A SIMULATED POINT CLOUD IMPLEMENTATION OF A MACHINE LEARNING SEGMENTATION AND CLASSIFICATION ALGORITHM

by

Junzhe Shen

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science

Department of Computer Graphics Technology

West Lafayette, Indiana

May 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Hazar Nicholas Dib, Chair

Department of Computer Graphics Technology

Dr. Tim McGraw

Department of Computer Graphics Technology

Dr. Ayman Habib

Lyles School of Civil Engineering

Approved by:

Dr. Nicoletta Adamo

To Mom, who taught me how to learn, behave, and live. She saved my life with hers. May her soul rest in peace.

To Dad, who always supports and respects me. He told me that life is not only about study and work, but also love and joy. He is a strong man who carried me through the darkest time.

To Wanqiu, the love of my life, who I met and fell in love with magically. To the midsummer night dream we shall never wake up from.

To my families and friends, without whom my life would not have been filled with happiness, hope and love.

To my dog, Baobao, who was always barking joyfully and cheering me up no matter what. He always stood by my side, but I failed to do the same when he passed away. May the simplest soul rest in peace.

ACKNOWLEDGMENTS

I would like to pay my special regards to my academic advisor, Prof. Hazar Nicholas Dib, who guided and encouraged me during the whole two years of graduate program. He has always believed in me and taken good care of me. Without his persistent help, my studies and thesis would not have been finished.

I would like to pay my special regards to my committee member, Prof. Tim McGraw, who gives the best lecture ever and is always patient with his students. He taught me the way of conducting precise and trustworthy research.

I would like to pay my special regards to my committee member, Prof. Ayman Habib, who is always strict but also nice to his students. It is whole-heartedly appreciated that your great advice for my study.

I wish to express my deepest gratitude to my classmates and colleagues, Xingyu Lei, Yuanpei Zhao, Varisht Raheja, Zongcheng Chu, and Nicholas Joseph Villani, who helped me through my studies and thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

As buildings have almost come to a saturation point in most developed countries, the management and maintenance of existing buildings have become the major problem of the field. Building Information Modeling (BIM) is the underlying technology to solve this problem. It is a 3D semantic representation of building construction and facilities that contributes to not only the design phase but also the construction and maintenance phases, such as life-cycle management and building energy performance measurement. This study aims at the processes of creating as-built BIM models, which are constructed after the design phase. Point cloud, a set of points in 3D space, is an intermediate product of as-built BIM models that is often acquired by 3D laser scanning and photogrammetry. A raw point cloud typically requires further procedures, e.g. registration, segmentation, classification, etc. In terms of segmentation and classification, machine learning methodologies are trending due to the enhanced speed of computation. However, supervised machine learning methodologies require labelling the training point clouds in advance, which is time-consuming and often leads to inevitable errors. And due to the complexity and uncertainty of real-world environments, the attributes of one point vary from the attributes of others. These situations make it difficult to analyze how one single attribute contributes to the result of segmentation and classification. This study developed a method of producing point clouds from a fast-generating 3D virtual indoor environment using procedural modeling. This research focused on two attributes of simulated point clouds, point density and the level of random errors. According to Silverman (1986), point density is associated with the point features around each output raster cell. The number of points within a neighborhood divided the area of the neighborhood is the point density. However, in this study, there was a little different. The point density was defined as the number of points on a surface divided by the surface area. And the unit is points per square meters (pts/m^2). This research compared the performances of a machine learning segmentation and classification algorithm on ten different point cloud datasets. The mean loss and accuracy of segmentation and classification were analyzed and evaluated to show how the point density and level of random errors affect the performance of the segmentation and classification models. Moreover, the real-world point cloud data were used as additional data to evaluate the applicability of produced models.

1. INTRODUCTION

Building Information Modeling (BIM) is developing at a high speed. Not only does it benefit building design and construction, but also life-cycle management, building energy performance measurement, and historic architecture maintenance and renovation. As Eastman et.al. (2011) and Pătrăucean et.al. (2015) described, a BIM model is a virtual representation of building constructions and facilities. It records all information during products' life cycle which includes digital designs, schedules, material, costs, and etc. In the case of historic and old buildings where blueprints and technologies such as CAD (Computer Aided Design) are non-existent, generating a BIM model for these buildings is not a straightforward process. According to a study in UK (Valero et.al., 2018), due to rapid climate change, historic reservations have the tendency to erode at an unwanted higher speed. Therefore, acquiring virtual representations is especially required for this kind of situation. 3D reconstruction is a general way of transferring real-world objects into virtual environments where they can be preserved. It is often used in the construction and maintenance phases for monitoring and progress surveillance. In general, 3D laser scanning and photogrammetry present a possible solution to this problem. “Photogrammetry is the technology of deriving 3D data from 2D images by mono-plotting (single-ray back projection), by stereo-imagery interpretation or by multi-image block adjustment” (Zhu, 2015, p.1). Despite its robustness and generalization, photogrammetry requires more data processing which takes a rather long time (Barazzetti, 2018). “Due to its accuracy, laser scanning is becoming an increasingly applied data acquisition method” (Rebolj et.al., 2017, p.324). However, the data acquisition method depends on the situation. A Light Detection and Ranging (LiDAR) method integrated with photogrammetry often brings better results in terms of accuracy and precision. Laser scanning (LS) is a unit that collect 3D point locations by shooting beams in certain directions. The location is calculated and recorded according to the beam angles and time difference between the emitted light and the reflected light. “LS is also referred to as LIDAR because it uses a laser to illuminate Earth's surface and a photodiode to register the backscatter radiation” (Zhu, 2015, p.1). The problem remains that laser scanners are still very expensive and “it is unlikely that they will become significantly more affordable in the foreseeable future” (Bechtold & Höfle, 2016, p.161).

LiDAR can be categorized by what equipment it is mounted on, e.g., tripod, ground vehicle, airplane, satellite. Airborne LiDAR has been frequently used in remote sensing and surveying. In terms of indoor environment measurement, especially Mechanical, Electrical & Plumbing (MEP) system inspection and configuration, stationary 3D scanners are the most common. Compared to LiDAR, photogrammetry has not been used that much for indoor implementations. The static terrestrial laser scanner is typically used for, e.g., industrial sites modeling, crack detection, site monitoring, historic cultural heritage documentation, and 3D city modeling. For laser scanning, two prevalent scanning techniques are referred to as "Scan-to-BIM" and "Scan-vs-BIM" (Bueno et.al., 2018). Scan-vs-BIM is the technology that compares the as-built environment and designed BIM models. Scan-to-BIM is known for using 3D scans to generate BIM models from existing buildings and facilities. The main implementations include controlling construction progress, quality, and life-cycle management (Bueno et.al., 2018). There are two kinds of BIM technologies, as-built BIM and as-designed BIM. The BIM-created in the design stage of a facility is called as-designed BIM, and the BIM that reflects a facility in its as-built condition is called as-built BIM (Pătrăucean et.al., 2015; Bosche et.al., 2015; Dore et.al., 2014). As-built BIMs are often generated when the facilities are built differently from the design (Pătrăucean et.al., 2015). Operation and maintenance of existing buildings, quality control, defect detection, and others are efficient benefits as-built BIM has brought to the industry (Carbonari et.al., 2014; Liu et.al., 2015). The technology of converting real-world objects to 3D models in a virtual environment not only benefits BIM technology but also the entertainment fields such as animation, film, and gaming. And it has been gaining popularity in industrial, consumer, healthcare, education, and governmental applications (Zhu, 2015).

Nevertheless, creating as-built BIM models is a difficult and time-consuming non-straightforward procedure. “As-built BIM creation of building interiors using scanned point clouds incurs critical difficulties: the complex design of indoor structures, not to mention obstacles, necessitates time-consuming manual operation and resultantly huge data sizes, which often leads to system slow-down or failure” (Jung et.al., 2014, p.68). As-built BIM can be error-prone due to anomalies, destructions, and mistakes made by workers during both constructions and after (Barazzetti, 2016). They are often the final products of point cloud processing. The process starts with data acquisition, which is the acquiring of point cloud data by LiDAR and photogrammetry. Before further processing, multiple scans of overlapping areas need registration. During registration, points are shifted and rotated according to the defined mapping coordinate. Automatic registration of point clouds remains a challenge due to the incompletion and similarity of buildings and other structures (Bueno et.al., 2018). After registration, point cloud data are further processed by procedures such as down sampling and noise removal. A point cloud sometimes contains millions of points, so it requires huge storage and heavy computation to display and process. According to the requirement of the project, the point clouds are down sampled to a smaller size to save storage and computation. For example, a point cloud with a point density of 10000 pts/m^2 can be down sampled to a point cloud with a point density of 500 pts/m^2 so that the new file size is approximately 1/20 of the original file size. Noises are the points introduced by unexpected and unwanted materials and objects in the scanned scene, e.g. mirror, glass, moving objects, etc. Noise removal partially eliminates points that do not contribute to the shape of the objects. The next processing step is the segmentation and classification, which is the most crucial step for semantic modeling. During the segmentation, points are grouped into different segments such as different planes and objects. During the classification, points are labelled according to rules. For instance, points can be labelled as the floor, the ceiling, the wall, the window, the door, and etc. In semantic modeling, models are associated with additional information other than geometric information. Segmentation and classification are interesting areas of point cloud processing and they are often associated with each other. Although the field has a trend of full automation, BIM engineers still need manual field measurement to ensure quality (Jung et.al., 2014). Machine learning segmentation and classification methods are developing fast with the progress of computation power and various algorithms. As a subset of artificial intelligence, machine learning algorithms use implicit rules and build mathematical models based on the training data. The training data are specified by users according to how they want the product of algorithms to be. In the training phase, the labelling of points is required for the algorithms to learn from. However, the labelling has been done manually and semi-automatically with human intervening. This is time-consuming and it is nearly impossible for human beings to avoid making mistakes. Although there have been a few automatic labelling methods, they can be error prone. Moreover, the clusters are decided and fixed after the labelling is done. Therefore, if operators need to add or change the clusters due to the project requirements, the labelling process needs to be revised again. Since the algorithm operates based on the implicit rules of the training data, the performance of the algorithm depends on the quality and quantity of the training data. A lot of researchers have addressed the importance of point cloud quality (Biswas et.al., 2015; Rebolj et.al., 2017; Zhang et.al., 2018). Nevertheless, due to the complexity and uncertainty of real-world environments, it is difficult to decouple the influence of each attributes of points, e.g., point density, level of random error, point cloud size, point distribution, etc.

The objective of this study focused on the segmentation and classification processes of as-built model generation. As-built models enable management and maintenance of construction and facilities. Moreover, it only requires the operator to do on-site data collection and engineers can access and analyze the data in the office. With the help of automation and enhanced computation power, as-built model becomes a cost-efficient and time-saving technology. Segmentation and classification are the most essential steps of as-built model generation because they are the fundaments of semantic modeling. Automatic segmentation and classification accelerate the whole process, but the result of as-built model depends on the accuracy of the classification. Therefore, it is important to develop segmentation and classification methodologies with higher accuracy.

In the following sections, the problem and the purpose of this study are explained in further detail.

1.1 Problem Statement

Point cloud processing, including data acquisition, registration, configuration, segmentation, classification, reconstruction, is time-consuming and expensive (Dimitrov & Golparvar-Fard, 2015). Inevitable errors exist within these tasks because of necessary human interference. The need for fast, low-cost, and reliable point cloud processing is in high demand. At fast changing construction sites, manual methods often fail to keep up with the requirements for monitoring and management (Dimitrov & Golparvar-Fard, 2015). Automation accelerates the process and helps users to match the required speed. Machine learning is one of the most dominant approaches in point cloud processing automation. However, the performance of a machine learning algorithm depends highly on the quality and quantity of the training data. Researchers have concluded that point cloud data quality is more important than the quantity (Weinmann et.al., 2015). The optimization of point cloud data can play a significant role in providing good segmentation and classification results. Segmentation and classification are crucial steps of semantic modeling, which is the most iconic feature in BIM. Although there have been a few existing applications that help with segmentation, e.g. Autodesk Recap Pro, the process of point cloud, especially point cloud segmentation and classification, is still mainly semi-auto with human interference.

Machine learning segmentation and classification methods are developing fast and have become the most common use of segmentation and classification methods in the field. Generally, machine learning methods are divided into two categories, supervised and unsupervised learning methods. "Supervised methods are the majority with a training phase mandatory and fundamental to guide the successive machine learning classification solution” (Grilli et.al., 2018, p. 339). Supervised learning relies on a set of provided training examples for the machine to learn how to deliver a correct result. Therefore, the result of the segmentation and classification done by algorithms associates with the provided human-labelled learning examples. The results are influenced by the quality of the training data and the density of the point cloud data (Grilli et.al., 2018). In researches like this, the semantic results are highly dependent on the abstract model trained using grammars on a 3D reconstructed model. Unfortunately, semi-auto and manual labelling by human operators often are associated with inevitable errors, which lead to vital consequences for the outcome of machine learning methods. Despite software helping the segmentation and classification with some level of automation, the task of labelling every point correctly in the dataset is rather challenging due to the variety of object types and spatial permutation of objects (Yousefhussien et.al., 2018). Different machine learning algorithms define clusters differently according to their own interests. Clusters are defined before the segmentation and classification process, which means once the model is trained, the clusters are final. If any projects require specific labelling other than the original ones, the labelling of training dataset needs to be revised. This repeating task is time-consuming but can be assisted by automatic labelling.

To study how point cloud attributes affect the performance of the segmentation and classification models, one must provide different sets of point cloud data with specially designed parameters. Real-world point cloud data acquisition takes a rather long time and it is difficult to control the scanning environment, hence it is difficult to develop control and experiment groups with respect to different attributes. For instance, the point density is not uniform through the scene due to the operation principles of LiDAR and photogrammetry. In LiDAR, the point density is affected by the distance from laser scanner to the hit point. The closer the scanned area is to the laser unit, the denser the point cloud is. In photogrammetry, feature extraction identifies the object's point in multiple images for 3D reconstruction. First of all, a point can be reconstructed if it appears in two different images. Secondly, a point can be reconstructed if it is extracted successfully in more than one images. Therefore, the point density depends on the distribution of extracted features that are visible in more than one data acquisition location. All these factors that affect the point density make it difficult to analyze its influence on the performance of the machine learning models.

Random errors are inevitable and associate with system measurements. In LiDAR, random errors of the point cloud are associated with the range error of the laser scanner, e.g., a FARO Focus 3D X 330 has a range error of ±2 mm. In photogrammetry, random errors are often caused by different distortion parameters along different images of the same objects, e.g. radial lens distortion, de-centering lens distortion, atmospheric refraction, etc. Therefore, the level of random errors changes through the whole point cloud, which makes it difficult to analyze its effect on the performance of the machine learning models. And in the real world, it is impossible to avoid random errors, hence the errors will always affect the segmentation and classification. However, it is simple to control the positional error in virtual simulation, which might help the related processing for real-world data.

1.2 Purpose Statement

Creating as-built models from point clouds brings benefits in the industrial business. Firstly, the number of buildings in developed cities have already reached their saturation. Constructing new buildings and facilities is not as important as reconstructing existing, problematic buildings. As-built models help engineers and architects to easily identify and diagnose problems, but the procedure of acquiring the as-built BIM models of existing buildings is not proficient and straightforward. Semantic information is the most important attribute of as-built models. The semantic process is done by segmentation and classification of point clouds. The result of machine learning segmentation and classification highly relies on the training dataset. Therefore, it is important to find the relation between the attributes of point cloud and the results of segmentation and classification.

