the choice of data and discuss, where applicable, how it was captured; highlighting any benefits and short-comings.

The process of classifying the data and preparing it for use within a propagation model will be described. This includes the classification of the point cloud, creation of individual class meshes and a novel method to create a better and simpler mesh-based representation of complicated structures like foliage.

A subsequent series of evaluation will be discussed which will review the differing outcomes, including providing visual interpretations, from using the pipeline created data within the propagation model. This will be based upon simulations of a person carrying UE as they walk past a base-station. It will include comparisons between existing hand-crafted ground-truth environment data, automatically classified environments and scenarios where a single generic

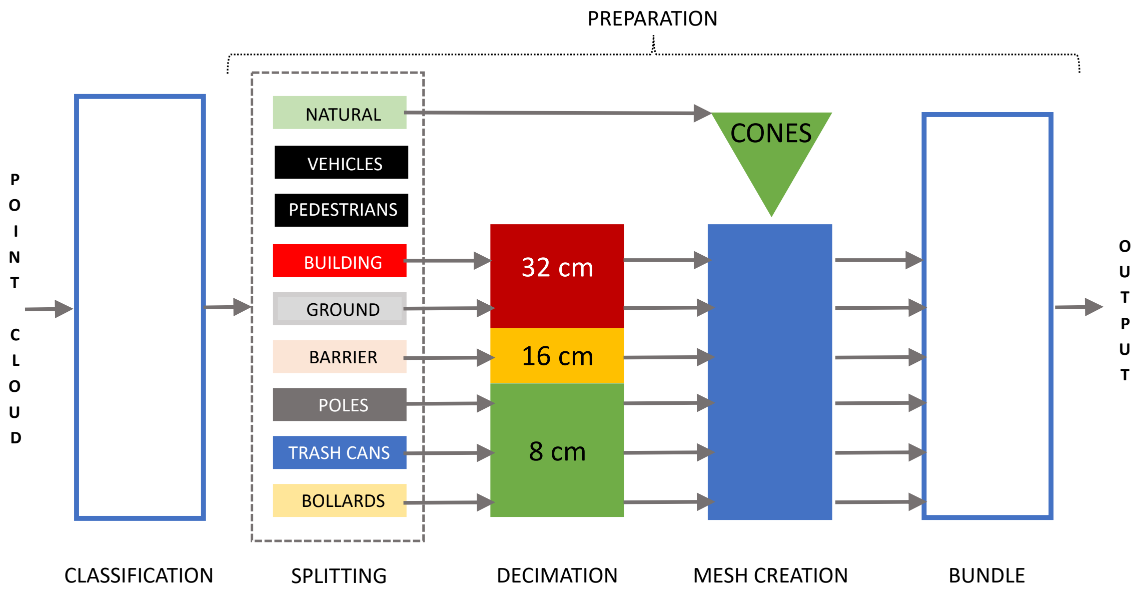


Figure 6.1: Block diagram of the automatic clutter classification pipeline.

material has been used to represent the environment. The defintion of the type of ground-truth used can be found in section 3.3. For the avoidance of doubt, the ground-truth utilised within the ProPhecy 3D multi-element modelling tool [109] consists of labelled polygon data and does not include and RF measurements. The generic material assigned was concrete whose characteristics restricted scatter and reflection.

The propagation experiments will suggest that the existing ground-truth data performs better than the classified data. However, it will discuss a number of reasons for this. Single generic material propagation testing will discussed. These will be found to be the most revealing set of tests. This is because they ask the most fundamental question of the research; whether a classification pipeline is actually needed or is it sufficient for raw LIDAR to be used.

## 6.1 Definition of the pipeline

An overview of the automatic clutter classification pipeline is given in Figure 6.1. It shows five main components split between two operations. The first operation is to classify the raw point cloud data, this is performed by the classification component and is discussed in section 6.1.1. This also includes a description of the data which was selected for input into the pipeline, and the model used. The remaining four components are post-classification processes and prepare the data for consumption by the propagation tool. The diagram illustrates that these components split the point cloud by label into a group of point clouds, reduce those of interest by sub-sampling them by a level deemed appropriate for their class type, create meshes from the sub-sampled

Figure 6.2: GeoSlam ZEB Go hand-held rotating LIDAR scanner [122].

results and bundle them into a package. These components are discussed further in section 6.1.2.

### 6.1.1 Data collection and its classification

The pipeline begins by accepting raw point cloud data as an input into the classification component. Classification is performed by the KPConv deep learning model. This model was previously discussed and evaluated in section 5.3; the outcome of which demonstrated that it was capable of providing good classification of street scene LIDAR from the NPM3D dataset [91], hence its use within the pipeline. For the purposes of a working pipeline, the KPConv model was trained using the NPM3D dataset using data captured from Paris and Lille, France. Training was performed with sub-sampled data at either 4 cm or 32 cm resolution. The sub-sampling set out to determine if there was any significant impact on the propagation model's results.

#### 6.1.1.1 Proof of concept

A small point cloud was captured detailing a couple of houses in a residential estate in the UK. It was used to visually evaluate how well the model performed with a slightly different environment, and not just French architecture. It was captured by the industrial sponsor of this thesis, Dr Ian Hindmarch, using a 'GeoSlam ZEB Go' hand-held rotating LIDAR scanner [122] which is capable of capturing 43,000 points per second with a range of 30 metres and field of view (FoV) of 360°in rotation and 270°to the sides. Furthermore, the ZEB Go has SLAM capability and is shown in Figure 6.2.

Figure 6.3 shows both the raw data captured using the LIDAR scanner and the resultant labelling from using the trained KPConv model which sub-sampled the raw data at a resolution of 4 cm. Whilst the labelling offers much clarity, it does highlight a couple of issues. The main being the confusion between buildings and barriers. A similar image (see Figure 5.8) was used in section 5.3.2.1 to illustrate this as a general problem, as this issue was also found to be present in the processing of the NPM3D dataset. Barriers do have similar characteristics to buildings within these datasets, they are both vertical and reasonably flat. However, there are a few properties which the model does not appear to be able to capitalise upon, including obvious height differences and, in the case of suburban houses, their shape e.g. the apex at the roof line and windows; although to be fair there are no windows embedded within the walls which were misclassified as barriers. Nonetheless, as permanent barriers and buildings have similar physical characteristics, it should not adversely affect propagation modelling, if they are considered the same material. This is an assumption based upon viewing the confusion matrix in Figure 5.7, section 5.3.2.1, which reveals that actual barriers have a minority representation within the environment; and of them, even less would be permanently installed. Relatively, the barrier class has a representation of 3.4% of what is available to the buildings class.

The issue with the misclassification of buildings and barriers extends to the model's confusion between them on the very same wall. This demonstrates that the model is not considering the object in its entirety. This certainly gives a plausible explanation into why the model is not able to use height to discriminate against the classes. Surprisingly, it is debatable whether this is a good thing or not. In the above example, it would be a benefit for the model to take a holistic view, and classify the structure as either one or the other. However, a more subtle classification effect, which can also be seen in Figure 6.3, shows foliage growing over a wall. Given the two very differing physical characteristics of these objects, it would be hard to justify taking such a holistic approach, especially if that resulted in foliage masking an entire wall and lessening its physical impact by making it somewhat transparent. This also vindicates the aforementioned approach to treat barriers as buildings within the propagation model as it is clear that, even with hand labelled data, foliage growing over a wall would not capture the environment accurately i.e. there is a need for tolerance within propagation modelling, irrespective of the labelling technique, because the model cannot look beyond the first thing it sees.

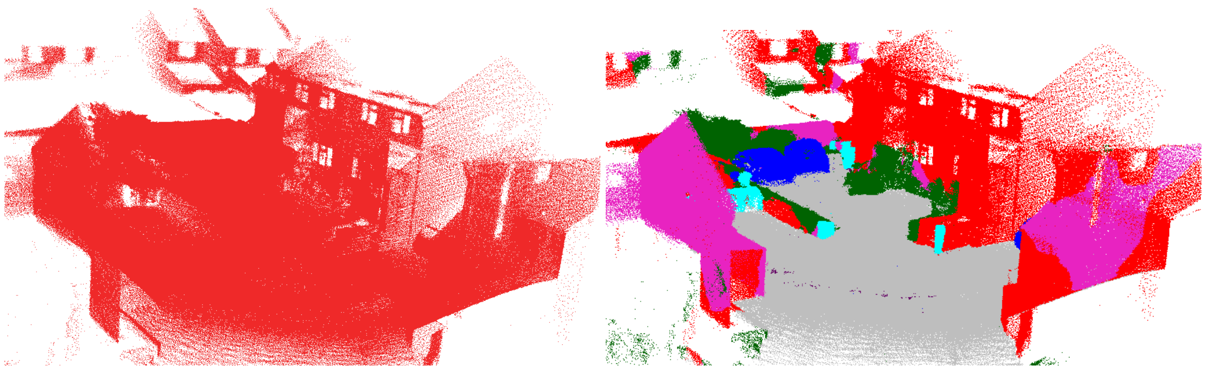


Figure 6.3: A small sample of suburban property captured for use as a proof of concept.

#### 6.1.1.2 Sources of data

The NPM3D dataset is an obvious source of data for the pipeline. Previous experimentation in chapter 5 already provides labelled point clouds which have been sub-sampled at the required resolution, together with the corresponding accuracy results and all without any additional effort. In essence, allowing the pipeline to bypass classification. Furthermore, the dataset contains the hand-labelled ground-truth which was used to train the pipeline's classification model. In this instance, no further classification is required and the ground-truth can be used to form an important baseline in order to compare with the related sub-sampled areas.

Figure 6.4 shows three labelled point clouds: i) the hand-labelled ground truth taken from the NPM3D dataset, ii) the classified benchmark which was discussed in section 5.3.2.1 and recreated the original scenario submitted by the authors of KPConv; and, iii) a classified experiment which used a resolution of 32 cm with retained edge points, sub-sampled at 16 cm. This sub-sampled experiment was discussed in section 5.3.2.5. The benchmark and sub-sampled point clouds have a validation accuracy of 87.34% and 75.66% respectively. These results were created as part of the investigation into using sub-sampling in discussed in sections 5.3.2 and 5.3.2.4.

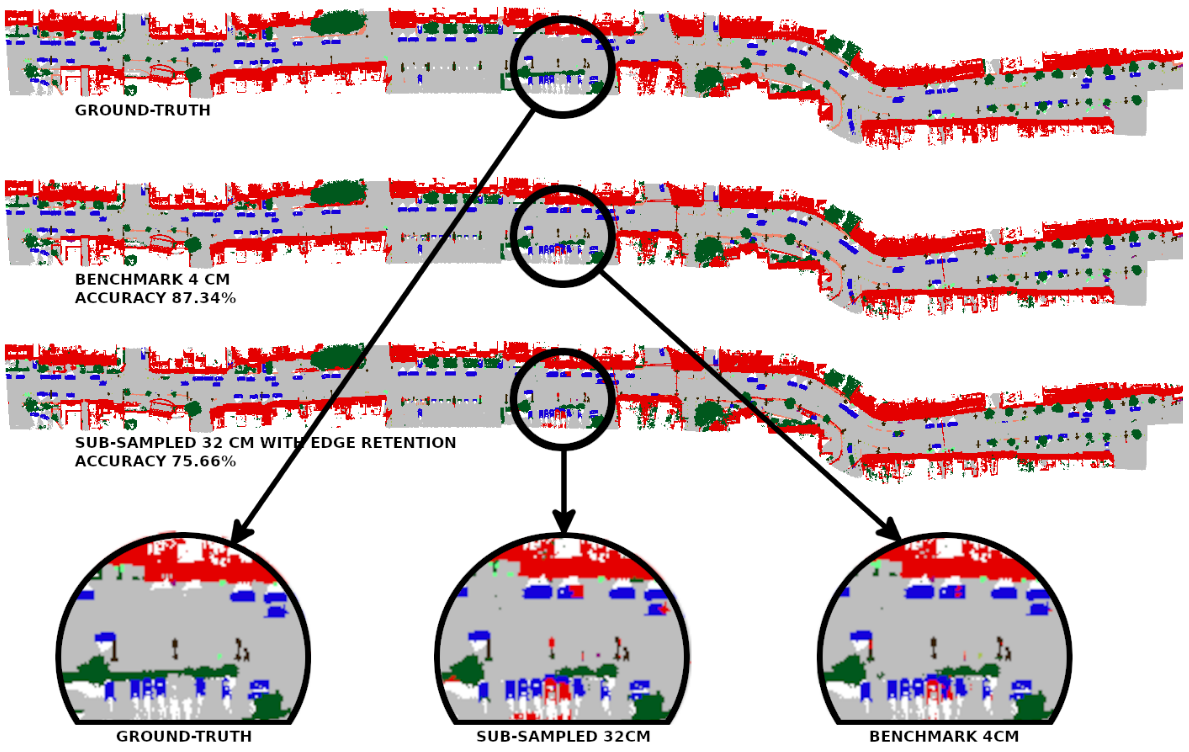


Figure 6.4: NPM3D classified point cloud data showing hand-labelled ground-truth (top), the benchmark and a sub-sampled version which retains edge points as discussed in section 5.3.2.5.

To highlight the differences, the figure also includes a zoomed in section from each. Surprisingly, the ground-truth has unclassified areas of the point cloud, which are labelled by the model. This can be seen by the four missing vehicles, which were given away by the LIDAR shadows. It implies that the model is performing slightly better than the accuracy figures suggest because the ground-truth has minor inaccuracies. Unfortunately, the sub-sampled version has misclassified a few vehicles as buildings.

To test the end-to-end process of the pipeline a new dataset was created. This was captured using the 'GeoSlam ZEB Go' hand-held rotating LIDAR scanner, previously used to obtain the proof of concept point cloud. The location of the dataset is a 460 metre section of Woodland Road, Bristol, UK. This was chosen because the propagation model already includes access to OS hand-labelled data, making it suitable for meaningful comparison. As with the proof of concept capture, the data has no associated ground-truth and has been validated by eye. It was therefore important for the propagation model to contain its own ground-truth to test against.

Scanning involved walking up one side of the Woodland Road and walking down the other, relying on the functionality of the scanner's SLAM. Figure 6.5 shows the captured point cloud. It has been coloured by capture time where blue is the start point, green marks the moment of crossing the road to then walk back down the other side and red marks the end point of the capture.

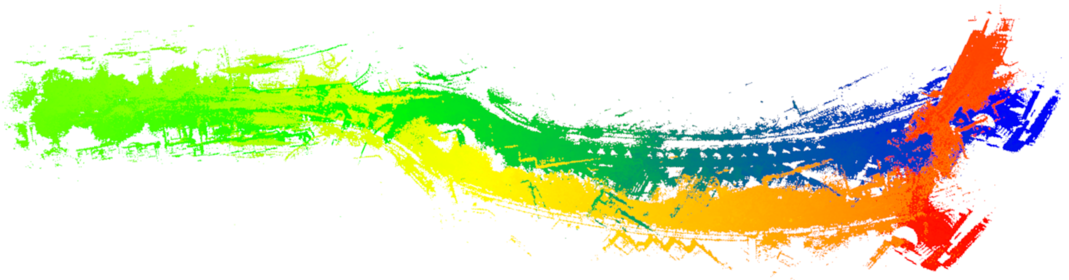


Figure 6.5: Raw LIDAR 460 metre section of Woodland Road, coloured by capture time.

Unfortunately, the SLAM functionality was unable to reflect the actual journey made. Consequently, the image shows the creation of a separate road, shown in yellow. Figure 6.6 highlights the problem by showing a zoomed-in section where the buildings, shown in blue, appear in the middle of the road. In reality, all blue points should be at the bottom of the image, with the yellow points at the top. One way to confirm this is to look at the shadows cast (or voids) on the objects which resemble cars. These should point towards the middle of the road and not away from it. A crude realignment was performed by first separating the point cloud into two. This was performed at the time-stamp where the road was crossed, representing the return point. One section was then rotated to overlay same structures on top of the other section. The aligned point cloud is shown in Figure 6.7.

An explanation for the failure to map the road correctly may be related to the range of the scanner, or more specifically, how range was affected by the technique used to scan the

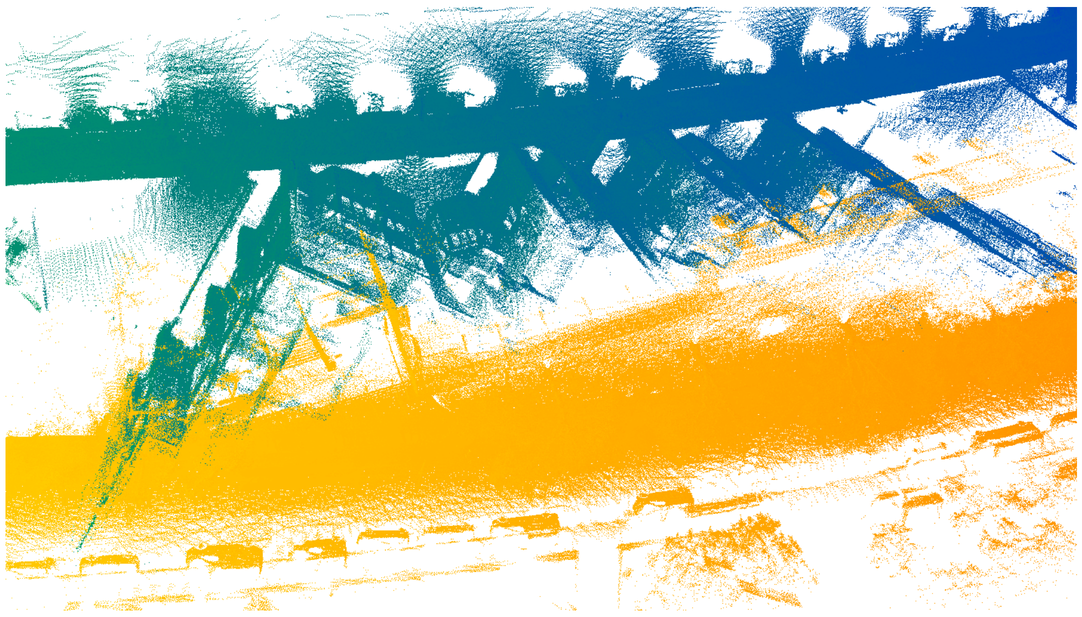


Figure 6.6: Zoomed in section of the raw LIDAR captured of Woodland Road.

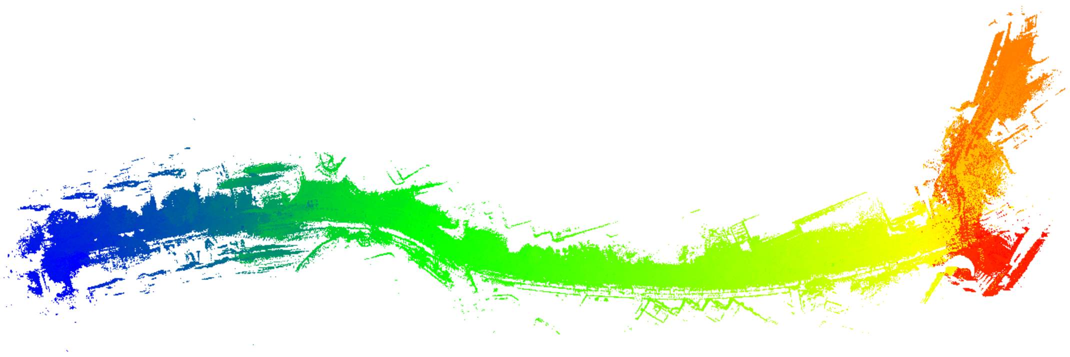
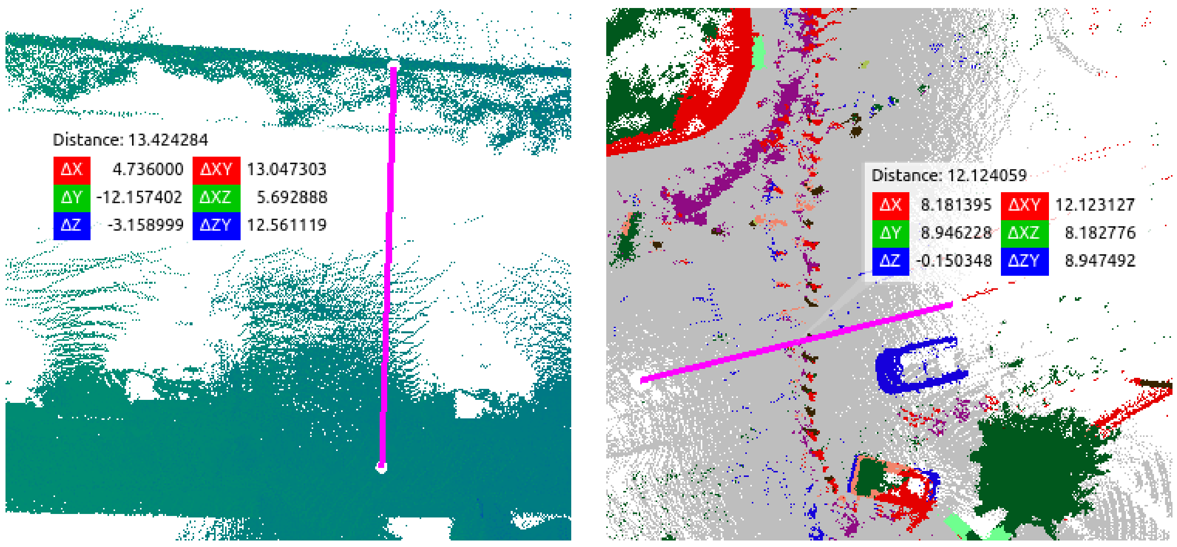


Figure 6.7: Realigned 460 metre section of Woodland Road.

environment on this occasion. As stated earlier, the 'GeoSlam ZEB Go' has a reported range of 30 metres. However, the effective range, during the capture, appears to be under 15 metres. This is shown in Figure 6.8 which measures the distance from the position of the scanner to the approximate extent of its capture. The discrepancy is because the scanner was never directly pointed across the road. Instead the equipment was used in a sweeping motion, predominantly pointing in the direction of travel, up the road. It cannot be ruled out that lack of range affected SLAM, especially if it was reliant upon anchor points, which is hinted at by the manufacturer's literature which suggests their algorithm uses scan lines to tackle difficult environments such as repetitive corridors [29] i.e. the domain is not mapped solely from GPS or accelerometer sources, but includes an element of taking visual cues from the scanned data.

The classification results from the aligned Woodland Road point cloud are shown in Figure


Distance: 13.424284
ΔX 4.736000 ΔXY 13.047303
ΔY -12.157402 ΔXZ 5.692888
ΔZ -3.158999 ΔZY 12.561119
Distance: 12.124059
ΔX 8.181395 ΔXY 12.123127
ΔY 8.946228 ΔXZ 8.182776
ΔZ -0.150348 ΔZY 8.947492

Raw LIDAR data Classified point cloud

Figure 6.8: Ruler measurements giving an indication of how the scanning technique used affected range.

6.9, with sections highlighted in figure 6.10. There is no ground-truth associated with this data and consequently, inspection was done by eye. The top-down view shown on the left hand side reveals two issues, one concerning the technique used to capture the data. The other, is caused by a slight misalignment left-over from when the two sections of data were realigned. Both issues concern the abundance of pedestrians captured who are shown, mainly, in purple. This is explained by how the data was collected, which involved the industrial sponsor walking approximately one metre behind the scanning activity. It appears that the sweeping technique performed inadvertently captured them. This simple error seems likely given the 'GeoSlam ZEB Go' scanner has a stated 270°FoV at the sides. Fortunately, most pedestrian objects are classified correctly, meaning they can be removed from the landscape further down the pipeline. However, the misaligned pedestrians crossing the road are misclassified as either bollards or buildings, and ultimately need to be cleaned from the environment.

The right hand side of Figure 6.10 highlights another classification error. It shows the incorrect labelling of a bus, which passed as the environment was being scanned. From visual inspection, the NPM3D training data does not appear to include large vehicles within the dataset, with the largest being a van. Even if it did, they would represent a small subset of the vehicle class. Furthermore, the rectangular definition of the bus, and the fact that it was moving away from the scanner, makes it easy to understand why it could be misclassified as a building; even if this result is a little disappointing. Moving objects in general pose a problem for the classifier as they are captured multiple times in slightly different locations. Fortunately, objects on the road can be hand-cleansed relatively easily from the environment, on the basis that they are very likely to be temporary structures e.g. vehicles. This also applied to misclassified pedestrians and

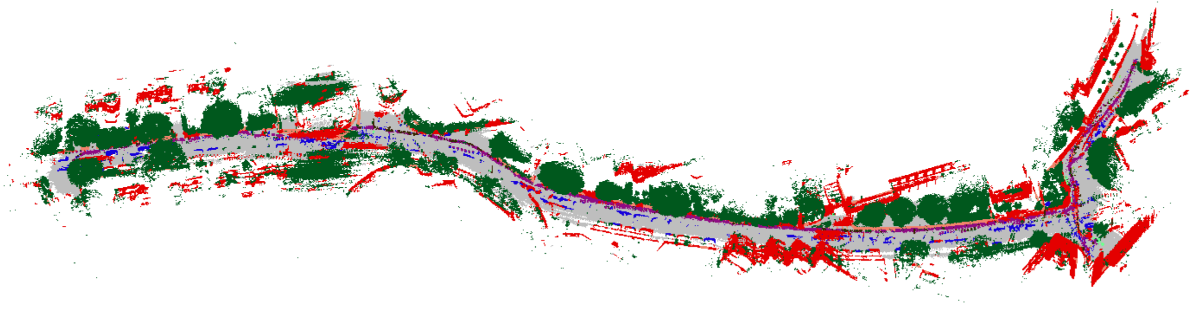


Figure 6.9: Classification result for Woodland Road, sub-sampled at a resolution of 32 cm with edge point retention.

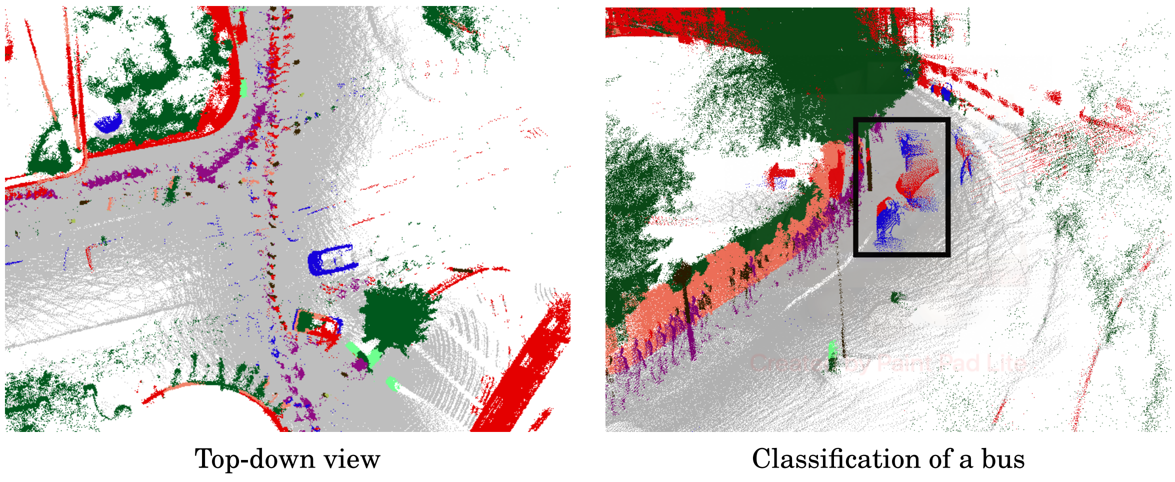


Figure 6.10: Sections from the Woodland Road classification result. The bus is show in the black rectangle, moving away from the scanner.

cars. Overall, these classification results look reasonable to the naked eye, especially considering the classifier achieved accuracy results of 75.66% when evaluating the NPM3D dataset, from an similar set-up.

### 6.1.2 Preparation for use in a propagation model

The second major section of the pipeline involves processing the classified output into a form suitable for the propagation tool to be able to use it as an environment. A number of significant steps need to be performed including splitting out the different classes, sub-sampling the point cloud and converting it into a mesh such that the environment is simplified.

#### 6.1.2.1 Splitting the classes

Splitting the classified point cloud by the creation of a point cloud for each class allows subsequent filtering of the data to remove classes that are not relevant to the model, such as the removal of vehicles and pedestrians. Furthermore, differing levels of attention can be given to the individual classes which remain. This will be seen with further sub-sampling and how natural objects such as foliage are handled. Splitting was achieved using a simple piece of Python code which filtered the classified point cloud by label.

#### 6.1.2.2 Sub-sampling/decimation

Even though sub-sampling formed an important part of classification, the actual results reflect every point in the original dataset meaning the point cloud contains the original number of points. This was achieved by the up-sampling section, performed at the latter stages, of the KPConv architecture.

One objective of the pipeline is to deliver the simplest representation of the environment which it can to the propagation tool. Class specific sub-sampling partly achieves this objective because any subsequent mesh created is based upon fewer points. Examples of where sub-sampling can have a large impact include decimating point clouds containing buildings and roads. This is because the points of these classes are typically very large but can be represented by simpler geometric shapes. On the other hand, performing less sub-sampling retains the intricate details of smaller features such as bollards.

The grid sub-sampling code used within the KPConv model was re-used to achieve class specific decimation. Building and roads were sub-sampled at a resolution of 32 cm, barriers at 16 cm with bollards, trash-cans and poles sub-sampled at 8 cm. The approach taken to determine this was to sub-sample all classes at a resolution of 32 cm and visually inspect them. Those which appeared to be broadly represented were left at that resolution, the others were sub-sampled again at 16 cm and then 8 cm. Foliage was also sub-sampled at 32 cm and is discussed further in section 6.1.2.4.

#### 6.1.2.3 Mesh creation and simplification

With the individual point clouds sub-sampled at the required resolution, meshes were created. This used a C++ implementation of the Greedy projection triangulation algorithm [61], available from the Point Cloud Library (PCL) [78]. Greedy projection looks for fringe points within a list of given points based on parameters including the maximum nearest neighbour, radius and surface angles. A radius of 5 metres was selected to reflect what seems to be a reasonable chunk of a road, with a maximum of 50 nearest neighbours. This allowed decimated point clouds to extend further whilst keeping the smaller objects constrained by the maximum number of points permitted. The surface angle was left at its default 45°setting. This was done to cater for the gradients within the terrain.

The resultant meshes were then simplified using the ‘Quadric Edge Collapse Decimation’ procedure available in MeshLab [66]. This was chosen because it is able to preserve the boundaries and was set to collapse the meshes by 99%. Meshlab includes a command line method for executing this procedure allowing it to be added to the pipeline script, although most mesh tools can perform similar simplification.

#### 6.1.2.4 Foliage clusters

Even with mesh simplification, natural objects such as foliage created too many mesh faces and appeared very spiky/noisy. A novel approach was taken to find a better representation of such objects. This involved finding the centroids of the earlier sub-sampled point cloud using k-means clustering. A trial and error approach was taken to determine a reasonable number of centroids to find. The final value selected was 3,000 centroids representing approximately 3 objects per metre, on each either side of the road; although in reality, the centroids were not evenly distributed.

These centroids, as well as the separated natural class point cloud were then loaded into Blender [15]. A Python script was then created within the Blender scripting section to create upside down cones at each centroid. The cone object was selected as it gives a rough representation of plant/tree objects when the cone is upside down. The in-built shrink wrap modifier function was then called from within the script. This wrapped the cone shape around the local points of the point cloud. These deformed cones where then used to represent foliage and are shown in Figure 6.11 which also shows the volume of relevant labelled points and the centroids found.

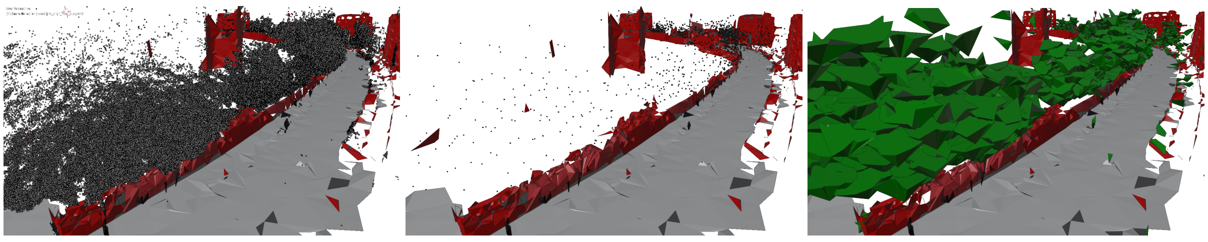


Figure 6.11: Illustration of the process of creating foliage cones.

#### 6.1.2.5 Final Blender file

The final meshes were then packaged up as a Blender project before being given to the industrial sponsor for evaluation within the propagation tool. Specifically, the following data was included:

- The ground and building classes sub-sampled at a resolution of 32 cm and converted to meshes.
- The barrier class sub-sampled at a resolution of 16 cm and converted to meshes.
- The bollards, trash can and pole classes sub-sampled at a resolution of 8 cm and converted to meshes.
- The cone meshes created to represent the foliage.

In total, five blender bundles were created for evaluation as environments within the propagation tool. Three are from the NPM3D dataset and possess varying degrees of quality, starting with the hand-labelled ground-truth through to a heavily sub-sampled point cloud. Two are from the data captured from Woodland Road, Bristol; where one is a cleaned version of the other. Specifically the bundles are:

- The ground truth of the NPM3D dataset.
- The benchmark experiment of the NPM3D dataset.
- The highly sub-sampled experiment with edge retention, also from the NPM3D dataset.
- The highly sub-sampled experiment with edge retention version from Woodland Road.
- A cleaned version of the Woodland Road which removed the bus and other artefacts present on the road.

Figure 6.12 shows the NPM3D related meshes and Figure 6.13 shows the meshes created from the cleaned version of Woodland Road. This removed objects (by hand) on the road, such as the bus shown in Figure 6.10, which were misclassified. All bundles removed objects labelled as either pedestrians, vehicles or unclassified. A section of the Woodland Road mesh can be seen in Figure 6.11 which represents shrink-wrapped cones as foliage. Figure 6.14 shows sections of each of the NPM3D meshes for comparison purposes, and show negligible differences. The bottom right hand corner of the three images differs slightly. The ground-truth describes that section as having a barrier (partition wall) with foliage growing on top. This is confirmed by Figure 6.15 which shows the hand-labelled ground-truth for that area, before being converted into a mesh. The benchmark classifies the partition wall as road, whilst the sub-sampled mesh labels it correctly as a partitioning wall. Both the benchmark and sub-sampled mesh see far more of the building behind the partition wall than the ground-truth. The explanation for this is that the ground-truth was hand-labelled and never classified using KPConv; whereas, the other two were.

Another difference, which reinforces the idea of sensitivity, or at least differing perceptions of the environment, is seen by the red power line crossing over the road in both the benchmark and sub-sampled meshes. Whilst well camouflaged, one even crosses above the aforementioned partitioning wall. These power lines are absent from within the ground-truth mesh. Again, suggesting the classifier has slightly better accuracy figures than reported in section 5.3.2. Figure 6.15 shows that these lines are not classified (light grey) in the ground-truth point cloud, explaining why they are missing from this mesh. However, these lines exist and have been classified by KPConv, even if they are misclassified as either ground or building. The dark grey patches in the road of each image in Figure 6.14 represent LIDAR shadows cast by the presence of vehicles during the scan i.e. areas where there are no points recorded because objects blocked the scanner. These can also be seen with the vehicles in Figure 6.15. These voids highlight a short-coming with the capture technique and subsequent removal of unwanted objects.

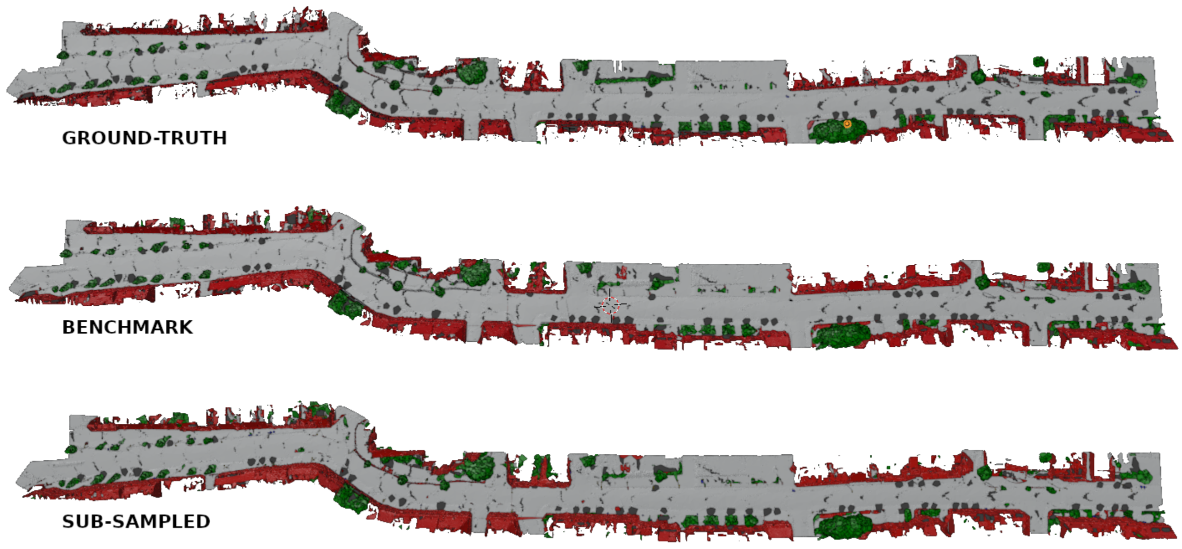


Figure 6.12: Meshes generated for the NPM3D based point clouds.