The purpose of this project is to find out how point density and the level of error in simulated point cloud affects the machine learning segmentation and classification results. Currently, the main use of simulate point cloud data is the test of segmentation and classification methodologies. For instance, Lari (2014) introduced simulated laser scanning data to evaluate the proposed segmentation approach and managed to provide a more accurate model with higher speed. This project explored the potential in simulated data for enhancing segmentation and classification accuracy and robustness for future implementation of real-world data. Moreover, the study revealed the relation between point cloud attributes and the machine learning segmentation and classification results by designing groups of data with different point density and with or without a level of error. The point density is non-uniform in a point cloud and depends on various aspects, e.g., the distribution of extracted features and hit points. This variation of point density in point clouds makes it difficult to analyze its effect on the segmentation and classification. However, this study proposed a point cloud simulation method called scattering. The points were simulated evenly and randomly with a same point density throughout the entire point cloud. This method decoupled point density from the point density variation which can be difficult for real-world scenarios and other simulation methods. The distribution of random error changes through the whole point cloud. This variation makes it difficult to analyze its effect on the segmentation and classification results. In real-world situations, the errors always affect the segmentation and classification. The proposed simulation method controlled the level of random errors, and it was evenly and randomly distributed throughout the whole point cloud. By doing this, the proposed method decoupled random errors from the random error variation, which is difficult for real-world scenarios. The level of error is described as ϵ which was defined as the average of the random error in the point cloud. Previous works in the field have not made their preparation of data rather transparent, instead, they intended to only show the best results. However, this study showed the direct relation between point cloud training data and segmentation results, which provided references for future works.

The significance of this research can be described in four aspects. Firstly, virtual simulations bring better accuracy and efficiency to the labelling part of point cloud processing. Efficiency is the key requirement in the monitoring and management of fast changing construction sites (Dimitrov & Golparvar-Fard, 2015). Generally, the first step of a machine learning segmentation is the labelling of the training point cloud datasets and the datasets for testing. Currently, this process is mostly semi-auto with some interference of human operators. This leads to all sorts of problems, e.g., wrongly labelled points, missing points, etc. However, the proposed method used a virtual computer-aided labelling method. A model was initially assigned with a label. Then the point cloud generated from the model was assigned with the same label automatically. Therefore, when a point cloud was generated, it was already a labelled point cloud data, which was directly used as the training and testing data for the segmentation and classification algorithm. Preparation for training and testing data became easier with the proposed simulation method. Therefore, it brought efficiency and robustness to multiple aspects of point clouds processing. Secondly, the proposed machine learning segmentation and classification models have the potential to be applied on real-world projects. The point cloud simulations were done as realistic as possible, only some aspects were configured in order to get better performance. The data structure of simulated point clouds was similar to that of real-world point clouds. Therefore, the proposed method can be applied on real-world point clouds, and vice versa. Thirdly, the proposed method decoupled point density from the point density variation, and random error from the random error variation. Although this is unlikely to be done in the near future for real-world projects, it is still important to evaluate how each attribute contributes to the results of the point cloud processing. Fourthly, LiDAR and photogrammetry are both qualified approaches to collect point cloud data. Photogrammetry is a transparent model, which means users have access to each step from original imagery to the final 3D reconstructed model. However, LiDAR is a non-transparent model. All the measuring and processing units are hidden in a black box. Users can only access the 3D coordinates and other output from the unit without knowing if there is something wrong with the data acquisition phase. This makes the quality control and quality assurance vital important for LiDAR system. In terms of acquisition, LiDAR directly records the 3D coordinates of points and the procedure can be done during the day or the night. Photogrammetry needs further computation from raw collection, which is complicated, and sometimes the matching procedure can be unreliable. Photogrammetry is a passive system, which means it requires sunlight for data acquisition, so it can only be done during the daytime. In terms of accuracy in the product, LiDAR provides better vertical accuracy and photogrammetry provides better planimetric accuracy. Moreover, photogrammetry provides higher redundancy so that users can check and determine how good the reconstruction is, while there is no inherent redundancy in LiDAR making it hard to check the reconstruction quality.

Photogrammetry provides semantic information based on 2D feature extraction on the original imagery, but it is difficult for LiDAR to derive semantic information because it only records positional and intensity data of points. Researchers have divided into two categories according to different data acquisition methods. The findings of LiDAR cannot be fully applied to photogrammetry, and the findings of photogrammetry cannot be fully applied to LiDAR. In this study, the point clouds were simulated using scattering. Points were randomly placed on the objects' surfaces to imitate the outcome of down sampling and other point cloud configurations. Therefore, the study released a more general finding, not specific to either of the two data acquisition methods.

By comparing the testing results of each point cloud datasets, including the mean loss and accuracy, this study revealed the direct relation between point density and the classification performance, and between the level of error and the classification performance.

1.3 Research Questions

This study focused on computer-simulated point cloud data. The research questions regarding how the point attributes affect the training and the outcome of the segmentation and classification method are listed as follows.

1. How does the level of random errors of simulated point clouds affect the performance of a machine learning segmentation and classification model in terms of the overall accuracy?
2. How does the point density of simulated point clouds affect the performance of a machine learning segmentation and classification model in terms of the overall accuracy?

1.4 Assumptions, Delimitations & Limitations

1.4.1 Assumptions

The proposed study used a point cloud simulation method based on point scattering algorithm. The labelling of points that are at the intersection of two planes and two objects is often vaguely defined because it can be assigned to either section. The scattering algorithm took a 3D polygon mesh as an input and was designed to output a point set scattering evenly and randomly on the surface of the mesh. An assumption was made that there was no point generated on the intersection line of two planes.

1.4.2 Delimitations

This study only focused on finding the relation between the attributes of the simulated point clouds and the performance of the models. The main testing results contained 10 different models trained and tested by simulated point clouds. Although real-world point cloud data were used to see how the final models perform on real-world point clouds, this study was not targeted to generate a better segmentation and classification model for real-world point clouds.

This study discussed how the attributes of simulated point cloud data affect the segmentation and classification result. Only two attributes were tested, the point density and the level of random errors. Other attributes were not within the scope of this study.

The study used an existing machine learning model to conduct the test. All models were trained using the same algorithm. The segmentation and classification results depend on two things, the machine learning algorithm and the point cloud data. However, the optimization of the algorithm was not within the scope of this study.

1.4.3 Limitations

Firstly, the approach of simulating point clouds in a virtual 3D scene was different from the approach of LiDAR and photogrammetry in real-world. The point cloud simulation method used in this study did not imitate the scanning structure of real scanners. Therefore, the experiment results only showed the effects of simulated data. The findings cannot be directly applied to real-world point cloud data.

Secondly, point cloud processing and implementation, including this study, are project-oriented. The trained segmentation and classification models cannot perform as expected on any point cloud projects. Therefore, the results of this study might differ from the results of other experiments.

Thirdly, the segmentation and classification model trained by simulated data cannot promise a better model than existing models, in terms of the accuracy when applied on real-world projects. Existing segmentation and classification methods often use real-world data for the training of models and then apply to real-world projects. There have been few studies that use simulated data for the training and application to real-world projects. The differences between simulated data and real-world data, e.g., point distribution, data structure, produce huge differences in the training and implementation phase, which leads to huge mean loss and low accuracy. The proposed models' performance on real-world point clouds was not expected to exceed the original model trained by real-world point clouds. Nevertheless, the proposed study was an exploration study to find the potential in simulated point cloud data to be a more significant role in real-world point cloud processing.

Fourthly, the clusters were designed by original methods, so the clusters cannot be added or changed after models are trained and generated. This study used existing machine learning segmentation and classification algorithms. The labelled parts, such as ceilings, floors, walls, were designed by the original method. Therefore, some of parts in the scene were labelled as clutter. If one needs to add or change the clusters, a new labelling process needs to be applied before the training session. This study did not create a different cluster set from the original method in order to compare some of the results with the original method.

Finally, the study used the overall accuracy of the classification as the criterion for evaluating the performance of the machine learning models. The overall accuracy was defined as the number of correctly labelled points divided by the number of all the points. This criterion has been used widely in the field, e.g. the original method of Charles et.al. (2017), Boulch et.al. (2018), etc. However, they also used measurements such as intersection over union (IoU) and per class accuracy. IoU is a statistic measurement used for evaluating the similarity and diversity of sample sets. According to Boulch et.al. (2018), $IoU = \frac{1}{|C|}\sum_{c \in C} IoU_c$, where IoU_C is the IoU per class. $IoU_c = \frac{T_C}{|\rho_C \cup P_C|}$ where T_C is the number of correctly labelling points of class, ρ_C is the group of points with true label c, and P_C is the group of points estimated as class c. They are different measurements for the performance of the models but they all can show how well the models work on the point clouds. This study did not use accuracy per class for every point cloud because the study focused on the overall performance of the models affected by the point density and the level of random errors.

2. LITERATURE REVIEW

Lots of research has been done in the point cloud processing areas, e.g., segmentation and classification, automatic registration, 3D reconstruction, 3D city modeling, and etc. In the following sections, point cloud related studies are reviewed in detail.

2.1 General Scan-to-BIM

"However, as-built BIM creation of building interiors using scanned point clouds incurs critical difficulties: the complex design of indoor structures, not to mention obstacles, necessitates time-consuming manual operation and resultantly huge data sizes, which often leads to system slow-down or failure" (Jung et.al., 2014, p.68). This summary by researchers introduces the major problems in point cloud implementation.

The point cloud implementation usually aims at converting point cloud data to as-built BIM models. The point cloud implementation usually starts with data acquisition using photogrammetry or LiDAR. Then the output from data acquisition requires further processing, e.g., registration, down sampling, noise removal, segmentation and classification, and model generation. Automatic registration of point clouds remains a challenge due to the incompletion and similarity of buildings and other structures (Bueno et.al., 2018). Thirdly, in general, point cloud data are segmented and classified to each cluster. For instance, there are clusters such as floor, ceiling, wall, window, door, etc. This process is the necessary anterior task to semantic modeling. In semantic modeling, models are built with labels, which provides information for Building Information Modeling related implementation.

Segmentation and classification are interesting areas of point cloud implementation, and they are often associated with one another. Although the field has a trend of full automation, the truth is, BIM engineers still need manual field measurement to ensure quality (Jung et.al., 2014). As-built BIM created by existing drawings can cause some errors due to anomalies, destructions, and mistakes made by workers during both constructions and after (Barazzetti, 2016). Moreover, machine learning segmentation and classification methods are developing fast with the progress of computation power and algorithms. However, in the training and testing phase, manual labelling of point cloud data is required for the algorithms to learn from. The problems with this process are as follows: Manual labelling is time-consuming, troublesome, and often brings errors in labelling. The clusters are decided and fixed after the labelling is done. Therefore, if operators need to add or change the clusters due to project requirements, the labelling process need to be manually revised again.

During the past decade, there have been plenty of studies related to Scan-to-BIM, despite their focusing points. Especially Luigi Barazzetti (2018), who has many remarkable works about this. He presented a machine learning algorithm that aims at solving point cloud occlusion for surfaces based on neural networks, which is a very common dilemma the field is facing. Existing technology about occlusion detection and filling were designed to solve mesh surfaces problems based on neighborhood points, topology, change of local curvature, and distribution of hole texture (Barazzetti, 2018). Earlier, he presented a semi-automated method to generate 3D parametric as-built models from point clouds based on NURBS (Nom-Uniform Rational B-Splines) curves and surfaces (Barazzetti, 2016). Automated reconstruction can be done by planar modeling as well as volumetric modeling. Bueno et.al. (2018) presented a novel 4-plane congruent algorithm for automatic rough point cloud registration. Jung et.al. (2014) presented a semi-automatic methodology to improve the productivity of generating as-built complex indoor objects. Their operation has three steps: segmentation, refinement of downsized data, and boundary tracing.

Detection of important building components is crucial but challenging. Quintanaa et.al. (2018) came out with a 6D-space approach that detects doors at any condition based on 3D laser scanned data of the indoor environment. 6D-space means that the data not only contains XYZ coordinates but also, RGB or HSV. Repair and maintenance of historic buildings is so far one of the most important fields that Scan-to-BIM has contributed. Because natural environments do severe damage to building fabric all the time, especially ancient buildings, Valero et.al. (2018) presented an algorithm based on the 2D Continuous Wavelet Transform (CWT) to automatically segment rubble masonry walls. Luigi Barazzetti et.al. (2015) managed to convert historic BIM into a finite element model for structural simulation using Cloud-to-BIM-to-FEM (Finite Element Model) methodology, since historic buildings' characterization lacks efficient procedures.

There are indeed lots of papers about structural components such as walls, ceilings, and doors, as listed previously. However, there are only a few studies carrying out related to secondary components. Antonio Adána, Blanca Quintanaa, Samuel A. Prietoa, Frédéric Bosché presented a 6D-based approach to detect smaller objects on the wall using dense colored 3D point clouds, such as sockets, switches, signs, and others. Semantically-rich 3D models were generated by the original consensus procedure after being segmented and detected in the color and geometric spaces (Adán et.al., 2018). One of the most important indoor secondary component areas, MEP, which stands for Mechanical, Electrical and Plumbing components, benefits a lot from early identification of differences between as-design and as-built (Bosché et.al., 2015).

2.2 Point cloud data acquisition and quality

There are two major approaches of acquiring point cloud data, LiDAR (Light Detection and Ranging) and photogrammetry. In general, LiDAR is more expensive to acquire and operate, but straight-forward in post-production. Photogrammetry is easy to access but takes a lot more computational power during the generation of point cloud data. Some of the researches based on LiDAR and photogrammetry are summarized in the following sections:

2.2.1 LiDAR related applications and techniques

2.2.1.1 Terrestrial laser scanning

For many years, Terrestrial laser scanning (TLS) has been used for flatness quality assessment (FQA). "However, existing TLS assisted FQA methods are all designed particularly for a certain type of surfaces, like construction surfaces and component surfaces" (Li et.al., 2020).

FQA is one of the tasks that photogrammetry and other image processing technologies have trouble with. According to researchers (Li et.al., 2020), "However, these methods cannot be applied to the FQA, because the deviations in elevation of the concrete surface cannot be accurately estimated based on image measurement." Automatic photogrammetry often relies on feature extraction of stereo images to reconstruct 3D models. Due to the flatness and colorless or concrete surfaces, it is hard for feature extractions. However, the laser scanning techniques are not constrained here. In the study, RANSAC algorithm was conducted for point cloud data processing. Then, the distances between points and the fitted plane created by those points were calculated as deviations in elevation of the plane. The fitted plane was calculated by 80% of the total points on the surface, and it was fitted *n* times to find the best one according to the residual. The normal vector of the plane was different according to whether or not it was a flat surface or a convex / concave one. A colored deviation map was created to show the flatness. Distance from the fitted plane was associated with a color. The color varied from negative distance to positive distance, which was determined by the direction of the normal. The approach started with the data preparation. During this phase, point cloud data were registered and were selected based on the type of the data, either construction surface data or component surface data. The FQA of component surface was aided by as-design BIM models. The as-designed models were first presented as polygon meshes. Secondly, the models were discretized into many dense points (Bosché, 2008 & Bosché, 2010), and these points were used to create the standard plane.

In general, ″due to the limitations of the data collection processes as well as the complexity of as-built scenes, automated 3D modeling still presents many challenges" (Dimitrov & Golparvar-Fard, 2015, p.32). The general problems of point cloud data acquisition are as follows:

Point cloud density: "Point clouds exhibit locally variable densities based on surface orientation and distance from the capture device" (Dimitrov & Golparvar-Fard, 2015, p.33). Due to the physics behind the scanning technology, lots of aspects affect the point cloud density. For instance, a surface relatively closer to the scanner would have a higher point cloud density compared to the one that is further. Even for the same plane, there is density variation along with the distance between the scanned area and the scanner. Moreover, the definition of point cloud density is often for a whole set of point cloud data where there is density variation inside of it. And different densities lead to different segmentation results.

Point cloud data size: Numerous point-model distance and intersection calculations are such a burden that a discretization of surfaces' field and filtration of data are truly required. Some of the works applied raw data elimination before analyzing to increase the processing speed (Wang et.al., 2015). The raw point cloud often brings problems to registration. Therefore, a major topic is the downsizing, down sampling of point cloud data. But the data downsizing is often realized by replacing voxels with estimated points (Wang et.al., 2015), which lead to errors and incompletion, caused sometimes by occlusion. The decimation of points is always important because the computational cost can be high. Also, after decimation, points should be able to provide enough information.

Surface material: “A given scene can contain a wide range of surface roughness, no priors about noise levels can be reliably used” (Dimitrov & Golparvar-Fard, 2015, p.33). And there are some extreme situations where mirrors and glasses are involved because their optical features make the scanner difficult, even impossible to capture them. These kinds of material often bring in noises and unexpected point cloud data. 3D scan data with color often comes with the lighting problem because material is associated with the lighting. Some of the studies used a camera flash to reduce color variations caused by non-homogeneous illumination at different locations (Quintana et.al., 2018).

Noise and Clutter: “A scene can contain small objects represented by very few points, moving objects (as the building occupants), and multiple objects in close proximity, making feature detection difficult” (Dimitrov & Golparvar-Fard, 2015, p.33). Multiple registrations also bring noises, since each registration often happens at different times, one after another.

Occlusion: “A scene can contain objects of significant size that occlude objects behind them. This produces incomplete and disjointed descriptions of the background surfaces” (Dimitrov & Golparvar-Fard, 2015, p.33). Problems with multiple hidden layer occlusion remain unsolved and challenging (Barazzetti, 2018). Occlusion often comes with a nonregular room structure. Nonregular indoor components are always difficult for as-built model registration due to lack of information in the dataset, such as non-rectangular doors (Quintana et.al., 2018), and so on. Point clouds occlusion recovery is one of the most important steps in as-built BIM.

Scale: “To be value-adding, automated modeling is required to efficiently fit within a greater engineering task. As point cloud sets get larger, segmentation methods need to be scalable in terms of time and memory complexity. Special attention needs to be paid to off-line processes—that can be run overnight without user supervision—and on-line responsive processes that leverage user interaction” (Dimitrov & Golparvar-Fard, 2015, p.33). Moreover, rotation is always a big obstruction for automatic detection and registration (Adán et.al., 2018). The algorithm must be adjusted and improved to adapt to this non-orthoimage.

Operation and maintenance of existing buildings, quality control, defect detection, and others are efficient benefits as-built BIM has brought to the industry (Carbonari et.al, 2015; Liu et.al, 2012). “However, as-built BIM creation of building interiors using scanned point clouds incurs critical difficulties: the complex design of indoor structures, not to mention obstacles, necessitates time-consuming manual operation and resultantly huge data sizes, which often leads to system slow-down or failure" (Jung et.al, 2014, p.68). Also, In the commercial area, the level of automation still has limitations for TLS (Terrestrial Laser Scanning) (Bosché et.al, 2015). That is why it is important to have a system that can provide a more accurate model with a higher speed.

Studies have been done to assess the importance of point cloud quality. However, the relation between the levels of point cloud quality and the success of Scan-vs-BIM is still not solved. Rebolj et.al. (2017) focused on the difference between point cloud quality and scanning methodologies, such as photogrammetry, videogrammetry, and range camera. But in photo and videogrammetry, only the density criterion has been verified since there is no direct correlation between camera resolution and the number of frames per second. The quality of point cloud will be determined in density and error (point position shift). Rebolj et.al (2017) provided standard criteria for point cloud quality in automated construction progress monitoring according to the successful identification of building elements of the four size classes described in construction phases. The four size classes are defined as large elements (L): Size $\geq 5m^2$, Medium elements (M): $1\ m^2 \leq$ Size $< 5\ m^2$, Small elements (S): $0.25\ m^2 \leq$ Size $< 1\ m^2$, Very small elements (XS): Size $< 0.25\ m^2$. Sizes are defined as the sum of the surface areas that an element projects onto three orthogonal planes of the element's local coordinate system. They also presented the relationship between the criteria and scanning methodology parameters. Compared to the traditional qualitative manner, the paper assessed the data in a quantitative manner.

2.2.1.2 Airborne laser scanning

LiDAR data based city modeling has proven itself to have promising results (Rottensteiner et.al., 2014). Generally, the automatic city-scale model generation is achieved by steps described as follows. Firstly, primitive geometric elements are extracted from point clouds as modeling cues. Secondly, these modeling cues are transferred into topological elements, they can both represent 3D rooftop models. CityGML is an open-source and XML-based application for generating virtual models. "In CityGML, 3D rooftop models can be differently represented according to the level-of-detail (LoD). A prismatic model of rooftop that is a height extrusion of a building footprint is defined as LoD1 in CityGML, while LoD2 requires a detailed representation of the primitive geometric and topological elements in a 3D rooftop model" (Jung et.al., 2017, p.2).

"A critical problem to hinder the automation of 3D rooftop model generation is that many portions of the object (rooftop) are unknown, and recovered with errors caused by the following reasons: Irregular point distribution, occlusion, and unreliable data analysis" (Jung et.al., 2017, p.2). Irregular point distribution is caused by the characteristics of laser scanners. The beam footprint and the space between each point are determined by multiple aspects, e.g., flight height, ground elevation, scanning frequency, and the angles between beam and objects' surfaces. Therefore, these system variables produce an irregular distribution to the outcome of point clouds, and as a result, errors in modeling cues. Occlusions are made due to multiple layers of objects on the line of the beam. Despite it being plausible for the laser to penetrate certain types of objects, the final receiving pulse energy depends on various things. In addition, due to limited flight lines especially for urban areas, it is impossible to avoid buildings blocked by other buildings and objects. In order to realize automation, a lot of algorithms are applied to raw point clouds to fulfill certain goals, e.g., object detection, segmentation and classification, feature extraction, etc. Therefore, errors in the raw data due to data size, occlusion, and noise result in the partial failure in the performance of these algorithms, and eventually, the failure in the modeling.

The inherent characteristic is one of the most distinguishable features of LiDAR compared to photogrammetry. But the inherent characteristic of LiDAR data associates with errors in modeling. There are three major types of modeling error using LiDAR data: shape deformation, boundary displacement, and orientation error (Jung & Sohn, 2019). Firstly, shape deformation is the difference between generated models and the reference models. The main reason for shape deformation is the inconsistency of point clouds from LiDAR. The physics principle of laser scanners creates gaps between points. Relatively, the further the scanner is to the scanned surface, the bigger the gap is. "Shape deformation can be caused by various reasons, such as scene complexity, data characteristics (resolution, signal-to-noise ratio, occlusion, and incomplete cue extraction), and characteristics of the rooftop modeling algorithms used (model-driven approach or data-driven approach)" (Jung & Sohn, 2019, p.158). The second modeling error is the boundary displacement. It is also caused by the discrete point distribution of LiDAR.

The LiDAR is not able to guarantee the exact capture of objects' boundaries. Therefore, the reconstructed geometry would be slightly smaller than the real object if the actual boundary is in the middle of two rows of scanned points. However, this kind of error often does not affect the overall shape of the reconstructed model (Jung & Sohn, 2019). The third main error is called the orientation error. "The building orientation is determined by analyzing angle distributions of initial rooftop boundaries which are generated by tracking irregularly distributed boundary points. Unlike optical imagery, building orientation errors caused by LiDAR data are not uniform across buildings but are subject to a relative angle between the scanner's flying direction and the orientation of a building of interest" (Jung & Sohn, 2019, p.158). This modeling error can be partially solved when integrated with photogrammetry.

Modeling constraints are functional as one of the approaches to addressing the aforementioned issues. "These constraints are used as prior knowledge on targeted rooftop structures: (1) for constructing the modeling cues to conform to Gestalt law (i.e., parallelism, symmetry, and orthogonality), and linking fragmented modeling cues in the frame of perceptual grouping; and (2) by determining optimal parametric rooftop model fit into part of rooftop objects through a trial-and-error of model section from a given primitive model database" (Jung et.al., 2017, p.3). These constraints were described as "explicit regularity" by Jung et.al. (2017) because they are fully defined. They introduced an "implicit regularity" to help constraint the modeling cue generation without fully expressing the relation between input and output. "This implicit regularity is used as a constraint for changing the geometric properties of the modeling cues and topological relations among adjacent modeling cues to conform to a regular pattern found in the given data" (Jung et.al., 2017, p.3).

Fast urbanization has demanded more from 3D city modeling for supporting a lot of applications, e.g., urban planning, flood simulation, emergency crowd simulation, and location-based services. Research has been vastly conducted to fully enable automated city-scaled 3D modeling. "One the most challenging task for 3D building model reconstruction is to regularize the noises introduced in the boundary of building object retrieved from a raw data with lack of knowledge on its true shape" (Jung et.al., 2017, p.1). They introduced a data-driven modeling approach to reconstruct city-scale models based on airborne laser scanning data. Firstly, the original rooftop objects were grouped into homogeneous point clouds according to their height and plane similarity. From there, modeling cues were extracted, e.g., line and plane primitives. Secondly, by using the Binary Space Partitioning (BSP) technique, the topological elements were recovered by iteratively partitioning and merging. Thirdly, by removing erroneous vertices or rectifying the geometric properties, implicit regularity was achieved with the deduction of errors in the modeling cues and topological elements. Given noisy information of the building boundary in a progressive manner, the proposed method was able to implicitly derive the shape regularity of the rooftops.

Updating 3D city models is more crucial than generating detailed city models that are out of date because analyses based on 3D city models requires current information about cities to conduct proper experiments and provide more accurate results. Thus, a few studies have been conducted to generate a progressive city modeling method, e.g., "by integrating the information retrieved from the existing model with new modeling cues extracted from single airborne imagery" (Jung & Sohn, 2019, p.158). Rapid development and changes in modern cities have brought extra dilemmas to the time-consuming and error-prone process of 3D city modeling. Most researches and applications have managed to reconstruct city models using various algorithms. "However, cities are dynamic systems that continuously change over time. Accordingly, their virtual representations need to be regularly updated in a timely manner to allow for accurate analysis" (Jung & Sohn, 2019). They proposed a fusion method to refine building rooftop by integrating optical imagery with previous models based on LiDAR. In the proposed study, a LIDAR-driven model (L-Model) by Jung et.al (2017) was used as a base model and was later refined by integrating it with airborne image features of the MCMC (Markov Chain Monte Carlo) framework. Images of rooftops were fed into two algorithms to extract features. "The shape of building rooftops can be well described by lines and corners" (Jung, J., & Sohn, G., 2019, p.160), therefore Kovesi's algorithm (Kovesi, 2011) was used to detect straight lines and an algorithm by Chabat et.al (1999) was used to detect corners. Then the extracted lines and corners were used as modeling cues for the refinement of models. After extracted, the 2D features needed to be transformed into 3D object spaces to match with the LiDAR-driven models. A concept of topological lines was generated between the L-Model and imagery. However, a transformation from 2D to 3D requires stereo images or multiple images, according to the collinearity equation. The study was limited to a single image, therefore, researchers back-projected the L-Model into the 2D image plane to establish the relation between I-Lines (I-Corner) and L-Model. Then, by using the height information of L-Model, the 3D coordinates of I-Lines and I-Corners were determined. In the end, an optimal model was selected after balancing model closeness and complexity.

Elevation data is crucial to the reconstruction of 3D objects, especially 3D city modeling. Biljecki et.al. (2017) introduced an approach of predicting the height of buildings using machine learning algorithms. The results satisfied the accuracy recommendations for the needs of some GIS analyses in terms of mean absolute error. One of the disadvantages of the approach is that the outcome of the proposed method of Biljecki et.al. (2017) would not be as accurate as those projects that have elevation data, but the researchers argued that their approach provided useful spatial analyses such as fast updating of newer constructions before the acquisition of elevation data. Their approach used supervised learning models based on predictors which are the attributes of buildings to estimate the height of the building in order to create 3D city models. Different predictive models were associated with the availability of the building attributes. They used a supervised learning method for classification and regression called Random Forests (Breiman, 2001). "It works by creating a number of decision trees on random subsets of data and uses averaging to improve the predictive accuracy and control over-fitting" (Biljecki et.al., 2017). An advantage of this method is its ability to assess the importance of different predictors, which works as a weight assigned to different predictors according to different situations. The attributes of this study included were building use, year of construction, number of stories above ground, the net internal area (sum of floor area in all units in a building), footprint area, shape complexity, number of neighboring objects, population density, average household size, and average income. Nevertheless, not all attributes were available for certain buildings. Then, after training the data, they used a point cloud of the City of Rotterdam as a benchmark (ground truth heights of buildings) to validate the results. They studied real-world cases of how building attributes affect the building heights to optimize their predictors in the model. The most associated predictors were stories, building age, net internal area, and population density. However, there were always outliers. In conclusion, they have shown the results that "several attributes available solely from 2D data can hint at a building's height. The achieved accuracy is comparable to many other instances used in research and practice. The experiments and the discussion show that these models could be useful in a number of spatial analyses" (Biljecki et.al., 2017).

2.3 Point cloud simulation

Due to the price and complexity of laser scanners, related studies often result in wasting a rather long time in the point cloud data acquisition phase. Although data acquisition is very crucial to project-oriented works, some studies that focus on other aspects such as point cloud data quality analysis do not need point cloud data of any specific environments. Researchers (Lari, 2014; Bechtold & Höfle, 2016) have been using simulated 3D scans for the acquisition of point cloud data. Bechtold & Höfle (2016) mentioned several possible cases for a laser scanning simulator, e.g., research and planning of scanning strategies, laser scanning teaching, and training, generation of artificial scan data for algorithm development, or sensor development and evaluation.

Despite the idea of laser scanning simulation, it is not new (Bechtold & Höfle, 2016). More and more simulated data have been brought into the system for testing. Lohani & Mishra (2007) developed a "2.5D" elevation map airborne laser scanning (ALS) simulator for education and general research which cannot be used to simulate terrestrial laser scanning (TLS) with realistic high-detail scenes. In terms of ALS simulator, Kukko and Hyypp¨a (2009) also created a laser scanning simulator that is capable of modeling beam divergence and full-waveform signal recording. Wang et. al (2013) created a system that interacts with a RIEGL VZ-1000 TLS system and a Tilia tree. The purpose of their work is to investigate how different scanning positions affect the derivation of plant characters, such as leaf area index. Lari (2014) generated simulated laser scanning data for his research using the Unity 3D game engine, however, this kind of method generates the point clouds without considering random errors in the laser scanning system measurements. Therefore, a forced erroring was applied after the simulation. The simulated data then was used for testing the segmentation outcome. Lari (2014) generated simulated laser scanning data for his research using the Unity 3D game engine. "Ray casting is one of the most important physical functions which are supported by this game engine" (Lari, 2014, p.147). Ray casting has been used frequently in Computer Graphics. Bechtold & Höfle (2016) developed a laser scanning simulation system name Heidelberg LiDAR Operations Simulator (HELIOS). As described, "HELIOS is implemented as a Java library and split up into a core component and multiple extension modules. And the objectives of the framework are teaching and training of laser scanning, development of new scanner hardware and scanning methods, or generation of artificial scan data sets to support the development of point cloud processing and analysis algorithms" (Bechtold & Höfle, 2016, p.161). In terms of platform classes, the HELIOS system has four of them: "1. Four-wheel ground vehicle with one steerable axle. 2. Helicopter/Multicopter. 3. Simple linearly interpolated movement along straight lines. 4. 'Dummy' platform without movement code (base class of all other platform classes, can be used to simulate stationary scanners)" (Bechtold & Höfle, 2016, p.163). There are two main parts that define a platform: The logic of simulation of platform type, e.g., airborne, vehicle, stationary, and the actual parameters of the platform, e.g., position, maximum speed, the height of the scanner (Bechtold & Höfle, 2016).

The distance calculation between simulated and real-world data is different. When using a 3D scanner, the range (distance) is calculated by the time of the pulse between the initial one and the reflected one. On the contrary, the computer records the hit point information and then calculates the Euclidean coordinates.

Another approach simulates point cloud by taking representative points directly from surfaces without casting any rays. Points are generated from the surface in order to maintain a fixed density. Occlusion is one of the biggest problems in point cloud implementation. By directly sampling from surface, occlusion can be avoided. Although the proposed approach is highly different from how a laser scanner works, the potential of an occlusion-free point cloud dataset may benefit the segmentation and classification method. In a study of Li et.al. (2020), they managed to carry out a method of assessing the flatness quality of construction surfaces and component surfaces. In their study, they generated the point cloud of as-design component models by discretization (Bosché, 2008 & Bosché, 2010). Then the dense points were used to create a standard surface for the calculation of the deviation of the scanned point cloud.