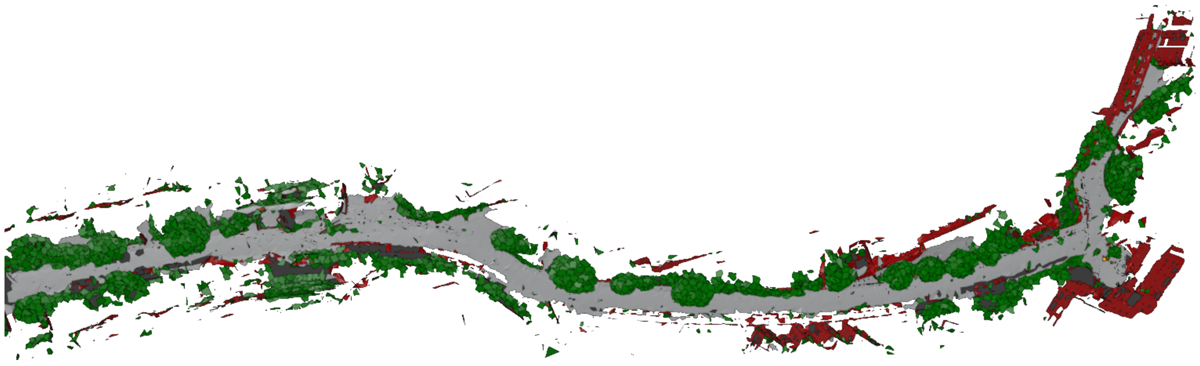
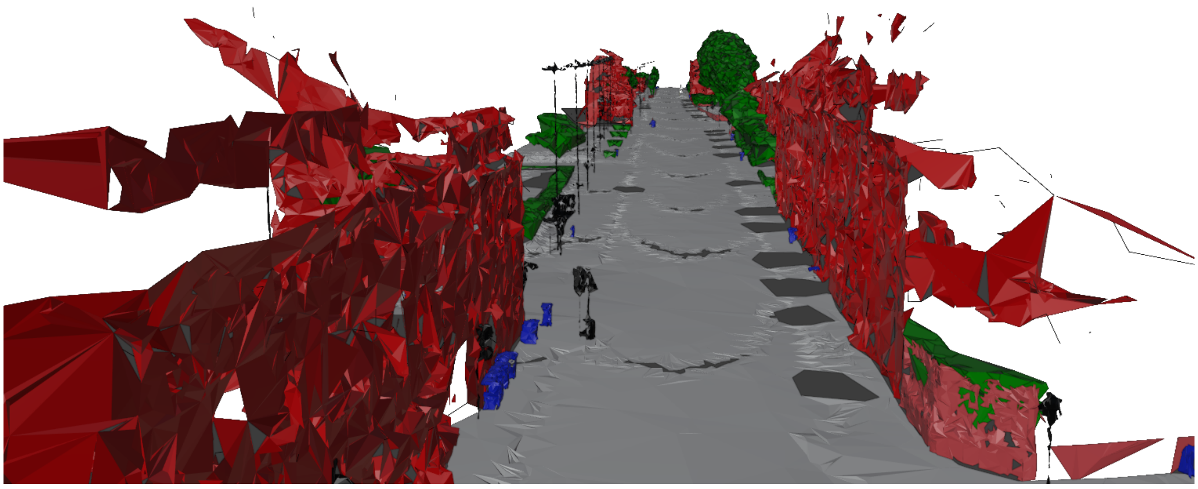


Figure 6.13: Meshes generated for the cleaned version of Woodland Road.

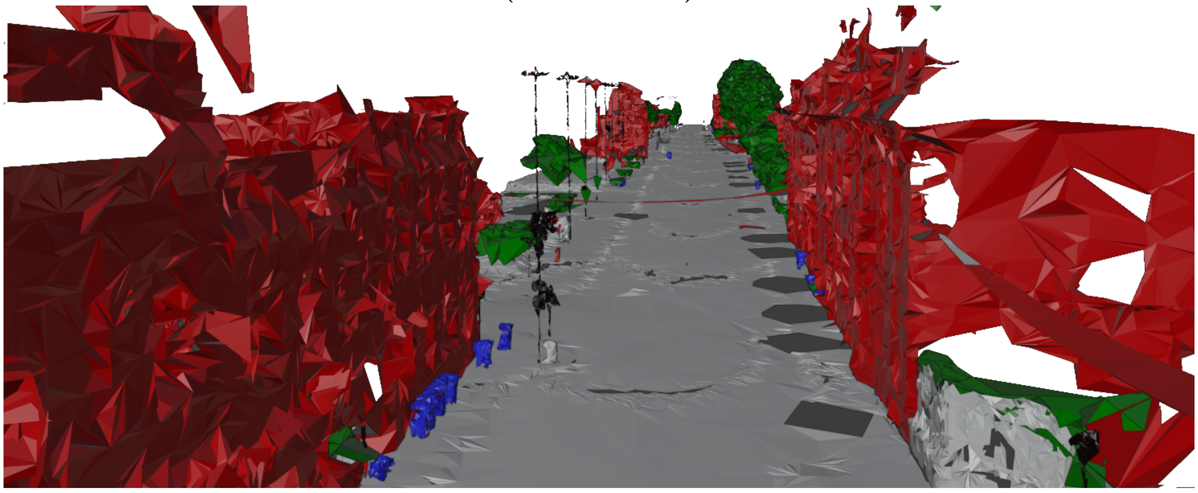
## 6.2 Propagation model testing

A series of experimentation was carried out in order to evaluate the performance of the pipeline meshes when employed as environments within the propagation tool. The objective was to compare the different meshes including any ground-truth data. In this respect, ground-truth falls into two areas. The first is the hand labelled ground-truth available from the NPM3D dataset. Whilst it did not need to go through the classification stage of the pipeline, the point cloud was subsequently converted into mesh form in the same manner as all the other point clouds.

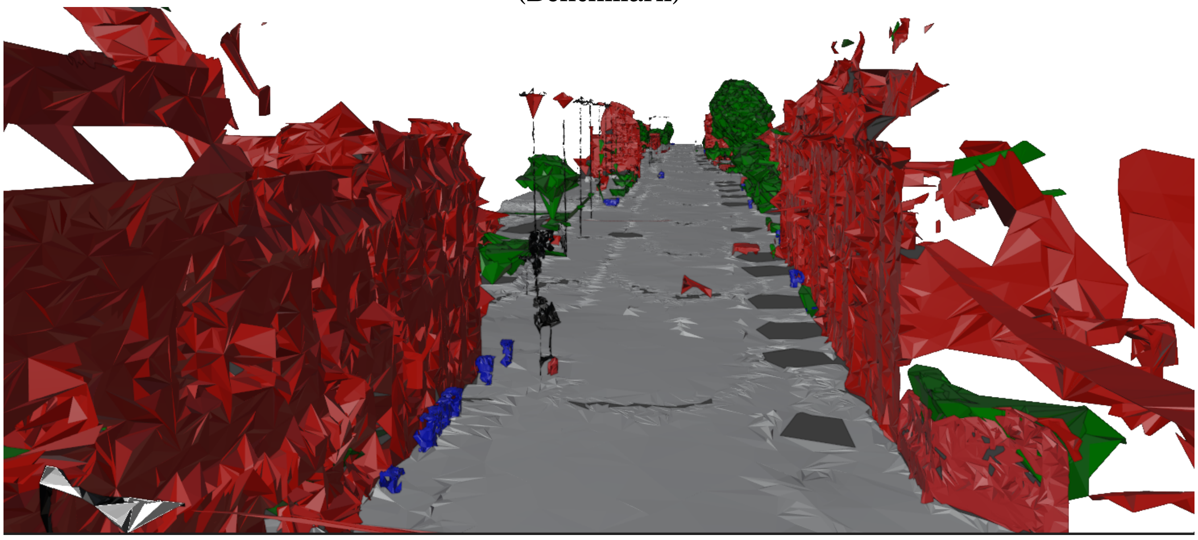
The second form of ground-truth comes from within the propagation tool itself and covers the area captured at Woodland Road, Bristol. It was created by the industrial sponsor using DTM and DEM LIDAR data captured by the Environment Agency, at a resolution of one metre. This



(Ground-truth)



(Benchmark)



(Sub-sampled)

Figure 6.14: Sections of the generated NPM3D meshes.

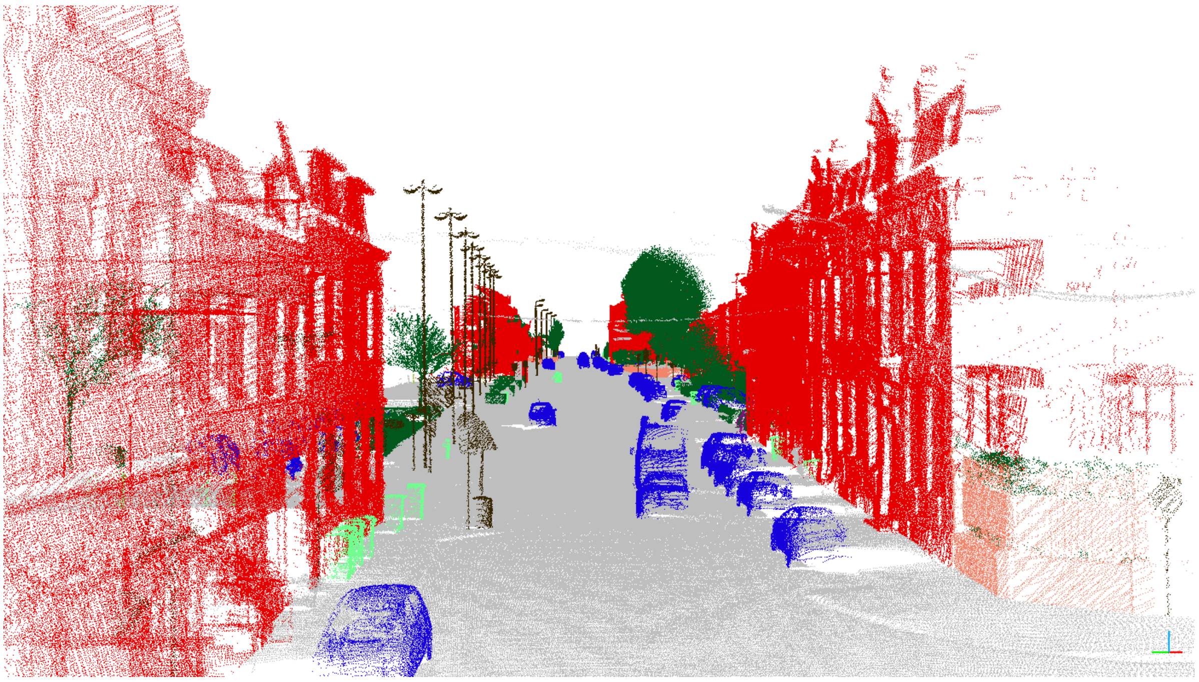


Figure 6.15: Section of hand-labelled ground-truth from the NPM3D dataset.

also includes the associated hand-labelled classifications. Both were discussed in section 4.2.1.1 which used this data as the basis of the dataset used for experimentation.

Triangulated meshes for this ground-truth were formed by fitting the outlines of buildings defined from map data available from Ordnance Survey, and creating height from the difference between the DTM and DEM information. Roof shapes were created which also reflected this height information, and a similar process was used for foliage. This ground-truth is referred to as OSGB through-out this section.

There are a couple of notable differences between the meshes created from the pipeline and the OSGB mesh. First, any artefacts which appeared in a $1m^2$ segment, defined by the classification data, were labelled as the same e.g. the facets of a tree and a wall would be labelled the same if they fell within the same segment. The result is a slightly imprecise but smoother set of meshes.

Then, there are classification differences, the meshes from the pipeline are based on nine classification points. At one metre resolution, the OSGB mesh would not be able to label objects such as barriers, bollards, poles and trash cans. Furthermore, the emphasis on what to classify is different. The NPM3D classification system has classes for buildings, the ground and nature; whereas, the OSGB classifications discriminate between roads, pavements and open green spaces, such as park land. This leads to differences in the material type assigned by the propagation model e.g. the OSGB mesh uses different materials for roads and pavements.

Three main sets of experiment were evaluated, one based upon the NPM3D environment and two from Woodland Road, Bristol. All involve placing a single base-station within the environment and simulating a person, with UE, walking past. Both the transmitter and receiver used isotropic antennas with a frequency of 2.4GHz (approximately 12.5 cm wavelength). The UE was maintained at a height of 1.5 metres, which represents the approximate height of a person. The base-station was placed on top of a building. Signal strength was then measured in 10 cm intervals as the person walked. Two sets of measurements were taken for each test, the dominant ray (which could be the LOS ray) and the combined effects of fast-fading. This is a measurement which includes the cumulative effects of constructive and destructive interference, caused by all of the scattered/reflected rays and any LOS ray.

The transmission was modelled at 0 dBm, which the industrial sponsor states is standard practice for ProPhecy users as it allows them to read path loss directly from the results. Consequently, the signal strength recorded is not a reflection of a real-word base-station which typically transmits at 30 to 35 dBm. In other words, the UE will be seen to have a connection to the base-station below the normal expected range. Adding 30 dB to the measurements would yield real-world results but this does not affect the outcome of these experiments, so has not been done.

The NPM3D environment had its base-station placed in a central location; whilst, the Bristol environment had the base-station placed in two different areas. One which gave the best overall coverage, and the other which tested how the rays propagated around a corner. All meshes, including the OSGB mesh, were tested with the appropriate material for their labelling. In addition, the sub-sampled meshes were also re-tested with a single generic material. The characteristics of the chosen material intentionally did not allow much scatter or reflection. For all results, the signal strength value, represented as a colour bar in the results section illustrations, has been normalised over the entire range of relevant values. This was performed in order to give a comparable view.

For the NPM3D environment the following meshes were used:

- The ground-truth.
- The benchmark.
- The sub-sampled version, at a resolution of 32 cm.
- The above sub-sampled version where a generic material has been assumed.

Likewise, for the Bristol environments the following meshes were used:

- The OSGB ground-truth.
- The sub-sampled version, at a resolution of 32 cm.
- A cleaned version of the above sub-sampled version.
- The sub-sampled version where a generic material has been assumed.

Table 6.1: Difference (dB) in signal strength from NPM3D ground-truth. Distance (metres) away from base-station is approximate.

| | **Dominant ray** | | | **Including fade** | | |
|---|---|---|---|---|---|---|
| **Distance (m)** | **Benchmark** | **Sub-sampled** | **Generic** | **Benchmark** | **Sub-sampled** | **Generic** |
| 140 | -2.25 | 1.41 | -5.37 | -1.42 | 2.39 | -3.82 |
| 120 | 1.65 | 9.25 | 11.97 | 5.77 | 13.59 | 22.78 |
| 100 | 2.88 | 6.45 | -2.98 | 3.42 | 6.15 | -0.29 |
| 80 | -0.51 | -1.24 | -17.22 | 1.06 | -0.79 | -17.67 |
| 60 | 3.35 | 2.70 | 7.71 | 3.23 | 4.09 | 3.16 |
| 40 | 6.21 | 4.22 | -3.28 | 3.38 | 1.79 | -6.12 |
| 20 | -0.25 | 0 | -0.00 | -0.33 | -0.25 | -0.27 |
| **base-station** | 0.31 | 0 | -0.00 | 0.32 | 0.32 | 0.29 |
| 20 | 0.61 | 1.71 | -2.76 | 0.76 | 3.40 | -0.96 |
| 40 | -1.97 | -2.33 | -21.04 | 1.61 | -5.42 | -23.52 |
| 60 | 0.81 | 0.45 | - | 6.18 | -0.09 | - |
| 80 | 0.61 | 0.28 | - | 7.90 | -0.88 | - |
| 100 | 1.26 | 0.96 | -7.82 | 2.80 | 0.92 | -4.05 |
| 120 | 1.16 | 0.88 | - | -0.65 | 5.20 | - |
| 140 | 0.78 | 0.51 | - | -5.78 | 4.42 | - |
| 160 | 1.44 | 1.17 | - | 2.91 | 3.48 | - |
| 180 | -3.17 | -3.42 | - | 2.71 | 0.11 | - |
| 200 | 2.19 | 1.93 | - | 6.76 | 4.60 | - |
| 220 | 2.27 | 2.03 | - | 4.66 | 2.52 | - |
| 240 | 2.28 | 2.04 | -6.50 | 6.11 | 3.10 | -4.64 |
| 260 | 2.28 | 2.05 | - | 9.33 | 5.20 | - |
| **mean** | 1.01 | 1.44 | -3.98 | 2.67 | 2.47 | -3.66 |

- The cleaned sub-sampled version where a generic material has been assumed.

Due to licensing issues with the ProPhecy RF propagation modelling tool, the tests were set-up and performed by the industrial sponsor.

## 6.3 NPM3D results

The results of the NPM3D meshes are presented here. Figure 6.16 contextually visualises the signal strength of the dominant ray, received at the UE as the person walked past the approximate placing of the base-station. The signal strength ranged from -80 dBm, when the person was near the transmitter, to -150 dBm at a distance; this includes areas with LOS.

It is noticeable that there exists a lot of drop out with the generic material mesh when the receiver does not have line-of-sight with the transmitter. The explanation for this is two-fold. First, trees no longer allow any rays to penetrate through them because they are classed as solid objects and therefore block the LOS. Secondly, the material chosen has low scatter and reflection characteristics, meaning any chance of rays finding their way to the UE is significantly decreased,

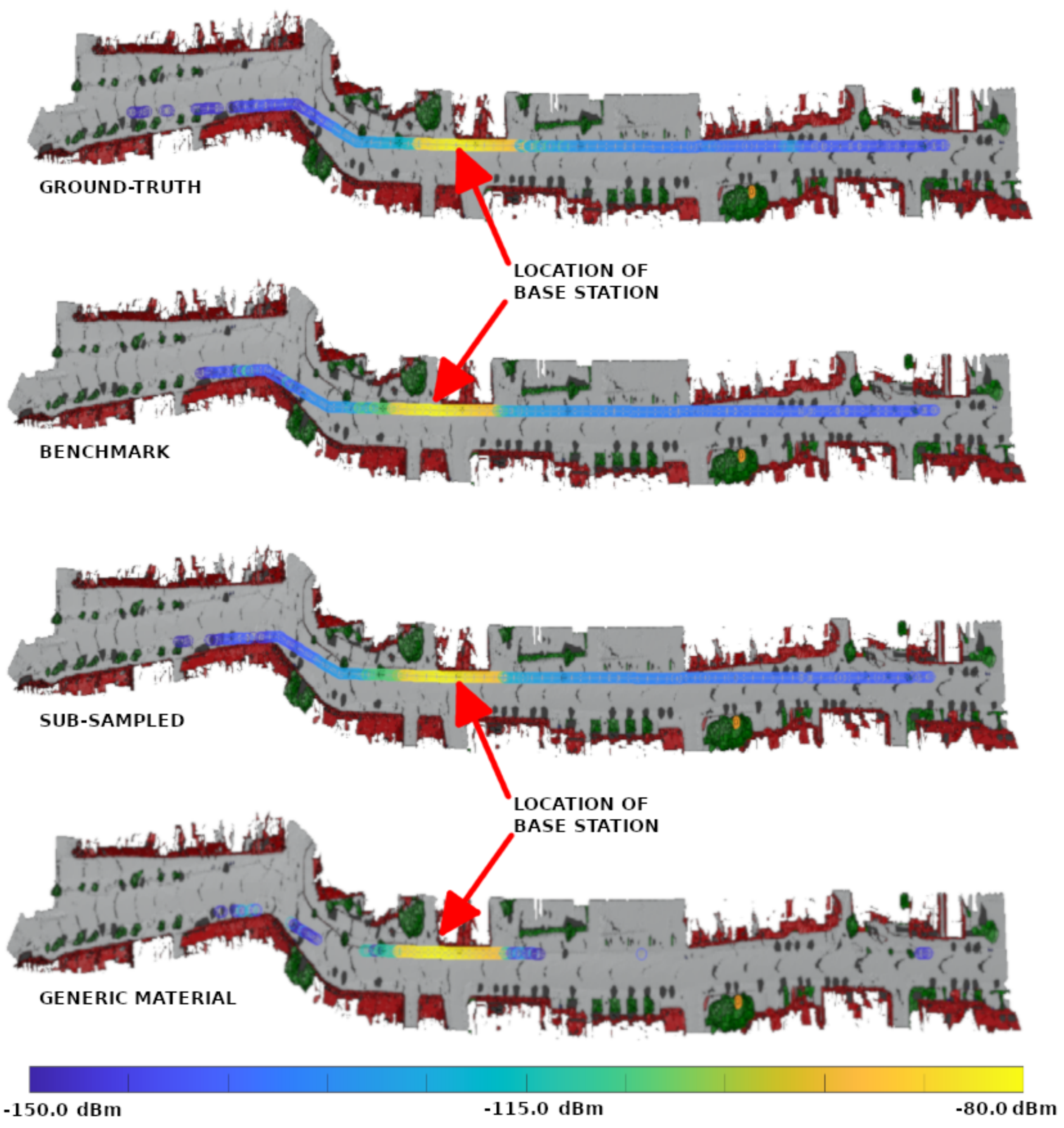


Figure 6.16: Signal strength (dominant ray) overlaid on top of the NPM3D mesh.

even by bouncing around the object. Even if the rays could find their way to the UE, they would be significantly attenuated.

This is also seen in the generic material fast fade measurements, which are shown in Figure 6.17. The fast fade signal strength measurement is a combination of all rays received including the LOS ray, if there is LOS. In this instance, the lack of scatter and reflection inhibits propagation significantly by reducing the amount of viable rays. It is noticeable that the fast fade signal strength range is lower than the single strongest ray (which tends to be the LOS) experiment. The implication of this is that, at one point or more, the constructive/destructive interference of the combined rays impacts the received signal negatively.

In both the strongest ray and fast-fade experiments there is negligible visual differences between the ground-truth, benchmark and sub-sampled meshes. Figures 6.18 and 6.19 plot the ground-truth differences of the dominant ray and fast fade signal strengths respectively. These figures show sample-average differences, when compared to the ground-truth, and are represented as distance away from the base-station. The sample-average is computed in blocks of 5% of the total measurements. Table 6.1 provides the averaged measurements used for the plots, and also gives the average difference across all measurements.

Unsurprisingly, both plots show good alignment of all scenarios when the UE is near the base-station. This is because they have good LOS. The benchmark and sub-sampled plots show good correlation overall, with their respective average measurements, for the dominant ray experiments, of 1.01 dB and 1.46 dB better than the ground-truth; and improvements of 2.67 dB and 2.47 dB for the fast-fade measurements, although fading appears to make the measurements much more variable than the ground-truth. They also appear to perform better as the UE starts to become less dependent on LOS advantages, as shown on the left of the base-station. However, the ground-truth in this instance has not been hand-crafted for the propagation model, and is instead a hand-labelled point cloud which has been converted into mesh form.

The signal strength difference plots also confirm the problems seen with the generic material which was seen visually in figures 6.16 and 6.17. The average difference measurements for the generic material was lower at 3.97 dB for the dominant ray and 3.66 dB for the fast-fade. This includes areas of the path taken where the measurements reduced by around 20 dB from the benchmark. Given that a 3 dB difference represents a halving of power, this reduction is quite significant and suggests that labelling point clouds with a single material is not a good idea and could be improved with better labelling. However, subsequent experimentation, using the Bristol dataset, will show that under certain conditions, generic materials can perform just as well as labelled datasets.

One final observation is that the signal strength, for the benchmark and sub-sampled experiments, tends to improve by around 2 dB as the UE moves further away from the base-station, when compared to the ground truth. Again, this can be attributed to the definition of the ground-truth i.e. it is not hand-crafted for the propagation model and is just hand-labelled version of the

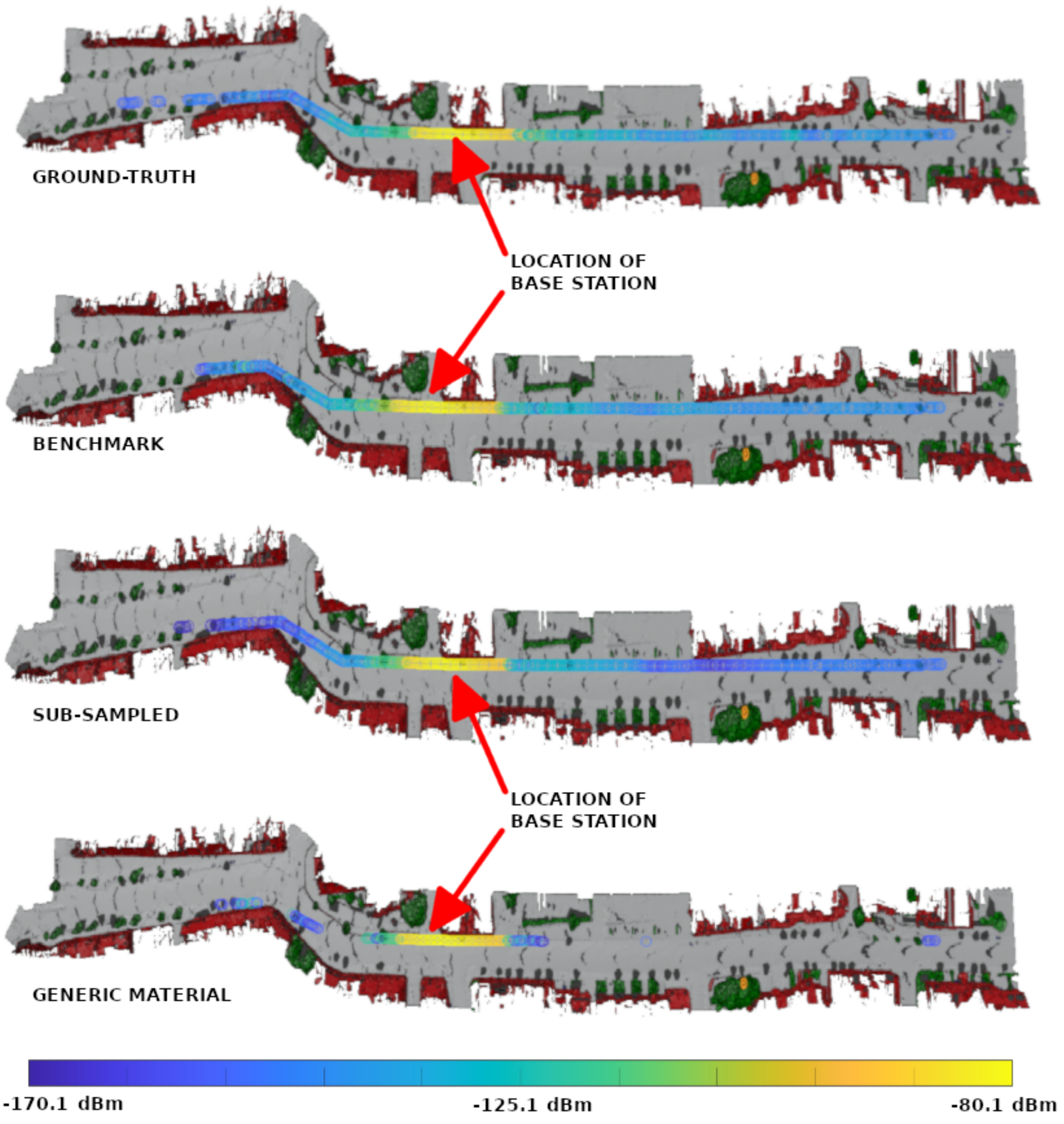


Figure 6.17: Signal strength (including fade) overlaid on top of the NPM3D mesh.

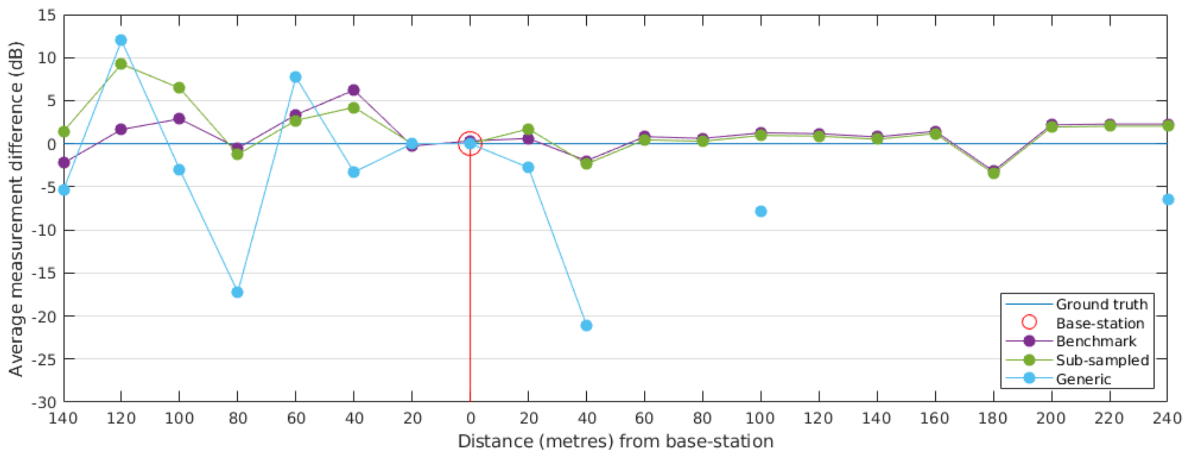


Figure 6.18: Difference (dB) in signal strength (dominant ray) from NPM3D ground-truth.

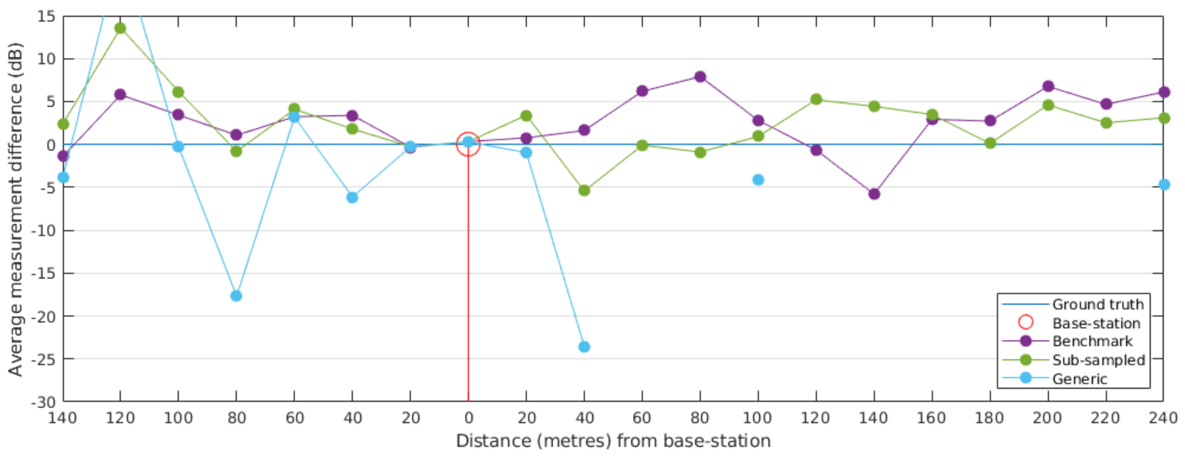


Figure 6.19: Difference (dB) in signal strength (including fade) from NPM3D ground-truth.

point cloud which has many shared labels as the other experiments.

What these NPM3D experiments show is i) that automatic classification is as good as hand-labelling, ii) that generic labelling can be problematic if trees are identified as buildings.

## 6.4 Bristol results with transmitter on Woodland Road

From an experimental point of view, the Woodland Road meshes have an advantage over the NPM3D meshes in that they can be compared with the OSGB mesh, which has been hand-crafted for use within the propagation model. Furthermore, the inclusion of misclassified objects such as an ill-timed passing of a bus makes for good comparison with a subsequently cleaned version. The terrain is also more interesting. It includes a side street (Cannocks Close) and, whilst not immediately apparent, the terrain is by no means flat.

The initial set of experiments for the Woodland Road meshes involved strategically placing the base-station in a location which gave the best overall coverage. The results of these experiments are presented here. Figures 6.20 and 6.21 illustrate the signal strength over the path taken for both the dominant ray and the fast fade measurements, respectively. The signal strength for the dominant ray measurements range from -70 dBm when close to the base-station to -150 dBm. The top image shows the OSGB ground-truth mesh and should be used as the standard to attain. The middle image shows the obstruction of the bus and how it affects the signal strength as the person moves away from the base-station. It clearly impacts the dominant ray measurements as it blocks the LOS. A view from within the propagation model, courtesy of the industrial sponsor, can be seen in Figure 6.22 which shows the loss of the strongest signal, forcing the tool to measure a weaker ray going forward. The fast fade version shows how the measurement of all scattered/reflected rays mitigates this to some extent.

The signal strength measurements for the generic material sub-sampled meshes are visualised in figures 6.23 and 6.24 which show the dominant ray and fast fade measurements over the path taken. Their range of measurements appear broadly similar to the fully labelled and ground-truth ranges; suggesting that material differences are negligible, when there is good line of sight.

Figures 6.25 and 6.26 plot the signal strength differences, from the ground-truth, for the dominant ray and fast fade measurements. Table 6.2 provides the numbers to support the charts. Whilst there is good LOS, there is little divergence between the different experiments. At approximately 72 metres from the base-station, the sub-sampled and generic experiment signal strength measurements start to diverge away from the ground-truth. The reason for this is the presence of the mis-classified bus which has blocked the LOS, see Figure 6.22. The subsequently cleaned versions perform better, for another 20 metres. They too diverge from the ground-truth as the road starts to bend around the corner, losing the benefit of the LOS.

The total average loss shows that cleaning the meshes, especially the bus helps both the generic and sub-sampled versions. Prior to cleaning both types of experiments had an average loss of circa 11.5 dB for the dominant ray results and a similar figure of circa 11.8 dB for the fast-fade results. However, when the data is cleaned both improve by around 3 dB, with a slight advantage going to the generic material over the sub-sampled material (8.14 vs 8.68). There is very little difference between the dominant ray and fast-fade results.

For these results, it is clear that the OSGB ground-truth mesh is consistently better than the other experiments. This could be for a number of reasons. First, there may be advantages with differing material types used. There may also benefits in having the more simplistic, one metre resolution, object classification. This extends to include the creation of a more straight-forward, and not necessarily any more accurate, mesh due to the low resolution. Consequently, a definitive statement of whether the enhanced performance, at the path edges, is because the OSGB is simply better or not cannot be made.

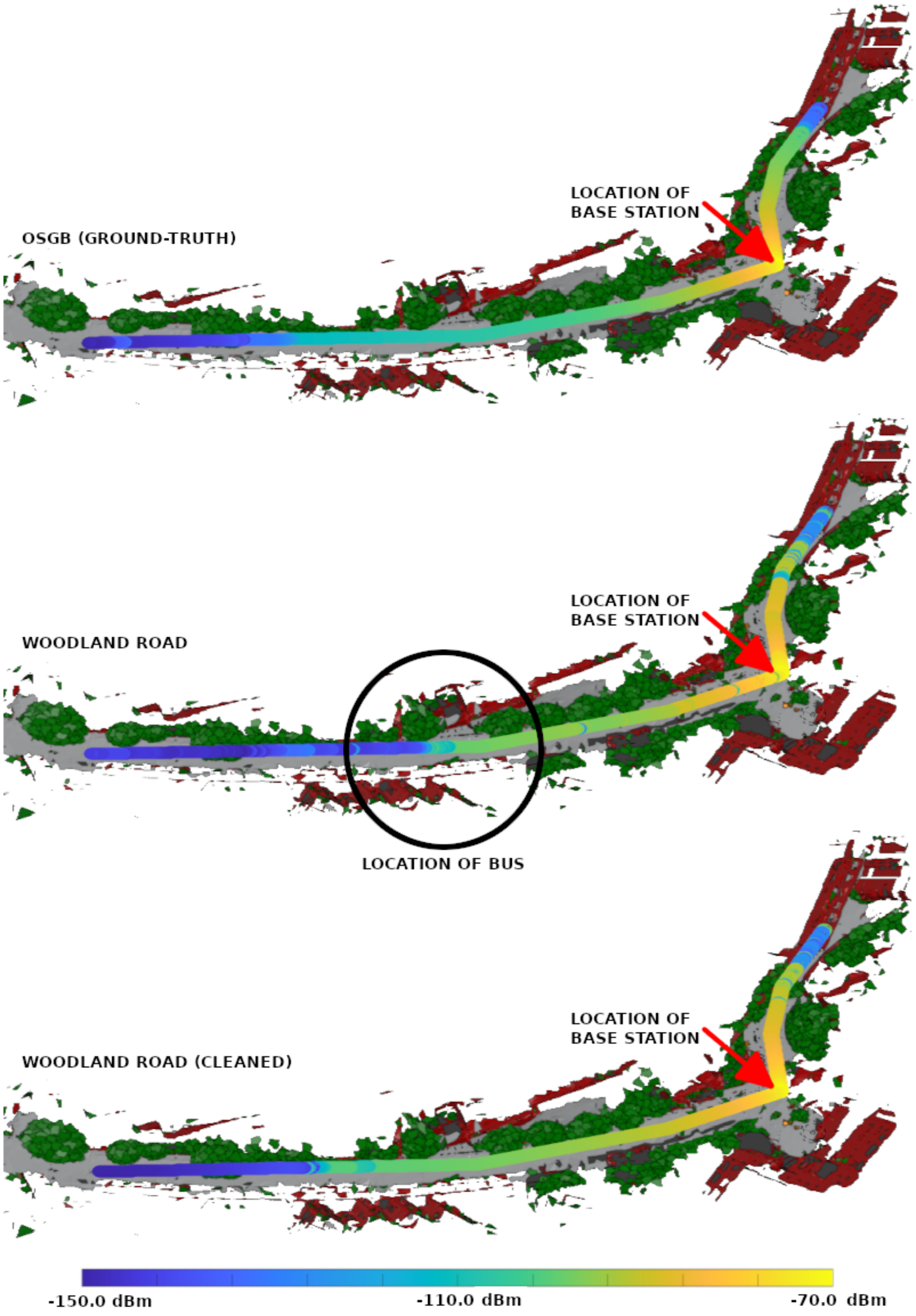


Figure 6.20: Signal strength (dominant ray) overlaid on top of the Woodland Road mesh.

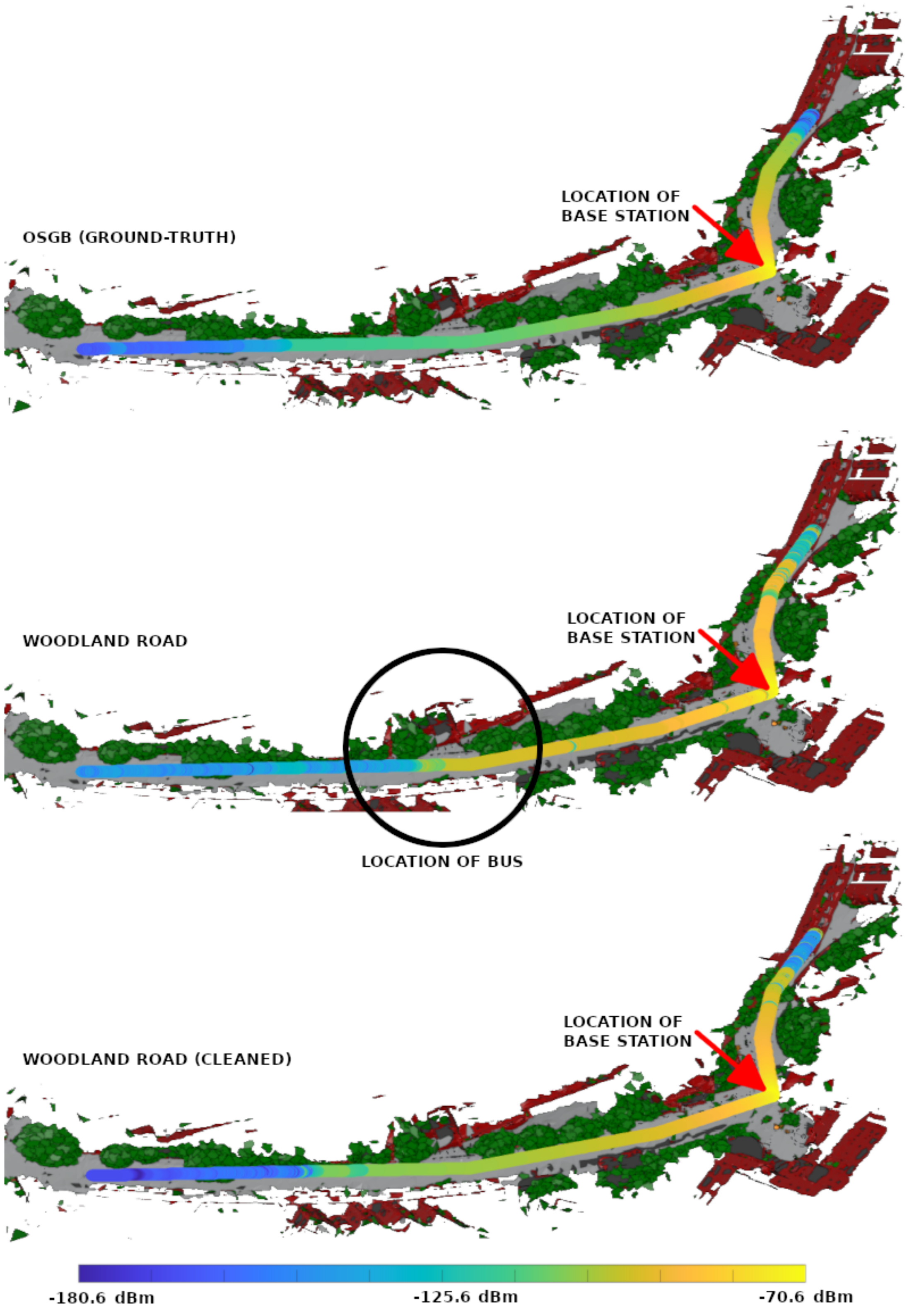


Figure 6.21: Signal strength (including fade) overlaid on top of the Woodland Road mesh.

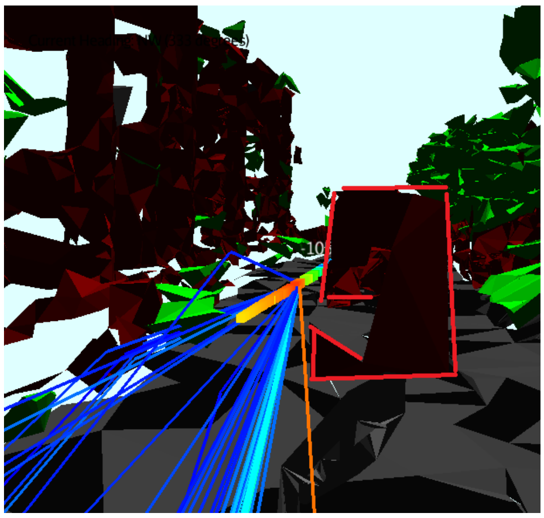


Figure 6.22: Image of the bus, as seen from within the propagation model.

Table 6.2: Difference (dB) in signal strength from Woodland Road ground-truth. Distance (metres) away from base-station is approximate.

| | Dominant ray | | | | Including fade | | | |
|---|---|---|---|---|---|---|---|---|
| **Distance (m)** | **Sub-sampled (cleaned)** | **Sub-sampled** | **Generic** | **Generic (cleaned)** | **Sub-sampled (cleaned)** | **Sub-sampled** | **Generic** | **Generic (cleaned)** |
| 60 | -14.71 | -10.45 | -9.52 | -23.22 | -14.93 | -10.09 | -8.57 | -22.87 |
| 48 | -11.37 | -7.82 | -7.41 | -16.41 | -12.74 | -8.19 | -7.68 | -16.91 |
| 36 | -0.59 | -4.73 | -4.96 | -0.63 | -0.59 | -4.49 | -5.12 | -0.62 |
| 24 | 0.34 | -1.90 | -0.92 | 0.29 | 0.32 | -1.93 | -0.98 | 0.27 |
| 12 | 1.04 | -1.08 | -0.53 | -0.40 | 1.00 | -1.14 | -0.56 | -0.44 |
| **base-station** | 1.09 | -1.32 | -1.20 | 0.51 | 1.05 | -1.46 | -1.38 | 0.47 |
| 12 | 0.30 | -1.21 | -1.09 | 0.26 | 0.33 | -1.17 | -1.01 | 0.29 |
| 24 | 0.17 | -1.27 | -0.88 | -0.50 | 0.16 | -1.30 | -0.89 | -0.51 |
| 36 | 0.12 | -2.80 | -2.67 | -2.19 | 0.11 | -2.85 | -2.70 | -2.20 |
| 48 | 0.097 | -3.48 | -2.04 | 0.08 | 0.12 | -3.37 | -2.06 | 0.11 |
| 60 | 0.08 | 0.00 | 0.08 | 0.07 | 0.13 | 0.06 | 0.12 | 0.12 |
| 72 | 0.07 | 0.00 | 0.07 | 0.06 | 0.10 | 0.03 | 0.09 | 0.09 |
| 84 | 0.06 | -3.61 | -11.71 | 0.05 | 0.07 | -3.62 | -12.92 | 0.06 |
| 96 | 0.05 | -30.57 | -34.35 | -0.94 | 0.06 | -30.61 | -33.37 | -0.93 |
| 108 | -3.16 | -39.01 | -39.35 | -13.19 | -3.18 | -40.55 | -38.56 | -13.37 |
| 120 | -22.54 | -35.42 | -30.40 | -44.56 | -22.31 | -36.45 | -31.22 | -45.61 |
| 132 | -29.70 | -23.67 | -26.91 | -37.82 | -29.96 | -24.96 | -27.89 | -40.56 |
| 144 | -25.09 | -23.80 | -20.31 | - | -20.75 | -23.00 | -20.53 | - |
| 156 | -24.89 | -19.54 | -17.83 | - | -22.34 | -20.11 | -19.44 | - |
| 168 | -27.88 | -17.80 | -17.94 | - | -30.26 | -18.58 | -18.72 | - |
| 180 | -25.79 | -13.28 | -13.80 | - | -26.29 | -15.88 | -13.90 | - |
| **mean** | -7.85 | -11.48 | -11.50 | -7.66 | -7.71 | -11.70 | -11.68 | -7.86 |

What the Woodland Road experiments show is i) the hand-crafted OSGB mesh performs better than the pipeline, but this may not translate well in the real-world as there are no physical measurements to accompany the ground-truth. Furthermore, the fact that the other experiments are highly correlated may reinforce the notion that the ground-truth is not reliable; ii) that cleaning the environment works iii) even though the generic material has similar results to the others, the pipeline had already purged most of the unwanted artefacts by the use of labelling so this will almost certainly not favour the generic material going forward.

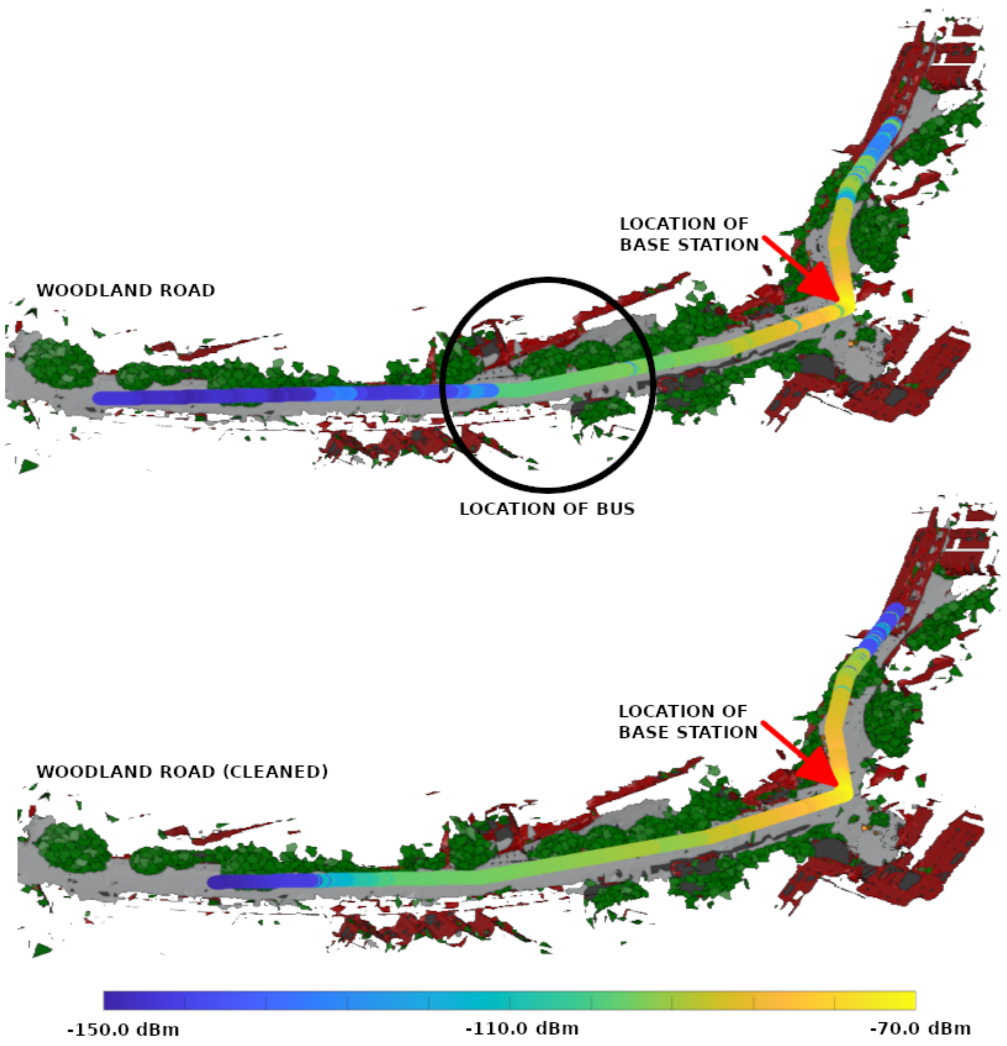


Figure 6.23: Signal stength (dominant ray) overlaid on top of the Woodland Road mesh with generic material.

## 6.5 Bristol results with transmitter on Cantocks Close

To place more emphasis on the make-up of the environment, and not the benefit that good LOS provides, a second set of experiments was performed using the Woodland Road meshes. In this instance, the base-station was moved to the side street (Cantocks Close). This was done so that it no longer provided the best coverage, and far fewer places where the UE has line-of-sight with the transmitter.

Figures 6.27 and 6.28 show the new position of the base-station for both the dominant ray and fast fade measurements. In both scenarios, there is a notable difference between the OSGB ground-truth and the pipeline created meshes where the ground-truth is able to perform better without LOS. This is evidenced by the signal strength as the UE leaves Cantocks Close and moves

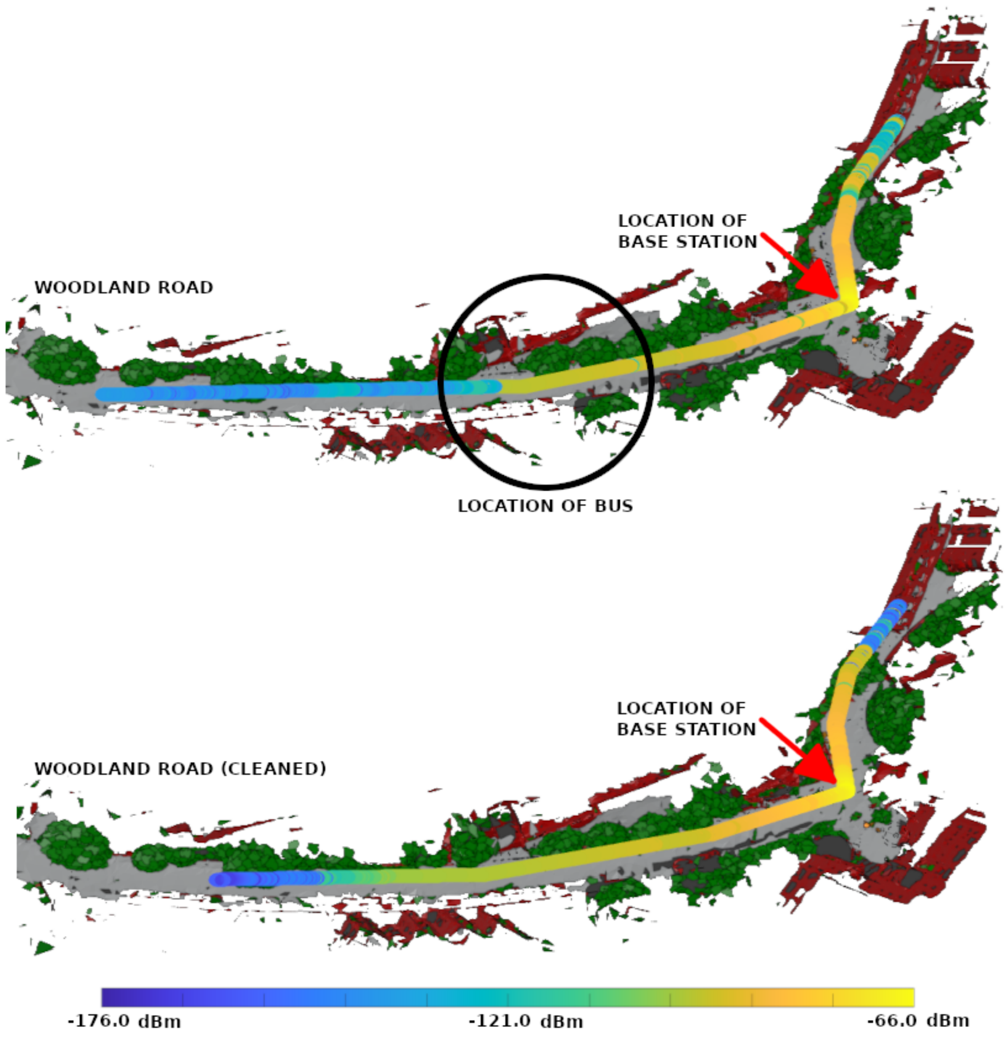


Figure 6.24: Signal stength (including fade) overlaid on top of the Woodland Road mesh with generic material.

up Woodland Road. This can also be seen in the signal strength differences shown in figures 6.29 and 6.30 and detailed in table 6.3. For the initial 48 metres, the relative signal strengths decrease significantly as LOS is lost. At the worst point, at least 30 dB is lost when compared to the ground-truth.

Again, the inclusion of the bus has some effect but it is by no means as clear cut as the original placement of the base-station in section 6.4 suggests. Notably, all charts of the pipeline based meshes for Cantock Close show sections which receive a significant boost in signal strength; whereas, the signal strength received by the UE in the OSGB ground-truth test appears to decline gracefully. What this suggests is that the OSGB meshes propagation path is restricted to following the roads. Whereas, the propagation path of the pipeline meshes is able to peek

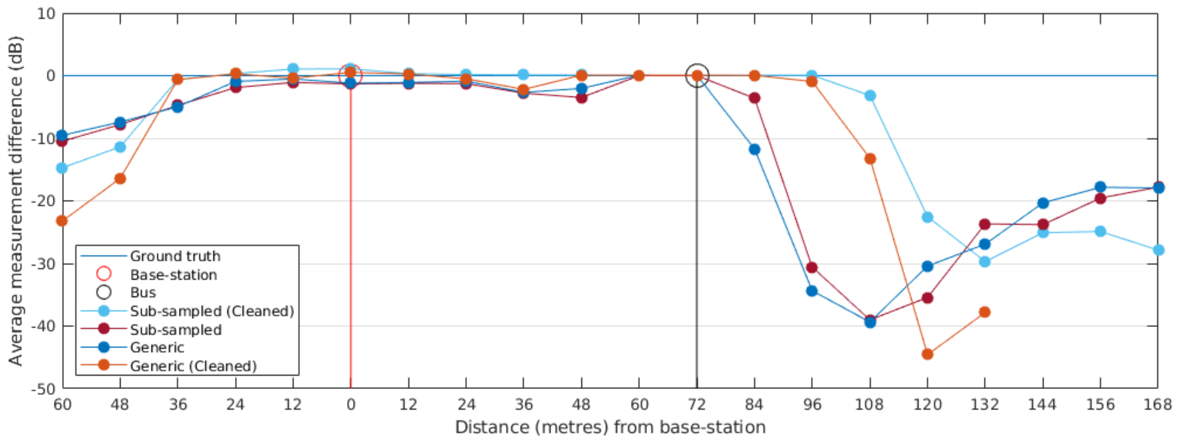


Figure 6.25: Difference (dB) in signal strength (dominant ray) from Woodland Road ground-truth.

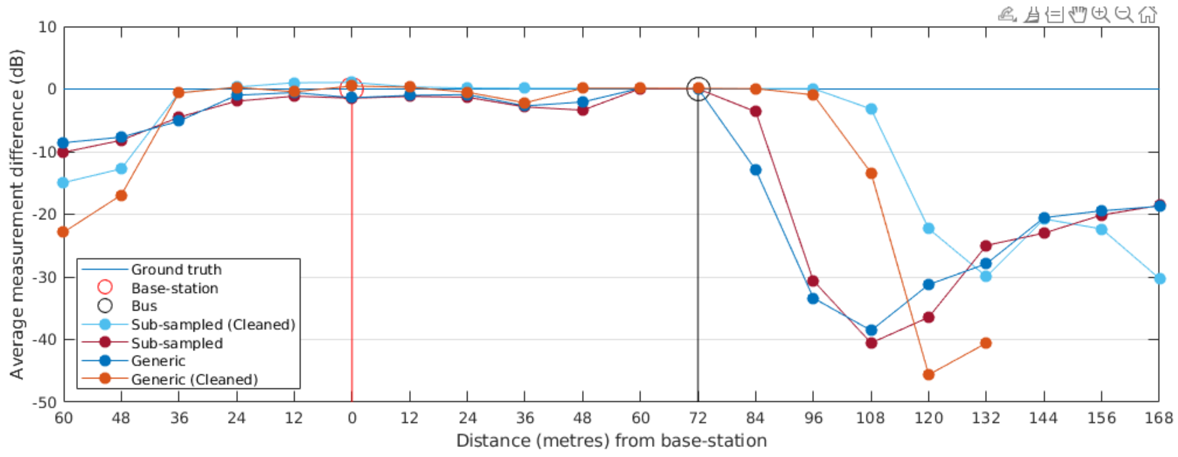


Figure 6.26: Difference (dB) in signal strength (including fade) from Woodland Road ground-truth.

through at certain locations giving LOS. This will almost certainly be a reflection of the two different techniques used to create the environments. In particular, that the hand-held scanner did not capture the true height of the objects. To a lesser degree, it is also possible that the higher resolution of the hand-held data gives rise to genuine areas where the rays can propagate. This may also explain why the bus has less impact. This along with the fact that the original experiment had good LOS to the bus, whereas, it was not true with the new location of the base-station.

The total average signal strength difference for each experiment shows that cleaning the environment does help, with both the sub-sampled and generic versions (shown in figures 6.31 and 6.32)performing better having been cleaned. The signal sub-sampled version prior to cleaning was lower at 10.05 dB versus 1.91 dB after cleaning. Similarly, the generic version had a difference of 15.35 versus a cleaned result of 11.12 dB. Small improvements were seen when fast fading was considered. These are worse than the experiments which placed the base-station on Woodland

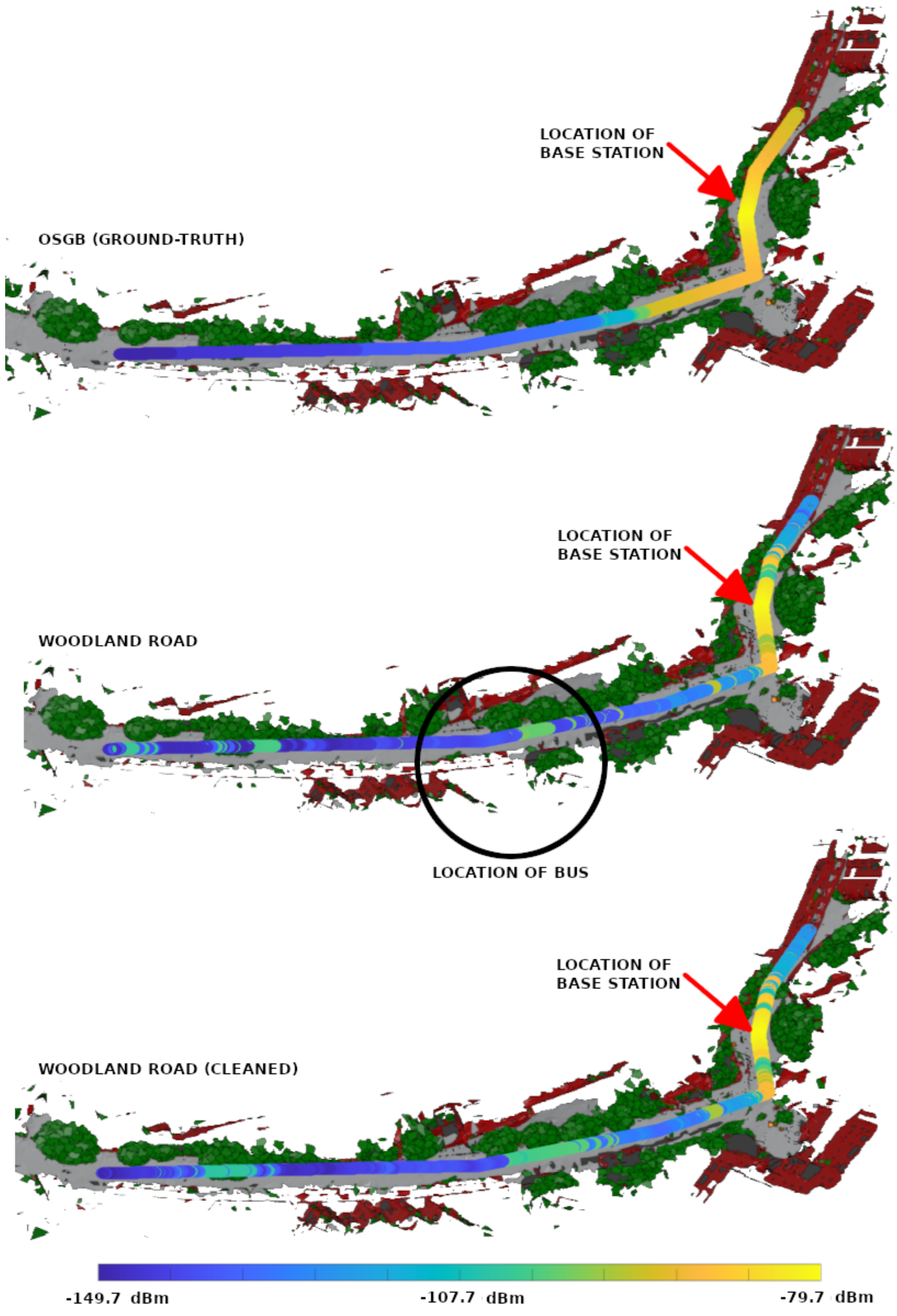


Figure 6.27: Signal stength (dominant ray) overlaid on top of the Woodland Road mesh. Base-station within the Cantocks

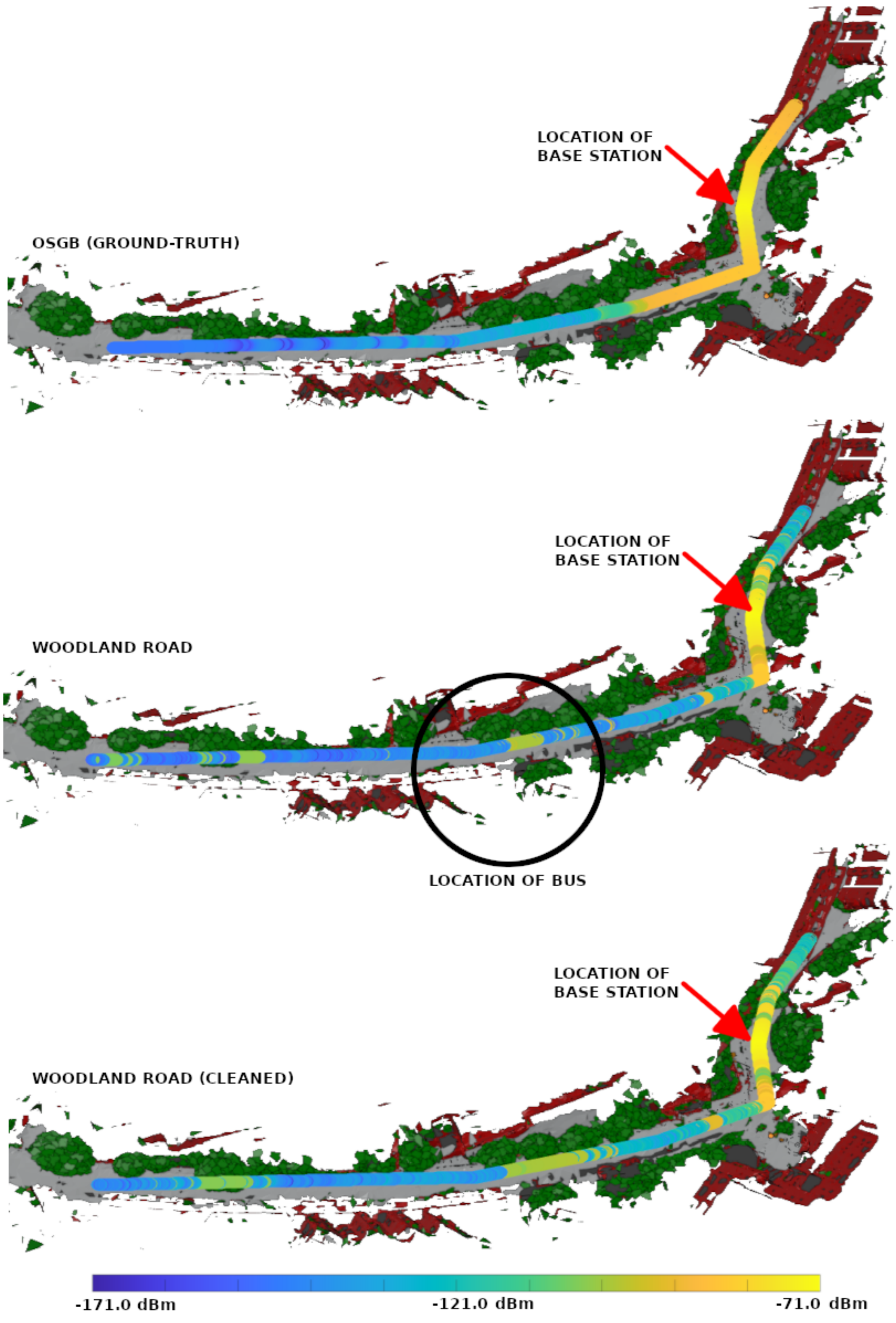


Figure 6.28: Signal stength (including fade) overlaid on top of the Woodland Road mesh. Base-station within the Cantocks

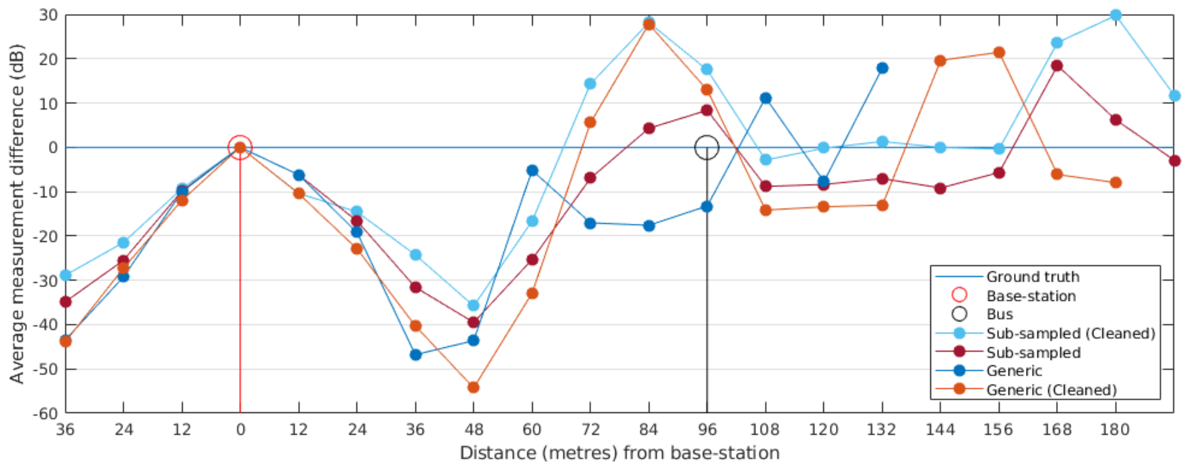


Figure 6.29: Difference (dB) in signal strength (dominant ray) from Cantocks ground-truth.

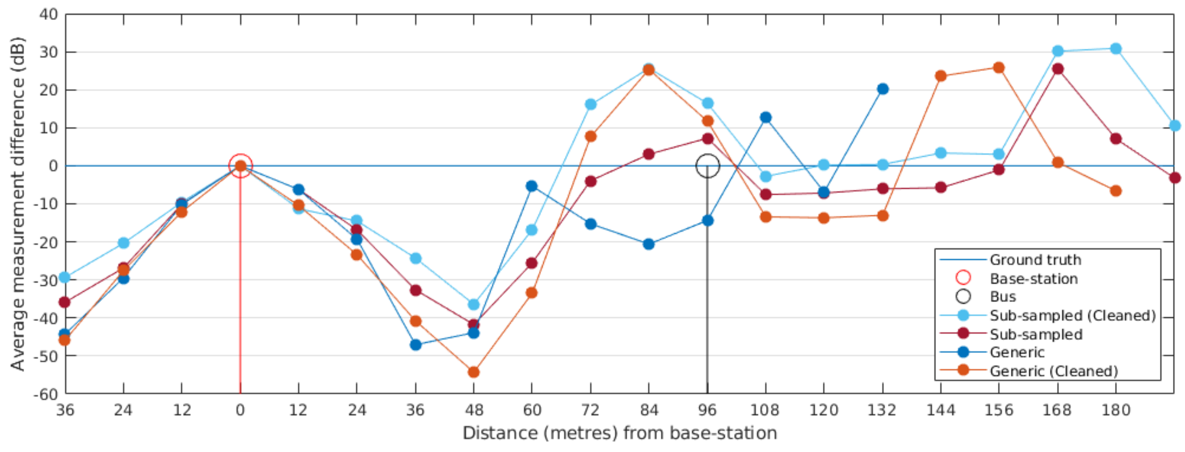


Figure 6.30: Difference (dB) in signal strength (including fade) from Cantocks ground-truth.

Road and the lack of good LOS explains this.

Before ruling out the pipeline over the meshes, the strengths and weaknesses of each approach needs to be taken into consideration. The OSGB ground-truth mesh is created by a much smoother profile and is populated with different material types defined in $1m^2$ segments. This is opposed to the much higher resolution and class availability from within the pipeline meshes.