2.4 Point cloud segmentation and classification

Point cloud acquisition, modeling, and data analysis is a time-consuming, expensive, and problematic task, and it is often done by human operators in current industry (Dimitrov & Golparvar-Fard, 2015). The need for low-cost, reliable, and fast automated point cloud data implementation is in high demand because in fast changing construction sites, manual methods often fail to keep up with the requirement for monitoring and management (Dimitrov & Golparvar-Fard, 2015).

The process of segmentation is not straightforward. "Starting with a point cloud of a scene—generated using laser scanners or image-based reconstruction methods—the user must first identify collections of points that belong to individual surfaces, and then, fit surfaces and solid geometry objects appropriate for the analysis" (Dimitrov & Golparvar-Fard, 2015, p.32). This process is also often done by human operators, which can be time-consuming and troublesome.

Common segmentation methods include edge-based segmentation, region growing, model fitting, hybrid method, and machine learning algorithm (Grilli et.al., 2017).

Edge-based segmentation algorithms normally have two steps, as described by Rabbani et.al. (2006): (1) edge detection to outlines the borders of different regions and (2) grouping of points inside the boundaries to deliver the final segments. Edges in a given depth map are defined by the points where changes in the local surface properties exceed a given threshold.

Region growing segmentation often starts from seed points and the model grows around with the seeds and find points with similar characteristics (Rabbani et.al., 2006; Jagannathan and Miller, 2007). Dimitrov & Golparvar-Fard (2015) proposed a region growing method for robust context-free segmentation of raw point cloud data based on geometrical continuities. Their method of segmentation starts with multi-scale feature detection, describing surface roughness and curvature around each 3D point, then seed finding and region growing steps are applied. (Dimitrov & Golparvar-Fard, 2015)

A model fitting segmentation method depends on the decomposition from man-made objects into geometric primitives. "Therefore, primitive shapes are fitted onto point cloud data and the points that conform to the mathematical representation of the primitive shape are labelled as one segment" (Grilli et.al., 2017, p. 340). A hybrid segmentation method is a method that combines more than one segmentation method.

Generally, machine learning methods are divided into two categories, supervised learning methods and unsupervised ones. "Supervised methods are the majority with a training phase mandatory and fundamental to guide the successive machine learning classification solution" (Grilli et.al., 2018, p. 339). Supervised learning relies on a set of provided training examples for the machine to learn how to deliver a correct result that it is designed to execute. Therefore, the result of the segmentation and classification done by algorithms associates with the provided human-labelled learning examples. And the results are always influenced by the quality of the training data and the density of the point cloud data (Grilli et.al., 2018). In such studies (Boulch et.al., 2014), the semantic result was highly dependent on the abstract model, trained using grammars on a 3D reconstructed model. Unfortunately, manual labelling cannot avoid errors due to human nature, which might lead to vital consequences for the machine learning methods. The task of labelling every point in the dataset is rather challenging due to the variety of object types, and spatial permutation of objects (Yousefhussien et.al., 2018).

Different machine learning algorithms define clusters differently according to their own interests. Even a single machine learning algorithm can have different sets of clusters. Clusters are defined before and during the labelling process, which means once the model is trained, the clusters are final. If any projects require specific labelling, the labelling of training dataset needs to be revised.

"A machine learning segmentation method is a scientific discipline concerned with the design and development of Artificial Intelligence algorithms that allow computers to take decisions based on empirical and training data" (Grilli et.al., 2017, p. 341). A noticeable relation must be built between the data and the observed variables. A feature with higher quality often also provides a better machine learning model. In general, there is supervised learning and unsupervised learning. "In machine learning, unsupervised learning is a class of problems in which one seeks to determine how the data are organized. It is distinguished from supervised learning (and reinforcement learning) as they rely on a set of provided training examples (features) to learn how to correctly perform a task" (Grilli et.al., 2017, p. 341). Since training a machine learning model takes a long time due to the computation load, researchers (Weinmann et.al., 2015) have noticed that the quality of data plays a much more significant role than the quantity of data. K-means clustering is one of the machine learning segmentation methods. "It is a method based on an algorithm able to classify or to group set of (3D) points into K groups using attributes /features. The grouping is done by minimizing the sum of squares of distances between point and the corresponding cluster centroid" (Grilli et.al., 2017, p. 341). On the contrary, hierarchical clustering methods compute features for every single point in the dataset based on geometrical characteristics. "They usually create a hierarchical decomposition of a dataset by iteratively splitting the dataset into smaller subsets until each subset consists of only one object” (Ng and Han, 1994, p. 123). Boulch et.al. (2017) presented a 2D deep segmentation network that took multiple snapshots of the point cloud for a pixel-wise labelling of each pair of them using fully convolutional networks. Then, back-projection was applied to deliver the labelling of 2D pixels to 3D points using efficient buffering. The interesting part of that research is the transformation from 3D points to 2D pixels and the back-projection. Since the 2D image space segmentation networks have been introduced a long time ago and proved to be rather efficient, the results can be perceived as trustworthy and time-saving. According to researchers, “Generally, such features – usually derived from the 3D-covariance matrix – are computed using the surrounding neighborhood of points. While these features capture local information, the process is usually time-consuming and requires the application at multiple scales combined with contextual methods in order to adequately describe the diversity of objects within a scene” (Yousefhussien et.al., 2018, p191). Hence, they introduced a 1D – fully convolutional network that consumes terrain-normalized points to perform labelling by implicitly learning from points’ contextual features. This method allowed semantic labelling on unordered point sets with varying densities.

2.5 Machine learning point cloud processing

Machine learning methods have been used frequently in point cloud processing, e.g., 3D city modeling, semantic modeling, etc.

3D city models have contributed to a wide range of fields, e.g., 3D mapping, building construction, flood simulation, urban planning, disaster management, and other large-scale analyses. LiDAR and photogrammetry provide various approaches to create 3D city models. ʺTraditionally, 3D rooftop models are derived through interaction with a user using photogrammetric workstations (e.g., multiple-view plotting or mono-plotting technology). This labor-intensive model generation process is tedious and time-consuming, which is not suitable for reconstructing rooftop models at city-scale” (Jung et.al., 2017, p.2). Recently, LiDAR has become a rather popular data acquisition tool due to its high density and accuracy. Computer Vision and LiDAR have made full automation possible and functional in the industry. “The automatic reconstruction of 3D geometric models of building rooftops has, for more than two decades, been considered a central research topic in the photogrammetry and computer vision fields, due to their prominence in virtual city models” (Jung & Sohn, 2019, p.157).

Biljecki et.al. (2017) introduced a supervised machine learning approach of predicting the height of buildings. The algorithm differentiated the building rooftop points and clutters, e.g., “tree canopy, tree branch, chimney, spike, miscellaneous objects on the roof, and the façade of contiguous buildings” (Park & Guildmann, 2019), so that the height of the building was only associated with the rooftop points.

Using footprints to estimate the height of the building often results in potential errors due to irrelevant objects that are assumed to be the rooftop. Researchers (Park & Guildmann, 2019) proposed a machine learning LiDAR point classification method to detect only the rooftop to be the estimation of the building height. Random Forest (RF) algorithm was used to classify points into four categories, e.g., rooftop, wall, ground, and high outlier. High outliers are the points that are higher than a user-defined height than the rooftop, which are often introduced by vegetation. And the outcome of this approach produced outcomes that were much closer to the ground truth. The proposed methodology consists of six main steps. First, LiDAR point clouds outside of the building footprint were eliminated from the original point clouds so that only the building point clouds were taken into consideration and each point only belonged to a single building. There were four classes of points: rooftop, wall, ground, and high outliers. Only rooftop points would be considered as predictors of building heights because others would underestimate or overestimate the building heights. Second, training and testing datasets were prepared according to high resolution 2D and 3D imagery. Third, buildings were grouped into different types of buildings and trained according to that. Fourth, the attributes that controlled the prediction were computed. Fifth, the Random Forest was selected to produce the final results. RF (Random Forest）classifier was trained using 11 features by growing decision trees. An advantage of RF is that it provides rankings for the predictors so that the features change according to the classification, and as a result it provides a more accurate performance. Finally, classification accuracies were analyzed. “The classification results are evaluated with four accuracy measures: overall accuracy (OA), producer's accuracy (PA), user's accuracy (UA), and Cohen's coefficient of agreement (Kappa). PA and UA are computed for each class. PA indicates the percentage of correctly classified samples within a ground-truth class. UA indicates the percentage of correctly classified samples within an assigned class. Kappa indicates a measure of classification reliability” (Park & Guildmann, 2019, p.79). The results of this study showed a mean overall accuracy (OA) of 96.5%. The overall classification accuracy increased after training separate classifiers for different types of buildings, e.g., residential buildings, commercial buildings. The OA of different types of buildings also differed from each other. 60.6% of the buildings had changes in building heights after the point clouds were classified. “The developed methodology differs from previous studies in that it utilizes open-source datasets, including LiDAR of low sampling resolution (less than 1 pts/m^2), lower complexity algorithm, and fewer predictors, but attains high classification accuracy (96.5%)” (Park & Guildmann, 2019).

3. RESEARCH DESIGN AND METHODOLOGY

3.1 Point cloud acquisition

In the point cloud data acquisition phase, both real-world and virtual simulated point cloud data were collected. The source and process of data collection are described in detail in the following sections.

3.1.1 Real-world point cloud data acquisition

In order to compare with the original method PointNet (Charles et.al., 2017), this study used the same real-world point clouds as the ones of the original method. In the original method, there were 6 areas of scanned indoor environments. Each area had multiple rooms, e.g. conference Room_1, hallway_1, office_1, etc. Each room had two datasets. The first dataset was called the Annotation, which contained each single point cloud of a single labelled object. The second dataset contained all the points in the room. For example, in the copyRoom_1, there was a dataset that contained all the points that belonged to the copyRoom_1. There was another dataset that divided the points into different point clouds of different labelled objects, e.g. beam_1, beam_2, column_1, wall_1, ceiling_1, table_1, chair_1, clutter_1, and etc. All points first stored in TXT files and modified to NPY (NumPy array) and H5 (Hierarchical Data Format) files for further processing. In each TXT file, 6 parameters were stored, including 3 positional parameters and RGB (red, green, and blue) parameters. However, this study only used positional parameters to train the models instead of all the 6 parameters. 0 weight was assigned to RGB parameters during the training and testing.

3.1.2 Simulated point cloud acquisition

The simulated point clouds were generated to train and test the segmentation and classification machine-learning models. The simulation process was described in detail in the following sections.

3.1.2.1 The mesh generation of labelled objects

The PointNet (Charles et.al., 2017) algorithm defines the clusters in 13 different labels, including “ceiling”, “floor”. “wall”, “beam”, “column”, “window”, “door”, “table”, “chair”, “sofa”, “bookcase”, “board”, and “clutter”. All the labelled objects except “clutter” were created in this step. Each kind of labelled model was created according to its characteristics under 3D laser scanning and photogrammetry. For example, room structure models such as “ceiling”, “floor”, “wall”, “beam”, and “column” were created using grid geometry, a simple layer primitive. The laser scanner and photogrammetry can only capture the inside structure of the room, and as a result, the point cloud of a wall was just a set of coplanar points. Other objects including “window”, “door”, “table”, “chair”, “sofa”, “bookcase”, “board” were imported into the scene as modified OBJ files. These OBJ files were created in advance according to real-world objects and modified based on their characteristics under LiDAR and photogrammetry. Each label had multiple types of objects and each created model had its own randomized position and size. For instance, there were 7 different models of doors, and each type of door was randomly chosen by a randomization algorithm to be put into the indoor environment. There were 5 different models of windows, 3 different models of bookcases, 3 different models of tables, and 6 different models of chairs. Models’ sizes were normalized to standard sizes according to the Architectural Graphic Standards, 12th Edition, (Hall & Giglio, 2016) then the generated models were given random shifts and rotations, as well as other random sizes.

Models were located in the indoor environments according to the Architectural Graphic Standard, 12th Edition (Hall & Giglio, 2016). Doors were fixed and located at one side of the walls, but the location within the wall was randomly selected. Other models changed locations according to the door. The as-designed models were imported as modified OBJ files as the representatives of the 3D objects. There were 3 different types of rooms, including small, medium, and large rooms. Each type of room had its own structural design and number of objects and facilities. For instance, small rooms had 1 door, 1 window, 1 bookcase, 1 table, 1 chair, 1 sofa, and other clutter. A table and a chair were located in the center area of the small room, while a sofa was located off the center. The size of the small rooms is defined as follows: The length and width is 4 meters to 6 meters and the height was 2.5 meters to 3.5 meters. The medium rooms had 1 door, 1 window, 1 bookcase, 4 tables, 4 chairs, and other clutter. The size of the medium rooms is defined as follows: The length and width is 6 meters to 8 meters and the height is 2.5 meters to 3.5 meters. The large rooms were designed to imitate classrooms. There was 1 door, 2 windows, 1 board, and multiple tables and chairs. The size of the large rooms is defined as follows: The length and width is 9 meters to 11 meters and the height is 2.5 meters to 4.0 meters. The randomization in locations, rotations, and sizes provided a variability of the training dataset so that the trained segmentation and classification models can apply to a wide range of different projects.

Despite the fact that this simulation method was different from the real-world scanning methods, there were some aspects that had to be taken care of in order to be further applied to real-world point clouds. As shown in Fig. 1, this was the backside of a bookcase which was placed against the wall. The cyan surface which had an RGB color of (0, 255, 255) was a side of the walls. The part where it intersected with the bookcase was deleted, and so was the intersection part of the bookcase. This procedure was to imitate how real-world scanning methods work, including LiDAR and photogrammetry, because the backside of an object cannot be captured.

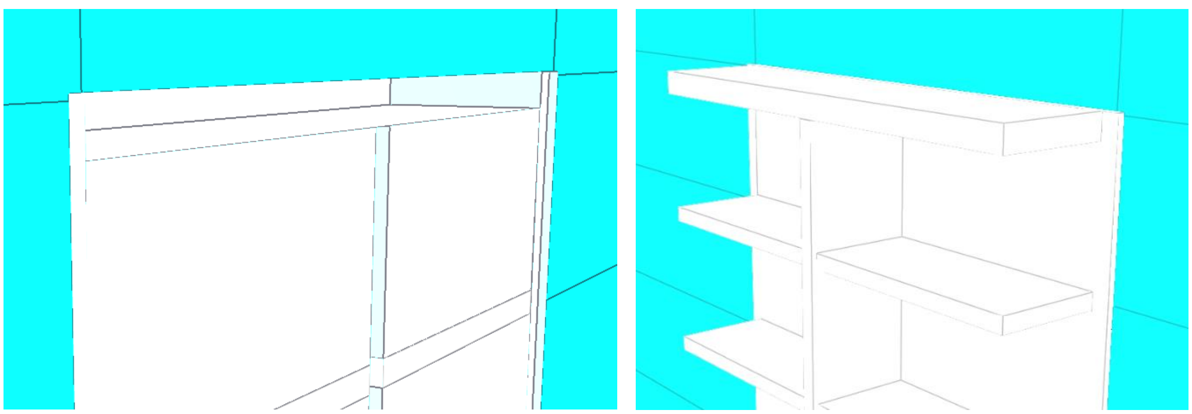

Figure 1. This shows the backside and frontside of a bookcase in the modeled environment. The part where it intersected with the bookcase was deleted, and so was the intersection part of the bookcase.

3.1.2.2 The simulation of clutter objects

There was one cluster, called clutter. The clutter included all the objects that were not given a label. The amount of clutter depended on the definition of the algorithm. Before the labelling of training and testing point clouds, users can change the number of clusters so that the number of clutters is changed. However, for this study, in order to be compared with the original method, the clusters were defined as the same as the original algorithm including 13 clusters.