What the Cantocks Close experiments show is i) the hand-crafted OSGB mesh performs better without LOS than the pipeline. Again, this may not translate well in the real-world ii) that cleaning the environment works iii) the technique used to capture Woodland Road and surrounding areas is flawed because it did not capture enough height to block erroneous LOS

Table 6.3: Difference (dB) in signal strength from Cantocks ground-truth. Distance (metres) away from base-station is approximate.

| | Dominant ray | | | | Including fade | | | |
|---|---|---|---|---|---|---|---|---|
| **Distance** | **Sub-sampled (cleaned)** | **Sub-sampled** | **Generic** | **Generic (cleaned)** | **Sub-sampled (cleaned)** | **Sub-sampled** | **Generic** | **Generic (cleaned)** |
| 36 | -28.89 | -34.90 | -43.49 | -43.93 | -29.33 | -35.88 | -44.23 | -45.84 |
| 24 | -21.47 | -25.49 | -29.26 | -27.29 | -20.37 | -26.85 | -29.47 | -27.56 |
| 12 | -9.31 | -9.87 | -10.07 | -11.97 | -9.55 | -10.08 | -10.16 | -12.09 |
| **base-station** | 0.028 | 0.01 | 0.01 | 0.02 | 0.02 | 0.01 | 0.01 | 0.02 |
| 12 | -10.36 | -6.20 | -6.20 | -10.42 | -11.30 | -6.21 | -6.21 | -10.44 |
| 24 | -14.57 | -16.60 | -19.20 | -22.88 | -14.50 | -16.85 | -19.30 | -23.36 |
| 36 | -24.25 | -31.57 | -46.78 | -40.28 | -24.24 | -32.59 | -47.03 | -40.73 |
| 48 | -35.79 | -39.50 | -43.62 | -54.28 | -36.42 | -41.74 | -43.87 | -54.28 |
| 60 | -16.63 | -25.30 | -5.16 | -32.90 | -16.82 | -25.54 | -5.35 | -33.51 |
| 72 | 14.29 | -6.96 | -17.04 | 5.54 | 16.02 | -3.92 | -15.23 | 7.61 |
| 84 | 28.14 | 4.29 | -17.58 | 27.80 | 25.66 | 3.04 | -20.58 | 25.33 |
| 96 | 17.49 | 8.35 | -13.25 | 12.94 | 16.43 | 7.24 | -14.36 | 11.74 |
| 108 | -2.86 | -8.82 | 11.19 | -14.16 | -2.76 | -7.57 | 12.62 | -13.42 |
| 120 | -0.17 | -8.36 | -7.69 | -13.41 | 0.20 | -7.17 | -6.89 | -13.66 |
| 132 | 1.31 | -7.05 | 17.86 | -13.04 | 0.35 | -6.05 | 20.30 | -13.01 |
| 144 | -0.03 | -9.16 | - | 19.63 | 3.35 | -5.78 | - | 23.56 |
| 156 | -0.37 | -5.66 | - | 21.44 | 3.00 | -1.10 | - | 25.92 |
| 168 | 23.65 | 18.43 | - | -6.14 | 30.10 | 25.53 | - | 0.93 |
| 180 | 29.74 | 6.19 | - | -7.96 | 30.89 | 7.07 | - | -6.63 |
| 196 | 11.73 | -2.90 | - | - | 10.68 | -3.20 | - | - |
| **mean** | -1.94 | -10.07 | -16.18 | -11.22 | -1.45 | -9.39 | -16.21 | -10.62 |

## 6.6 Discussion

The creation of a pipeline to classify and prepare raw point cloud data has raised some interesting considerations. These ultimately question the level of accuracy required by the propagation tool, or even if it is attainable. The proof-of-concept, discussed in section 6.1.1.1, posed the question of whether classification should be holistic or not. In the example given, a wall had foliage growing over it. Taking this to the extreme, should a wall, which is covered entirely with foliage, be classified as foliage or not; and is it even possible for a classifier to determine that a wall lies beneath it. The current classifier would elect to label the entire structure as foliage.

The physical characteristics used with the propagation tool for foliage differ significantly from a wall. Foliage allows rays to pass through it with some absorption; whereas, a wall would absorb, reflect and scatter. It would not allow rays to pass through in the same way. In the extreme example given, a hidden wall would become semi-transparent. This is not just a result of the classification component within the pipeline. The OSGB ground-truth used $1m^2$ segments which meant that if the facets of a tree and building lie within the same segment, then they would be classed as the same thing.

The OSGB ground-truth mesh also raises the question about resolution. Compared to the meshes created by the pipeline, the OSGB mesh can be considered somewhat coarse. Does the propagation model really need the higher resolution offered by the pipeline? The results given in sections 6.4 and 6.5 show that the OSGB mesh performs better within the propagation tool. However, it is hard to be sure without seeing real-world performances. Could the simplicity and smoothness of the OSGB mesh bias the tool in favour of it, and does it matter, given that the environment is constantly changing? A simple way to answer this is to consider the impact that a $1m^2$ segment, which is classified as a building but only contains 55% of an actual building, has on the the propagation model's ability to trace the LOS. It will likely block it, when in reality it

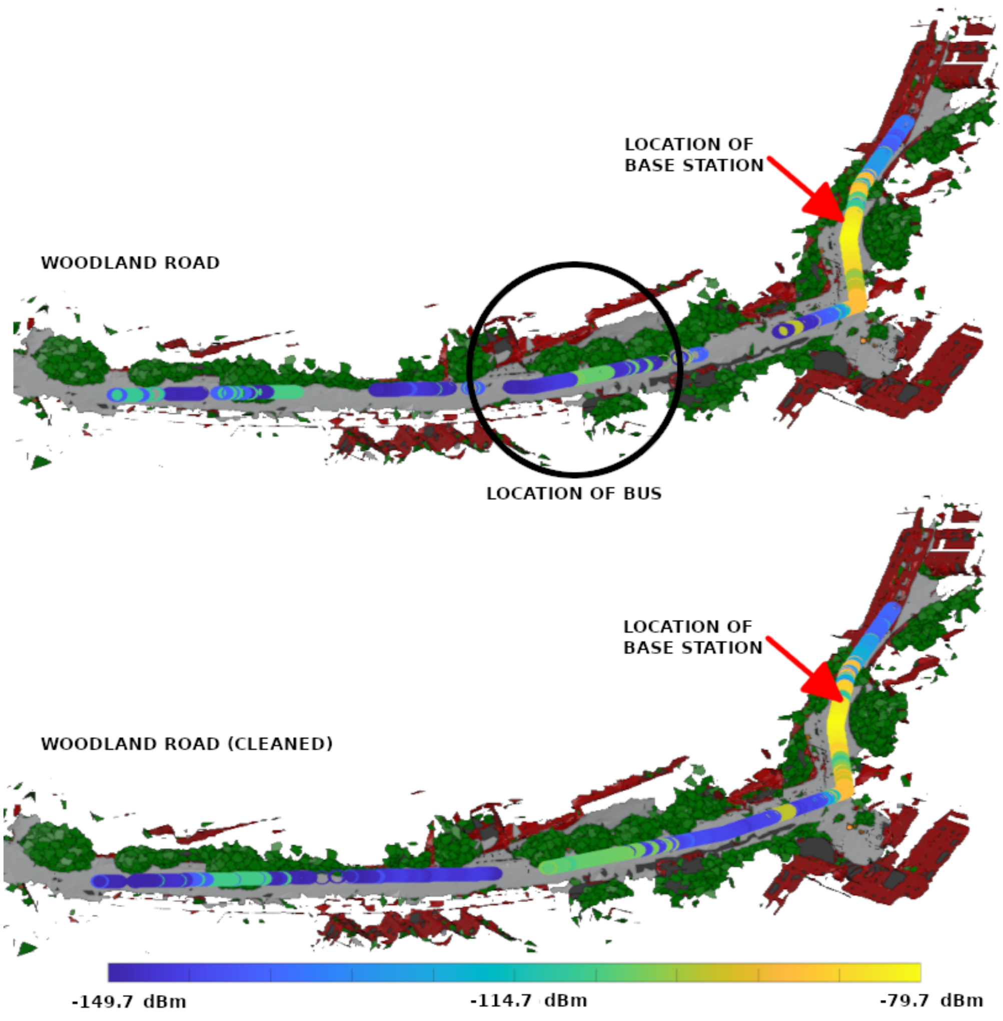


Figure 6.31: Signal stength (dominant ray) overlaid on top of the Woodland Road mesh. Basestation within the Cantocks with generic material

should have passed it. Higher resolution is clearly a benefit in this instance, as the definition of free space is as important, if not more so, as the correct assignment of the physical characteristics of clutter.

The OSGB mesh also had fewer classifications than the pipeline meshes. Furthermore, there was a different emphasis given to those classifications, including three out of the five discriminating against the ground and, consequently, being assigned different material types. In contrast, the output from the classification component in the pipeline was formed from nine classifications, with only one focused upon the ground. Four labels, poles, barriers, bollards and trash cans, would not be considered in the OSGB mesh because they would be too small to fit the $1m^2$ criteria. All of which could affect the propagation of rays within the model. Of the four classes

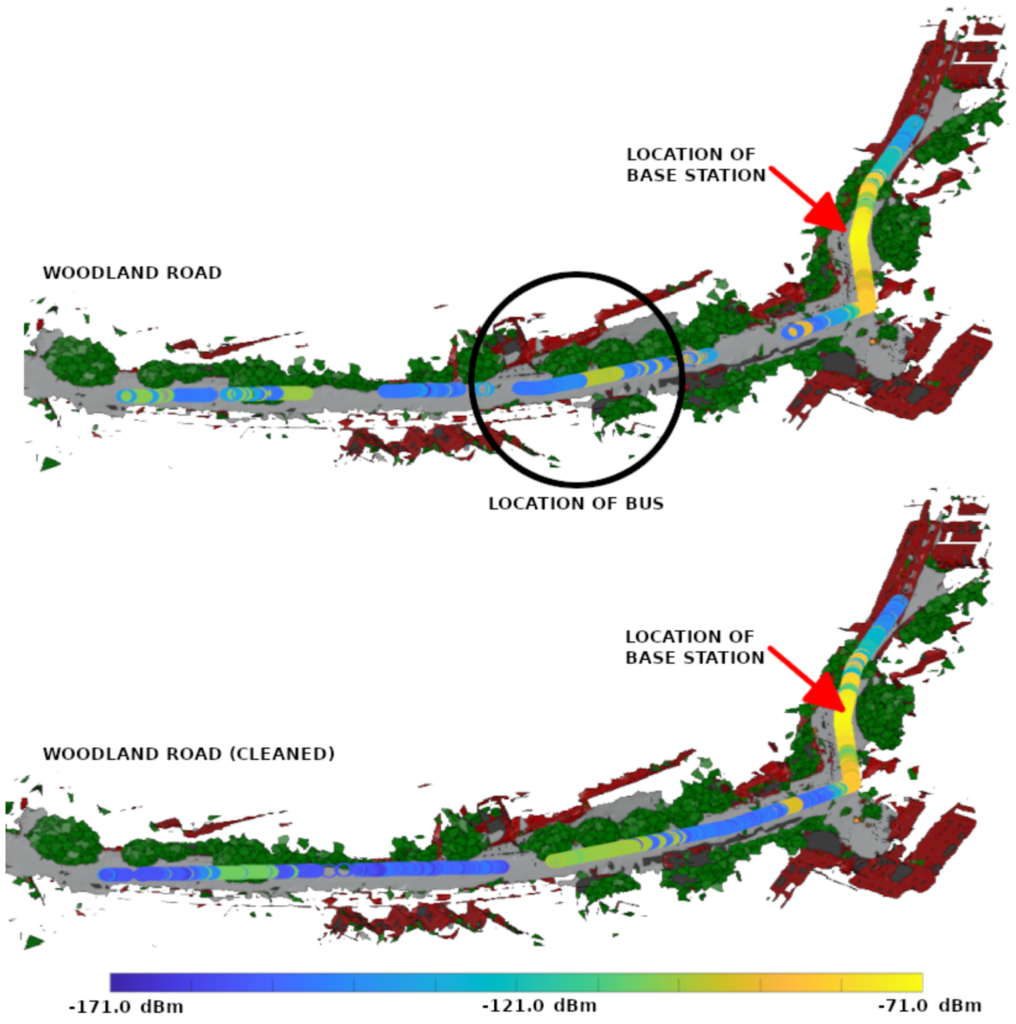


Figure 6.32: Signal stength (including fade) overlaid on top of the Woodland Road mesh. Base-station within the Cantocks with generic material

remaining, two are not included in the final meshes and therefore not relevant for comparison. These are pedestrians and vehicles, which one assumes were objects that were also ignored when the OSGB data was hand-labelled. The other two were buildings and foliage.

The design objective of the pipeline was to create an automated process that could be scripted, and in the main, this was achieved. It was never designed to find the perfect solution and whilst the results are encouraging, the processes could be improved. In particular, creating a mesh with fewer faces whilst not affecting the object's structural integrity. Some effort was placed on this aim and can be seen by the novel way the foliage was described.

It is easy to assume that a road could be represented by fewer meshes, the reality differs when its gradient and curvature are considered. Nonetheless, the pipeline should be viewed as an early prototype. That said, and implied in earlier statements, the propagation tool needs to require higher levels of accuracy to make this pursuit worthwhile i.e. is there a significant positive impact if a process can accurately define the environment, especially where the environment itself is dynamic. Whilst there may be a computational trade-off in managing the complexity of an environment, even if the propagation tool did not benefit from higher resolution, it makes sense to provide the most accurate description of the environment.

The results from the experiments in section 6.3, which focused upon the NPM3D dataset, show negligible differences in performance between the ground-truth data and extreme sub-sampled data. In this instance, the accuracy levels for the hand-labelled, benchmark and sub-sampled datasets were all over 80%. This demonstrates the utility of automatic classification. Furthermore, it was found that the ground-truth point cloud had not classified all objects whilst the others had, suggesting that the ground-truth does not possess 100% accuracy.

One problem the NPM3D ground-truth did reveal was the shadows left by objects which were subsequently removed from the environment. These shadows are actually voids in the dataset where an object blocked the scan of the LIDAR. It was also seen that the airborne LIDAR technique used to create the OSGB ground-truth gave a better representation of the height of objects. Accurate height was not achieved by the hand-held scanner, potentially permitting propagation of the rays over unfinished buildings.

Probably the most compelling reason to classify the data is not just to label objects which will remain within the environment, but to remove dynamic objects such as motor vehicles and pedestrians. This has been adequately demonstrated by the misclassified bus, discussed in sections 6.4 and 6.5, which adversely affects the propagation. It can also be seen when the environment has been assigned the same generic material type. This effect is best illustrated by figures 6.16 and 6.17 which reveals significant loss in propagation when the rays meet brick-like foliage. Extrapolating this out to include the dynamic clutter, which would not be removed without labelling, and it is possible to see why labelling to remove objects is just as important as labelling to leave them in.

## 6.7 Summary

This chapter defined an automatic clutter classification pipeline which is capable of classifying raw LIDAR point clouds before converting them into a suitable mesh form for use within a RF propagation tool. It gave an overview of three different point clouds. One was used as a proof-of-concept to demonstrate that the classification model was able to perform to a reasonable level accurately of a UK environment. This was needed because the model was trained using French architecture. The others were fed into the pipeline for subsequent evaluation as environments within the propagation tool. A description of how the data was obtained and the motivation for using the data was given. This highlighted several problems with the technique used for hand-capturing the Woodland Road point cloud including the failure of the scanner to locate itself on the road and the duplication of the industrial sponsor as they followed behind the scanner.

The stages of the pipeline were given including the classification component, the techniques used to create and simplify the general meshes. This included the introduction of a simple, but effective, technique to form simplified meshes representing foliage. This was achieved by shrink-wrapping cone shapes around large collections of data points.

An evaluation of the pipeline meshes and the existing ground-truth was then performed. Propagation tests were set-up which simulated a person carrying UE as they walked past a base-station. The results were presented including providing visual illustrations of the path taken by the UE. This included a colour-coded representation of the signal strength received.

The set of experiments which concerned the NPM3D data showed that the ground-truth, benchmark and sub-sampled tests performed in line with each other, with the latter two having an average improvements in signal strength, over the ground-truth, of 1.01 dB and 2.67 dB. The main reason for this is the similarity of those datasets to that of the ground-truth i.e. whilst the ground-truth was hand-labelled it was specifically designed for use within the propagation model and consequently, was processed in the pipeline in the same way as the others. The accuracy of all three types of dataset was also in excess of 80%. When a generic material was assigned to all objects, large gaps appeared where the rays were not able to propagate around concrete trees.

The Woodland Road experiments placed the base-station in two strategic locations, the first having good LOS with the second being located on a side street (Cantocks Close). When compared with the OSGB ground-truth, the results were lower, especially when the experiments included the mis-classified bus. The cleaned versions gave average signal strength differences of circa 8 dB. This extended to improvements where the cleaned generic material test outperformed the original, not cleaned, sub-sampled test. Again, the generic material performed slightly worse than the fully labelled environments. For the Cantocks Close base-station location, the results diverged more with a loss of circa $-11$ dB from the ground-truth. Importantly, it revealed a significant flaw in the data capture. This is the problem where scanning an environment from street-level did not capture the height or depth information correctly i.e. the base-station was located and elevated on a side-street, there was no way of capturing the detail in the area which connected the base-station to Woodland Road, through private land. Consequently, false LOS was given which boosted the signal.

Whilst the OSGB ground-truth appeared better, it is worth stressing that it does not necessary reflect the real-world for several reasons. This includes the fact that the resolution of the ground-truth environment was $1m^2$, meaning the classification were crude and could favour propagation. Also, the low resolution meant that is was incapable of including small street furniture, which the pipeline created environments were able to capture. In its favour, the ground-truth was found to have better classification emphasis of the materials representing the ground; although, it had fewer classes overall.

One of the surprising realisations came from the results which focussed upon the use of a single generic material. This generic material was used to describe every object within the environment. Coincidentally, a misclassified bus, which was initially thought to be an annoyance, gave a first insight into what would happen if the raw LIDAR was simply converted into a mesh and placed within the propagation model. It was misclassified because the training data appeared to lack high sided vehicles and was assigned the material properties of a building. The relevant test results revealed that the bus was indeed problematic for ray propagation. The single generic material reinforced this by showing the effects of treating trees and other objects as if they were the same as buildings. Those results showed large RF shadows where the scattered rays were unable to propagate around these structures. The realisation was that it was essential to label the point cloud, not just to select the best material properties for objects within the propagation tool, but in order to remove unwanted objects such as vehicles and pedestrians. In this instance, the origin of the generic material was a labelled point-cloud. The results of a true unlabelled environment would be significantly worse.

# Chapter 7

# Conclusion and recommendations

RF propagation modelling is already a viable alternative to the acquisition of measurements via physical site-survey, which is both time consuming and expensive to perform. A basic overview of the mathematics surrounding RF propagation was given in chapter 2. This was done in order to demonstrate the mathematical nature of the propagation, which forms the basis for modelling. It was noted that whilst propagation modelling tools can be restricted by computational power and resources, one area which could be improved upon is the creation of the environments which the tools are modelling.

A justification for automatic classification was given in chapter 3 which focussed mainly upon reducing the time/cost overheads and combating the lack of scalability with human-based labelling. An overview of appropriate sources of data was then given and a description of prior work, which have classified trees, buildings and roads using image-based datasets was presented.

Machine and deep learning techniques and architectures were reviewed in chapter 4. In particular, the incredible progress of deep learning, in the field of image processing, was shown. This chapter set the scene for employing such techniques as classifiers for the aforementioned data sources. An series of experimentation was performed using CNN and capsule networks on a newly created RGB-D dataset. This was done as a proof of concept and for architectural comparison. These achieved accuracy results in excess of 80% but had limited classification labelling, were too shallow to exploit capsule networks and used low resolution data, which resulted in coarse classifications. Another short-coming was the lack of experimentation with hyper-parameters which had an adverse impact on one of the tests. These short-comings were taken on-board when performing subsequent testing.

LIDAR point clouds became the focal point of the thesis thereafter. Chapter 5 was dedicated to deep learning architectures which were able to process point cloud data directly. A series of experimentation was performed which attempted to characterise the point cloud data. This was achieved by sub-sampling the data which also allowed evaluation of the time taken to train the network. The use of a slightly dated architecture provided great insight into the underlying structure of the point cloud, yielding classification accuracy over 80% whilst decimating the data from 2,048 point per object to 32. However, the model could only work with single objects leading to the use of a synthetic dataset where it appeared the objects were too diverse e.g. a plane versus a table. It was also not clear whether the classifier was using the scale of an object over its shape. Furthermore, the synthetic dataset did not reflect real-world conditions; including the uneven distribution of points, inclusion of noise and point omissions.

A second set of point cloud experiments were performed using a state-of-the-art model with real-world data. This was also based around perceived efficiencies which could be gained via sub-sampling. In this instance though, emphasis was placed upon intelligently decided which points to keep. Retained edge features allowed the model to achieve accuracy results of circa 80% (range 74.1% to 82.55%) for the most decimated point cloud, compared to the benchmark of 87.34%. This was achieved with a fraction of the points used, sometimes as low as 5%. These experiment were performed to meet objective 1.1.2 which considers the impact of sub-sampling point cloud data.

An automatic clutter classification pipeline was proposed in chapter 6. This pipeline meets objective 1.1.1 by providing the mechanism for raw LIDAR point cloud to be classified and prepared for use within a propagation modelling tool. Part of that pipeline also provides a simplified mesh which meets objective 1.1.4. As an example, a novel centroid-based process was used to significantly reduce the mesh representation of complicated structures such as foliage. It simply shrink-wrapped an upside down cone shape around large clusters of points.

A series of experimentation was performed to test i) the pipeline and ii) whether automatic classification works for propagation modelling. Furthermore, positive results from applying the output from the pipeline within the propagation modelling tool met objectives 1.1.3 and 1.1.5, These are concerned with whether automatic classification could replace hand-labelled environments and whether labelling should be used.

The first set of experiments directly compared hand-labelled with automatically classified point cloud data. The results show all data types performed within 3 dBm and not in favour of the hand-labelled data. Additionally, automatic classification proved to be fast and therefore scalable process, taking only a few hours to label many millions of points. The second set of experiments used a new Woodland Road dataset and compared automatically classified data with a hand-crafted environment which is the basis for 1.1.3. The results of these tests were slightly worse with total average signal strength losses, when compared to the ground truth, of circa $-8$ dBm for a favourable placing of the base-station, and, circa $-11$ dBm for a poorly located base-station. The latter revealed a flaw in the data capture technique which meant that the dataset did not include sufficient height and depth information. Whilst these results are not as good as the ground-truth, the signal strength differences are still reasonable. There are also questions over whether the design on the ground-truth reflects the real-world environment. This is discussed further in section 6.6. Furthermore, the results meet part 2 of 1.1.3 by providing a fast and efficient scalable solution.

Objective 1.1.5 asks to justify whether either hand-labelled or classified data should be used in preference to not labelling. Experimentation with the use of a single generic material demonstrated that it was always preferable to label an environment. Not only for what is included in the final environment, but in order to remove undesirable objects such as pedestrians and vehicles, which were shown to significantly affect propagation. This point was adequately demonstrated, in section 6.4, when a misclassified bus was left in the environment as it blocked LOS propagation.

## 7.1 Recommendations

There are three main recommendation which can be made:

- **Vertical and street captured LIDAR**. Section 4.2 performs tests using data captured using vertical aerial photography. Whilst is does not have the resolution of the data captured using the hand-held scanner it provides two useful pieces of information i) the height of buildings and ii) and detail of objects that are set back from the street. A more robust dataset could be created by combining the vertical data with the street data.

- **Pipeline meshes**. The high resolution of the point cloud data gave rise to some very complicated meshes. Some of these were addressed e.g. foliage meshes, and others were simply smoothed. However, when compare to the ground-truth, these were still very complicated. It is not say that the resolution details should not be maintained, just a better technique for simplifying meshes should be considered.

- **Ground-truth**. The ground-truth used in the propagation model was low resolution, this made genuine comparisons to the benefit of having high resolutions LIDAR difficult to assess. A better ground-truth should be created.

# BIBLIOGRAPHY

[1] *Fundamentals of VHF and UHF Propagation*, John Wiley & Sons, Ltd, ch. 2, pp. 15–31.

[2] *Reconstructing 3D buildings from aerial LiDAR with AI: details*.
`https://medium.com/geoai/reconstructing-3d-buildings-from-aerial-lidar-with-ai-details-6a81cb3079c0`.
Accessed: 2019-01-28.

[3] *Tesla P100-PCIE-16GB specifications*.
`https://www.nvidia.com/en-us/data-center/tesla-p100/`.
Accessed: 2020-12-28.

[4] B. A. OLSHAUSEN AND D. FIELD, *Emergence of simple-cell receptive field properties by learning a sparse code for natural images*, 381 (1996), pp. 607–9.

[5] A. ABEDINIA, M. HAHNB, AND F. SAMADZADEGANA, *An investigation into the registration of LIDAR intensity data and aerial images using the SIFT approach*, (2008).

[6] R. ACHANTA, A. SHAJI, K. SMITH, A. LUCCHI, P. FUA, AND S. SÜSSTRUNK, *SLIC Superpixels Compared to State-of-the-Art Superpixel Methods*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 34 (2012), pp. 2274–2282.

[7] J. B. ANDERSEN, T. S. RAPPAPORT, AND S. YOSHIDA, *Propagation measurements and models for wireless communications channels*, IEEE Communications Magazine, 33 (1995), pp. 42–49.

[8] APHEX34, *Typical convolutional neural network architecture*.
`https://commons.wikimedia.org/w/index.php?curid=45679374`.
Accessed on 17/01/2018.

[9] G. E. ATHANASIADOU, A. R. NIX, AND J. P. MCGEEHAN, *A ray tracing algorithm for microcellular wideband propagation modelling*, in 1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century, vol. 1, Jul 1995, pp. 261–265 vol.1.

[10] G. E. Athanasiadou, A. R. Nix, and J. P. McGeehan, *A microcellular ray-tracing propagation model and evaluation of its narrow-band and wide-band predictions*, IEEE Journal on Selected Areas in Communications, 18 (2000), pp. 322–335.

[11] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, *Speeded-Up Robust Features (SURF)*, Computer Vision and Image Understanding, 110 (2008), pp. 346–359.
Similarity Matching in Computer Vision and Multimedia.

[12] D. Bazazian, J. R. Casas, and J. Ruiz-Hidalgo, *Fast and Robust Edge Extraction in Unorganized Point Clouds*, in 2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2015, pp. 1–8.

[13] P. Belhumeur, J. Hespanha, and D. Kriegman, *Eigenfaces vs. Fisherfaces: recognition using class specific linear projection*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 19 (1997), pp. 711–720.

[14] Y. Ben-Shabat, M. Lindenbaum, and A. Fischer, *3DmFV: Three-Dimensional Point Cloud Classification in Real-Time Using Convolutional Neural Networks*, IEEE Robotics and Automation Letters, 3 (2018), pp. 3145–3152.

[15] Blender, *Blender, a free and open source 3D creation suite*.
`https://www.blender.org/`.
Accessed on 03/10/2020.

[16] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, *YOLOv4: Optimal Speed and Accuracy of Object Detection*, 2020.

[17] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein, *Learning shape correspondence with anisotropic convolutional neural networks*, in Advances in Neural Information Processing Systems 29, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds., Curran Associates, Inc., 2016, pp. 3189–3197.

[18] J. Bowers, R. Wang, L. yi Wei, and D. Maletz, *Parallel Poisson disk sampling with spectrum analysis on surfaces*, in In SIGGRAPH Asia '10, 2010.

[19] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, *Spectral Networks and Deep Locally Connected Networks on Graphs*, (2013).

[20] D. R. Bull and F. Zhang, *Chapter 2 - The human visual system*, in Intelligent Image and Video Compression (Second Edition), D. R. Bull and F. Zhang, eds., Academic Press, Oxford, second edition ed., 2021, pp. 17–58.

[21] M. Defferrard, X. Bresson, and P. Vandergheynst, *Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering*, CoRR, abs/1606.09375 (2016).

[22] I. ELKHRACHY, *Feature Extraction of Laser Scan Data Based on Geometric Properties*, Journal of the Indian Society of Remote Sensing, 45 (2017), pp. 1–10.

[23] ENV, *CASI Multispectral Imagery*.
`https://data.gov.uk/dataset/casi-multispectral-imagery`.
Accessed on 27/09/2017.

[24] ——, *LIDAR Point Cloud*.
`https://data.gov.uk/dataset/lidar-point-cloud`.
Accessed on 27/09/2017.

[25] ——, *Vertical Aerial Photography*.
`https://data.gov.uk/dataset/vertical-aerial-photography`.
Accessed on 27/09/2017.

[26] M. ERIKSON, *Species classification of individually segmented tree crowns in high-resolution aerial images using radiometric and morphologic image measures*, Remote Sensing of Environment, 91 (2004), pp. 469–477.

[27] ERS, *E.R.S. Vector Raytracing Capabilities: Simple Projective Environment Model Solution Design Description*, 2016.

[28] D. FIELD, *What Is the Goal of Sensory Coding?*, 6 (1994), pp. 559–601.

[29] GEOSLAM, *What is SLAM?*
`https://geoslam.com/what-is-slam/`.
Accessed on 23/02/2021.

[30] A. GÉRON, *Handson-ML*.
`https://github.com/ageron/handson-ml`, 2017.

[31] R. B. GIRSHICK, *Fast R-CNN*, CoRR, abs/1504.08083 (2015).

[32] R. B. GIRSHICK, J. DONAHUE, T. DARRELL, AND J. MALIK, *Rich feature hierarchies for accurate object detection and semantic segmentation*, CoRR, abs/1311.2524 (2013).

[33] GISGEOGRAPHY, *Multispectral vs Hyperspectral imagery explained*.
`http://gisgeography.com/multispectral-vs-hyperspectral-imagery-explained/`.
Accessed on 27/09/2017.

[34] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep Learning*, Adaptive computation and machine learning, MIT Press, 2016.

[35] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative Adversarial Nets*, in Advances in Neural Information Processing Systems 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds., Curran Associates, Inc., 2014, pp. 2672–2680.

[36] I. GOODFELLOW, J. SHLENS, AND C. SZEGEDY, *Explaining and Harnessing Adversarial Examples*, in International Conference on Learning Representations, 2015.

[37] F. A. GOUGEON, *A crown-following approach to the automatic delineation of individual tree crowns in high spatial resolution aerial images*, Canadian journal of remote sensing, 21 (1995), pp. 274–284.

[38] X.-F. HAN, J. S. JIN, M.-J. WANG, W. JIANG, L. GAO, AND L. XIAO, *A review of algorithms for filtering the 3D point cloud*, Signal Processing: Image Communication, 57 (2017), pp. 103–112.

[39] D.-C. HE AND L. WANG, *Texture unit, texture spectrum, and texture analysis*, IEEE transactions on Geoscience and Remote Sensing, 28 (1990), pp. 509–512.

[40] K. HE, G. GKIOXARI, P. DOLLÁR, AND R. GIRSHICK, *Mask R-CNN*, in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988.

[41] K. HE, X. ZHANG, S. REN, AND J. SUN, *Deep Residual Learning for Image Recognition*, in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[42] M. HEIKKILÄ, M. PIETIKÄINEN, AND C. SCHMID, *Description of Interest Regions with Center-Symmetric Local Binary Patterns*, in Computer Vision, Graphics and Image Processing, P. Kalra and S. Peleg, eds., vol. 4338 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2006, pp. 58–69.

[43] P. R. HILL, C. N. CANAGARAJAH, AND D. R. BULL, *Image segmentation using a texture gradient based watershed transform*, IEEE Transactions on Image Processing, 12 (2003), pp. 1618–1633.

[44] G. HINTON, S. SABOUR, AND N. FROSST, *Matrix capsules with EM routing*, 2018.

[45] G. E. HINTON, A. KRIZHEVSKY, AND S. D. WANG, *Transforming Auto-Encoders*, in Artificial Neural Networks and Machine Learning – ICANN 2011, T. Honkela, W. Duch, M. Girolami, and S. Kaski, eds., Berlin, Heidelberg, 2011, Springer Berlin Heidelberg, pp. 44–51.

[46] G. E. HINTON AND R. R. SALAKHUTDINOV, *Reducing the Dimensionality of Data with Neural Networks*, Science, 313 (2006), pp. 504–507.

[47] S. IOFFE AND C. SZEGEDY, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, in Proceedings of the 32nd International Conference on Machine Learning, F. Bach and D. Blei, eds., vol. 37 of Proceedings of Machine Learning Research, Lille, France, 07–09 Jul 2015, PMLR, pp. 448–456.

[48] M. JADERBERG, K. SIMONYAN, A. ZISSERMAN, AND K. KAVUKCUOGLU, *Spatial Transformer Networks*, in Advances in Neural Information Processing Systems 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds., Curran Associates, Inc., 2015, pp. 2017–2025.

[49] I. JOLLIFFE, *Principal Component Analysis*, Springer Series in Statistics, Springer, 2002.

[50] S. KIM, G. SCHREUDER, R. J. MCGAUGHEY, AND H.-E. ANDERSEN, *Individual tree species identification using LIDAR intensity data*, in ASPRS 2008 Annual Conference Portland, Oregon April, 2008.

[51] T. N. KIPF AND M. WELLING, *Semi-Supervised Classification with Graph Convolutional Networks*, CoRR, abs/1609.02907 (2016).

[52] S. KODORS, A. RATKEVICS, A. RAUSIS, AND J. BULS, *Building Recognition Using LiDAR and Energy Minimization Approach*, Procedia Computer Science, 43 (2015), pp. 109–117.
ICTE in Regional Development, December 2014, Valmiera, Latvia.

[53] I. KORPELA, B. DAHLIN, H. SCHÄFER, E. BRUUN, F. HAAPANIEMI, J. HONKASALO, S. ILVESNIEMI, V. KUUTTI, M. LINKOSALMI, J. MUSTONEN, ET AL., *Single-tree forest inventory using lidar and aerial images for 3D treetop positioning, species recognition, height and crown width estimation*, in Proceedings of ISPRS workshop on laser scanning, 2007, pp. 227–233.

[54] K. KRAUS AND N. PFEIFER, *Determination of terrain models in wooded areas with airborne laser scanner data*, ISPRS Journal of Photogrammetry and Remote Sensing, 53 (1998), pp. 193–203.

[55] A. KRIZHEVSKY AND G. HINTON, *Learning multiple layers of features from tiny images*, Computer Science Department, University of Toronto, Tech. Rep, 1 (2009).

[56] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *ImageNet Classification with Deep Convolutional Neural Networks*, in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds., Curran Associates, Inc., 2012, pp. 1097–1105.

[57] Y. LECUN, *The MNIST Database of Handwritten Digits*, http://yann.lecun.com/exdb/mnist/.

[58] Y. LeCun, L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik, *Comparison of learning algorithms for handwritten digit recognition*, in International Conference on Artificial Neural Networks, F. Fogelman and P. Gallinari, eds., Paris, 1995, EC2 & Cie, pp. 53–60.

[59] D. G. Lowe, *Object Recognition from Local Scale-Invariant Features*, in ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2, Washington, DC, USA, 1999, IEEE Computer Society.

[60] ——, *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, 60 (2004), pp. 91–110.

[61] Z. C. Marton, R. B. Rusu, and M. Beetz, *On Fast Surface Reconstruction Methods for Large and Noisy Datasets*, in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, May 12-17 2009.

[62] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst, *Geodesic Convolutional Neural Networks on Riemannian Manifolds*, in Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), ICCVW '15, Washington, DC, USA, 2015, IEEE Computer Society, pp. 832–840.

[63] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst, *ShapeNet: Convolutional Neural Networks on Non-Euclidean Manifolds*, CoRR, abs/1501.06297 (2015).

[64] D. Maturana and S. Scherer, *VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition*, in Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, September 2015, p. 922 – 928.

[65] J. W. McKown and R. L. Hamilton, *Ray tracing as a design tool for radio networks*, IEEE Network, 5 (1991), pp. 27–30.

[66] MeshLab, *MeshLab the open source system for processing and editing 3D triangular meshes*.
`https://www.meshlab.net/index.html`.
Accessed on 01/10/2020.

[67] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, *RangeNet++: Fast and Accurate LiDAR Semantic Segmentation*, in IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2019.

[68] D. P. Mitchell, *Generating Antialiased Images at Low Sampling Densities*, in Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques,

SIGGRAPH '87, New York, NY, USA, 1987, Association for Computing Machinery, p. 65–72.

[69] V. MNIH AND G. E. HINTON, *Learning to detect roads in high-resolution aerial images*, in European Conference on Computer Vision, Springer, 2010, pp. 210–223.

[70] F. MONTI, D. BOSCAINI, J. MASCI, E. RODOLA, J. SVOBODA, AND M. M. BRONSTEIN, *Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs*, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[71] NASA, *Elements of aerial photography*.
`https://web.archive.org/web/20110317194519/http://rst.gsfc.nasa.gov/Sect10/Sect10_1.html`.
Accessed on 27/09/2017.

[72] K. H. NG, E. K. TAMEH, A. DOUFEXI, M. HUNUKUMBURE, AND A. R. NIX, *Efficient multielement ray tracing with site-specific comparisons using measured MIMO channel data*, IEEE Transactions on Vehicular Technology, 56 (2007), pp. 1019–1032.

[73] T. OJALA, M. PIETIKAINEN, AND D. HARWOOD, *Performance evaluation of texture measures with classification based on Kullback discrimination of distributions*, in Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on, vol. 1, IEEE, 1994, pp. 582–585.

[74] T. OJALA, M. PIETIKÄINEN, AND T. MÄENPÄÄ, *Multiresolution grayscale and rotation invariant texture classification with local binary patterns*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 24 (2002), p. 2002.

[75] B. A. OLSHAUSEN AND D. J. FIELD, *Sparse coding with an overcomplete basis set: A strategy employed by V1?*, Vision Research, 37 (1997), pp. 3311–3325.

[76] N. PAPERNOT, P. D. MCDANIEL, AND I. J. GOODFELLOW, *Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples*, CoRR, abs/1605.07277 (2016).

[77] J. D. PARSONS, *The Mobile Radio Propagation Channel*, John Wiley & Sons, Ltd, 2001.

[78] PCL, *Fast triangulation of unordered point clouds*.
`https://pcl.readthedocs.io/projects/tutorials/en/latest/greedy_projection.html`.
Accessed on 01/10/2020.

[79] M. PIETIKÄINEN, *Computer vision using local binary patterns*, Springer, London New York, 2011.

[80] PROVISION, *ProPhecy: Operator's Manual Version 2.3*, 2008.

[81] C. R. QI, H. SU, K. MO, AND L. J. GUIBAS, *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*, CoRR, abs/1612.00593 (2016).

[82] C. R. QI, H. SU, M. NIESSNER, A. DAI, M. YAN, AND L. J. GUIBAS, *Volumetric and Multi-View CNNs for Object Classification on 3D Data*, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[83] C. R. QI, L. YI, H. SU, AND L. J. GUIBAS, *PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space*, CoRR, abs/1706.02413 (2017).

[84] J. REDMON, S. DIVVALA, R. GIRSHICK, AND A. FARHADI, *You Only Look Once: Unified, Real-Time Object Detection*, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[85] J. REDMON AND A. FARHADI, *YOLO9000: Better, Faster, Stronger*, in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6517–6525.

[86] J. REDMON AND A. FARHADI, *YOLOv3: An Incremental Improvement*, CoRR, abs/1804.02767 (2018).

[87] S. REN, K. HE, R. B. GIRSHICK, AND J. SUN, *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*, CoRR, abs/1506.01497 (2015).

[88] G. RIEGLER, A. OSMAN ULUSOY, AND A. GEIGER, *OctNet: Learning Deep 3D Representations at High Resolutions*, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[89] X. ROYNARD, J. DESCHAUD, AND F. GOULETTE, *Paris-Lille-3D: a large and high-quality ground truth urban point cloud dataset for automatic segmentation and classification*, CoRR, abs/1712.00032 (2017).

[90] X. ROYNARD, J.-E. DESCHAUD, AND F. GOULETTE, *Paris-Lille-3D: Benchmarking suite*. `https://npm3d.fr/paris-lille-3d`.
Accessed: 2021-01-20.

[91] ——, *Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification*, The International Journal of Robotics Research, 37 (2018), pp. 545–557.

[92] O. RUSSAKOVSKY, J. DENG, H. SU, J. KRAUSE, S. SATHEESH, S. MA, Z. HUANG, A. KARPATHY, A. KHOSLA, M. BERNSTEIN, A. C. BERG, AND L. FEI-FEI, *ImageNet Large Scale Visual Recognition Challenge*, International Journal of Computer Vision (IJCV), 115 (2015), pp. 211–252.

[93] S. Sabour, N. Frosst, and G. E. Hinton, *Dynamic Routing Between Capsules*, in Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., Curran Associates, Inc., 2017, pp. 3859–3869.

[94] T. K. Sarkar, Z. Ji, K. Kim, A. Medouri, and M. Salazar-Palma, *A survey of various propagation models for mobile communication*, IEEE Antennas and Propagation Magazine, 45 (2003), pp. 51–82.

[95] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset, *PCRNet: Point Cloud Registration Network using PointNet Encoding*, 2019.

[96] SatelliteImagingCorporation, *About Orthorectification*. https://www.satimagingcorp.com/services/orthorectification/. Accessed on 27/09/2017.

[97] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, Computational and Biological Learning Society, 2015, pp. 1–14.

[98] Y. Singh, *Comparison of Okumura, Hata and COST-231 Models on the Basis of Path Loss and Signal Strength*, 59 (2012), pp. 37–41.

[99] L. Sirovich and M. Kirby, *Low-dimensional procedure for the characterization of human faces*, J. Opt. Soc. Am. A, 4 (1987), pp. 519–524.

[100] W. Smith, P. Engineer, and D. S. W. Director, *3-D Object recognition from point clouds Dr. Bingcai Zhang, Engineering Fellow*, 2011.

[101] C. Song, P. Li, and F. Yang, *Multivariate texture measured by local binary pattern for multispectral image classification*, in Geoscience and Remote Sensing Symposium, 2006. IGARSS 2006. IEEE International Conference on, IEEE, 2006, pp. 2145–2148.

[102] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, J. Mach. Learn. Res., 15 (2014), pp. 1929–1958.

[103] B. St-Onge, J. Jumelet, M. Cobello, and C. Véga, *Measuring individual tree height using a combination of stereophotogrammetry and lidar*, Canadian Journal of Forest Research, 34 (2004), pp. 2122–2130.

[104] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M. Yang, and J. Kautz, *SPLATNet: Sparse Lattice Networks for Point Cloud Processing*, CoRR, abs/1802.08275 (2018).

[105] H. SU, S. MAJI, E. KALOGERAKIS, AND E. LEARNED-MILLER, *Multi-View Convolutional Neural Networks for 3D Shape Recognition*, in The IEEE International Conference on Computer Vision (ICCV), December 2015.

[106] B. SUN, *Experiment on visualising PointNet*.
`https://github.com/GitBoSun/PointNet_vis`, 2017.

[107] C. SZEGEDY, W. LIU, Y. JIA, P. SERMANET, S. REED, D. ANGUELOV, D. ERHAN, V. VANHOUCKE, AND A. RABINOVICH, *Going Deeper With Convolutions*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.

[108] C. SZEGEDY, W. ZAREMBA, I. SUTSKEVER, J. BRUNA, D. ERHAN, I. GOODFELLOW, AND R. FERGUS, *Intriguing properties of neural networks*, in International Conference on Learning Representations, 2014.

[109] E. TAMEH, A. NIX, AND M. BEACH, *A 3-D integrated macro and microcellular propagation model, based on the use of photogrammetric terrain and building data*, in 1997 IEEE 47th Vehicular Technology Conference. Technology in Motion, vol. 3, 1997, pp. 1957–1961 vol.3.

[110] J. R. TAYLOR AND S. T. LOVELL, *Mapping public and private spaces of urban agriculture in Chicago through the analysis of high-resolution aerial images in Google Earth*, Landscape and Urban Planning, 108 (2012), pp. 57–70.

[111] H. THOMAS, *KPConv: Flexible and Deformable Convolution for Point Clouds*.
`https://github.com/PRBonn/lidar-bonnetal`, 2019.
Accessed: 2020-03-01.

[112] H. THOMAS, C. R. QI, J.-E. DESCHAUD, B. MARCOTEGUI, F. GOULETTE, AND L. J. GUIBAS, *KPConv: Flexible and Deformable Convolution for Point Clouds*, Proceedings of the IEEE International Conference on Computer Vision, (2019).

[113] J. TOMPSON, R. GOROSHIN, A. JAIN, Y. LECUN, AND C. BREGLER, *Efficient Object Localization Using Convolutional Networks*, CoRR, abs/1411.4280 (2014).

[114] P. VIOLA AND M. JONES, *Rapid object detection using a boosted cascade of simple features*, in Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, vol. 1, 2001, pp. I–I.

[115] P. VIOLA AND M. JONES, *Robust Real-time Object Detection*, in International Journal of Computer Vision, 2001.

[116] L. WANG AND D.-C. HE, *Texture classification using texture spectrum*, Pattern Recognition, 23 (1990), pp. 905–910.

[117] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, *O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis*, ACM Transactions on Graphics (SIGGRAPH), 36 (2017).

[118] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, *Dynamic Graph CNN for Learning on Point Clouds*, ACM Transactions on Graphics (TOG), (2019).

[119] B. Wu, A. Wan, X. Yue, and K. Keutzer, *Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud*, ICRA, (2018).

[120] E. Xi, S. Bing, and Y. Jin, *Capsule Network Performance on Complex Data*, arXiv e-prints, (2017), p. arXiv:1712.03480.

[121] J. Yellott, *Spectral consequences of photoreceptor sampling in the rhesus retina*, Science, 221 (1983), pp. 382–385.

[122] ZEBGo, *Rotating hand-held LIDAR scanner*.
`https://geoslam.com/solutions/zeb-go`.
Accessed on 23/02/2021.

[123] M. D. Zeiler and R. Fergus, *Visualizing and Understanding Convolutional Networks*, in Computer Vision – ECCV 2014, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds., Cham, 2014, Springer International Publishing, pp. 818–833.

[124] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, *Deconvolutional networks*, in In CVPR, 2010.

[125] S. Zhang, D. Kong, E. Mellios, G. Hilton, A. Nix, T. A. Thomas, and A. Ghosh, *Impact of BS antenna number and array geometry on single-user LTE-A data throughputs in realistic Macro and Pico cellular environments*, in 2015 IEEE Wireless Communications and Networking Conference (WCNC), March 2015, pp. 1464–1469.

[126] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, *3D Point-Capsule Networks*, CoRR, abs/1812.10775 (2018).

[127] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao, *3D ShapeNets: A deep representation for volumetric shapes*, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015, pp. 1912–1920.

# Appendix A

## Full results from edge only experimentation

The full results of the experiments which retained only the edge features are presented here. The associated graphs can be found in section 5.3.2.3 of chapter 5.

Table A.1: Validation accuracy results for the edge only set of experiments - KNN=5

| **KNN** | **Threshold** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **5** | **5** | 75.08 | 98.79 | 95.75 | 81.17 | 65.16 | 79.39 | 47.24 | 27.76 | 95.47 | 85.02 |
| **5** | **6** | 75.77 | 98.76 | 95.53 | 78.53 | 62.42 | 80.01 | 49.50 | 37.03 | 96.30 | 83.86 |
| **5** | **7** | 76.85 | 98.68 | 95.25 | 79.80 | 64.77 | 77.78 | 47.69 | 50.08 | 95.97 | 81.63 |
| **5** | **8** | 73.23 | 98.60 | 94.83 | 76.32 | 58.47 | 73.61 | 46.78 | 35.04 | 94.59 | 80.82 |
| **5** | **9** | 74.49 | 98.44 | 94.37 | 76.97 | 62.23 | 75.47 | 47.09 | 44.07 | 95.04 | 76.77 |
| **5** | **10** | 67.86 | 98.28 | 93.89 | 73.34 | 58.06 | 72.72 | 44.89 | 2.81 | 93.43 | 73.33 |
| **5** | **11** | 70.24 | 98.12 | 93.63 | 72.39 | 63.12 | 70.30 | 47.07 | 22.32 | 93.80 | 71.40 |
| **5** | **12** | 66.19 | 97.97 | 93.07 | 72.09 | 60.54 | 66.56 | 40.58 | 3.04 | 92.01 | 69.83 |
| **5** | **13** | 65.51 | 97.79 | 92.40 | 70.56 | 54.37 | 65.58 | 40.18 | 11.44 | 90.76 | 66.56 |
| **5** | **14** | 62.11 | 97.56 | 91.90 | 65.18 | 49.55 | 63.26 | 36.32 | 0.79 | 89.56 | 64.89 |
| **5** | **15** | 60.75 | 97.34 | 90.94 | 64.00 | 44.43 | 57.89 | 31.98 | 10.20 | 87.88 | 62.10 |
| **5** | **16** | 61.33 | 97.00 | 90.39 | 59.19 | 47.00 | 62.41 | 32.14 | 16.73 | 87.10 | 60.05 |
| **5** | **17** | 55.03 | 96.70 | 89.12 | 53.04 | 38.21 | 50.46 | 27.20 | 0.00 | 83.81 | 56.71 |
| **5** | **18** | 53.43 | 96.13 | 87.52 | 52.60 | 35.87 | 51.71 | 22.96 | 0.00 | 82.31 | 51.81 |
| **5** | **19** | 49.26 | 95.62 | 86.24 | 45.00 | 37.22 | 28.88 | 21.61 | 0.00 | 79.97 | 48.80 |
| **5** | **20** | 45.19 | 95.24 | 83.98 | 37.82 | 32.46 | 17.77 | 18.88 | 0.00 | 76.95 | 43.59 |
| **5** | **21** | 41.97 | 94.21 | 81.47 | 31.18 | 24.51 | 21.29 | 16.57 | 0.00 | 70.24 | 38.27 |
| **5** | **22** | 36.43 | 93.30 | 79.40 | 23.01 | 8.64 | 12.56 | 11.52 | 0.00 | 66.55 | 32.87 |
| **5** | **23** | 30.96 | 91.61 | 76.34 | 15.69 | 2.19 | 0.81 | 5.83 | 0.00 | 59.02 | 27.11 |
| **5** | **24** | 28.01 | 89.61 | 72.29 | 12.86 | 0.00 | 0.57 | 4.18 | 0.00 | 50.81 | 21.74 |
| **5** | **25** | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

Table A.2: Validation accuracy results for the edge only set of experiments - KNN=10

| **KNN** | **Threshold** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **10** | **5** | 80.66 | 98.68 | 96.01 | 78.81 | 67.31 | 83.80 | 53.58 | 65.85 | 96.18 | 85.72 |
| **10** | **6** | 79.34 | 98.43 | 95.51 | 76.62 | 63.98 | 79.76 | 49.34 | 71.26 | 95.17 | 83.98 |
| **10** | **7** | 76.37 | 98.16 | 95.04 | 74.04 | 59.02 | 78.70 | 48.52 | 57.39 | 94.47 | 81.93 |
| **10** | **8** | 72.77 | 97.88 | 94.44 | 72.22 | 50.62 | 73.33 | 48.84 | 46.28 | 93.28 | 78.08 |
| **10** | **9** | 74.52 | 97.69 | 94.04 | 71.95 | 56.12 | 73.71 | 45.18 | 63.59 | 91.96 | 76.43 |
| **10** | **10** | 70.91 | 97.37 | 93.14 | 69.93 | 53.76 | 64.83 | 42.47 | 53.93 | 88.91 | 73.88 |
| **10** | **11** | 65.90 | 96.97 | 92.64 | 66.70 | 43.71 | 65.57 | 40.50 | 28.15 | 87.14 | 71.70 |
| **10** | **12** | 63.71 | 96.45 | 91.58 | 63.02 | 41.99 | 59.18 | 37.31 | 30.57 | 84.70 | 68.55 |
| **10** | **13** | 62.43 | 96.28 | 90.82 | 59.78 | 35.72 | 58.78 | 33.22 | 37.97 | 85.10 | 64.24 |
| **10** | **14** | 59.08 | 95.93 | 89.47 | 58.45 | 36.97 | 55.05 | 30.44 | 22.56 | 82.32 | 60.51 |
| **10** | **15** | 55.63 | 95.36 | 88.44 | 57.74 | 36.65 | 52.50 | 27.15 | 2.50 | 82.18 | 58.16 |
| **10** | **16** | 54.69 | 94.71 | 86.73 | 52.82 | 38.08 | 51.83 | 26.35 | 8.21 | 80.34 | 53.18 |
| **10** | **17** | 50.57 | 93.88 | 84.96 | 49.62 | 32.81 | 42.98 | 21.04 | 5.05 | 76.62 | 48.17 |
| **10** | **18** | 46.96 | 92.96 | 82.85 | 47.31 | 25.93 | 37.27 | 19.04 | 0.71 | 73.96 | 42.59 |
| **10** | **19** | 44.05 | 92.09 | 80.58 | 47.75 | 23.85 | 24.38 | 16.53 | 0.62 | 70.59 | 40.03 |
| **10** | **20** | 40.50 | 90.53 | 78.01 | 41.02 | 13.90 | 22.73 | 13.94 | 0.00 | 68.66 | 35.67 |
| **10** | **21** | 37.03 | 88.37 | 75.42 | 38.61 | 8.00 | 16.23 | 13.56 | 0.00 | 63.76 | 29.29 |
| **10** | **22** | 33.55 | 86.22 | 73.18 | 32.35 | 3.10 | 11.19 | 6.99 | 3.47 | 58.77 | 26.65 |
| **10** | **23** | 29.63 | 84.41 | 69.58 | 26.37 | 0.00 | 7.36 | 2.82 | 0.00 | 53.27 | 22.90 |
| **10** | **24** | 21.98 | 77.71 | 61.77 | 9.56 | 0.00 | 0.00 | 0.32 | 0.00 | 32.63 | 15.85 |
| **10** | **25** | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

Table A.3: Validation accuracy results for the edge only set of experiments - KNN=25

| KNN | Threshold | mean | ground | buildings | poles | bollards | trash_cans | barriers | pedestrians | cars | natural |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **25** | **5** | 76.84 | 97.60 | 95.10 | 67.37 | 57.59 | 77.09 | 47.04 | 77.50 | 89.63 | 82.63 |
| **25** | **6** | 73.06 | 96.82 | 94.24 | 63.26 | 49.44 | 72.19 | 43.76 | 69.91 | 85.30 | 82.59 |
| **25** | **7** | 69.29 | 95.83 | 93.23 | 57.14 | 45.17 | 62.17 | 37.61 | 73.78 | 81.16 | 77.47 |
| **25** | **8** | 65.04 | 95.24 | 92.57 | 54.00 | 33.43 | 58.19 | 32.90 | 66.84 | 77.99 | 74.19 |
| **25** | **9** | 58.74 | 93.86 | 90.75 | 51.10 | 22.32 | 53.26 | 27.97 | 42.42 | 73.77 | 73.26 |
| **25** | **10** | 56.06 | 92.58 | 89.60 | 44.21 | 21.66 | 47.49 | 25.43 | 46.28 | 69.45 | 67.83 |
| **25** | **11** | 52.12 | 91.59 | 87.22 | 43.70 | 17.81 | 46.44 | 20.44 | 33.21 | 65.33 | 63.32 |
| **25** | **12** | 52.09 | 90.07 | 85.39 | 43.37 | 17.71 | 44.56 | 19.13 | 45.01 | 61.74 | 61.80 |
| **25** | **13** | 46.65 | 88.63 | 82.71 | 39.68 | 7.27 | 38.37 | 13.65 | 35.27 | 59.78 | 54.46 |
| **25** | **14** | 42.82 | 85.67 | 79.73 | 38.92 | 7.70 | 34.26 | 12.87 | 20.12 | 56.69 | 49.38 |
| **25** | **15** | 42.34 | 85.37 | 79.04 | 36.34 | 4.29 | 30.24 | 15.02 | 28.85 | 54.45 | 47.44 |
| **25** | **16** | 37.58 | 81.07 | 74.65 | 35.43 | 0.00 | 26.07 | 10.92 | 19.77 | 48.87 | 41.42 |
| **25** | **17** | 33.62 | 76.83 | 69.84 | 28.61 | 0.00 | 17.97 | 6.26 | 20.30 | 46.08 | 36.66 |
| **25** | **18** | 29.64 | 71.60 | 67.13 | 27.69 | 0.00 | 6.42 | 4.92 | 19.02 | 42.20 | 27.82 |
| **25** | **19** | 29.76 | 68.32 | 65.48 | 18.20 | 0.00 | 4.70 | 5.23 | 43.60 | 41.92 | 20.34 |
| **25** | **20** | 24.16 | 60.88 | 58.79 | 14.67 | 0.00 | 2.45 | 3.27 | 25.55 | 34.80 | 17.04 |
| **25** | **21** | 20.48 | 53.76 | 53.75 | 10.39 | 0.00 | 0.00 | 2.25 | 18.16 | 32.90 | 13.09 |
| **25** | **22** | 18.11 | 50.50 | 47.50 | 7.51 | 0.00 | 2.42 | 0.85 | 14.98 | 27.33 | 11.95 |
| **25** | **23** | 16.80 | 39.12 | 42.11 | 4.67 | 0.00 | 0.00 | 1.04 | 30.85 | 24.24 | 9.13 |
| **25** | **24** | 11.73 | 35.34 | 39.76 | 1.69 | 0.00 | 0.28 | 0.35 | 0.00 | 20.36 | 7.80 |
| **25** | **25** | 10.74 | 32.24 | 34.46 | 2.31 | 0.00 | 0.00 | 0.63 | 0.00 | 20.58 | 6.43 |

Table A.4: Validation accuracy results for the edge only set of experiments - KNN=50

| **KNN** | **Threshold** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **50** | **5** | 63.67 | 95.18 | 94.32 | 57.90 | 27.77 | 65.93 | 38.38 | 43.35 | 70.29 | 79.89 |
| **50** | **6** | 58.16 | 93.81 | 93.57 | 46.20 | 16.42 | 64.04 | 30.83 | 37.58 | 64.10 | 76.92 |
| **50** | **7** | 52.44 | 91.85 | 92.10 | 47.07 | 11.59 | 49.84 | 28.08 | 20.23 | 58.06 | 73.10 |
| **50** | **8** | 49.93 | 88.88 | 89.27 | 38.27 | 9.81 | 47.94 | 23.59 | 32.03 | 51.08 | 68.51 |
| **50** | **9** | 47.04 | 87.14 | 87.13 | 35.24 | 9.88 | 43.01 | 23.04 | 21.61 | 48.03 | 68.28 |
| **50** | **10** | 44.20 | 84.69 | 84.30 | 37.80 | 4.72 | 40.15 | 18.46 | 20.13 | 43.46 | 64.12 |
| **50** | **11** | 40.24 | 82.41 | 82.44 | 33.87 | 0.51 | 33.35 | 17.90 | 16.67 | 40.26 | 54.71 |
| **50** | **12** | 37.51 | 74.69 | 73.41 | 29.51 | 0.06 | 34.87 | 15.99 | 14.73 | 36.44 | 57.85 |
| **50** | **13** | 34.62 | 72.24 | 73.02 | 29.49 | 0.00 | 31.71 | 11.52 | 15.84 | 34.93 | 42.81 |
| **50** | **14** | 33.05 | 67.07 | 67.29 | 26.15 | 0.00 | 26.32 | 11.20 | 16.60 | 30.50 | 52.33 |
| **50** | **15** | 26.42 | 56.87 | 61.42 | 18.11 | 0.00 | 14.12 | 5.77 | 15.90 | 25.56 | 40.06 |
| **50** | **16** | 25.63 | 54.40 | 60.08 | 16.15 | 0.00 | 7.72 | 4.95 | 12.44 | 24.40 | 50.56 |
| **50** | **17** | 21.79 | 45.54 | 54.19 | 12.09 | 0.00 | 4.84 | 3.93 | 12.01 | 22.99 | 40.47 |
| **50** | **18** | 17.57 | 36.20 | 49.16 | 8.75 | 0.00 | 3.31 | 3.28 | 10.63 | 19.91 | 26.92 |
| **50** | **19** | 15.52 | 30.19 | 48.50 | 6.10 | 0.00 | 0.34 | 3.10 | 8.82 | 17.91 | 24.76 |
| **50** | **20** | 13.74 | 31.98 | 42.24 | 7.01 | 0.00 | 0.00 | 2.56 | 6.82 | 19.68 | 13.35 |
| **50** | **21** | 11.77 | 22.62 | 39.74 | 4.54 | 0.00 | 0.45 | 3.08 | 6.61 | 17.65 | 11.20 |
| **50** | **22** | 10.42 | 16.59 | 38.27 | 2.75 | 0.00 | 0.00 | 2.04 | 7.47 | 14.43 | 12.20 |
| **50** | **23** | 7.95 | 10.01 | 31.78 | 2.19 | 0.00 | 0.17 | 1.58 | 4.84 | 13.19 | 7.80 |
| **50** | **24** | 8.35 | 8.15 | 28.34 | 2.79 | 0.00 | 0.00 | 2.68 | 15.07 | 11.71 | 6.39 |
| **50** | **25** | 6.68 | 4.63 | 27.21 | 0.23 | 0.00 | 0.00 | 1.95 | 12.91 | 7.30 | 5.88 |

Table A.5: Validation accuracy results for the edge only set of experiments - KNN=75

| KNN | Threshold | mean | ground | buildings | poles | bollards | trash_cans | barriers | pedestrians | cars | natural |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **75** | **5** | 60.47 | 90.91 | 92.23 | 50.85 | 21.51 | 65.59 | 37.57 | 60.04 | 55.91 | 69.63 |
| **75** | **6** | 52.65 | 89.50 | 90.93 | 43.36 | 11.90 | 53.33 | 31.64 | 29.98 | 47.86 | 75.36 |
| **75** | **7** | 48.45 | 85.99 | 87.61 | 38.39 | 10.96 | 48.60 | 27.94 | 22.26 | 43.39 | 70.95 |
| **75** | **8** | 44.26 | 82.48 | 84.30 | 33.42 | 10.02 | 43.88 | 24.24 | 14.55 | 38.92 | 66.54 |
| **75** | **9** | 40.68 | 81.48 | 86.70 | 30.58 | 1.48 | 33.84 | 18.34 | 21.35 | 35.11 | 57.26 |
| **75** | **10** | 37.21 | 75.31 | 78.22 | 26.56 | 0.00 | 29.89 | 15.14 | 18.32 | 32.55 | 58.90 |
| **75** | **11** | 33.59 | 70.20 | 74.97 | 23.06 | 0.00 | 24.99 | 13.94 | 16.97 | 29.31 | 48.88 |
| **75** | **12** | 30.38 | 61.91 | 67.59 | 20.31 | 0.00 | 21.74 | 8.57 | 12.00 | 25.54 | 55.79 |
| **75** | **13** | 25.55 | 49.07 | 59.89 | 15.09 | 0.00 | 14.47 | 7.11 | 12.84 | 22.74 | 48.69 |
| **75** | **14** | 23.68 | 45.15 | 57.28 | 13.96 | 0.00 | 14.14 | 6.71 | 9.41 | 21.27 | 45.17 |
| **75** | **15** | 20.37 | 35.58 | 52.64 | 12.69 | 0.00 | 7.05 | 6.53 | 8.46 | 18.91 | 41.49 |
| **75** | **16** | 20.24 | 34.60 | 51.81 | 11.03 | 0.00 | 7.59 | 5.58 | 11.16 | 17.93 | 42.51 |
| **75** | **17** | 17.30 | 26.57 | 48.96 | 13.95 | 0.00 | 3.67 | 5.96 | 7.90 | 16.00 | 32.65 |
| **75** | **18** | 15.38 | 24.87 | 46.84 | 9.35 | 0.00 | 1.52 | 5.21 | 4.81 | 15.49 | 30.33 |
| **75** | **19** | 11.68 | 14.75 | 45.10 | 5.44 | 0.00 | 0.00 | 4.54 | 3.53 | 13.48 | 18.28 |
| **75** | **20** | 12.15 | 11.41 | 42.42 | 5.45 | 0.00 | 0.19 | 3.93 | 11.85 | 13.08 | 20.96 |
| **75** | **21** | 10.17 | 9.29 | 40.42 | 5.63 | 0.00 | 0.00 | 4.04 | 6.25 | 11.30 | 14.56 |
| **75** | **22** | 8.60 | 5.54 | 35.73 | 6.08 | 0.00 | 0.00 | 3.78 | 3.19 | 10.95 | 12.14 |
| **75** | **23** | 8.41 | 3.96 | 35.35 | 4.59 | 0.00 | 0.00 | 2.79 | 7.07 | 11.00 | 10.91 |
| **75** | **24** | 5.66 | 2.51 | 25.34 | 2.62 | 0.00 | 0.00 | 3.57 | 2.08 | 8.66 | 6.17 |
| **75** | **25** | 4.74 | 0.13 | 23.07 | 0.00 | 0.00 | 0.00 | 2.40 | 2.97 | 8.30 | 5.75 |

Table A.6: Validation accuracy results for the edge only set of experiments - KNN=100

| KNN | Threshold | mean | ground | buildings | poles | bollards | trash_cans | barriers | pedestrians | cars | natural |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **100** | **5** | 55.10 | 89.01 | 90.84 | 41.00 | 22.81 | 60.24 | 35.67 | 36.23 | 46.69 | 73.41 |
| **100** | **6** | 49.31 | 85.05 | 87.95 | 35.19 | 14.66 | 54.45 | 30.96 | 26.61 | 40.12 | 68.84 |
| **100** | **7** | 48.23 | 83.78 | 88.01 | 34.98 | 10.46 | 50.47 | 25.75 | 36.84 | 35.39 | 68.39 |
| **100** | **8** | 42.02 | 80.59 | 85.72 | 28.04 | 6.86 | 39.18 | 20.28 | 21.78 | 33.09 | 62.67 |
| **100** | **9** | 35.46 | 74.42 | 82.59 | 24.15 | 0.24 | 35.99 | 16.60 | 7.84 | 29.50 | 47.85 |
| **100** | **10** | 33.36 | 68.08 | 74.87 | 20.92 | 1.83 | 26.22 | 12.42 | 12.26 | 27.89 | 55.76 |
| **100** | **11** | 29.85 | 60.47 | 68.94 | 19.41 | 0.08 | 20.47 | 10.66 | 10.74 | 25.25 | 52.67 |
| **100** | **12** | 24.63 | 47.51 | 59.01 | 17.62 | 0.00 | 15.26 | 9.55 | 5.80 | 22.85 | 44.06 |
| **100** | **13** | 23.04 | 41.63 | 56.81 | 14.30 | 0.00 | 10.67 | 7.56 | 7.21 | 21.12 | 48.02 |
| **100** | **14** | 21.60 | 38.81 | 55.32 | 10.73 | 0.00 | 13.94 | 7.68 | 9.26 | 18.88 | 39.75 |
| **100** | **15** | 19.33 | 30.91 | 52.52 | 9.37 | 0.00 | 7.71 | 5.88 | 7.56 | 15.98 | 44.03 |
| **100** | **16** | 17.33 | 28.33 | 50.90 | 10.00 | 0.00 | 6.59 | 4.84 | 7.82 | 14.92 | 32.56 |
| **100** | **17** | 16.04 | 22.58 | 48.19 | 7.89 | 0.00 | 1.04 | 5.39 | 9.77 | 14.83 | 34.66 |
| **100** | **18** | 15.16 | 21.81 | 48.42 | 7.07 | 0.00 | 0.94 | 5.10 | 5.26 | 13.96 | 33.91 |
| **100** | **19** | 13.47 | 14.65 | 44.84 | 5.98 | 0.00 | 0.00 | 6.08 | 10.48 | 11.62 | 27.55 |
| **100** | **20** | 12.17 | 10.45 | 44.33 | 4.37 | 0.00 | 0.19 | 3.30 | 10.63 | 11.62 | 24.59 |
| **100** | **21** | 11.01 | 6.91 | 42.82 | 5.09 | 0.00 | 0.60 | 2.62 | 10.78 | 9.92 | 20.36 |
| **100** | **22** | 9.12 | 4.92 | 42.70 | 2.78 | 0.00 | 0.01 | 3.38 | 3.60 | 8.94 | 15.71 |
| **100** | **23** | 7.58 | 4.14 | 35.08 | 3.13 | 0.00 | 0.00 | 2.02 | 4.90 | 9.12 | 9.79 |
| **100** | **24** | 6.43 | 0.51 | 26.77 | 0.00 | 0.00 | 0.00 | 1.64 | 13.50 | 8.61 | 6.81 |
| **100** | **25** | 4.32 | 0.55 | 22.20 | 0.00 | 0.00 | 0.00 | 1.91 | 2.09 | 6.55 | 5.56 |

Table A.7: Validation accuracy results for the edge only set of experiments - KNN=125

| **KNN** | **Threshold** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **125** | **5** | 55.98 | 89.35 | 92.74 | 46.19 | 21.57 | 63.57 | 33.29 | 40.92 | 41.54 | 74.68 |
| **125** | **6** | 48.52 | 84.60 | 88.81 | 30.05 | 14.90 | 47.98 | 30.18 | 32.14 | 36.31 | 71.67 |
| **125** | **7** | 43.14 | 81.58 | 90.24 | 26.20 | 12.19 | 45.63 | 26.53 | 18.35 | 31.91 | 55.62 |
| **125** | **8** | 39.73 | 77.61 | 86.32 | 20.54 | 8.09 | 37.89 | 22.89 | 20.33 | 29.23 | 54.64 |
| **125** | **9** | 34.25 | 69.88 | 76.81 | 22.14 | 2.29 | 31.49 | 15.29 | 6.61 | 27.58 | 56.17 |
| **125** | **10** | 30.18 | 63.60 | 72.38 | 20.03 | 0.19 | 22.59 | 10.59 | 3.29 | 25.01 | 53.91 |
| **125** | **11** | 26.80 | 50.97 | 62.79 | 16.01 | 0.79 | 20.01 | 9.28 | 6.09 | 22.98 | 52.25 |
| **125** | **12** | 23.47 | 45.98 | 62.28 | 7.49 | 0.04 | 15.71 | 7.43 | 8.57 | 19.85 | 43.89 |
| **125** | **13** | 21.81 | 40.44 | 59.18 | 8.22 | 0.06 | 10.25 | 6.66 | 9.88 | 18.87 | 42.73 |
| **125** | **14** | 20.01 | 36.02 | 56.43 | 6.52 | 0.00 | 8.89 | 6.33 | 6.24 | 17.24 | 42.40 |
| **125** | **15** | 18.74 | 31.19 | 52.72 | 9.37 | 0.00 | 6.74 | 5.90 | 6.73 | 16.66 | 39.36 |
| **125** | **16** | 17.76 | 29.80 | 51.08 | 7.29 | 0.00 | 5.31 | 5.26 | 8.69 | 16.04 | 36.40 |
| **125** | **17** | 16.04 | 24.38 | 49.92 | 5.32 | 0.00 | 3.41 | 4.75 | 9.18 | 14.02 | 33.35 |
| **125** | **18** | 14.64 | 18.31 | 48.39 | 4.21 | 0.00 | 1.00 | 4.88 | 10.40 | 12.58 | 32.00 |
| **125** | **19** | 13.25 | 11.90 | 45.88 | 5.06 | 0.00 | 0.00 | 5.51 | 8.64 | 12.31 | 29.92 |
| **125** | **20** | 11.23 | 9.02 | 43.52 | 3.06 | 0.00 | 0.00 | 3.29 | 7.72 | 10.89 | 23.61 |
| **125** | **21** | 10.87 | 5.33 | 43.64 | 2.63 | 0.00 | 0.00 | 3.41 | 6.87 | 9.72 | 26.24 |
| **125** | **22** | 7.78 | 2.25 | 37.80 | 2.42 | 0.00 | 0.00 | 1.97 | 3.83 | 9.91 | 11.87 |
| **125** | **23** | 8.27 | 1.33 | 37.61 | 1.98 | 0.00 | 0.81 | 2.05 | 7.87 | 8.70 | 14.06 |
| **125** | **24** | 7.30 | 0.42 | 34.86 | 1.31 | 0.00 | 0.61 | 2.61 | 8.32 | 7.04 | 10.50 |
| **125** | **25** | 5.14 | 1.40 | 23.54 | 1.26 | 0.00 | 0.00 | 2.95 | 4.92 | 5.87 | 6.29 |

Table A.8: Validation accuracy results for the edge only set of experiments - KNN=150

| KNN | Threshold | mean | ground | buildings | poles | bollards | trash_cans | barriers | pedestrians | cars | natural |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **150** | **5** | 54.17 | 87.06 | 91.89 | 37.71 | 19.11 | 58.05 | 35.78 | 47.96 | 38.23 | 71.72 |
| **150** | **6** | 47.04 | 84.16 | 91.31 | 22.81 | 21.41 | 44.66 | 34.35 | 27.51 | 33.47 | 63.64 |
| **150** | **7** | 41.80 | 77.92 | 85.31 | 20.70 | 16.18 | 53.17 | 26.70 | 7.53 | 30.55 | 58.18 |
| **150** | **8** | 37.07 | 74.42 | 84.98 | 22.37 | 10.25 | 39.01 | 19.81 | 8.90 | 28.94 | 44.92 |
| **150** | **9** | 31.26 | 67.07 | 79.19 | 13.66 | 5.63 | 25.36 | 15.74 | 3.49 | 25.73 | 45.48 |
| **150** | **10** | 27.02 | 55.79 | 68.48 | 11.65 | 0.40 | 23.29 | 12.10 | 3.76 | 22.31 | 45.38 |
| **150** | **11** | 25.90 | 51.96 | 66.42 | 10.45 | 0.14 | 20.02 | 8.20 | 6.61 | 20.94 | 48.35 |
| **150** | **12** | 24.75 | 50.96 | 66.55 | 9.73 | 1.24 | 12.04 | 7.56 | 4.56 | 18.98 | 51.08 |
| **150** | **13** | 21.77 | 43.41 | 60.81 | 8.07 | 0.67 | 7.92 | 6.96 | 5.16 | 18.76 | 44.12 |
| **150** | **14** | 20.67 | 41.38 | 58.82 | 8.93 | 0.00 | 6.39 | 5.72 | 3.88 | 17.88 | 43.01 |
| **150** | **15** | 17.97 | 31.36 | 53.78 | 6.72 | 0.00 | 3.98 | 6.13 | 5.67 | 16.86 | 37.26 |
| **150** | **16** | 17.39 | 24.80 | 49.46 | 4.38 | 0.00 | 4.71 | 4.69 | 15.03 | 15.54 | 37.87 |
| **150** | **17** | 14.41 | 17.72 | 47.93 | 4.03 | 0.00 | 0.89 | 4.28 | 9.04 | 14.86 | 30.92 |
| **150** | **18** | 13.96 | 15.12 | 46.44 | 3.38 | 0.00 | 1.01 | 4.40 | 10.62 | 12.69 | 31.96 |
| **150** | **19** | 12.16 | 8.65 | 45.15 | 2.82 | 0.04 | 0.41 | 3.99 | 5.22 | 11.26 | 31.90 |
| **150** | **20** | 10.84 | 5.39 | 45.32 | 1.86 | 0.00 | 0.45 | 3.65 | 4.36 | 10.42 | 26.15 |
| **150** | **21** | 9.98 | 3.47 | 41.76 | 1.87 | 0.00 | 0.00 | 3.42 | 7.61 | 9.38 | 22.32 |
| **150** | **22** | 8.85 | 1.50 | 39.59 | 1.76 | 0.00 | 0.00 | 4.04 | 4.97 | 9.14 | 18.61 |
| **150** | **23** | 8.28 | 2.18 | 38.68 | 1.95 | 0.00 | 0.00 | 2.57 | 4.25 | 8.01 | 16.85 |
| **150** | **24** | 5.93 | 0.82 | 27.64 | 0.70 | 0.00 | 0.00 | 3.15 | 6.16 | 6.75 | 8.13 |
| **150** | **25** | 4.62 | 0.84 | 22.02 | 0.30 | 0.00 | 0.14 | 2.36 | 4.46 | 5.48 | 5.95 |

# Appendix B

# Full results from combined edges with sub-sampling

The results of the experiments which combined the edge points with sub-sampling are presented here, ordered by level of severity. The associated edges points have been sub-sampled at half the resolution of the main sub-sampling. For example, the edges are sub-sampled at a resolution of 16 cm when the main dataset has been sub-sampled at a resolution of 32 cm. Each section contains a table of results showing the number of points used, the mean accuracy and accuracy on a class by class basis. Two further charts are included for each sub-sampled experiment. These are plots of the results on a class by class basis from the perspectives of flatness threshold and k-nearest neighbours respectively.