There were multiple models belonging to the clutters, e.g. the PC (personal computer), the laptop, the book, the cup, the keyboard, the mouse, the light, etc. The occlusion by these clutters was taken into consideration. For instance, the pc and the laptop were associated with the table, which was one of the labelled clutters in the algorithm. The mentioned clutter objects were only generated on the upside surface of the tables instead of anywhere in the indoor environments.

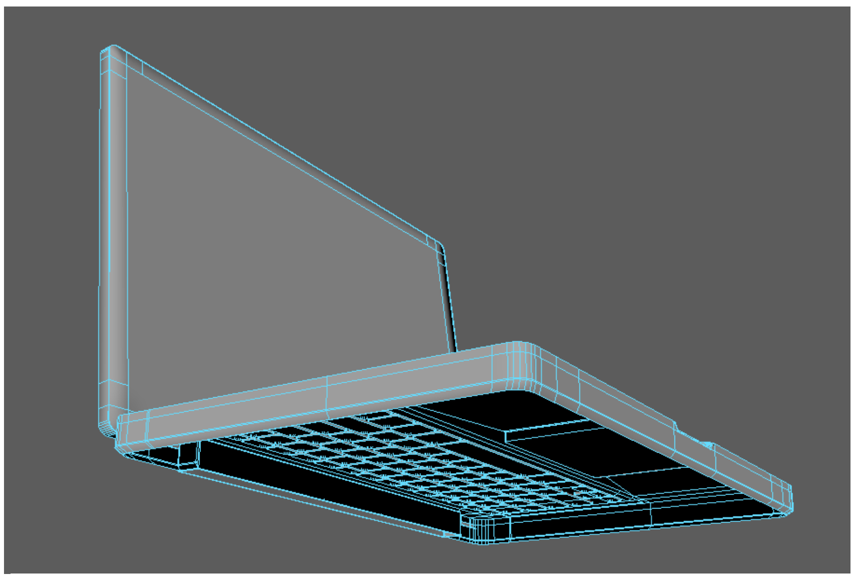

Figure 2. This shows a laptop mesh. The downside of the laptop was deleted, because it was placed on the table.

Moreover, by following the rules of LiDAR and photogrammetry, only the visible parts of the objects were generated, as shown in Figure 2. The downside of the laptop collided with the upside surface of the table. Therefore, the intersection part was deleted, and so was the part on the table. The process of defining the modeling of a laptop is described as follows. First, the bounding box of the table was extracted to be the original location of the laptop. Second, a random shift and rotation were applied to the table, as well as a scale change. Third, the parts of the table where the laptop intersected with were deleted. Each simulation of indoor environments resulted in different locations and a different number of items on the table, which added to the variety of the point cloud dataset. Other clutter objects were generated in the similar way.

Model modifications were applied before and while the models were imported into the virtual environment. During the model modifications, the structures of original models were adjusted to imitate a scanning situation of the object according to how they would be captured and behave in real-world situation.

3.1.2.3 The generation of point clouds

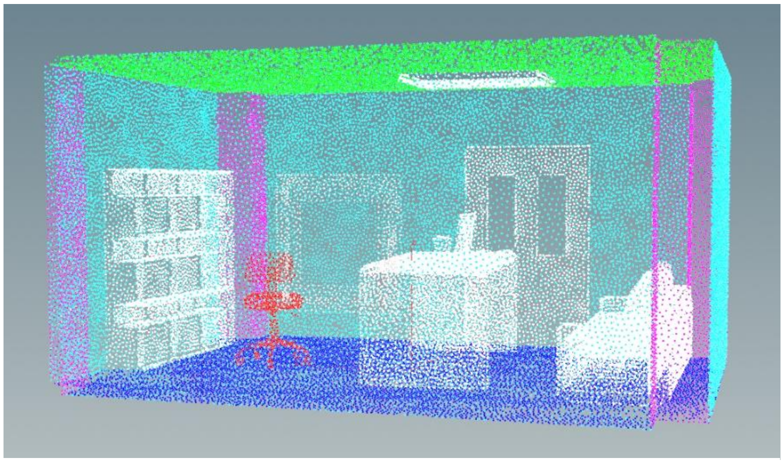

Figure 3. This shows the point cloud of a small room with a point density of 500 pts/m^2 (points/m^2) and no random error.

After the meshes were generated, the scattering algorithm was applied to all the meshes. The scattering of points meant that the surfaces were discretized into many dense points (Bosché, 2008 & 2010). The point density was defined as the number of points on a surface divided by the area of the surface. Therefore, the number of points was the multiplication of the summation of the surface areas and the user-defined point density. The points were evenly and randomly scattered around the surfaces of the objects. Figure 3 shows the point cloud of a small room with a point density of 500 pts/m^2 (points/m^2) and no random error. Each simulated point cloud dataset contained 4 areas of simulated environments. Area_1, area_2, and area_3 were the point clouds for training. Area_4 was the point clouds for testing. Area_1 had 40 small rooms, area_2 had 40 medium rooms, and area_3 had 40 large rooms. Area_4 had 10 small rooms, 10 medium rooms, and 10 large rooms. There were 10 simulated point cloud datasets, including 50 pts/m^2 without random error, 100 pts/m^2 without random error, 500 pts/m^2 without random error, 1000 pts/m^2 without random error, and 5000 pts/m^2 without random error; 50 pts/m^2 with 5 mm ϵ, 100 pts/m^2 with 5 mm ϵ, 500 pts/m^2 with 5 mm ϵ, 1000 pts/m^2 with 5 mm ϵ, and 5000 pts/m^2 with 5

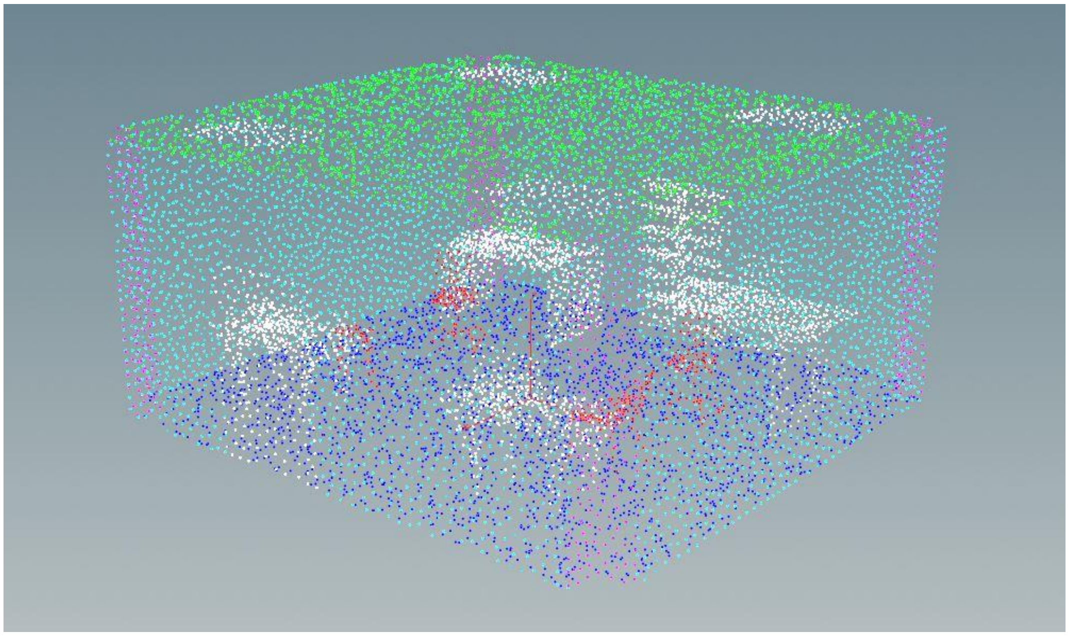

Figure 4. This is an example of the point cloud of a medium room with a point density of 50 pts/m^2 without random error.

mm ϵ. The ϵ was defined as the average of the random error in the point clouds. Figure 4 shows an example of the point cloud of a medium room with a point density of 50 pts/m^2 without random error. Figure 5 shows an example of the point cloud of a medium room with a point density of 100 pts/m^2 without random error. Figure 6 shows an example of the point cloud of a medium room with a point density of 500 pts/m^2 without random error. Figure 7 shows an example of the point cloud of a medium room with a point density of 1000 pts/m^2 without random error. Figure 8 shows an example of the point cloud of a medium room with a point density of 5000 pts/m^2 without random error. Figure 9 shows an example of the point cloud of a large room with a point density of 50 pts/m^2 without random error. Figure 10 shows an example of the point cloud of a large room with a point density of 100 pts/m^2 without random error. Figure 11 shows an example of the point cloud of a large room with a point density of 500 pts/m^2

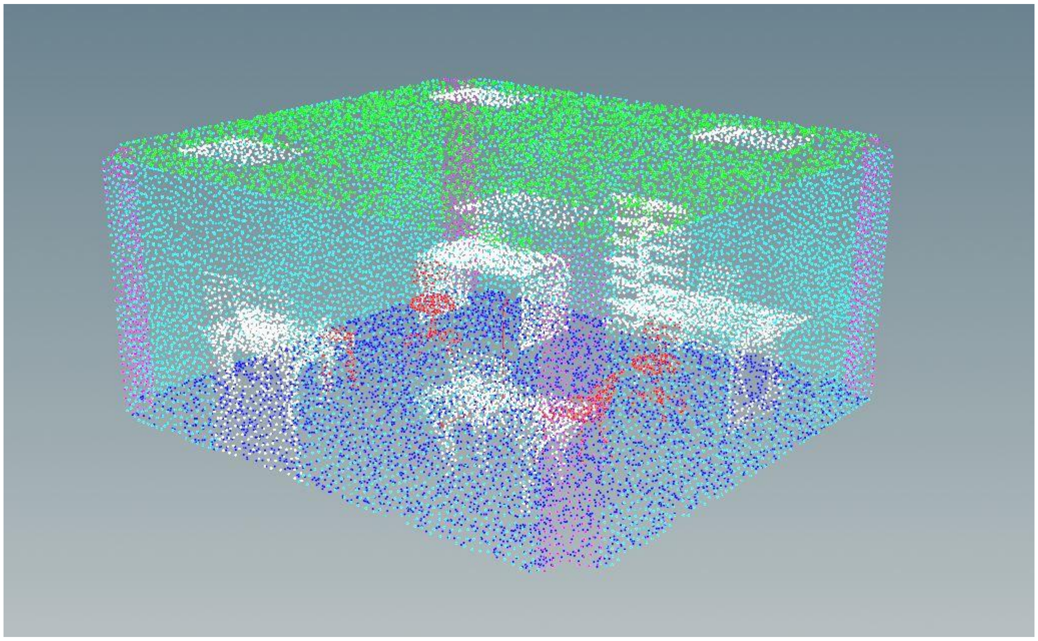

Figure 5. This is an example of the point cloud of a medium room with a point density of 100 pts/m^2 without random error.

without random error. Figure 12 shows an example of the point cloud of a large room with a point density of 1000 pts/m^2 without random error. Figure 13 shows an example of the point cloud of a large room with a point density of 5000 pts/m^2 without random error.

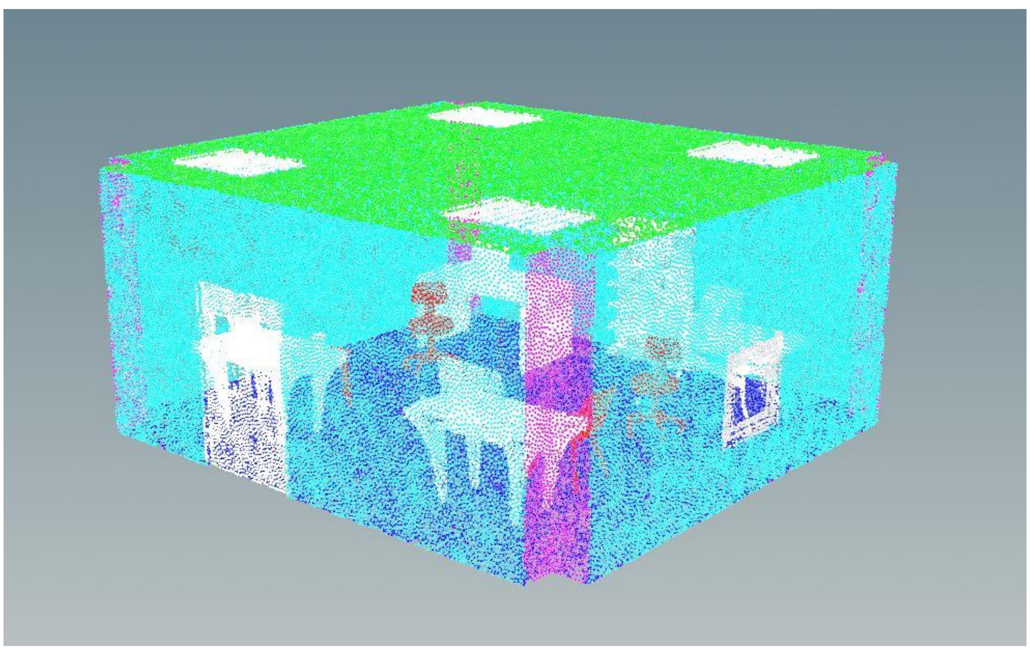

Figure 6. This is an example of the point cloud of a medium room with a point density of 500 pts/m^2 without random error.

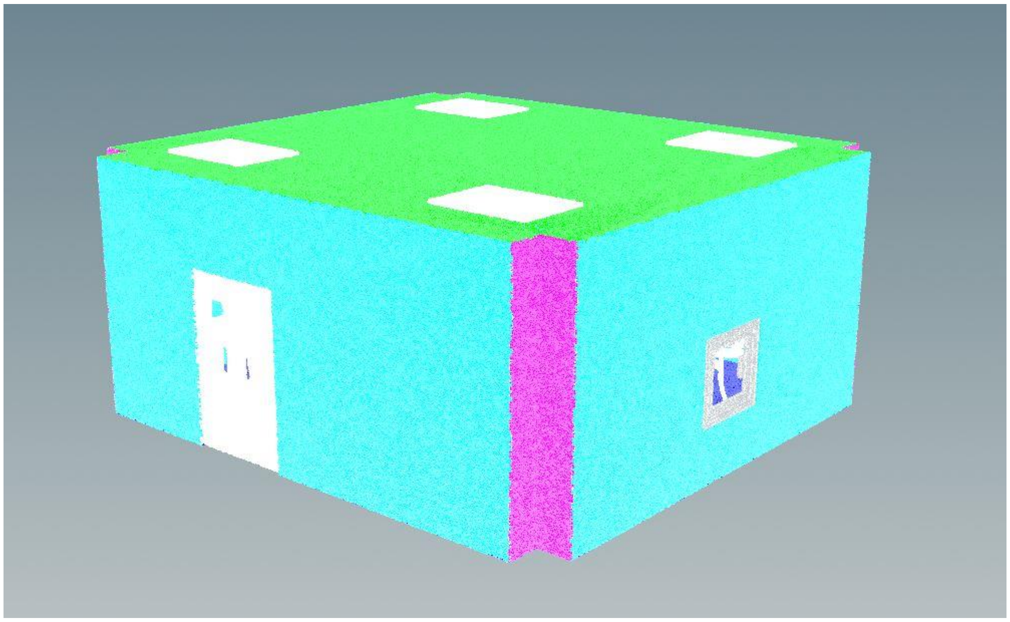

Figure 7. This is an example of the point cloud of a medium room with a point density of 5000 pts/m^2 without random error.

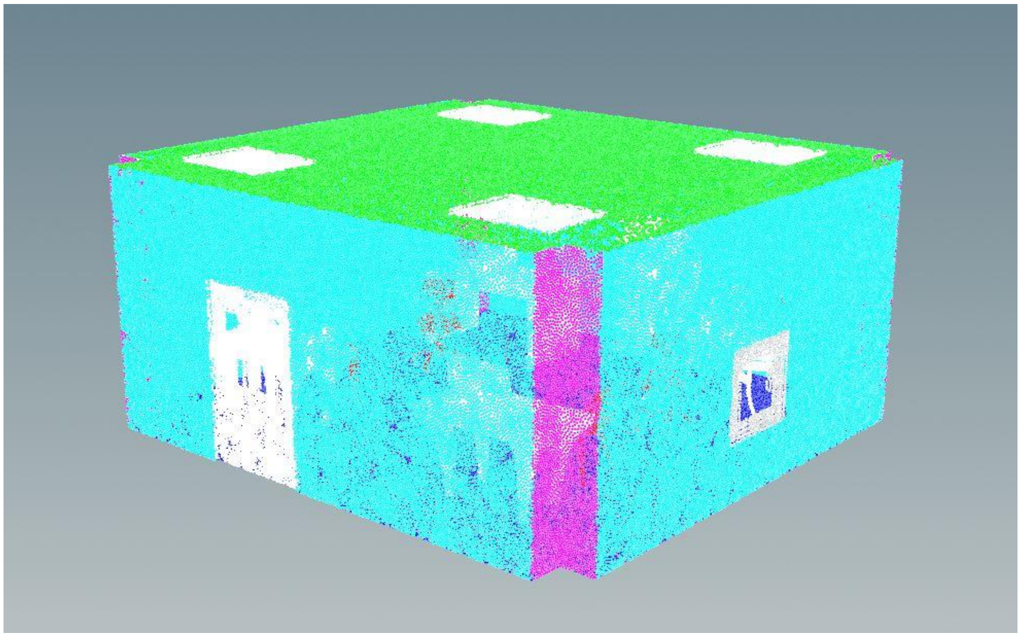

Figure 8. This is an example of the point cloud of a medium room with a point density of 1000 pts/m^2 without random error.

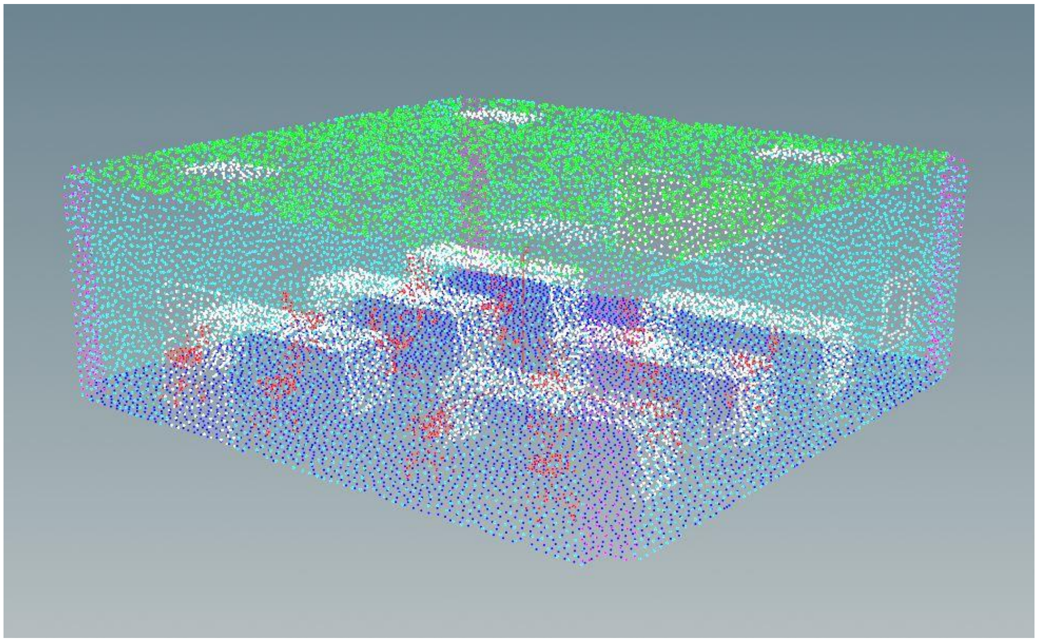

Figure 9. This is an example of the point cloud of a large room with a point density of 50 pts/m^2 without random error.

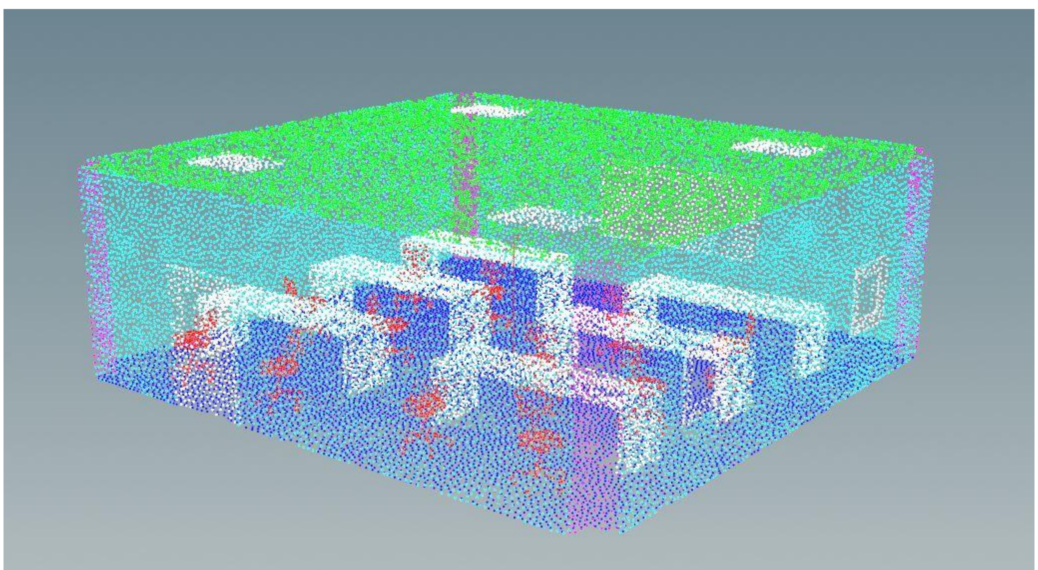

Figure 10. This is an example of the point cloud of a large room with a point density of 100 pts/m^2 without random error.

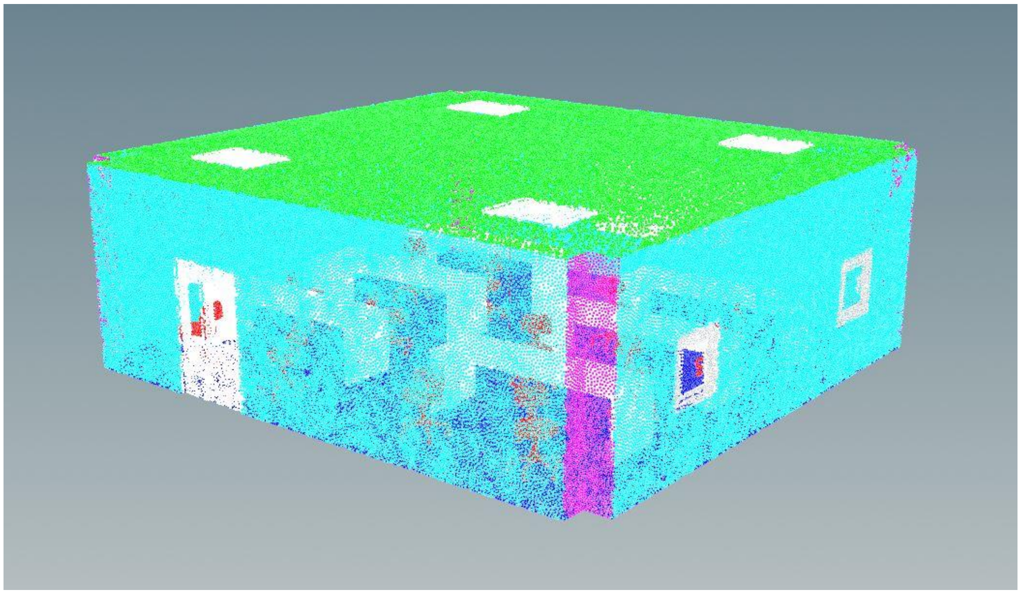

Figure 11. This is an example of the point cloud of a large room with a point density of 500 pts/m^2 without random error.

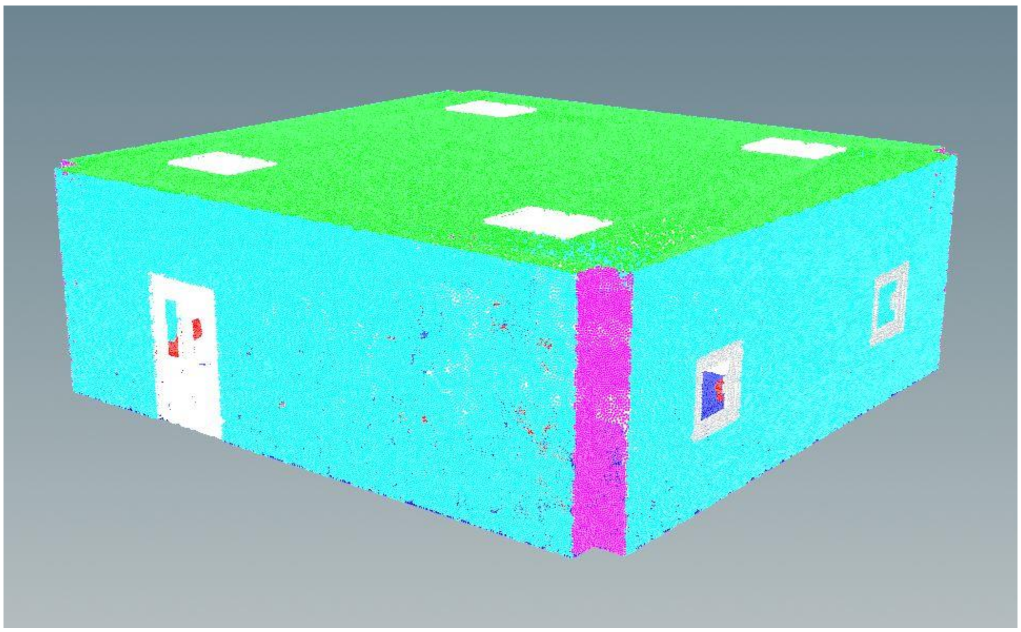

Figure 12. This is an example of the point cloud of a large room with a point density of 1000 pts/m^2 without random error.

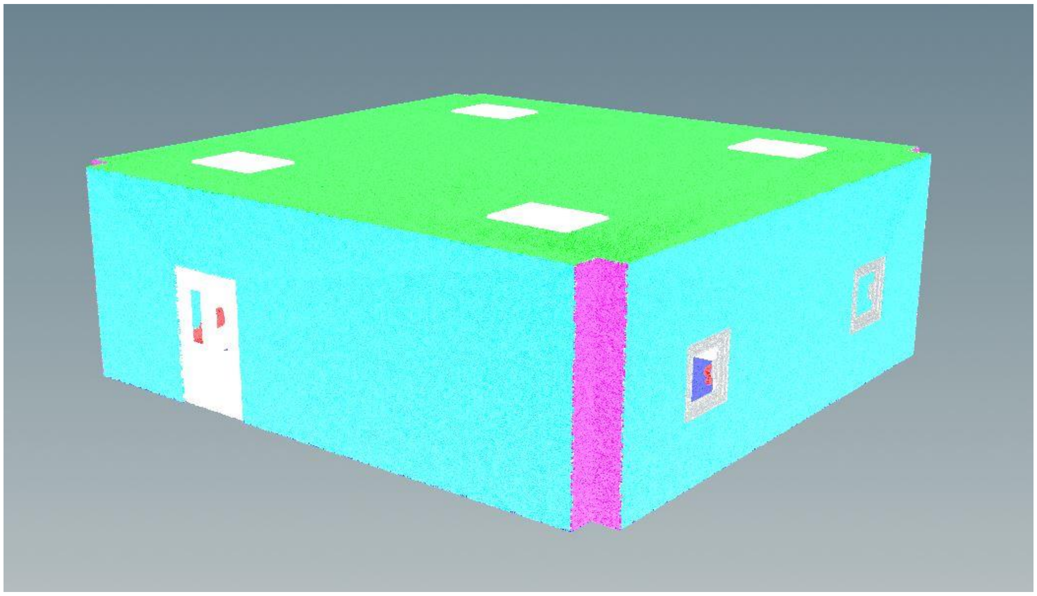

Figure 13. This is an example of the point cloud of a large room with a point density of 5000 pts/m^2 without random error.

3.1.2.4 Output the point clouds files

This study only discussed how points' positional parameters affect the semantic segmentation results. A PTS file is one typical formats of the point cloud file. It provides 7 parameters for each point, which are XYZ, intensity and RGB values. Not all laser scanners can get RGB data of scanner points and the essence of point cloud data is in the positional parameters. And photogrammetry method does not provide an intensity parameter. The intensity depends on the objects' reflection of the beam, which provides additional information about the objects. However, in order to ensure the generality and testability, this study only trained the models by position data, setting the weight of color to 0 without any intensity data.

```
0.805082 -2.40315 3 0 0 0
1.05852 -1.77485 3 0 0 0
-0.222918 -1.4655 3 0 0 0
1.54598 -0.732547 3 0 0 0
-1.56978 0.831267 3 0 0 0
0.175879 1.7608 3 0 0 0
1.3421 2.94715 3 0 0 0
-1.12023 2.21357 3 0 0 0
-0.64368 0.80496 3 0 0 0
0.986123 2.02646 3 0 0 0
-1.27476 1.07611 3 0 0 0
-0.371948 2.68082 3 0 0 0
-1.97141 -0.0599797 3 0 0 0
-2.04813 -2.9759 3 0 0 0
2.19497 0.203985 3 0 0 0
0.565183 -2.22086 3 0 0 0
-2.43552 -0.420821 3 0 0 0
```

Figure 14. An example of
point cloud file data structure.

The data structure of the point cloud data was a 6-D-parameter that consisted of point coordinates and RGB colors (which were set to (0, 0, 0)). The data files were output as TXT files. Every row consisted of 6 numbers and the number of rows is the number of points. An example data structure is listed in Figure 14.

3.1.3 Point cloud data training preparation

The TXT files contained 3D Cartesian coordinates and an RGB array. According to the original algorithm, the TXT files needed to be transformed to another file format called an NPY (NumPy array) file as an intermediate file format for further transformation. The NPY did not only serve as an intermediate file format but was used for testing. Then the NPY file needed to be transformed to the H5 (Hierarchical Data Format) file, which was the input for the segmentation and classification algorithm.

3.2 Segmentation and classification method

In this study, a machine learning algorithm, introduced by PointNet (Charles et.al., 2017), was used to perform the segmentation and classification. The algorithm was not adjusted or optimized for this study. There were 10 point cloud dataset for training and testing. Each point cloud dataset had 4 areas. Area_1, Area_2, and Area_3 were used as training data. They were input into the algorithm as H5 files. Area_4 was used as the testing data and input as NPY files.

3.2.1 Deep learning segmentation and classification framework

The PointNet (Charles et.al., 2017) is a deep learning framework that treats raw point cloud data. The algorithm performs the semantic segmentation of a 3D environment. The model outputs *n x m* scores for each of the *n* points and each of the *m* semantic subcategories (Charles et.al., 2017). The algorithm (Charles et.al., 2017) defines the input as a subset of Euclidean space points that have three main properties; unordered, interaction among points, and invariance under transformation. The input point sets are unordered unlike pixel matrix and voxel matrix, therefore, the permutation of point sets should remain the same during input. Secondly, points are not isolated but rather capable of capturing local structures from nearby points. Thirdly, points set should not be modified during any transformation and rotation (Charles et.al., 2017). "The network has three key modules: the max pooling layer as a symmetric function to aggregate information from all the points, a local and global information combination structure, and two joint alignment networks that align both input points and point features" (Charles et.al., 2017, p.3). "The classification network takes *n* points as input, applies input and feature transformations, and then aggregates point features by max pooling" (Charles et.al., 2017, p.3). The max pooling method is a down sampling process. It takes representatives of groups of objects, reducing the dimensionality of the regions. "The output is classification scores for k classes. The segmentation network is an extension to the classification net. It concatenates global and local features and outputs per point scores" (Charles et.al., 2017, p.3). According to PointNet (Charles et.al., 2017), rooms were sampled into blocks with an area of 1m by 1m. The trained segmentation method predicted per-point class in each block. "Each point is represented by a 9-dim vector of XYZ, RGB and normalized location as to the room (from 0 to 1)" (Charles et.al., 2017, p.7). In the original work, the researchers sampled 4096 points in each block in a rush during training time. And at test time, all points were tested. A K-fold strategy (Armeni et.al., 2016) was used for train and test.