## B.1 Edge points with sub-sampling at 32 cm

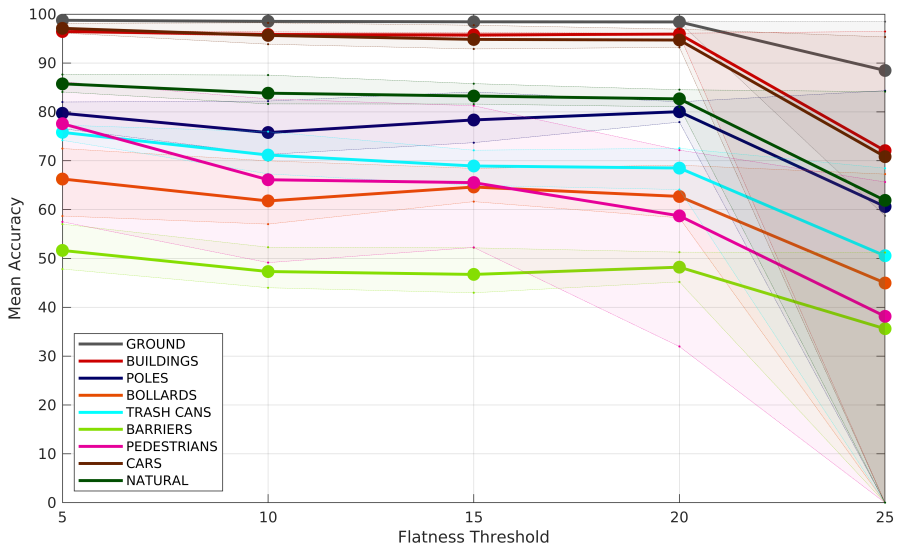


Figure B.1: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 32 cm.

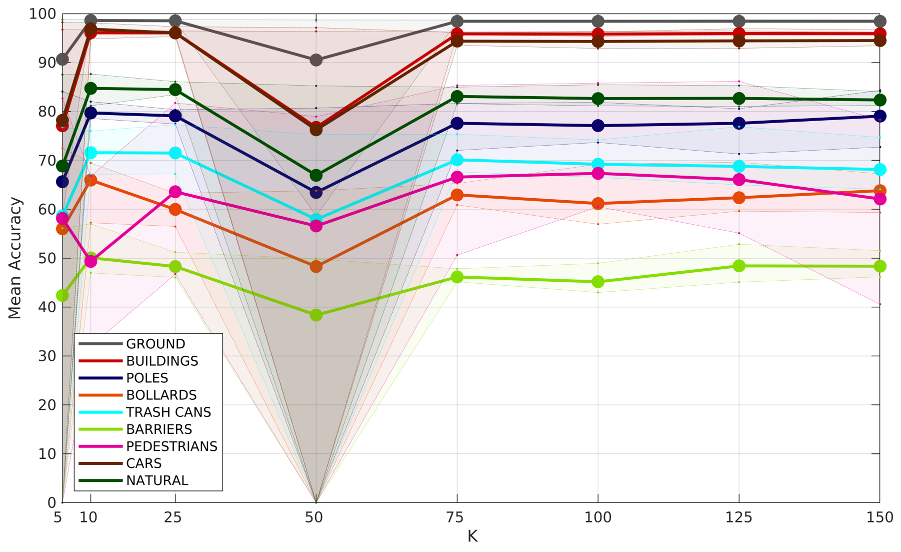


Figure B.2: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 32 cm.

Table B.1: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 32 cm.

| **Sub-sampling** | | | | **Number of points** | | | | | | | | **Accuracy** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **32** | **5** | **5** | 2040100 | 1132272 | 2799836 | 1750596 | 80.61 | 98.79 | 96.72 | 81.83 | 72.50 | 76.76 | 56.10 | 57.50 | 98.12 | 87.16 |
| **32** | **5** | **10** | 1566209 | 917922 | 2228893 | 1387049 | 82.66 | 98.72 | 96.38 | 82.21 | 69.98 | 75.94 | 52.29 | 82.66 | 98.22 | 87.53 |
| **32** | **5** | **15** | 1160159 | 705768 | 1718648 | 1039969 | 81.37 | 98.66 | 96.23 | 84.09 | 68.43 | 68.18 | 52.17 | 81.25 | 97.74 | 85.63 |
| **32** | **5** | **20** | 836053 | 496980 | 1254525 | 723981 | 79.73 | 98.49 | 96.05 | 80.07 | 69.09 | 72.55 | 51.28 | 69.18 | 96.94 | 83.88 |
| **32** | **5** | **25** | 674172 | 352450 | 945149 | 539085 | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **32** | **10** | **5** | 2101258 | 1172545 | 3025107 | 1775224 | 81.29 | 98.79 | 96.75 | 82.02 | 68.42 | 76.01 | 57.02 | 66.79 | 98.14 | 87.66 |
| **32** | **10** | **10** | 1578881 | 947794 | 2448700 | 1376567 | 78.03 | 98.68 | 96.32 | 79.20 | 69.48 | 74.39 | 51.44 | 49.14 | 98.02 | 85.60 |
| **32** | **10** | **15** | 1163177 | 728281 | 1898607 | 1033060 | 77.12 | 98.61 | 95.82 | 78.79 | 67.52 | 71.33 | 47.19 | 52.25 | 96.83 | 85.78 |
| **32** | **10** | **20** | 856766 | 526952 | 1404935 | 747801 | 74.43 | 98.54 | 95.77 | 79.89 | 67.18 | 68.96 | 47.72 | 31.97 | 96.42 | 83.47 |
| **32** | **10** | **25** | 683446 | 375107 | 1025290 | 556332 | 74.10 | 98.45 | 95.96 | 78.57 | 57.23 | 67.17 | 46.98 | 46.52 | 94.85 | 81.15 |
| **32** | **25** | **5** | 1854044 | 1061981 | 3003524 | 1543865 | 80.93 | 98.76 | 96.40 | 80.37 | 61.44 | 77.28 | 48.93 | 81.73 | 97.34 | 86.09 |
| **32** | **25** | **10** | 1335586 | 807401 | 2394043 | 1123905 | 77.80 | 98.57 | 95.96 | 77.43 | 57.03 | 71.10 | 46.70 | 72.73 | 96.55 | 84.15 |
| **32** | **25** | **15** | 999620 | 602740 | 1806271 | 842182 | 76.99 | 98.46 | 95.78 | 79.82 | 63.28 | 72.15 | 46.06 | 58.30 | 95.60 | 83.45 |
| **32** | **25** | **20** | 788637 | 452219 | 1327955 | 654712 | 76.87 | 98.48 | 95.91 | 79.17 | 61.58 | 69.77 | 48.58 | 58.37 | 95.49 | 84.53 |
| **32** | **25** | **25** | 678489 | 358546 | 1013717 | 543749 | 74.98 | 98.46 | 96.46 | 78.73 | 56.49 | 67.21 | 51.22 | 46.74 | 95.33 | 84.14 |
| **32** | **50** | **5** | 1623171 | 921294 | 2884500 | 1320073 | 80.09 | 98.71 | 96.36 | 80.57 | 58.68 | 75.29 | 49.83 | 78.96 | 97.14 | 85.24 |
| **32** | **50** | **10** | 1227007 | 695264 | 2291348 | 984617 | 76.30 | 98.51 | 95.72 | 77.55 | 58.07 | 72.18 | 45.32 | 60.06 | 95.13 | 84.20 |
| **32** | **50** | **15** | 974625 | 543616 | 1726930 | 774260 | 78.20 | 98.36 | 95.61 | 78.39 | 63.79 | 70.74 | 47.42 | 71.68 | 95.12 | 82.64 |
| **32** | **50** | **20** | 793510 | 429457 | 1275267 | 630825 | 78.36 | 98.40 | 95.98 | 80.69 | 60.93 | 71.43 | 49.21 | 72.17 | 93.79 | 82.67 |
| **32** | **50** | **25** | 684026 | 352675 | 989818 | 539627 | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **32** | **75** | **5** | 1542578 | 850124 | 2809511 | 1236588 | 81.07 | 98.70 | 96.25 | 79.78 | 65.29 | 75.34 | 47.83 | 85.33 | 96.15 | 84.97 |
| **32** | **75** | **10** | 1215097 | 659250 | 2245201 | 944532 | 77.33 | 98.46 | 95.69 | 72.03 | 61.88 | 70.34 | 45.16 | 75.53 | 93.85 | 83.00 |
| **32** | **75** | **15** | 990787 | 529497 | 1720142 | 759691 | 76.47 | 98.37 | 95.65 | 74.49 | 64.87 | 69.20 | 45.91 | 64.65 | 93.55 | 81.60 |
| **32** | **75** | **20** | 807581 | 426422 | 1274266 | 628141 | 75.66 | 98.34 | 95.82 | 79.97 | 60.89 | 67.65 | 45.58 | 56.76 | 94.14 | 81.82 |
| **32** | **75** | **25** | 688529 | 351747 | 985523 | 539153 | **75.66** | **98.37** | **95.91** | **81.66** | **61.78** | **68.04** | **46.30** | **50.63** | **94.20** | **84.00** |
| **32** | **100** | **5** | 1512814 | 816265 | 2759974 | 1202157 | 81.48 | 98.71 | 96.36 | 78.36 | 69.21 | 74.18 | 48.95 | 85.80 | 96.28 | 85.46 |
| **32** | **100** | **10** | 1218200 | 642787 | 2232692 | 926992 | 75.31 | 98.43 | 95.46 | 73.66 | 57.59 | 69.67 | 44.00 | 63.37 | 93.93 | 81.63 |
| **32** | **100** | **15** | 1005453 | 524645 | 1733746 | 754723 | 75.40 | 98.35 | 95.39 | 73.70 | 63.85 | 68.67 | 42.97 | 60.52 | 92.88 | 82.24 |
| **32** | **100** | **20** | 817959 | 425116 | 1285499 | 627795 | 75.47 | 98.35 | 95.93 | 77.91 | 58.31 | 66.97 | 45.20 | 61.39 | 94.03 | 81.15 |
| **32** | **100** | **25** | 692281 | 351406 | 987059 | 539301 | 76.37 | 98.40 | 96.03 | 81.87 | 56.93 | 66.52 | 44.72 | 65.65 | 94.46 | 82.73 |
| **32** | **125** | **5** | 1500739 | 798166 | 2724263 | 1185755 | 82.36 | 98.76 | 96.48 | 78.23 | 69.63 | 76.89 | 52.85 | 86.17 | 96.94 | 85.30 |
| **32** | **125** | **10** | 1224083 | 633944 | 2231724 | 917787 | 75.21 | 98.38 | 95.61 | 71.30 | 60.78 | 68.39 | 46.73 | 58.12 | 94.82 | 82.75 |
| **32** | **125** | **15** | 1016328 | 522195 | 1749686 | 752706 | 76.23 | 98.34 | 95.76 | 77.99 | 61.63 | 64.98 | 45.09 | 66.72 | 92.91 | 82.61 |
| **32** | **125** | **20** | 827618 | 424775 | 1297588 | 628094 | 76.37 | 98.34 | 95.83 | 80.55 | 59.61 | 66.62 | 47.98 | 64.19 | 93.21 | 81.02 |
| **32** | **125** | **25** | 695209 | 351383 | 988920 | 540065 | 75.79 | 98.41 | 96.08 | 79.85 | 60.17 | 66.97 | 49.45 | 55.12 | 94.34 | 81.73 |
| **32** | **150** | **5** | 1496267 | 786758 | 2699105 | 1176607 | 80.20 | 98.74 | 96.24 | 76.57 | 64.93 | 74.68 | 51.54 | 78.30 | 96.72 | 84.07 |
| **32** | **150** | **10** | 1229729 | 628578 | 2234150 | 913326 | 76.00 | 98.46 | 95.60 | 72.71 | 59.34 | 67.31 | 46.92 | 67.24 | 94.65 | 81.73 |
| **32** | **150** | **15** | 1025494 | 520328 | 1765237 | 751930 | 77.23 | 98.33 | 95.74 | 79.55 | 63.54 | 66.13 | 47.01 | 68.73 | 94.09 | 81.98 |
| **32** | **150** | **20** | 834456 | 424460 | 1309531 | 628878 | 76.31 | 98.33 | 96.10 | 82.06 | 63.96 | 64.10 | 50.16 | 55.73 | 93.61 | 82.75 |
| **32** | **150** | **25** | 697126 | 351025 | 991437 | 540980 | 75.09 | 98.36 | 95.92 | 84.34 | 67.26 | 68.47 | 46.18 | 40.55 | 93.44 | 81.25 |
| **32** | **150** | **25** | 1535441 | 798022 | 2143867 | 1259415 | 82.55 | 98.77 | 96.77 | 86.96 | 69.35 | 74.90 | 55.09 | 78.79 | 97.54 | 84.82 |

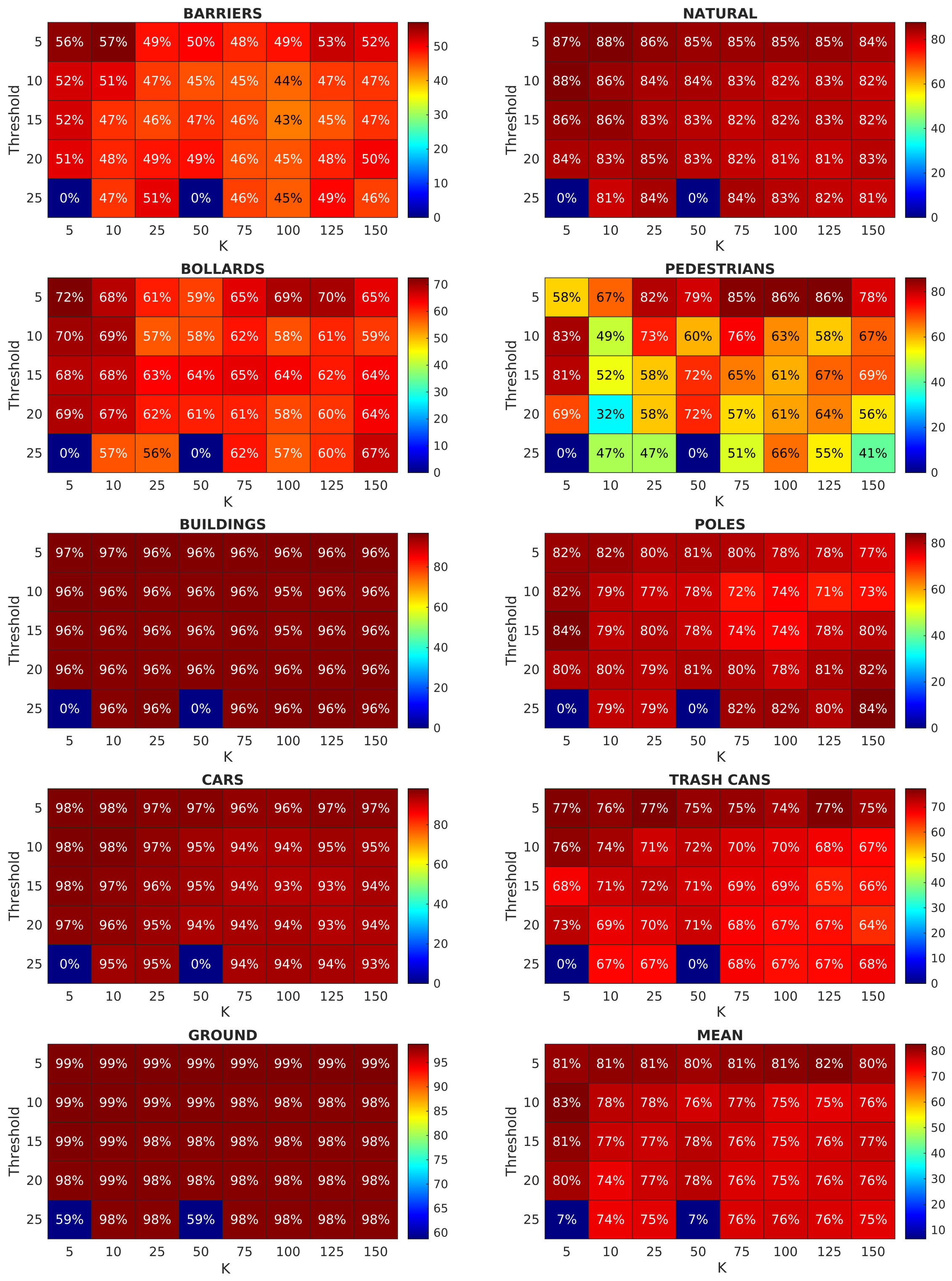


Figure B.3: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 32 cm) point cloud.

## B.2 Edge points with sub-sampling at 28 cm

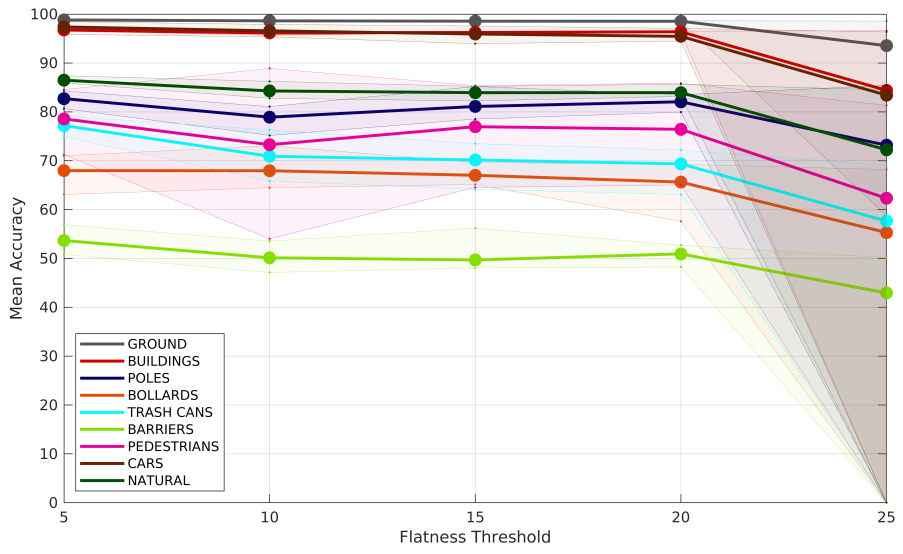


Figure B.4: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 28 cm.

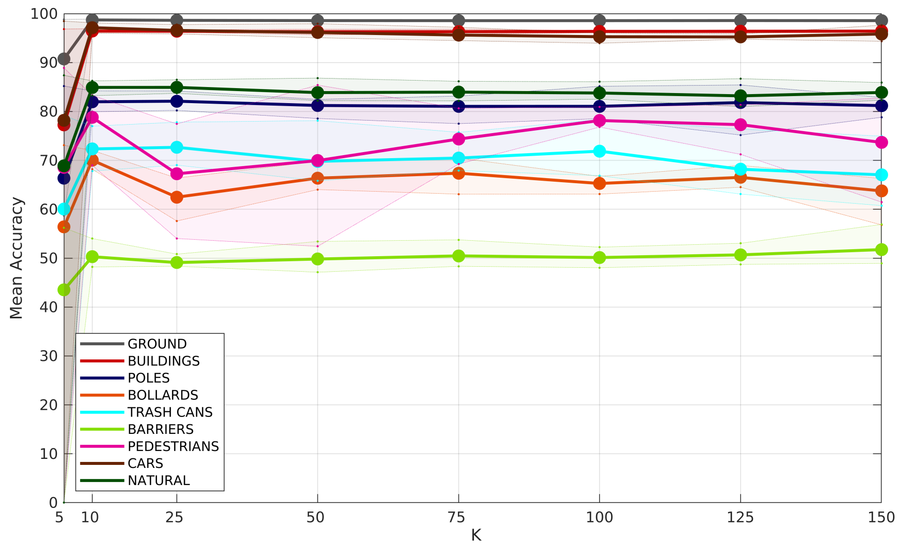


Figure B.5: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 28 cm.

Table B.2: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 28 cm.

| Sub-sampling | | | | Number of points | | | Accuracy | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **28** | **5** | **5** | 2500317 | 1413280 | 3463550 | 2189983 | 83.91 | 98.84 | 96.86 | 84.31 | 70.47 | 79.16 | 55.11 | 84.57 | 98.47 | 87.39 |
| **28** | **5** | **10** | 1891120 | 1130309 | 2736973 | 1704985 | 83.60 | 98.81 | 96.54 | 81.07 | 73.09 | 76.31 | 53.56 | 88.89 | 97.90 | 86.25 |
| **28** | **5** | **15** | 1391904 | 859121 | 2097610 | 1262774 | 82.75 | 98.71 | 96.51 | 85.20 | 69.23 | 72.68 | 56.21 | 83.56 | 97.57 | 85.12 |
| **28** | **5** | **20** | 1020271 | 607106 | 1536613 | 887642 | 82.08 | 98.61 | 96.39 | 81.14 | 69.21 | 72.03 | 52.72 | 85.87 | 97.09 | 85.66 |
| **28** | **5** | **25** | 849368 | 446036 | 1188822 | 686702 | 6.52 | 58.71 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **28** | **10** | **5** | 2577401 | 1465647 | 3748749 | 2228163 | 83.30 | 98.90 | 96.90 | 84.20 | 71.01 | 77.00 | 54.01 | 83.30 | 98.12 | 86.26 |
| **28** | **10** | **10** | 1913494 | 1171682 | 3017411 | 1703165 | 80.90 | 98.81 | 96.53 | 81.08 | 72.04 | 74.95 | 52.44 | 69.00 | 97.82 | 85.43 |
| **28** | **10** | **15** | 1405604 | 893142 | 2328885 | 1266341 | 80.88 | 98.70 | 96.16 | 81.17 | 69.63 | 71.89 | 48.21 | 79.52 | 97.41 | 85.21 |
| **28** | **10** | **20** | 1048431 | 646908 | 1724189 | 921806 | 80.61 | 98.65 | 96.20 | 79.95 | 69.28 | 70.03 | 48.24 | 82.69 | 96.01 | 84.47 |
| **28** | **10** | **25** | 859199 | 471691 | 1280034 | 705811 | 80.23 | 98.55 | 96.39 | 83.50 | 68.20 | 67.81 | 48.57 | 79.45 | 96.36 | 83.23 |
| **28** | **25** | **5** | 2256925 | 1319385 | 3704396 | 1926405 | 81.22 | 98.83 | 96.70 | 82.88 | 64.47 | 77.81 | 50.83 | 75.22 | 97.77 | 86.48 |
| **28** | **25** | **10** | 1611621 | 992634 | 2932893 | 1388184 | 77.70 | 98.66 | 96.23 | 80.22 | 66.38 | 73.06 | 48.35 | 54.02 | 96.90 | 85.49 |
| **28** | **25** | **15** | 1216095 | 741309 | 2209040 | 1043754 | 78.62 | 98.55 | 96.22 | 80.67 | 65.14 | 73.49 | 48.66 | 64.54 | 95.86 | 84.46 |
| **28** | **25** | **20** | 977032 | 561936 | 1635005 | 820047 | 77.71 | 98.61 | 96.37 | 82.55 | 57.60 | 69.98 | 49.22 | 65.06 | 96.33 | 83.71 |
| **28** | **25** | **25** | 855106 | 454566 | 1270730 | 693557 | 79.27 | 98.54 | 96.50 | 84.19 | 58.77 | 69.03 | 48.46 | 77.46 | 96.02 | 84.49 |
| **28** | **50** | **5** | 1961811 | 1138399 | 3536975 | 1638968 | 83.01 | 98.75 | 96.82 | 82.29 | 69.81 | 78.13 | 53.42 | 83.10 | 97.95 | 86.83 |
| **28** | **50** | **10** | 1478698 | 853299 | 2789568 | 1217776 | 76.62 | 98.61 | 95.69 | 78.56 | 65.52 | 70.78 | 47.12 | 54.11 | 95.92 | 83.28 |
| **28** | **50** | **15** | 1186802 | 669744 | 2104356 | 963418 | 80.25 | 98.50 | 96.20 | 81.61 | 65.97 | 66.91 | 48.12 | 85.39 | 96.24 | 83.34 |
| **28** | **50** | **20** | 982210 | 535059 | 1570267 | 792903 | 79.51 | 98.56 | 96.40 | 82.47 | 66.51 | 67.38 | 50.81 | 74.76 | 95.08 | 83.64 |
| **28** | **50** | **25** | 861289 | 447950 | 1242784 | 689233 | 76.21 | 98.51 | 96.44 | 81.21 | 64.03 | 65.81 | 49.64 | 52.42 | 95.58 | 82.21 |
| **28** | **75** | **5** | 1862209 | 1047977 | 3436008 | 1534181 | 82.17 | 98.79 | 96.81 | 81.51 | 70.27 | 75.73 | 53.75 | 79.30 | 97.23 | 86.16 |
| **28** | **75** | **10** | 1464613 | 808375 | 2725516 | 1168359 | 80.13 | 98.54 | 95.81 | 77.51 | 70.48 | 69.09 | 49.00 | 80.64 | 96.29 | 83.80 |
| **28** | **75** | **15** | 1203762 | 652487 | 2092484 | 945793 | 79.04 | 98.53 | 96.17 | 80.48 | 65.68 | 70.92 | 49.64 | 71.36 | 94.99 | 83.59 |
| **28** | **75** | **20** | 996928 | 531126 | 1567504 | 790302 | 79.43 | 98.47 | 96.34 | 82.57 | 67.35 | 68.72 | 51.61 | 71.27 | 94.49 | 84.07 |
| **28** | **75** | **25** | 865957 | 446879 | 1237230 | 688828 | 78.23 | 98.54 | 96.48 | 83.14 | 63.09 | 67.91 | 48.34 | 69.26 | 95.13 | 82.17 |
| **28** | **100** | **5** | 1825938 | 1005408 | 3369696 | 1491160 | 81.19 | 98.79 | 96.56 | 80.66 | 63.10 | 78.37 | 52.26 | 78.76 | 96.07 | 86.09 |
| **28** | **100** | **10** | 1467601 | 787664 | 2706850 | 1147116 | 79.47 | 98.51 | 96.12 | 78.85 | 66.73 | 68.88 | 50.26 | 76.81 | 96.05 | 82.98 |
| **28** | **100** | **15** | 1219999 | 646017 | 2105637 | 940196 | 79.43 | 98.49 | 96.29 | 78.53 | 65.59 | 73.09 | 48.05 | 77.16 | 93.96 | 83.72 |
| **28** | **100** | **20** | 1008392 | 529544 | 1578490 | 789756 | 80.64 | 98.53 | 96.49 | 82.06 | 65.09 | 72.19 | 51.65 | 80.95 | 95.22 | 83.59 |
| **28** | **100** | **25** | 870141 | 446523 | 1238063 | 689147 | 79.56 | 98.55 | 96.46 | 85.15 | 65.88 | 66.87 | 48.42 | 77.07 | 95.13 | 82.52 |
| **28** | **125** | **5** | 1811817 | 982296 | 3322378 | 1469690 | 81.28 | 98.81 | 96.69 | 83.77 | 68.98 | 76.40 | 53.05 | 71.23 | 95.85 | 86.71 |
| **28** | **125** | **10** | 1474135 | 776880 | 2704592 | 1136000 | 78.61 | 98.57 | 96.07 | 75.18 | 64.90 | 65.83 | 48.73 | 80.06 | 95.36 | 82.79 |
| **28** | **125** | **15** | 1232117 | 642471 | 2123525 | 937133 | 79.08 | 98.53 | 96.32 | 81.24 | 68.63 | 64.16 | 49.69 | 75.47 | 95.44 | 82.24 |
| **28** | **125** | **20** | 1018583 | 529097 | 1591767 | 790003 | 80.29 | 98.47 | 96.46 | 83.51 | 64.51 | 71.43 | 51.76 | 78.41 | 94.79 | 83.24 |
| **28** | **125** | **25** | 872872 | 446338 | 1239758 | 689680 | 79.60 | 98.58 | 96.43 | 85.42 | 65.60 | 63.10 | 50.16 | 81.34 | 94.78 | 80.99 |
| **28** | **150** | **5** | 1806243 | 967416 | 3289721 | 1457497 | 81.29 | 98.84 | 96.69 | 82.05 | 65.69 | 74.95 | 56.90 | 72.98 | 97.64 | 85.91 |
| **28** | **150** | **10** | 1480826 | 769936 | 2706348 | 1130381 | 80.18 | 98.55 | 96.23 | 78.81 | 64.47 | 68.41 | 51.59 | 82.75 | 96.53 | 84.24 |
| **28** | **150** | **15** | 1241790 | 639999 | 2141220 | 935815 | 79.60 | 98.50 | 96.17 | 79.95 | 66.29 | 68.03 | 48.94 | 78.72 | 96.02 | 83.78 |
| **28** | **150** | **20** | 1025772 | 528686 | 1604358 | 791107 | 78.63 | 98.49 | 96.52 | 82.31 | 65.53 | 63.09 | 51.49 | 72.44 | 94.66 | 83.17 |
| **28** | **150** | **25** | 874752 | 445977 | 1242614 | 690677 | 75.95 | 98.50 | 96.60 | 82.88 | 56.80 | 60.74 | 49.85 | 61.46 | 94.36 | 82.31 |

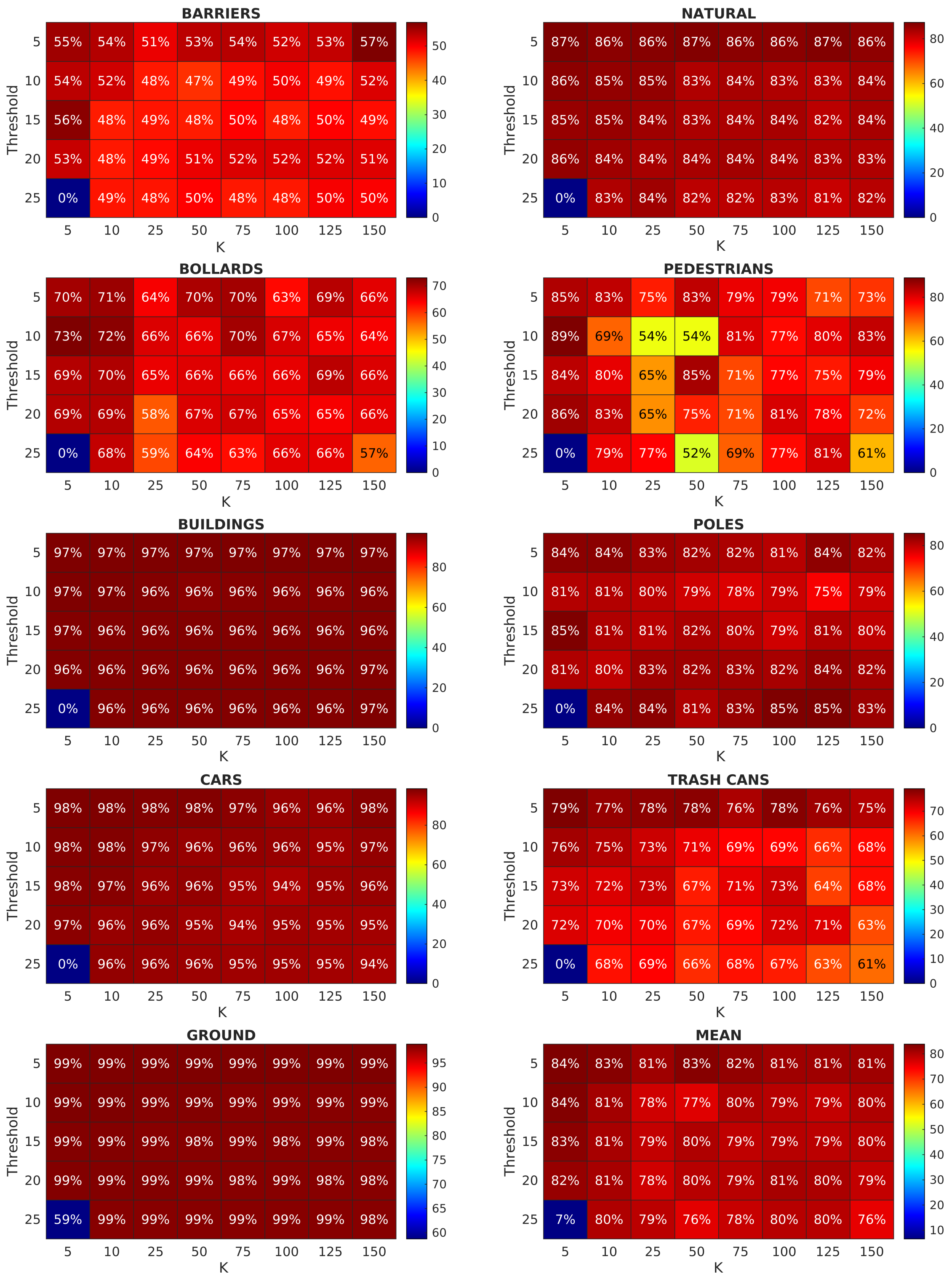


Figure B.6: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 28 cm) point cloud.

## B.3 Edge points with sub-sampling at 24 cm

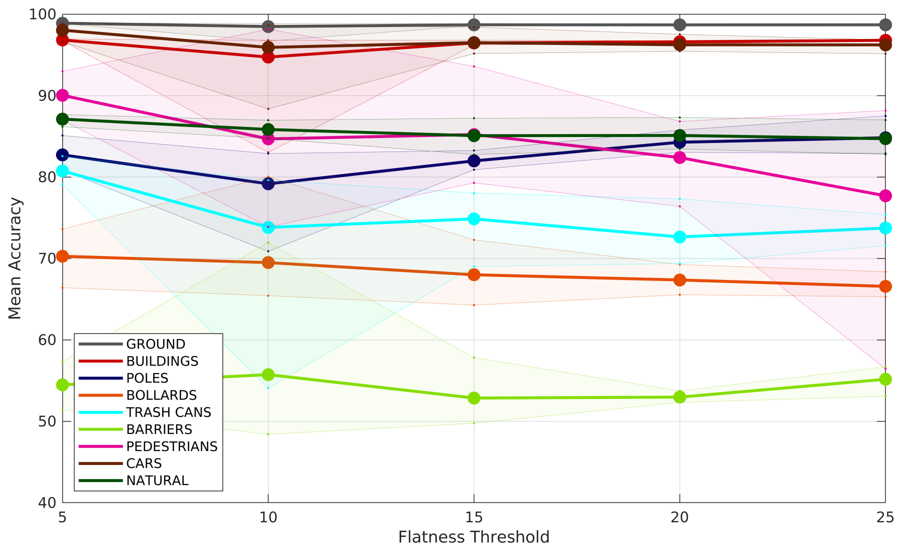


Figure B.7: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 24 cm.