3.3 Experiment design and evaluation metrics

The steps of the experiment and evaluation metrics are described in detail as follows.

In total, 10 models were trained in no particular order. The mean loss and accuracy were outputted during the training and the evaluated mean loss and the evaluated accuracy were outputted during the testing.

The first segmentation and classification model was trained by the point cloud dataset (Point density: 50 pts / m^2; Random error: None). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were calculated

The second segmentation and classification model was trained by the point cloud dataset (Point density: 100 pts / m^2; Random error: None). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The third segmentation and classification model was trained by the point cloud dataset (Point density: 500 pts / m^2; Random error: None). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The fourth segmentation and classification model was trained by the point cloud dataset (Point density: 1000 pts / m^2; Random error: None). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The fifth segmentation and classification model was trained by the point cloud dataset (Point density: 5000 pts / m^2; Random error: None). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The sixth segmentation and classification model was trained by the point cloud dataset (Point density: 50 pts / m^2; Random error: 5 mm ϵ). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The seventh segmentation and classification model was trained by the point cloud dataset (Point density: 100 pts / m^2; Random error: 5 mm ϵ). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The eighth segmentation and classification model was trained by the point cloud dataset (Point density: 500 pts / m^2; Random error: 5 mm ϵ). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The ninth segmentation and classification model was trained by the point cloud dataset (Point density: 1000 pts / m^2; Random error: 5 mm ϵ). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

The tenth segmentation and classification model was trained by the point cloud dataset (Point density: 5000 pts / m^2; Random error: 5 mm ϵ). Area_1, Area_2, and Area_3 were used as the training point clouds. The mean loss and the accuracy of the trained model were calculated, tested by Area_1, Area_2, and Area 3. Area_4 was used as the testing point cloud. The evaluated mean loss and the evaluated accuracy of the trained model were also calculated by implementing the model on Area_4.

After all the models were trained. Each model was also tested by other datasets in order to find the relation between the results and two point cloud attributes: point density and random error (with or without).

The first segmentation and classification model was trained by the point cloud dataset (Point density: 50 pts / m^2; Random error: None). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 100 pts / m^2; Random error: None. 2. Point density: 500 pts / m^2; Random error: None. 3. Point density: 1000 pts / m^2; Random error: None. 4. Point density: 5000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 50 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The second segmentation and classification model was trained by the point cloud dataset (Point density: 100 pts / m^2; Random error: None). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: None. 2. Point density: 500 pts / m^2; Random error: None. 3. Point density: 1000 pts / m^2; Random error: None. 4. Point density: 5000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 100 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The third segmentation and classification model was trained by the point cloud dataset (Point density: 500 pts / m^2; Random error: None). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: None. 2. Point density: 100 pts / m^2; Random error: None. 3. Point density: 1000 pts / m^2; Random error: None. 4. Point density: 5000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 500 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The fourth segmentation and classification model was trained by the point cloud dataset (Point density: 1000 pts / m^2; Random error: None). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: None. 2. Point density: 100 pts / m^2; Random error: None. 3. Point density: 500 pts / m^2; Random error: None. 4. Point density: 5000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 1000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The fifth segmentation and classification model was trained by the point cloud dataset (Point density: 5000 pts / m^2; Random error: None). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: None. 2. Point density: 100 pts / m^2; Random error: None. 3. Point density: 500 pts / m^2; Random error: None. 4. Point density: 1000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 5000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The sixth segmentation and classification model was trained by the point cloud dataset (Point density: 50 pts / m^2; Random error: 5 mm ϵ). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 100 pts / m^2; Random error: 5 mm ϵ. 2. Point density: 500 pts / m^2; Random error: 5 mm ϵ. 3. Point density: 1000 pts / m^2; Random error: 5 mm ϵ. 4. Point density: 5000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 50 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The seventh segmentation and classification model was trained by the point cloud dataset (Point density: 100 pts / m^2; Random error: 5 mm ϵ). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: 5 mm ϵ. 2. Point density: 500 pts / m^2; Random error: 5 mm ϵ. 3. Point density: 1000 pts / m^2; Random error: 5 mm ϵ. 4. Point density: 5000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 100 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The eighth segmentation and classification model was trained by the point cloud dataset (Point density: 500 pts / m^2; Random error: 5 mm ϵ). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: 5 mm ϵ. 2. Point density: 100 pts / m^2; Random error: 5 mm ϵ. 3. Point density: 1000 pts / m^2; Random error: 5 mm ϵ. 4. Point density: 5000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 500 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The ninth segmentation and classification model was trained by the point cloud dataset (Point density: 1000 pts / m^2; Random error: 5 mm ϵ). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: 5 mm ϵ. 2. Point density: 100 pts / m^2; Random error: 5 mm ϵ. 3. Point density: 500 pts / m^2; Random error: 5 mm ϵ. 4. Point density: 5000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 1000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

The tenth segmentation and classification model was trained by the point cloud dataset (Point density: 5000 pts / m^2; Random error: 5 mm ϵ). Firstly, it was tested by Area_4 of 4 other point cloud datasets: 1. Point density: 50 pts / m^2; Random error: 5 mm ϵ. 2. Point density: 100 pts / m^2; Random error: 5 mm ϵ. 3. Point density: 500 pts / m^2; Random error: 5 mm ϵ. 4. Point density: 1000 pts / m^2; Random error: 5 mm ϵ. The evaluated mean loss and the evaluated accuracy of the trained model were considered output and compared to find out the relation

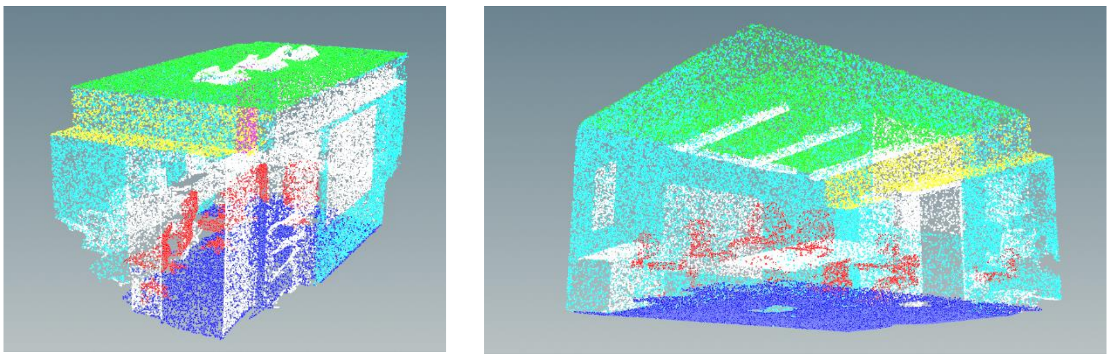

Figure 15. The figures above show the visualizations of one office and one conference room of the original data from Charles et.al (2017). Different colored points represent different clusters.

between the results and point density. Secondly, the trained model was tested by Area_4 of another point cloud dataset: Point density: 5000 pts / m^2; Random error: None. The evaluated mean loss and the evaluated accuracy of the trained model were output and compared to find out the relation between the results and the level of random error.

Table 1. The table above shows that there are 13 clusters and the colors for their representations. The names of the colors according to the RGB values are subjective and they are only for visualization purpose.

number	label	RGB	Color
1	'ceiling'	[0,255,0]	Green
2	'floor'	[0,0,255]	Blue
3	'wall'	[0,255,255]	Cyan
4	'beam'	[255,255,0]	Yellow
5	'column'	[255,0,255]	Magenta
6	'window'	[100,100,255]	Medium slate blue
7	'door'	[200,200,100]	Dark khaki
8	'table'	[170,120,200]	Medium Orchid
9	'chair'	[255,0,0]	Red
10	'sofa'	[200,100,100]	Indian red
11	'bookcase’	[10,200,100]	Medium sea green
12	'board'	[200,200,200]	Silver
13	'clutter'	[50,50,50]	Dim gray

Both quantitative and qualitative results were carried out through the study. The qualitative results were in the form of visual semantic representations as shown in Figure 15. The color of the points indicates the labels of the points. As shown in Table 1, each label was represented by an RGB array. The corresponding real-world colors were listed at the right side of the RGB values. The quantitative results were conducted based on the mean loss and the accuracy. A loss is a number indicating how bad the model's prediction is on one example. A mean loss indicates how bad the model’s prediction is on the whole dataset in average. Unlike accuracy, it is a sum of the errors made for each example in training or testing sets. An accuracy is used to measure the algorithm’s performance. It is the measure of how accurate the model's prediction is compared to the true data. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss on average, across all examples. Higher loss is worse (bad prediction) for any model. The mean loss and the accuracy were calculated based on the above-mentioned quality indicators for the evaluation of segmentation results. A low accuracy and huge loss mean that there are huge errors on a lot of the points. Low accuracy and low loss mean that there is little error on a lot of the points. A great accuracy with low mean loss that there are low errors on some data, which is the best case. The mean loss and accuracy were calculated in the experiment by the equations below:

$$Mean\ loss = \frac{\text{loss sum}}{\text{the number of batches}}$$

$$Accuracy = \frac{\text{total correct}}{\text{total seen}}$$

"Loss sum" is the summation of all losses in all the batches in a point cloud. "Total seen" means the labelled points in the point cloud, and "total correct" means the correctly labelled points in the point cloud.

The testing results of the experiments are listed in section 4.

4. EXPERIMENT RESULTS

The final results of different sets of segmentation and classification models were compared between models and within themselves. The final comparison results included the evaluated mean loss and the evaluated accuracy between models with different point density. Every model was tested by point cloud data with the same point density as the training dataset. Additionally, every model was tested by various point cloud data with different point density and different level of random errors. For example, the machine learning model trained by a point cloud dataset with the point density of 1000 pts/m^2 was tested not only by a point cloud dataset with the point density of 1000 pts/m^2, but also other point cloud datasets with different point density, e.g., 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 5000 pts/m^2.

The testing results of the machine learning models trained by different point cloud data were compared and evaluated in the following sections. In the evaluation section, in terms of point cloud density, there are two main blocks. The first block consists of the evaluation results of each trained segmentation and classification model tested by the point cloud data with the same point density as the related training point cloud data. The second block consists of the evaluation results of each trained segmentation and classification model tested by different point cloud data with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2. The evaluation of the level of error also has two main blocks. The first block consists of the evaluation results of the segmentation and classification model trained by errorless (without adding a ϵ) point cloud data. This was tested by both the errorless point cloud data and the point cloud data with adding a 5 mm ϵ. The second block consists of the evaluation results of the segmentation and classification model trained by the point cloud data with a 5 mm ϵ. It was also tested by both the errorless point cloud data and the point cloud data with adding a 5 mm ϵ.

4.1 Test results

There were two main sections of results, the first one being the mean loss and accuracy results output while training the models. They included the mean loss and accuracy tested by the training data from Area_1, Area_2 and Area_3. The second main section are the evaluated mean loss and the evaluated accuracy, tested by the testing data from Area_4. The mean loss and accuracy were the evaluation for the training dataset and the evaluated mean loss and evaluated accuracy were the evaluation for the testing dataset. The mean loss, accuracy, evaluated mean loss, and evaluated accuracy were displayed and recorded after each epoch.

The significant number of digits of the direct output mean loss and accuracy of the testing results was firstly set as 6 to make sure there was no rounding-up problem. Then the significant number of digits was chosen to be 4 because an accuracy difference less than 0.01% was ignored in this study. And according to a number of studies (Rottensteiner et.al., 2014; Charles et.al., 2017) regarding the accuracy of classification and object detection, the significant number of digits was often set below 4.

Table 2 shows the mean loss, accuracy, the evaluated mean loss, and accuracy after each epoch of 10 trained models. Table 2 shows the mean loss, accuracy and the evaluated mean loss and accuracy after each epoch of 4 point cloud datasets with different point density including: 50 pts/m^2, 50 pts/m^2 with a 5 mm ϵ, 100 pts/m^2 , 100 pts/m^2 with a 5 mm ϵ. Table 3 shows the mean loss, accuracy and the evaluated mean loss and accuracy after each epoch of 4 point cloud datasets with different point density, including 500 pts/m^2, 500 pts/m^2 with a 5 mm ϵ, 1000 pts/m^2 , 1000 pts/m^2 with a 5 mm ϵ. Table 4 shows the mean loss, accuracy and the evaluated mean loss and accuracy after each epoch of 2 point cloud datasets with different point density, including 5000 pts/m^2, 5000 pts/m^2 with a 5 mm ϵ. The max epoch was set to 20 after a few test trainings. The evaluated accuracy reached almost the maximum at the 20th epoch, and the models encountered overfitting problems when continuing training after the 20th epoch. The row indicates each epoch and the column indicates different training dataset, e.g. different point density: 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 and with or without a level of error.

Table 2. The mean loss and accuracy and the evaluated mean loss and accuracy after each epoch of 4 point cloud datasets with different point density, including 50 pts/m^2, 50 pts/m^2 with a 5 mm ϵ, 100 pts/m^2 , 100 pts/m^2 with a 5 mm ϵ.

Epoch	Output results	50 pts/m^2	50 pts/m^2 + 5 mm ϵ	100 pts/m^2	100 pts/m^2 + 5 mm ϵ
1	Mean Loss	0.353421	0.351447	0.322527	0.318329
	Accuracy	0.883021	0.883121	0.89353	0.895664
	Evaluated Mean Loss	0.247315	0.306197	0.213028	0.214336
	Evaluated Accuracy	0.918296	0.900873	0.926838	0.930647
2	Mean Loss	0.230158	0.232009	0.205491	0.205674
	Accuracy	0.920276	0.918816	0.929793	0.929973
	Evaluated Mean Loss	0.189294	0.234032	0.155536	0.169215
	Evaluated Accuracy	0.929138	0.915078	0.94389	0.936886
3	Mean Loss	0.177794	0.180044	0.154271	0.153984
	Accuracy	0.935933	0.934947	0.945277	0.944527
	Evaluated Mean Loss	0.155587	0.15518	0.14624	0.121257
	Evaluated Accuracy	0.946072	0.94154	0.947472	0.958715
4	Mean Loss	0.151212	0.152259	0.127718	0.130696
	Accuracy	0.945373	0.944866	0.954633	0.953296
	Evaluated Mean Loss	0.153832	0.149322	0.112793	0.111294
	Evaluated Accuracy	0.9442	0.946953	0.958999	0.958979
5	Mean Loss	0.139666	0.135508	0.117515	0.113883
	Accuracy	0.949258	0.950409	0.958244	0.959203
	Evaluated Mean Loss	0.129746	0.132832	0.10007	0.107652
	Evaluated Accuracy	0.953081	0.953698	0.964247	0.960649
6	Mean Loss	0.129794	0.127802	0.108363	0.103931
	Accuracy	0.952503	0.95338	0.961308	0.962603
	Evaluated Mean Loss	0.123052	0.117849	0.114584	0.097472
	Evaluated Accuracy	0.95576	0.955819	0.958108	0.963151
7	Mean Loss	0.116004	0.117317	0.099861	0.101498
	Accuracy	0.957051	0.956778	0.964086	0.963206
	Evaluated Mean Loss	0.124417	0.112198	0.080958	0.084177
	Evaluated Accuracy	0.955611	0.957806	0.971411	0.967069
8	Mean Loss	0.111863	0.111683	0.091879	0.092293
	Accuracy	0.958662	0.958514	0.966342	0.966023
	Evaluated Mean Loss	0.117916	0.132865	0.08766	0.103311
	Evaluated Accuracy	0.958231	0.954333	0.96823	0.960821
9	Mean Loss	0.10844	0.103638	0.085603	0.084884
	Accuracy	0.959373	0.961038	0.968653	0.96884
	Evaluated Mean Loss	0.10793	0.089097	0.07681	0.075172
	Evaluated Accuracy	0.961222	0.96538	0.971158	0.973208
10	Mean Loss	0.102084	0.100311	0.084488	0.084456
	Accuracy	0.961469	0.962328	0.969053	0.968694
	Evaluated Mean Loss	0.11477	0.101631	0.074044	0.083601
	Evaluated Accuracy	0.957926	0.962232	0.971688	0.970065
11	Mean Loss	0.097816	0.095114	0.08042	0.077869
	Accuracy	0.962868	0.963992	0.970287	0.970953