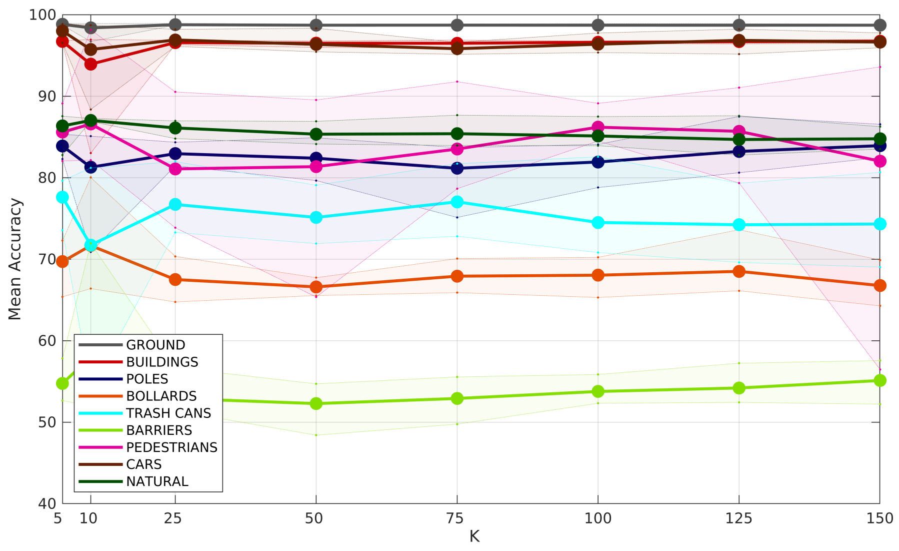
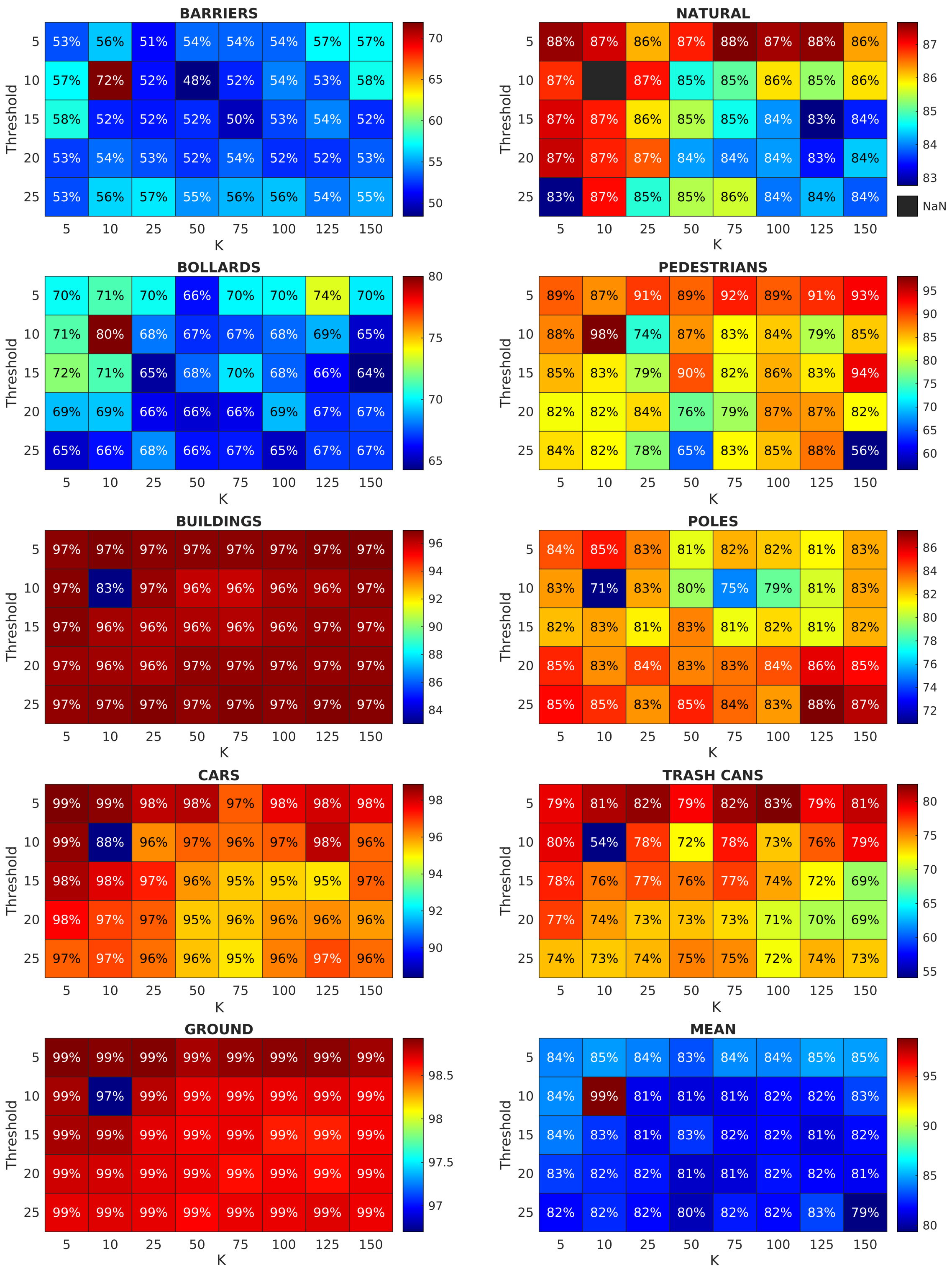


Figure B.8: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 24 cm.

Table B.3: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 24 cm.

| **Sub-sampling** | | | | **Number of points** | | | | | | | | **Accuracy** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **24** | **5** | **5** | 3149980 | 1814728 | 4377983 | 2803069 | 84.24 | 98.93 | 96.78 | 84.07 | 70.26 | 79.47 | 53.10 | 89.11 | 98.87 | 87.55 |
| **24** | **5** | **10** | 2340943 | 1427430 | 3431807 | 2142741 | 84.41 | 98.85 | 96.80 | 82.88 | 71.43 | 79.63 | 57.14 | 87.52 | 98.64 | 86.84 |
| **24** | **5** | **15** | 1716621 | 1073586 | 2613535 | 1571537 | 84.12 | 98.84 | 96.81 | 82.29 | 72.28 | 78.03 | 57.82 | 85.43 | 98.41 | 87.22 |
| **24** | **5** | **20** | 1287782 | 763669 | 1925352 | 1124362 | 82.90 | 98.75 | 96.52 | 84.83 | 69.14 | 77.32 | 52.65 | 82.05 | 97.53 | 87.31 |
| **24** | **5** | **25** | 1106693 | 583755 | 1532293 | 906560 | 81.76 | 98.70 | 96.67 | 85.33 | 65.38 | 73.55 | 53.07 | 83.82 | 96.54 | 82.79 |
| **24** | **10** | **5** | 3258277 | 1888802 | 4753156 | 2868469 | 84.71 | 98.91 | 96.94 | 85.09 | 71.32 | 81.20 | 55.94 | 86.99 | 98.71 | 87.26 |
| **24** | **10** | **10** | 2386165 | 1488375 | 3804222 | 2158722 | 82.45 | 98.84 | 96.70 | 83.01 | 70.87 | 80.03 | 54.08 | 71.95 | 98.15 | 88.37 |
| **24** | **10** | **15** | 1750533 | 1123845 | 2920009 | 1595317 | 82.77 | 98.84 | 96.44 | 82.74 | 71.34 | 75.63 | 51.97 | 83.17 | 97.85 | 86.94 |
| **24** | **10** | **20** | 1324917 | 817527 | 2164227 | 1172751 | 82.41 | 98.74 | 96.49 | 83.03 | 69.26 | 74.45 | 53.74 | 82.15 | 96.88 | 86.90 |
| **24** | **10** | **25** | 1116823 | 612625 | 1636718 | 927509 | 82.49 | 98.72 | 96.74 | 84.69 | 66.39 | 73.28 | 56.27 | 82.45 | 96.83 | 87.01 |
| **24** | **25** | **5** | 2827358 | 1689614 | 4674526 | 2466387 | 84.17 | 98.92 | 96.78 | 83.25 | 70.34 | 81.97 | 51.40 | 90.52 | 98.20 | 86.18 |
| **24** | **25** | **10** | 2003050 | 1255506 | 3673293 | 1761905 | 81.35 | 98.80 | 96.57 | 82.67 | 68.08 | 77.54 | 51.60 | 73.86 | 96.11 | 86.97 |
| **24** | **25** | **15** | 1526951 | 937805 | 2762573 | 1331631 | 81.37 | 98.71 | 96.32 | 81.45 | 64.75 | 76.95 | 51.72 | 79.28 | 97.24 | 85.94 |
| **24** | **25** | **20** | 1249933 | 719895 | 2061705 | 1062035 | 82.11 | 98.71 | 96.37 | 84.35 | 65.96 | 73.28 | 53.16 | 83.94 | 96.59 | 86.63 |
| **24** | **25** | **25** | 1113679 | 595580 | 1631255 | 915965 | 81.82 | 98.70 | 96.86 | 83.01 | 68.37 | 73.81 | 56.68 | 77.77 | 96.37 | 84.79 |
| **24** | **50** | **5** | 2439814 | 1447722 | 4433892 | 2088314 | 83.29 | 98.84 | 96.77 | 80.85 | 66.40 | 79.07 | 53.71 | 88.71 | 98.33 | 86.91 |
| **24** | **50** | **10** | 1832148 | 1077048 | 3464192 | 1547947 | 81.06 | 98.70 | 95.98 | 79.65 | 66.88 | 71.91 | 48.40 | 86.79 | 96.51 | 84.73 |
| **24** | **50** | **15** | 1488028 | 848170 | 2619200 | 1234842 | 82.76 | 98.67 | 96.26 | 83.25 | 67.73 | 75.75 | 52.19 | 89.53 | 95.99 | 85.46 |
| **24** | **50** | **20** | 1254873 | 688054 | 1979666 | 1031151 | 80.68 | 98.69 | 96.73 | 83.28 | 65.54 | 73.48 | 52.40 | 76.39 | 95.44 | 84.13 |
| **24** | **50** | **25** | 1120918 | 588374 | 1598786 | 911689 | 80.35 | 98.65 | 96.73 | 84.90 | 66.44 | 75.41 | 54.70 | 65.35 | 95.51 | 85.45 |
| **24** | **75** | **5** | 2307731 | 1330339 | 4289924 | 1953078 | 84.39 | 98.88 | 96.75 | 82.49 | 70.07 | 81.71 | 53.58 | 91.78 | 96.61 | 87.67 |
| **24** | **75** | **10** | 1810746 | 1017884 | 3372806 | 1485090 | 81.29 | 98.70 | 95.94 | 75.12 | 67.28 | 78.36 | 52.01 | 82.68 | 96.42 | 85.06 |
| **24** | **75** | **15** | 1504982 | 826365 | 2597400 | 1212811 | 81.62 | 98.69 | 96.22 | 80.96 | 69.76 | 77.25 | 49.76 | 81.82 | 95.45 | 84.68 |
| **24** | **75** | **20** | 1270414 | 683173 | 1972147 | 1026964 | 81.02 | 98.62 | 96.66 | 83.49 | 65.89 | 72.81 | 53.64 | 78.65 | 95.50 | 83.95 |
| **24** | **75** | **25** | 1126167 | 586777 | 1591436 | 911064 | 82.20 | 98.69 | 96.89 | 83.69 | 66.57 | 75.04 | 55.55 | 82.64 | 95.14 | 85.60 |
| **24** | **100** | **5** | 2259572 | 1274684 | 4197410 | 1896461 | 84.27 | 98.90 | 96.74 | 82.07 | 70.23 | 82.56 | 53.56 | 89.12 | 97.75 | 87.49 |
| **24** | **100** | **10** | 1812983 | 991553 | 3344514 | 1457833 | 81.82 | 98.69 | 96.44 | 78.79 | 67.82 | 73.34 | 54.27 | 84.50 | 96.60 | 85.96 |
| **24** | **100** | **15** | 1521925 | 817808 | 2608820 | 1204907 | 81.86 | 98.59 | 96.48 | 81.80 | 67.71 | 74.24 | 52.84 | 85.63 | 95.35 | 84.09 |
| **24** | **100** | **20** | 1282411 | 680517 | 1982806 | 1026042 | 82.07 | 98.67 | 96.70 | 84.08 | 69.11 | 70.81 | 52.32 | 86.82 | 95.97 | 84.12 |
| **24** | **100** | **25** | 1130224 | 586115 | 1591706 | 911086 | 81.79 | 98.69 | 96.74 | 82.89 | 65.30 | 71.56 | 55.85 | 84.90 | 96.22 | 83.92 |
| **24** | **125** | **5** | 2241038 | 1243573 | 4131625 | 1868272 | 84.88 | 98.90 | 96.86 | 81.33 | 73.61 | 79.32 | 57.23 | 91.05 | 98.00 | 87.59 |
| **24** | **125** | **10** | 1819600 | 977509 | 3338036 | 1443051 | 81.87 | 98.71 | 96.41 | 80.61 | 68.95 | 76.34 | 52.90 | 79.34 | 98.22 | 85.33 |
| **24** | **125** | **15** | 1535753 | 812858 | 2627645 | 1200839 | 81.08 | 98.58 | 96.67 | 80.89 | 66.12 | 71.96 | 54.33 | 83.27 | 95.16 | 82.77 |
| **24** | **125** | **20** | 1292876 | 679478 | 1997011 | 1025962 | 81.78 | 98.62 | 96.67 | 85.77 | 66.77 | 69.60 | 52.43 | 86.62 | 96.07 | 83.44 |
| **24** | **125** | **25** | 1132874 | 585885 | 1593027 | 911719 | 83.04 | 98.72 | 96.86 | 87.51 | 67.07 | 73.89 | 54.08 | 88.16 | 96.79 | 84.28 |
| **24** | **150** | **5** | 2233168 | 1223228 | 4087271 | 1851946 | 84.81 | 98.86 | 96.95 | 82.58 | 69.89 | 80.65 | 57.31 | 92.98 | 97.77 | 86.26 |
| **24** | **150** | **10** | 1827092 | 968529 | 3338289 | 1435474 | 83.04 | 98.68 | 96.69 | 82.70 | 65.43 | 79.27 | 57.56 | 84.60 | 96.50 | 85.96 |
| **24** | **150** | **15** | 1546914 | 809781 | 2646775 | 1198821 | 81.87 | 98.66 | 96.52 | 82.47 | 64.27 | 69.02 | 52.21 | 93.59 | 96.56 | 83.51 |
| **24** | **150** | **20** | 1300740 | 678751 | 2010509 | 1027047 | 81.49 | 98.68 | 96.72 | 85.34 | 67.17 | 69.37 | 53.39 | 82.48 | 95.94 | 84.34 |
| **24** | **150** | **25** | 1134803 | 585295 | 1595826 | 912749 | 79.35 | 98.68 | 96.80 | 86.55 | 67.05 | 73.26 | 55.08 | 56.47 | 96.43 | 83.79 |



**BARRIERS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 53% | 56% | 51% | 54% | 54% | 54% | 57% | 57% |
| 10 | 57% | 72% | 52% | 48% | 52% | 54% | 53% | 58% |
| 15 | 58% | 52% | 52% | 52% | 50% | 53% | 54% | 52% |
| 20 | 53% | 54% | 53% | 52% | 54% | 52% | 52% | 53% |
| 25 | 53% | 56% | 57% | 55% | 56% | 56% | 54% | 55% |

**NATURAL**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 88% | 87% | 86% | 87% | 88% | 87% | 88% | 86% |
| 10 | 87% | ■ | 87% | 85% | 85% | 86% | 85% | 86% |
| 15 | 87% | 87% | 86% | 85% | 85% | 84% | 83% | 84% |
| 20 | 87% | 87% | 87% | 84% | 84% | 84% | 83% | 84% |
| 25 | 83% | 87% | 85% | 85% | 86% | 84% | 84% | 84% |

**BOLLARDS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 70% | 71% | 70% | 66% | 70% | 70% | 74% | 70% |
| 10 | 71% | 80% | 68% | 67% | 67% | 68% | 69% | 65% |
| 15 | 72% | 71% | 65% | 68% | 70% | 68% | 66% | 64% |
| 20 | 69% | 69% | 66% | 66% | 66% | 69% | 67% | 67% |
| 25 | 65% | 66% | 68% | 66% | 67% | 65% | 67% | 67% |

**PEDESTRIANS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 89% | 87% | 91% | 89% | 92% | 89% | 91% | 93% |
| 10 | 88% | 98% | 74% | 87% | 83% | 84% | 79% | 85% |
| 15 | 85% | 83% | 79% | 90% | 82% | 86% | 83% | 94% |
| 20 | 82% | 82% | 84% | 76% | 79% | 87% | 87% | 82% |
| 25 | 84% | 82% | 78% | 65% | 83% | 85% | 88% | 56% |

**BUILDINGS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 97% | 97% | 97% | 97% | 97% | 97% | 97% | 97% |
| 10 | 97% | 83% | 97% | 96% | 96% | 96% | 96% | 97% |
| 15 | 97% | 96% | 96% | 96% | 96% | 96% | 97% | 97% |
| 20 | 97% | 96% | 96% | 97% | 97% | 97% | 97% | 97% |
| 25 | 97% | 97% | 97% | 97% | 97% | 97% | 97% | 97% |

**POLES**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 84% | 85% | 83% | 81% | 82% | 82% | 81% | 83% |
| 10 | 83% | 71% | 83% | 80% | 75% | 79% | 81% | 83% |
| 15 | 82% | 83% | 81% | 83% | 81% | 82% | 81% | 82% |
| 20 | 85% | 83% | 84% | 83% | 83% | 84% | 86% | 85% |
| 25 | 85% | 85% | 83% | 85% | 84% | 83% | 88% | 87% |

**CARS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 99% | 99% | 98% | 98% | 97% | 98% | 98% | 98% |
| 10 | 99% | 88% | 96% | 97% | 96% | 97% | 98% | 96% |
| 15 | 98% | 98% | 97% | 96% | 95% | 95% | 95% | 97% |
| 20 | 98% | 97% | 97% | 95% | 96% | 96% | 96% | 96% |
| 25 | 97% | 97% | 96% | 96% | 95% | 96% | 97% | 96% |

**TRASH CANS**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 79% | 81% | 82% | 79% | 82% | 83% | 79% | 81% |
| 10 | 80% | 54% | 78% | 72% | 78% | 73% | 76% | 79% |
| 15 | 78% | 76% | 77% | 76% | 77% | 74% | 72% | 69% |
| 20 | 77% | 74% | 73% | 73% | 73% | 71% | 70% | 69% |
| 25 | 74% | 73% | 74% | 75% | 75% | 72% | 74% | 73% |

**GROUND**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 99% | 99% | 99% | 99% | 99% | 99% | 99% | 99% |
| 10 | 99% | 97% | 99% | 99% | 99% | 99% | 99% | 99% |
| 15 | 99% | 99% | 99% | 99% | 99% | 99% | 99% | 99% |
| 20 | 99% | 99% | 99% | 99% | 99% | 99% | 99% | 99% |
| 25 | 99% | 99% | 99% | 99% | 99% | 99% | 99% | 99% |

**MEAN**

| Threshold | 5 | 10 | 25 | 50 | 75 | 100 | 125 | 150 |
|---|---|---|---|---|---|---|---|---|
| 5 | 84% | 85% | 84% | 83% | 84% | 84% | 85% | 85% |
| 10 | 84% | 99% | 81% | 81% | 81% | 82% | 82% | 83% |
| 15 | 84% | 83% | 81% | 83% | 82% | 82% | 81% | 82% |
| 20 | 83% | 82% | 82% | 81% | 81% | 82% | 82% | 81% |
| 25 | 82% | 82% | 82% | 80% | 82% | 82% | 83% | 79% |

Figure B.9: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 24 cm) point cloud.

## B.4 Edge points with sub-sampling at 20 cm

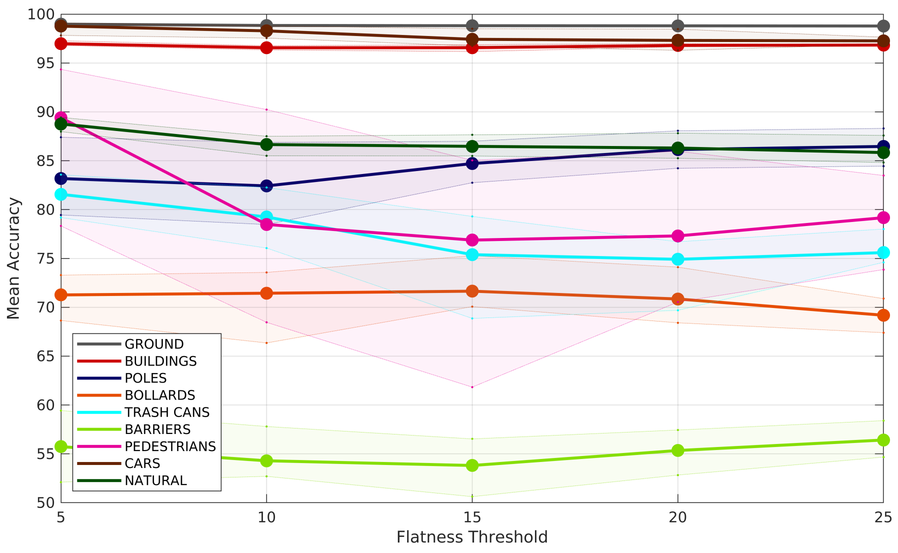


Figure B.10: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 20 cm.

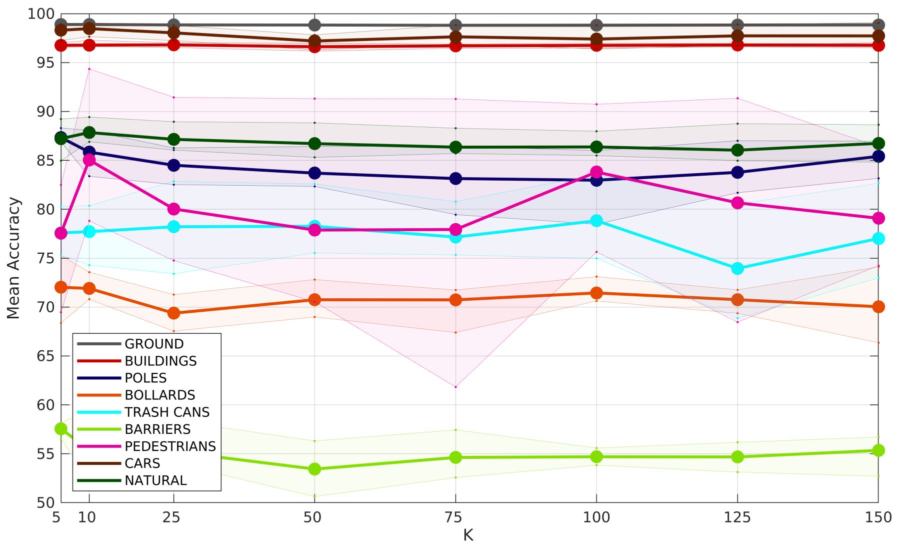


Figure B.11: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 20 cm.

Table B.4: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 20 cm.

| Sub-sampling | | | Number of points | | | | Accuracy | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **20** | **5** | **5** | 4073748 | 2410468 | 5737765 | 3711751 | 84.62 | 98.98 | 97.03 | 87.40 | 73.29 | 80.29 | 58.17 | 78.32 | 98.92 | 89.22 |
| **20** | **5** | **10** | 2975723 | 1860805 | 4458605 | 2775929 | 84.65 | 98.90 | 96.77 | 86.87 | 73.40 | 79.74 | 57.80 | 82.49 | 98.76 | 87.14 |
| **20** | **5** | **15** | 2188834 | 1383548 | 3375496 | 2023315 | 82.85 | 98.95 | 96.55 | 86.99 | 75.28 | 76.07 | 56.53 | 69.48 | 98.14 | 87.65 |
| **20** | **5** | **20** | 1695830 | 996087 | 2513725 | 1486536 | 83.22 | 98.86 | 96.66 | 87.10 | 69.78 | 76.57 | 57.14 | 77.25 | 98.47 | 87.16 |
| **20** | **5** | **25** | 1505130 | 794678 | 2068487 | 1250886 | 83.12 | 98.77 | 96.81 | 88.31 | 68.38 | 75.26 | 58.08 | 80.23 | 97.25 | 84.96 |
| **20** | **10** | **5** | 4242636 | 2524066 | 6257539 | 3829779 | 86.17 | 98.99 | 97.28 | 84.83 | 72.99 | 79.18 | 59.43 | 94.35 | 99.07 | 89.43 |
| **20** | **10** | **10** | 3062840 | 1955993 | 4977958 | 2835356 | 84.03 | 98.91 | 96.63 | 83.37 | 73.57 | 80.36 | 55.43 | 81.76 | 98.77 | 87.50 |
| **20** | **10** | **15** | 2254155 | 1463464 | 3802413 | 2084748 | 83.19 | 98.84 | 96.63 | 86.14 | 70.80 | 74.29 | 52.29 | 84.31 | 98.51 | 86.90 |
| **20** | **10** | **20** | 1743977 | 1071339 | 2828907 | 1554561 | 84.06 | 98.87 | 96.66 | 88.06 | 71.27 | 76.71 | 52.83 | 85.97 | 98.37 | 87.82 |
| **20** | **10** | **25** | 1515575 | 827118 | 2188261 | 1273603 | 83.61 | 98.85 | 96.74 | 86.82 | 70.89 | 78.01 | 57.10 | 78.81 | 97.65 | 87.59 |
| **20** | **25** | **5** | 3659236 | 2241885 | 6120672 | 3280151 | 85.18 | 98.99 | 97.03 | 83.11 | 71.27 | 82.86 | 54.17 | 91.45 | 98.74 | 88.95 |
| **20** | **25** | **10** | 2572837 | 1638668 | 4769310 | 2320283 | 82.69 | 98.86 | 96.73 | 82.52 | 69.60 | 80.21 | 55.31 | 74.75 | 98.81 | 87.43 |
| **20** | **25** | **15** | 1989844 | 1226643 | 3588514 | 1766420 | 82.61 | 98.76 | 96.54 | 84.83 | 70.07 | 79.30 | 54.02 | 75.13 | 97.68 | 87.12 |
| **20** | **25** | **20** | 1665703 | 957918 | 2709384 | 1434312 | 81.84 | 98.85 | 96.86 | 85.78 | 68.41 | 73.40 | 54.21 | 75.33 | 97.71 | 86.05 |
| **20** | **25** | **25** | 1513166 | 810879 | 2190422 | 1263208 | 83.35 | 98.80 | 96.90 | 86.27 | 67.54 | 75.29 | 58.40 | 83.49 | 97.25 | 86.19 |
| **20** | **50** | **5** | 3131365 | 1904259 | 5758587 | 2763435 | 84.71 | 98.96 | 96.80 | 82.34 | 71.65 | 82.59 | 52.11 | 91.31 | 97.83 | 88.84 |
| **20** | **50** | **10** | 2346708 | 1402340 | 4452528 | 2040332 | 82.33 | 98.84 | 96.34 | 82.75 | 72.81 | 79.90 | 52.70 | 73.38 | 97.66 | 86.62 |
| **20** | **50** | **15** | 1936507 | 1110393 | 3382509 | 1644186 | 82.16 | 98.82 | 96.17 | 82.75 | 70.35 | 76.88 | 50.62 | 80.27 | 97.05 | 86.55 |
| **20** | **50** | **20** | 1670905 | 917297 | 2600890 | 1396957 | 81.34 | 98.78 | 96.98 | 84.23 | 68.98 | 75.53 | 55.44 | 70.54 | 96.30 | 85.30 |
| **20** | **50** | **25** | 1521580 | 802686 | 2151206 | 1258743 | 82.44 | 98.79 | 96.81 | 86.42 | 69.96 | 76.40 | 56.31 | 73.86 | 97.19 | 86.25 |
| **20** | **75** | **5** | 2952055 | 1743191 | 5541908 | 2578047 | 84.42 | 98.95 | 96.88 | 79.45 | 71.75 | 80.78 | 53.59 | 91.28 | 98.84 | 88.29 |
| **20** | **75** | **10** | 2314075 | 1321594 | 4314405 | 1956154 | 81.90 | 98.79 | 96.46 | 80.56 | 71.44 | 76.50 | 53.43 | 75.26 | 98.46 | 86.23 |
| **20** | **75** | **15** | 1951657 | 1080113 | 3342544 | 1615225 | 80.39 | 98.78 | 96.50 | 82.82 | 71.72 | 76.61 | 52.57 | 61.83 | 96.90 | 85.77 |
| **20** | **75** | **20** | 1686035 | 909936 | 2585590 | 1391292 | 83.40 | 98.74 | 96.88 | 86.40 | 71.37 | 76.57 | 57.44 | 80.53 | 96.99 | 85.70 |
| **20** | **75** | **25** | 1526922 | 800513 | 2140833 | 1257966 | 82.71 | 98.79 | 96.82 | 86.46 | 67.40 | 75.33 | 56.08 | 80.76 | 96.98 | 85.76 |
| **20** | **100** | **5** | 2886316 | 1667092 | 5404119 | 2499442 | 84.97 | 98.97 | 96.97 | 81.31 | 70.82 | 83.57 | 55.54 | 90.74 | 98.82 | 87.98 |
| **20** | **100** | **10** | 2314219 | 1284855 | 4266996 | 1919385 | 84.10 | 98.80 | 96.48 | 78.47 | 73.11 | 82.26 | 53.83 | 90.24 | 97.55 | 86.15 |
| **20** | **100** | **15** | 1968953 | 1067897 | 3348420 | 1603714 | 83.56 | 98.77 | 96.70 | 84.63 | 71.55 | 78.12 | 54.36 | 85.04 | 97.36 | 85.49 |
| **20** | **100** | **20** | 1698220 | 905790 | 2594913 | 1389211 | 82.06 | 98.76 | 96.70 | 84.45 | 71.12 | 75.21 | 54.15 | 75.64 | 96.39 | 86.10 |
| **20** | **100** | **25** | 1531159 | 799368 | 2140305 | 1257993 | 82.59 | 98.76 | 96.92 | 86.00 | 70.62 | 74.98 | 55.59 | 77.38 | 96.90 | 86.14 |
| **20** | **125** | **5** | 2859779 | 1623411 | 5308133 | 2459449 | 84.89 | 98.98 | 96.90 | 82.74 | 69.71 | 80.54 | 56.17 | 91.35 | 98.90 | 88.75 |
| **20** | **125** | **10** | 2320665 | 1265531 | 4252269 | 1899783 | 81.10 | 98.82 | 96.60 | 81.69 | 71.24 | 76.06 | 53.13 | 68.46 | 98.37 | 85.50 |
| **20** | **125** | **15** | 1983204 | 1060811 | 3365579 | 1597346 | 82.19 | 98.82 | 96.60 | 82.99 | 71.70 | 68.86 | 53.76 | 84.50 | 96.75 | 85.74 |
| **20** | **125** | **20** | 1708733 | 904105 | 2608766 | 1388315 | 82.37 | 98.80 | 96.95 | 87.01 | 71.76 | 69.69 | 55.65 | 78.94 | 97.25 | 85.25 |
| **20** | **125** | **25** | 1533869 | 798751 | 2141311 | 1258546 | 82.35 | 98.77 | 96.92 | 84.45 | 69.36 | 74.60 | 54.67 | 80.04 | 97.40 | 84.97 |
| **20** | **150** | **5** | 2848428 | 1594349 | 5243910 | 2436060 | 84.70 | 99.01 | 96.94 | 84.24 | 68.65 | 82.66 | 56.71 | 86.40 | 99.08 | 88.64 |
| **20** | **150** | **10** | 2328564 | 1253266 | 4248340 | 1888621 | 82.51 | 98.83 | 96.50 | 83.16 | 66.35 | 78.93 | 52.70 | 81.50 | 97.94 | 86.67 |
| **20** | **150** | **15** | 1994870 | 1056232 | 3385486 | 1594199 | 82.37 | 98.81 | 96.88 | 86.55 | 71.72 | 72.98 | 56.33 | 74.46 | 97.04 | 86.55 |
| **20** | **150** | **20** | 1716772 | 902588 | 2622616 | 1389174 | 82.84 | 98.80 | 96.73 | 86.13 | 74.11 | 75.60 | 55.88 | 74.24 | 97.04 | 87.03 |
| **20** | **150** | **25** | 1535441 | 798022 | 2143867 | 1259415 | 82.55 | 98.77 | 96.77 | 86.96 | 69.35 | 74.90 | 55.09 | 78.79 | 97.54 | 84.82 |

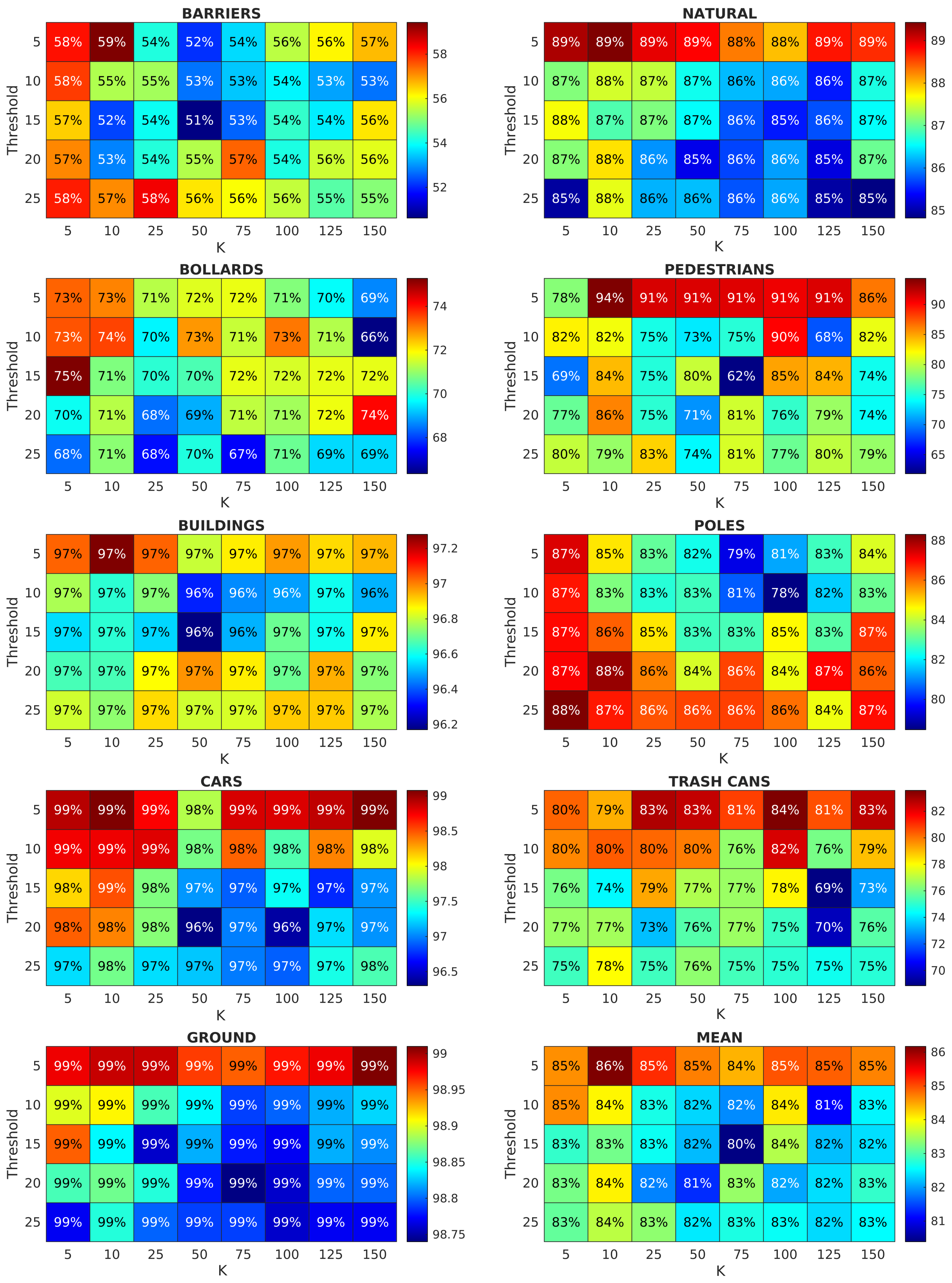


Figure B.12: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 20 cm) point cloud.

## B.5 Edge points with sub-sampling at 16 cm

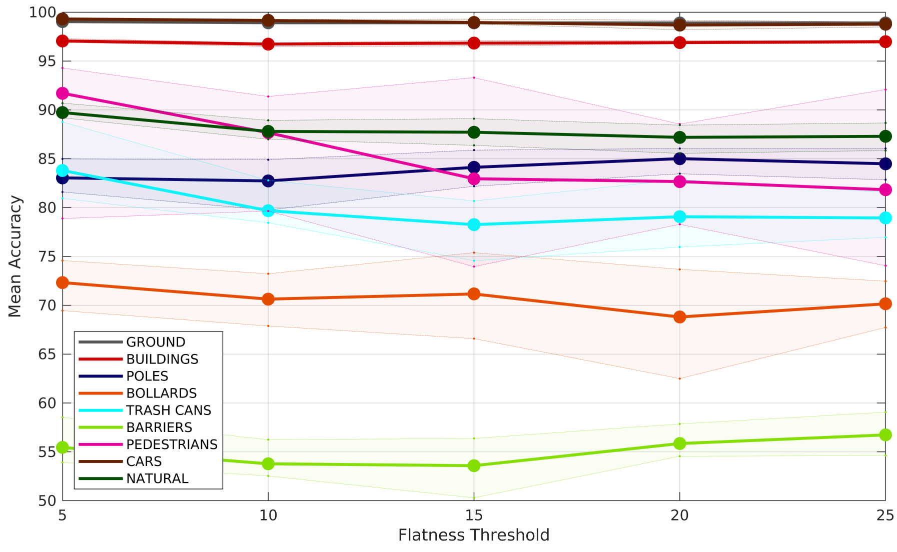


Figure B.13: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 16 cm.

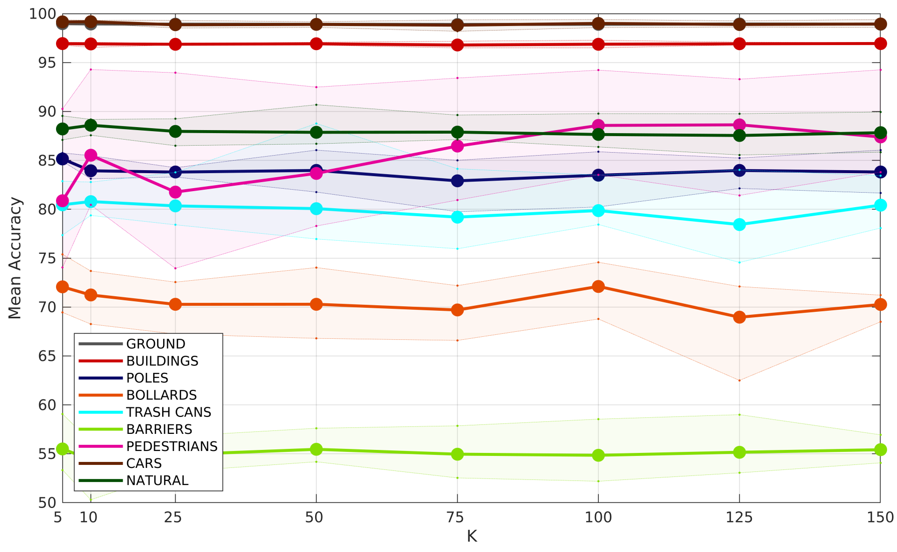


Figure B.14: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 16 cm.

Table B.5: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 16 cm.

| **Sub-sampling** | | | **Number of points** | | | | **Accuracy** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **16** | **5** | **5** | 5482180 | 3367421 | 7859291 | 5132515 | 83.78 | 99.08 | 96.96 | 84.98 | 69.45 | 81.82 | 53.92 | 78.88 | 99.36 | 89.55 |
| **16** | **5** | **10** | 3949893 | 2552201 | 6044226 | 3758961 | 84.68 | 99.01 | 96.78 | 84.90 | 71.03 | 78.71 | 53.33 | 90.26 | 99.18 | 88.93 |
| **16** | **5** | **15** | 2949428 | 1891201 | 4556955 | 2748834 | 84.07 | 98.97 | 96.95 | 85.03 | 75.39 | 77.36 | 56.37 | 79.37 | 99.25 | 87.94 |
| **16** | **5** | **20** | 2381410 | 1397671 | 3455941 | 2102531 | 84.60 | 98.94 | 96.94 | 85.76 | 73.68 | 82.87 | 54.80 | 81.78 | 99.17 | 87.50 |
| **16** | **5** | **25** | 2180972 | 1172701 | 2951107 | 1847455 | 83.60 | 98.89 | 97.07 | 85.05 | 70.79 | 81.56 | 59.06 | 74.05 | 98.81 | 87.10 |
| **16** | **10** | **5** | 5786287 | 3562405 | 8628661 | 5373502 | 85.69 | 99.00 | 97.02 | 83.18 | 73.69 | 80.94 | 54.54 | 94.29 | 99.33 | 89.19 |
| **16** | **10** | **10** | 4126797 | 2713865 | 6822300 | 3909822 | 85.37 | 98.97 | 96.83 | 84.07 | 73.23 | 82.78 | 56.25 | 89.40 | 99.25 | 87.57 |
| **16** | **10** | **15** | 3065259 | 2023154 | 5185781 | 2874845 | 83.33 | 98.94 | 96.55 | 83.12 | 70.84 | 80.68 | 50.30 | 81.14 | 99.29 | 89.10 |
| **16** | **10** | **20** | 2443286 | 1505245 | 3884578 | 2196841 | 84.06 | 98.92 | 97.08 | 85.48 | 70.15 | 80.17 | 54.97 | 82.35 | 98.99 | 88.44 |
| **16** | **10** | **25** | 2191712 | 1208804 | 3089225 | 1872385 | 83.50 | 98.90 | 97.06 | 83.81 | 68.26 | 79.37 | 55.97 | 80.45 | 99.01 | 88.66 |
| **16** | **25** | **5** | 4975606 | 3147500 | 8388177 | 4597976 | 85.84 | 98.99 | 96.91 | 83.30 | 72.55 | 83.77 | 54.52 | 93.96 | 99.31 | 89.25 |
| **16** | **25** | **10** | 3479258 | 2268890 | 6480227 | 3224801 | 83.18 | 98.96 | 96.81 | 83.80 | 67.88 | 80.55 | 53.18 | 79.65 | 98.90 | 88.86 |
| **16** | **25** | **15** | 2743513 | 1712411 | 4889527 | 2486199 | 82.80 | 98.90 | 96.83 | 83.50 | 72.03 | 79.34 | 54.13 | 73.96 | 98.89 | 87.61 |
| **16** | **25** | **20** | 2359530 | 1371143 | 3749526 | 2065695 | 83.30 | 98.91 | 96.86 | 84.25 | 67.24 | 79.65 | 56.79 | 79.73 | 98.69 | 87.61 |
| **16** | **25** | **25** | 2189763 | 1194389 | 3103228 | 1863094 | 83.62 | 98.87 | 96.96 | 84.17 | 71.69 | 78.41 | 55.92 | 81.52 | 98.53 | 86.50 |
| **16** | **50** | **5** | 4221698 | 2653568 | 7810907 | 3853200 | 86.33 | 99.05 | 96.92 | 81.76 | 74.04 | 88.77 | 54.24 | 92.49 | 99.02 | 90.69 |
| **16** | **50** | **10** | 3160898 | 1937585 | 5972432 | 2838533 | 83.94 | 98.87 | 96.79 | 83.86 | 68.93 | 79.02 | 54.18 | 87.54 | 99.18 | 87.09 |
| **16** | **50** | **15** | 2664248 | 1552646 | 4574879 | 2322315 | 83.40 | 98.87 | 97.08 | 84.72 | 70.99 | 77.43 | 55.45 | 79.67 | 98.79 | 87.60 |
| **16** | **50** | **20** | 2362562 | 1317342 | 3598874 | 2019040 | 82.61 | 98.88 | 96.80 | 83.46 | 66.80 | 78.15 | 55.79 | 78.29 | 98.60 | 86.69 |
| **16** | **50** | **25** | 2198744 | 1185142 | 3054542 | 1858268 | 83.75 | 98.89 | 97.01 | 86.05 | 70.68 | 76.96 | 57.61 | 80.36 | 98.94 | 87.27 |
| **16** | **75** | **5** | 3957703 | 2419686 | 7462569 | 3582850 | 86.11 | 99.01 | 97.19 | 83.58 | 72.19 | 84.14 | 56.52 | 93.42 | 99.36 | 89.63 |
| **16** | **75** | **10** | 3107502 | 1823323 | 5745470 | 2718474 | 83.69 | 98.84 | 96.50 | 79.75 | 68.60 | 78.67 | 52.53 | 91.35 | 99.21 | 87.79 |
| **16** | **75** | **15** | 2674229 | 1509651 | 4496619 | 2280579 | 82.80 | 98.89 | 96.62 | 82.18 | 66.59 | 76.78 | 53.13 | 84.53 | 98.86 | 87.67 |
| **16** | **75** | **20** | 2376597 | 1305557 | 3567781 | 2009496 | 83.57 | 98.88 | 96.88 | 85.01 | 70.05 | 75.96 | 57.86 | 82.05 | 98.20 | 87.22 |
| **16** | **75** | **25** | 2204404 | 1181937 | 3040994 | 1857022 | 83.61 | 98.87 | 96.79 | 84.02 | 71.08 | 80.45 | 54.75 | 80.94 | 98.49 | 87.13 |
| **16** | **100** | **5** | 3858430 | 2308782 | 7241989 | 3465760 | 86.44 | 99.04 | 97.29 | 81.60 | 74.58 | 83.50 | 58.54 | 94.23 | 99.40 | 89.78 |
| **16** | **100** | **10** | 3101389 | 1770041 | 5659819 | 2664829 | 83.69 | 98.88 | 96.50 | 80.24 | 72.89 | 78.44 | 53.24 | 86.75 | 99.26 | 86.99 |
| **16** | **100** | **15** | 2689637 | 1491837 | 4486691 | 2262770 | 84.04 | 98.87 | 96.86 | 85.88 | 71.84 | 79.31 | 52.19 | 86.27 | 98.81 | 86.37 |
| **16** | **100** | **20** | 2387831 | 1298995 | 3570662 | 2005162 | 83.69 | 98.86 | 96.75 | 84.83 | 68.78 | 79.24 | 55.64 | 83.53 | 98.58 | 87.02 |
| **16** | **100** | **25** | 2208440 | 1179881 | 3038615 | 1856531 | 85.09 | 98.92 | 96.99 | 84.85 | 72.47 | 78.85 | 54.62 | 92.08 | 98.98 | 88.07 |
| **16** | **125** | **5** | 3816346 | 2245143 | 7088710 | 3404445 | 85.69 | 99.05 | 97.09 | 82.13 | 71.81 | 84.02 | 55.99 | 92.07 | 99.27 | 89.77 |
| **16** | **125** | **10** | 3105924 | 1741347 | 5626020 | 2635938 | 84.63 | 98.91 | 96.81 | 83.56 | 72.10 | 79.28 | 53.19 | 91.37 | 99.15 | 87.29 |
| **16** | **125** | **15** | 2703294 | 1480673 | 4498265 | 2251956 | 84.14 | 98.92 | 96.80 | 83.83 | 70.67 | 74.56 | 53.05 | 93.30 | 98.80 | 87.35 |
| **16** | **125** | **20** | 2397344 | 1295269 | 3580882 | 2003363 | 82.65 | 98.93 | 96.89 | 85.24 | 62.50 | 76.50 | 54.55 | 85.02 | 98.64 | 85.55 |
| **16** | **125** | **25** | 2210736 | 1178543 | 3038433 | 1856916 | 83.70 | 98.90 | 97.00 | 85.07 | 67.73 | 77.84 | 59.00 | 81.42 | 98.63 | 87.76 |
| **16** | **150** | **5** | 3796390 | 2201827 | 6986498 | 3367740 | 85.84 | 99.04 | 97.05 | 83.80 | 70.27 | 83.44 | 55.33 | 94.26 | 99.40 | 89.94 |
| **16** | **150** | **10** | 3113681 | 1723316 | 5611508 | 2618811 | 83.79 | 98.89 | 96.86 | 81.66 | 70.40 | 80.03 | 54.32 | 84.99 | 99.09 | 87.84 |
| **16** | **150** | **15** | 2714565 | 1473022 | 4515681 | 2245910 | 84.27 | 98.93 | 96.95 | 84.70 | 70.92 | 80.57 | 54.07 | 85.37 | 98.87 | 88.06 |
| **16** | **150** | **20** | 2404587 | 1292577 | 3593578 | 2003124 | 84.89 | 98.87 | 96.88 | 86.05 | 71.22 | 79.98 | 56.40 | 88.55 | 98.59 | 87.47 |
| **16** | **150** | **25** | 2211733 | 1177422 | 3040120 | 1857727 | 83.41 | 98.92 | 97.01 | 82.85 | 68.49 | 78.08 | 56.94 | 83.80 | 98.77 | 85.84 |

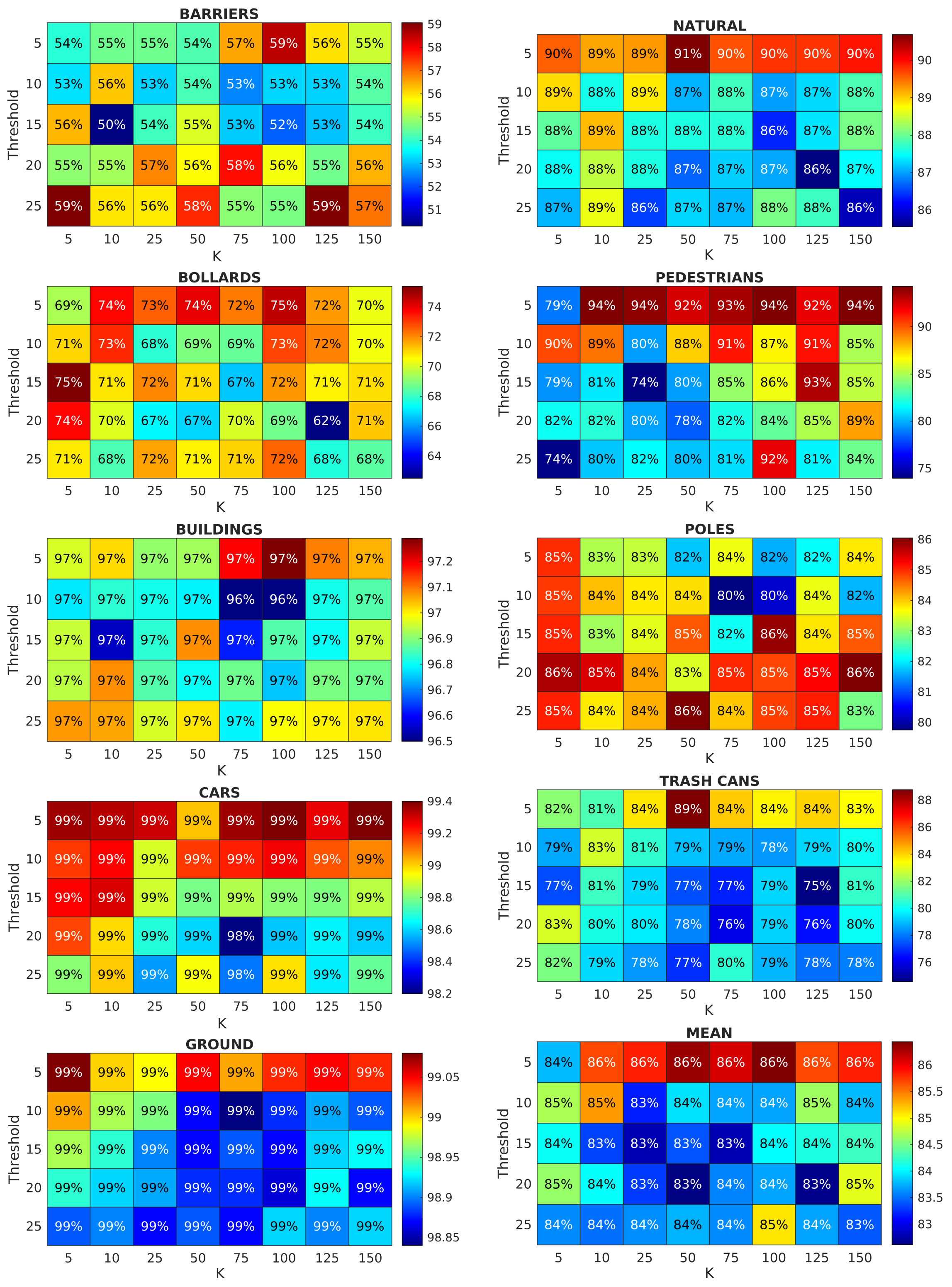


Figure B.15: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 16 cm) point cloud.

## B.6 Edge points with sub-sampling at 12 cm

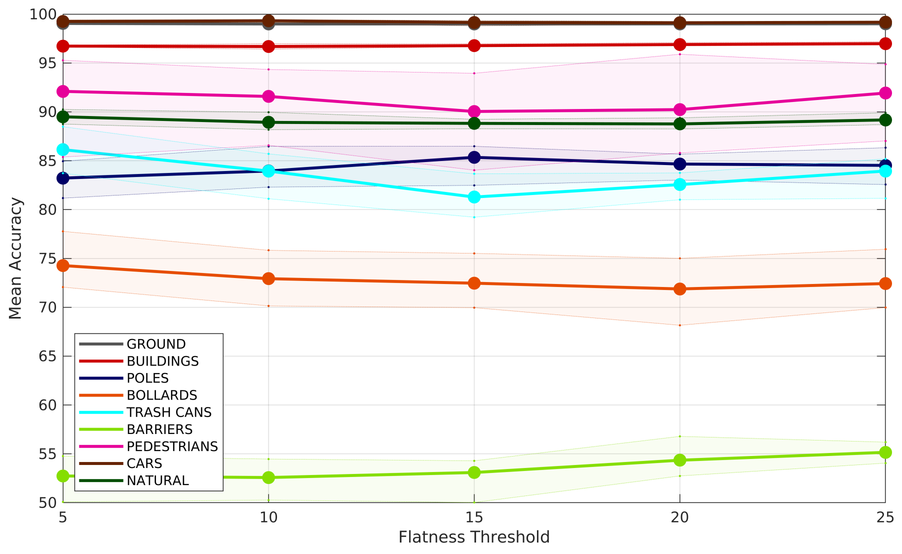


Figure B.16: Edge retained experimentation giving mean accuracy per classification, averaged by flatness threshold. Sub-sampled at a resolution of 12 cm.

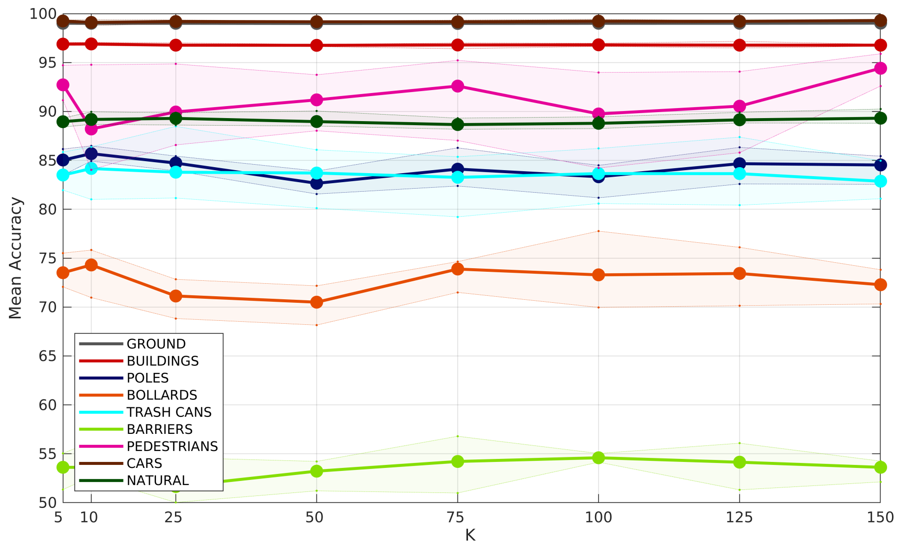


Figure B.17: Edge retained experimentation giving mean accuracy per classification, averaged by number of neighbours. Sub-sampled at a resolution of 12 cm.

Table B.6: Validation accuracy results for the edge retained set of experiments, sub-sampled at a resolution of 12 cm.

| Sub-sampling | | | | Number of points | | | | | | | | Accuracy | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resolution (cm)** | **KNN** | **Threshold** | **Lille1_1** | **Lille2** | **Paris** | **Lille1_2(V)** | **mean** | **ground** | **buildings** | **poles** | **bollards** | **trash_cans** | **barriers** | **pedestrians** | **cars** | **natural** |
| **12** | **5** | **5** | 7781781 | 5007828 | 11459541 | 7544356 | 85.82 | 99.07 | 96.80 | 84.04 | 72.06 | 85.85 | 51.35 | 94.73 | 99.12 | 89.33 |
| **12** | **5** | **10** | 5601402 | 3734860 | 8739827 | 5464031 | 85.90 | 98.98 | 96.84 | 85.80 | 73.70 | 82.77 | 54.47 | 92.22 | 99.31 | 89.04 |
| **12** | **5** | **15** | 4321433 | 2777882 | 6606572 | 4083905 | 85.83 | 99.02 | 96.86 | 86.16 | 75.52 | 81.96 | 53.28 | 91.16 | 99.35 | 89.15 |
| **12** | **5** | **20** | 3670633 | 2142153 | 5173471 | 3299593 | 85.98 | 99.03 | 97.09 | 85.67 | 73.66 | 83.75 | 53.94 | 93.05 | 99.24 | 88.41 |
| **12** | **5** | **25** | 3461184 | 1893035 | 4607355 | 3024783 | 85.64 | 98.98 | 96.84 | 83.49 | 72.62 | 83.15 | 55.05 | 92.49 | 99.23 | 88.87 |
| **12** | **10** | **5** | 8382185 | 5392881 | 12718472 | 8080480 | 85.49 | 99.02 | 96.75 | 84.95 | 75.30 | 86.38 | 53.16 | 85.38 | 99.14 | 89.33 |
| **12** | **10** | **10** | 5943898 | 4040300 | 10015669 | 5812228 | 86.39 | 99.03 | 96.97 | 86.47 | 75.83 | 85.70 | 53.28 | 90.88 | 99.38 | 89.96 |
| **12** | **10** | **15** | 4512121 | 3012444 | 7606848 | 4321464 | 85.01 | 99.00 | 96.82 | 86.48 | 74.43 | 83.66 | 52.62 | 84.03 | 98.79 | 89.24 |
| **12** | **10** | **20** | 3746910 | 2296183 | 5776813 | 3429161 | 84.90 | 99.00 | 96.89 | 85.61 | 75.01 | 81.01 | 52.74 | 86.03 | 99.11 | 88.69 |
| **12** | **10** | **25** | 3471919 | 1932790 | 4766034 | 3051463 | 86.10 | 99.02 | 97.06 | 84.93 | 70.97 | 84.08 | 56.20 | 94.77 | 99.12 | 88.70 |
| **12** | **25** | **5** | 7241475 | 4766387 | 12305975 | 6952192 | 85.75 | 99.05 | 96.64 | 84.66 | 72.79 | 88.48 | 50.08 | 90.97 | 99.24 | 89.87 |
| **12** | **25** | **10** | 5067678 | 3387496 | 9444815 | 4862765 | 84.74 | 99.00 | 96.67 | 84.46 | 72.84 | 83.98 | 50.27 | 86.58 | 99.36 | 89.47 |
| **12** | **25** | **15** | 4109544 | 2589396 | 7182330 | 3829484 | 84.70 | 98.99 | 96.69 | 85.44 | 71.24 | 81.70 | 50.02 | 90.34 | 99.28 | 88.61 |
| **12** | **25** | **20** | 3653773 | 2138532 | 5637069 | 3283037 | 84.52 | 99.00 | 96.88 | 83.93 | 68.83 | 83.62 | 53.30 | 87.01 | 98.98 | 89.16 |
| **12** | **25** | **25** | 3469598 | 1921346 | 4800121 | 3043078 | 85.59 | 99.04 | 97.02 | 85.13 | 69.97 | 81.15 | 54.61 | 94.86 | 99.17 | 89.35 |
| **12** | **50** | **5** | 6102501 | 3990832 | 11328729 | 5799142 | 85.52 | 99.04 | 96.63 | 81.56 | 72.17 | 86.08 | 51.21 | 93.75 | 99.19 | 90.05 |
| **12** | **50** | **10** | 4596210 | 2890921 | 8564060 | 4290590 | 84.95 | 99.01 | 96.81 | 82.30 | 71.57 | 85.37 | 53.71 | 88.03 | 99.27 | 88.51 |
| **12** | **50** | **15** | 3986673 | 2358642 | 6660989 | 3593835 | 84.60 | 99.00 | 96.66 | 82.48 | 70.45 | 80.12 | 52.85 | 91.81 | 99.20 | 88.80 |
| **12** | **50** | **20** | 3651048 | 2064941 | 5409047 | 3219894 | 84.75 | 98.99 | 96.86 | 83.03 | 68.16 | 81.77 | 54.22 | 91.85 | 99.11 | 88.73 |
| **12** | **50** | **25** | 3478306 | 1911466 | 4738573 | 3037251 | 85.28 | 98.95 | 96.87 | 83.91 | 70.15 | 85.19 | 54.12 | 90.48 | 99.10 | 88.72 |
| **12** | **75** | **5** | 5683036 | 3619918 | 10712657 | 5370280 | 85.73 | 99.01 | 96.73 | 82.52 | 74.13 | 83.72 | 53.11 | 94.28 | 99.34 | 88.75 |
| **12** | **75** | **10** | 4500158 | 2716236 | 8150667 | 4105155 | 85.39 | 98.97 | 96.42 | 82.38 | 74.55 | 85.37 | 50.99 | 92.51 | 99.15 | 88.18 |
| **12** | **75** | **15** | 3982548 | 2291738 | 6497448 | 3529878 | 85.41 | 99.00 | 96.96 | 86.28 | 71.50 | 79.22 | 54.08 | 93.95 | 99.26 | 88.44 |
| **12** | **75** | **20** | 3660944 | 2045553 | 5346748 | 3204234 | 86.39 | 99.00 | 96.88 | 84.49 | 74.65 | 82.84 | 56.79 | 95.23 | 99.03 | 88.60 |
| **12** | **75** | **25** | 3483664 | 1906469 | 4718524 | 3035289 | 85.80 | 99.03 | 97.01 | 84.87 | 74.60 | 85.17 | 56.10 | 87.03 | 99.07 | 89.32 |
| **12** | **100** | **5** | 5518843 | 3444507 | 10317412 | 5178491 | 86.19 | 99.06 | 96.84 | 81.17 | 77.77 | 86.22 | 54.77 | 91.01 | 99.37 | 89.45 |
| **12** | **100** | **10** | 4479419 | 2633296 | 7977976 | 4021988 | 85.80 | 98.98 | 96.65 | 84.09 | 72.33 | 83.66 | 54.46 | 93.99 | 99.43 | 88.64 |
| **12** | **100** | **15** | 3991991 | 2262915 | 6450336 | 3500165 | 84.02 | 99.00 | 96.72 | 84.32 | 69.96 | 80.59 | 54.14 | 84.22 | 98.96 | 88.27 |
| **12** | **100** | **20** | 3669352 | 2033891 | 5334631 | 3195912 | 85.15 | 98.97 | 96.86 | 84.49 | 73.67 | 83.44 | 54.55 | 87.00 | 99.15 | 88.25 |
| **12** | **100** | **25** | 3487067 | 1903046 | 4711715 | 3034224 | 85.76 | 99.05 | 96.98 | 82.56 | 72.73 | 84.35 | 55.03 | 92.54 | 99.24 | 89.34 |
| **12** | **125** | **5** | 5445198 | 3341695 | 10043203 | 5075001 | 86.15 | 99.03 | 96.66 | 82.59 | 76.11 | 87.38 | 54.01 | 91.39 | 99.26 | 88.93 |
| **12** | **125** | **10** | 4478020 | 2588043 | 7898895 | 3976751 | 85.14 | 98.97 | 96.49 | 83.54 | 70.14 | 83.57 | 51.31 | 94.08 | 99.30 | 88.82 |
| **12** | **125** | **15** | 4002043 | 2244309 | 6442049 | 3482210 | 85.72 | 99.02 | 96.74 | 86.18 | 74.25 | 80.42 | 54.29 | 92.20 | 99.22 | 89.15 |
| **12** | **125** | **20** | 3676378 | 2027051 | 5336910 | 3191043 | 84.78 | 99.02 | 96.85 | 84.67 | 70.72 | 82.96 | 54.99 | 85.79 | 99.10 | 88.90 |
| **12** | **125** | **25** | 3488581 | 1900691 | 4709485 | 3033869 | 86.31 | 99.03 | 97.17 | 86.33 | 75.94 | 83.88 | 56.08 | 89.25 | 99.23 | 89.91 |
| **12** | **150** | **5** | 5408946 | 3272309 | 9857327 | 5012297 | 86.46 | 99.12 | 96.80 | 84.31 | 73.82 | 84.96 | 54.20 | 95.28 | 99.41 | 90.25 |
| **12** | **150** | **10** | 4481838 | 2559992 | 7857072 | 3948468 | 85.17 | 99.02 | 96.67 | 82.55 | 72.46 | 81.11 | 52.12 | 94.34 | 99.44 | 88.80 |
| **12** | **150** | **15** | 4011332 | 2232849 | 6448499 | 3471105 | 85.59 | 98.99 | 96.75 | 85.43 | 72.39 | 82.59 | 53.45 | 92.60 | 99.15 | 88.92 |
| **12** | **150** | **20** | 3681961 | 2021888 | 5342911 | 3189119 | 85.71 | 99.01 | 96.80 | 85.40 | 70.32 | 81.08 | 54.25 | 95.90 | 99.26 | 89.39 |
| **12** | **150** | **25** | 3488643 | 1898668 | 4710258 | 3034349 | 86.04 | 99.01 | 96.88 | 84.96 | 72.45 | 84.59 | 54.05 | 93.99 | 99.26 | 89.19 |

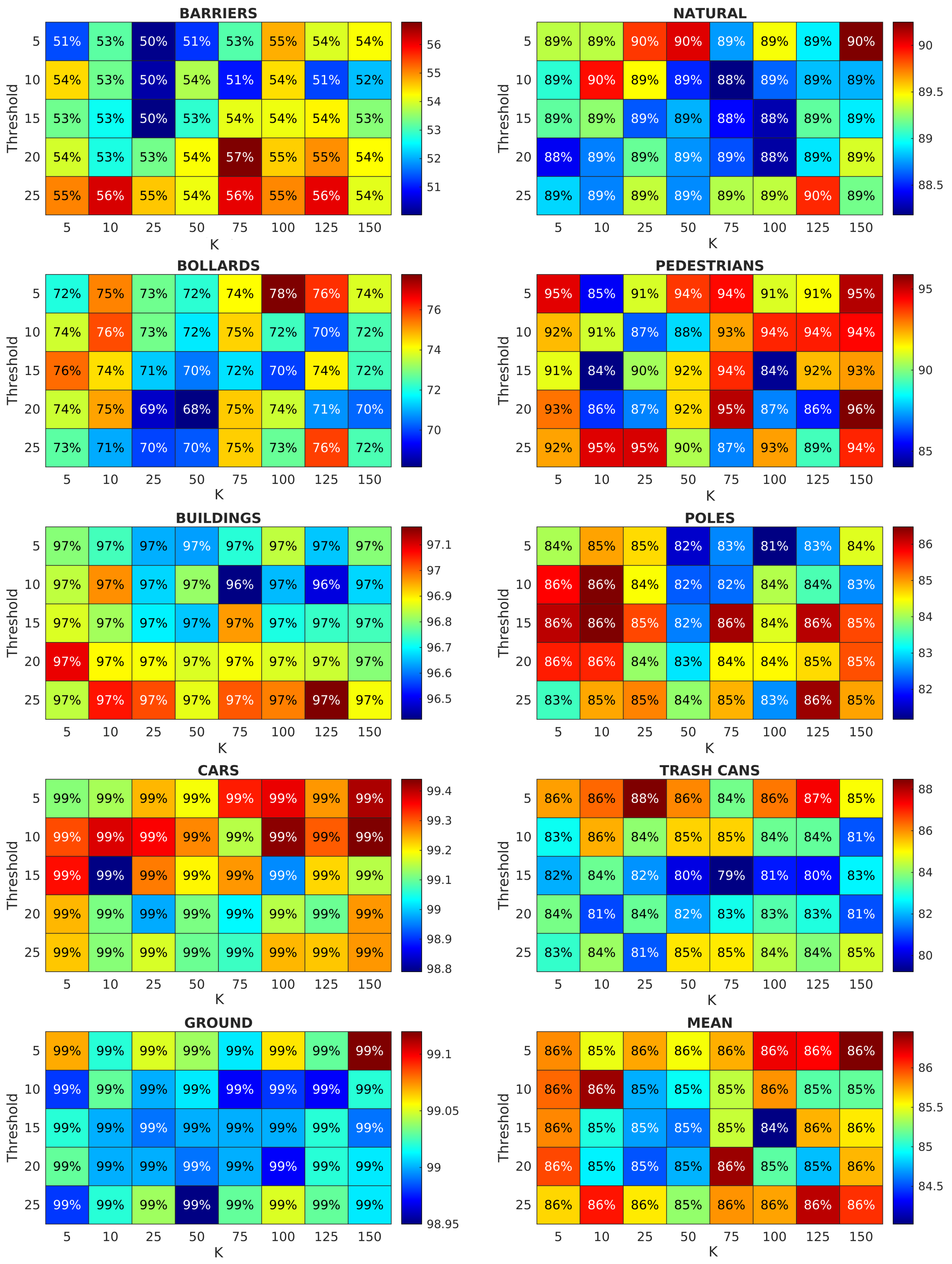


Figure B.18: Accuracy results over both KNN and threshold ranges for classification of retained edges combined with a grid sub-sampled (resolution 12 cm) point cloud.

# Appendix C

# Related Papers

## C.1 Accepted

### C.1.1 A system to automatically classify LIDAR for use within RF propagation modelling

Conference - IEEE Computer Society, Artificial Intelligence for Industries

Abstract - Many technologies and applications now necessitate an awareness of their geographical surroundings, typically employing an array of sensors to capture the environment. A key application is telecommunication network planning which benefits from the utilisation of RF propagation tools which incorporate representations of target environments typically sourced from high resolution aerial photography and/or LIDAR point clouds. However, the amount of data associated with LIDAR scanning can be very large, permutation invariant and clustered. Manually classifying this data, to maximise its utility in a propagation model, is not easily scaleable; being both labour intensive and time consuming. This paper describes a system which facilitates the automatic classification of point cloud data and its subsequent translation as wireframe meshes into a propagation model. Testing of automatically classified versus hand-labelled clutter results in comparable performance, with the average difference across all measurements of the automated approach outperforming hand-labelled data by circa 2.5 dB.

J. WORSEY , S. ARMOUR , I. HINDMARCH , AND D. BULL , A system to automatically classify LIDAR for use within RF propagation modelling, in 2021 IEEE Computer Society Artificial Intelligence for Industries, Sept. 2021

### C.1.2 Observations from using a portable LIDAR scanner to capture RF propagation modelling environments

Conference - IEEE 5G World Forum

Abstract - Automated assessment of a 3D geographical environment is highly desirable for many reasons including the creation of accurate environments for RF propagation modelling tools to assist in network planning, and for the rapid deployment of communication aids in areas affected by war or natural disasters. This paper presents some of the observations and limitations of using a portable Laser Imaging, Detection and Ranging (LIDAR) scanner to capture the data points. The point cloud was subsequently classified using a deep learning network before being translated into mesh based environments. Observations include issues with the capturing technique, where a single person can be captured multiple times; a lack of scanner range which permitted false free space line-of-site (LOS) propagation and unexpected classification issues which impacted radio wave propagation. By addressing these observations, cheap and portable LIDAR scanners can provide a viable technique to assist network planning.

J. WORSEY , S. ARMOUR , I. HINDMARCH , AND D. BULL , Observations from using a portable LIDAR scanner to capture RF propagation modelling environments, in 2021 IEEE 4th 5G World Forum (5GWF) (5G-WF'21), Montreal, Canada, Oct. 2021.