Table 2 continued

	Evaluated Mean Loss	0.093181	0.089985	0.069682	0.077972
	Evaluated Accuracy	0.965304	0.967054	0.973995	0.971252
12	Mean Loss	0.092095	0.091487	0.063458	0.065321
	Accuracy	0.964932	0.964986	0.976208	0.9754
	Evaluated Mean Loss	0.087163	0.113405	0.070034	0.058057
	Evaluated Accuracy	0.967019	0.956953	0.971925	0.977335
13	Mean Loss	0.084699	0.079027	0.059953	0.059397
	Accuracy	0.967745	0.969595	0.97727	0.977399
	Evaluated Mean Loss	0.072922	0.069767	0.082149	0.056318
	Evaluated Accuracy	0.972268	0.972772	0.967552	0.977644
14	Mean Loss	0.071268	0.071761	0.057933	0.05699
	Accuracy	0.972325	0.971974	0.977832	0.978156
	Evaluated Mean Loss	0.084506	0.073257	0.068777	0.059953
	Evaluated Accuracy	0.965127	0.971301	0.972128	0.976957
15	Mean Loss	0.070647	0.069794	0.056697	0.057823
	Accuracy	0.972401	0.972657	0.97842	0.977929
	Evaluated Mean Loss	0.065307	0.067228	0.054594	0.056667
	Evaluated Accuracy	0.974773	0.972742	0.978989	0.979073
16	Mean Loss	0.069319	0.067953	0.055921	0.055845
	Accuracy	0.972943	0.973159	0.978425	0.978748
	Evaluated Mean Loss	0.066397	0.072759	0.056957	0.050605
	Evaluated Accuracy	0.973948	0.971896	0.9784	0.980879
17	Mean Loss	0.068161	0.066731	0.052686	0.053446
	Accuracy	0.97319	0.973596	0.979661	0.97931
	Evaluated Mean Loss	0.064598	0.074008	0.055509	0.049538
	Evaluated Accuracy	0.974523	0.972008	0.978034	0.980496
18	Mean Loss	0.06584	0.064968	0.05312	0.052224
	Accuracy	0.974026	0.974252	0.979579	0.979812
	Evaluated Mean Loss	0.073284	0.083867	0.051588	0.049362
	Evaluated Accuracy	0.972696	0.967046	0.980624	0.981041
19	Mean Loss	0.064988	0.065183	0.052211	0.052063
	Accuracy	0.974232	0.974056	0.979726	0.979884
	Evaluated Mean Loss	0.069003	0.064316	0.048689	0.055883
	Evaluated Accuracy	0.971253	0.974045	0.980635	0.978931
20	Mean Loss	0.063948	0.062526	0.051328	0.051014
	Accuracy	0.974649	0.975102	0.979938	0.980132
	Evaluated Mean Loss	0.062197	0.064441	0.056112	0.059853
	Evaluated Accuracy	0.974567	0.974841	0.977507	0.977121

Table 3. The mean loss and accuracy and the evaluated mean loss and accuracy after each epoch of 4 point cloud datasets with different point density, including 500 pts/m^2, 500 pts/m^2 with a 5 mm ϵ, 1000 pts/m^2, 1000 pts/m^2 with a 5 mm ϵ.

Epoch	Output results	500 pts/m^2	500 pts/m^2 + 5 mm ϵ	1000 pts/m^2	1000 pts/m^2 + 5 mm ϵ
1	Mean Loss	0.297174	0.303645	0.297578	0.301577
	Accuracy	0.900685	0.899548	0.902515	0.900537
	Evaluated Mean Loss	0.194418	0.268461	0.197965	0.276726
	Evaluated Accuracy	0.931482	0.916452	0.934131	0.907685
2	Mean Loss	0.168584	0.182759	0.17457	0.171347
	Accuracy	0.941285	0.936711	0.939125	0.9401
	Evaluated Mean Loss	0.1186	0.127619	0.132449	0.139708
	Evaluated Accuracy	0.959136	0.954162	0.951584	0.94996
3	Mean Loss	0.127361	0.128535	0.125747	0.125688
	Accuracy	0.956043	0.955551	0.956224	0.956903
	Evaluated Mean Loss	0.111535	0.139232	0.106608	0.214224
	Evaluated Accuracy	0.960473	0.949231	0.964799	0.929882
4	Mean Loss	0.109523	0.111471	0.110453	0.107712
	Accuracy	0.962109	0.961666	0.961973	0.963676
	Evaluated Mean Loss	0.099124	0.105201	0.089203	0.133182
	Evaluated Accuracy	0.965788	0.962129	0.968528	0.958725
5	Mean Loss	0.096141	0.096614	0.09743	0.096246
	Accuracy	0.966515	0.966902	0.966505	0.966993
	Evaluated Mean Loss	0.089089	0.080304	0.083741	0.087072
	Evaluated Accuracy	0.968669	0.97175	0.971362	0.968962
6	Mean Loss	0.089295	0.092075	0.090533	0.087021
	Accuracy	0.969024	0.968351	0.968867	0.97015
	Evaluated Mean Loss	0.083337	0.091896	0.077138	0.07777
	Evaluated Accuracy	0.971589	0.966539	0.971524	0.97189
7	Mean Loss	0.081917	0.082339	0.081026	0.083026
	Accuracy	0.971416	0.971346	0.971832	0.971547
	Evaluated Mean Loss	0.075724	0.08883	0.076394	0.092806
	Evaluated Accuracy	0.971444	0.968368	0.972035	0.970118
8	Mean Loss	0.079046	0.080439	0.079766	0.077114
	Accuracy	0.972356	0.972104	0.972389	0.973387
	Evaluated Mean Loss	0.068929	0.071842	0.095179	0.089052
	Evaluated Accuracy	0.974354	0.973464	0.964935	0.9705
9	Mean Loss	0.075709	0.073839	0.07424	0.071748
	Accuracy	0.973541	0.974303	0.974069	0.975323
	Evaluated Mean Loss	0.061322	0.071473	0.06944	0.064339
	Evaluated Accuracy	0.97819	0.974701	0.975518	0.976885

Table 3 continued

10	Mean Loss	0.071788	0.068061	0.069007	0.068636
	Accuracy	0.974842	0.975738	0.975716	0.97622
	Evaluated Mean Loss	0.0622	0.071444	0.075539	0.075116
	Evaluated Accuracy	0.977936	0.97677	0.973274	0.971991
11	Mean Loss	0.066244	0.067653	0.064316	0.068538
	Accuracy	0.976476	0.976178	0.977233	0.976283
	Evaluated Mean Loss	0.064792	0.065311	0.060886	0.058117
	Evaluated Accuracy	0.975537	0.975419	0.977743	0.978361
12	Mean Loss	0.051989	0.052615	0.051723	0.050885
	Accuracy	0.981325	0.981188	0.981675	0.98211
	Evaluated Mean Loss	0.052723	0.048964	0.046934	0.07427
	Evaluated Accuracy	0.981111	0.98221	0.982803	0.975589
13	Mean Loss	0.051436	0.050624	0.049614	0.048451
	Accuracy	0.98154	0.982005	0.982389	0.98294
	Evaluated Mean Loss	0.049166	0.055845	0.046155	0.046692
	Evaluated Accuracy	0.982335	0.979288	0.983009	0.983478
14	Mean Loss	0.048291	0.04742	0.046998	0.046034
	Accuracy	0.982625	0.983076	0.983374	0.983698
	Evaluated Mean Loss	0.05251	0.052028	0.044787	0.048849
	Evaluated Accuracy	0.981211	0.981516	0.983717	0.9825
15	Mean Loss	0.047606	0.046691	0.045391	0.044337
	Accuracy	0.982872	0.98326	0.983767	0.984409
	Evaluated Mean Loss	0.049546	0.043595	0.04299	0.041847
	Evaluated Accuracy	0.983073	0.98421	0.983956	0.985165
16	Mean Loss	0.046771	0.045752	0.045356	0.044769
	Accuracy	0.983093	0.983638	0.98388	0.984248
	Evaluated Mean Loss	0.061442	0.045723	0.038632	0.040321
	Evaluated Accuracy	0.97635	0.983325	0.986034	0.985141
17	Mean Loss	0.044676	0.043237	0.042998	0.041975
	Accuracy	0.983807	0.984343	0.984658	0.985132
	Evaluated Mean Loss	0.039383	0.04155	0.038394	0.03958
	Evaluated Accuracy	0.985937	0.98517	0.986481	0.985327
18	Mean Loss	0.043496	0.043701	0.042218	0.042434
	Accuracy	0.984163	0.984243	0.9848	0.984949
	Evaluated Mean Loss	0.042463	0.042805	0.043643	0.046702
	Evaluated Accuracy	0.984672	0.984297	0.984083	0.983064
19	Mean Loss	0.041323	0.043912	0.039527	0.040293
	Accuracy	0.984975	0.984016	0.985887	0.985739
	Evaluated Mean Loss	0.043284	0.040643	0.038502	0.053898
	Evaluated Accuracy	0.984341	0.985182	0.985861	0.980961

Table 3 continued

20	Mean Loss	0.041898	0.041915	0.040292	0.040039
	Accuracy	0.984685	0.984896	0.98561	0.985871
	Evaluated Mean Loss	0.04046	0.039649	0.038686	0.041403
	Evaluated Accuracy	0.985244	0.98564	0.985451	0.985603

Table 4. The mean loss and accuracy and the evaluated mean loss and accuracy after each epoch of 2 point cloud datasets with different point density, including 5000 pts/m^2, 5000 pts/m^2 with a 5 mm ϵ.

Epoch	Output results	5000 pts/m^2	5000 pts/m^2 + 5 mm ϵ
1	Mean Loss	0.292543	0.298256
	Accuracy	0.903753	0.901926
	Evaluated Mean Loss	0.171619	0.177342
	Evaluated Accuracy	0.938807	0.937808
2	Mean Loss	0.161389	0.166703
	Accuracy	0.943256	0.941365
	Evaluated Mean Loss	0.132767	0.114456
	Evaluated Accuracy	0.951304	0.95862
3	Mean Loss	0.120138	0.122044
	Accuracy	0.959023	0.957856
	Evaluated Mean Loss	0.137915	0.089014
	Evaluated Accuracy	0.952152	0.969447
4	Mean Loss	0.103122	0.105161
	Accuracy	0.964851	0.963862
	Evaluated Mean Loss	0.097819	0.094388
	Evaluated Accuracy	0.966996	0.967649
5	Mean Loss	0.090989	0.093293
	Accuracy	0.968882	0.968024
	Evaluated Mean Loss	0.094929	0.110131
	Evaluated Accuracy	0.966466	0.959791
6	Mean Loss	0.082333	0.081687
	Accuracy	0.971783	0.972059
	Evaluated Mean Loss	0.080264	0.083923
	Evaluated Accuracy	0.971981	0.971254
7	Mean Loss	0.076931	0.075969
	Accuracy	0.973694	0.974
	Evaluated Mean Loss	0.071581	0.077376
	Evaluated Accuracy	0.974826	0.971384

Table 4 continued

8	Mean Loss	0.072282	0.070137
	Accuracy	0.975416	0.975723
	Evaluated Mean Loss	0.059724	0.066408
	Evaluated Accuracy	0.978911	0.975398
9	Mean Loss	0.066541	0.070066
	Accuracy	0.976991	0.976014
	Evaluated Mean Loss	0.055188	0.066955
	Evaluated Accuracy	0.980927	0.974398
10	Mean Loss	0.062587	0.064531
	Accuracy	0.978427	0.977407
	Evaluated Mean Loss	0.161899	0.062519
	Evaluated Accuracy	0.954057	0.978164
11	Mean Loss	0.059561	0.062548
	Accuracy	0.979635	0.978513
	Evaluated Mean Loss	0.058576	0.060378
	Evaluated Accuracy	0.979366	0.978056
12	Mean Loss	0.046365	0.046363
	Accuracy	0.983959	0.984094
	Evaluated Mean Loss	0.046569	0.04339
	Evaluated Accuracy	0.981454	0.984404
13	Mean Loss	0.044048	0.044153
	Accuracy	0.984665	0.984631
	Evaluated Mean Loss	0.044907	0.049328
	Evaluated Accuracy	0.98306	0.982964
14	Mean Loss	0.042003	0.042403
	Accuracy	0.985436	0.985399
	Evaluated Mean Loss	0.036292	0.037274
	Evaluated Accuracy	0.987435	0.986649
15	Mean Loss	0.039826	0.039719
	Accuracy	0.986144	0.986312
	Evaluated Mean Loss	0.038393	0.035577
	Evaluated Accuracy	0.985941	0.987498
16	Mean Loss	0.039102	0.038933
	Accuracy	0.986485	0.986481
	Evaluated Mean Loss	0.037033	0.045101
	Evaluated Accuracy	0.985637	0.985833
17	Mean Loss	0.037525	0.037731
	Accuracy	0.987009	0.986952
	Evaluated Mean Loss	0.035665	0.033036
	Evaluated Accuracy	0.987236	0.988014

Table 4 continued

18	Mean Loss	0.037904	0.036682
	Accuracy	0.986881	0.987241
	Evaluated Mean Loss	0.037622	0.037177
	Evaluated Accuracy	0.986627	0.986765
19	Mean Loss	0.036929	0.035995
	Accuracy	0.987192	0.987483
	Evaluated Mean Loss	0.03382	0.039089
	Evaluated Accuracy	0.988071	0.986409
20	Mean Loss	0.034655	0.034986
	Accuracy	0.987951	0.987823
	Evaluated Mean Loss	0.030144	0.038553
	Evaluated Accuracy	0.989452	0.986497

Table 5 shows the evaluated mean loss and accuracy of each model (without random error) tested by different point density datasets. For example, the third column shows the evaluated mean loss and accuracy of the model trained by a 50 pts/m^2 point density point cloud without error, tested by other 4 different point densities (without random error) datasets.

Table 5. The evaluated mean loss and accuracy of each model (without random error) tested by different point density datasets.

Testing Dataset	Output results	50 pts/m^2	100 pts/m^2	500 pts/m^2	1000 pts/m^2	5000 pts/m^2
50 pts/m^2	Evaluated mean loss	0.062197	0.435696	0.336238	0.540385	0.448696
	Evaluated accuracy	0.974567	0.900721	0.910607	0.892106	0.890980
100 pts/m^2	Evaluated mean loss	0.206970	0.056112	0.250923	0.432514	0.366324
	Evaluated accuracy	0.931166	0.977507	0.929784	0.912553	0.909621
500 pts/m^2	Evaluated mean loss	0.211475	0.372242	0.040460	0.391709	0.346514
	Evaluated accuracy	0.931920	0.919755	0.985244	0.921949	0.917268
1000 pts/m^2	Evaluated mean loss	0.214208	0.371572	0.217392	0.038686	0.334647
	Evaluated accuracy	0.931725	0.920620	0.940598	0.985451	0.918455
5000 pts/m^2	Evaluated mean loss	0.208316	0.369093	0.212643	0.379705	0.030144
	Evaluated accuracy	0.933770	0.922644	0.942198	0.925458	0.989452

Table 6 shows the evaluated mean loss and accuracy of each model (with 5 mm ϵ) tested by different point density datasets (with 5 mm ϵ). The ϵ was introduced by adding random shift to each point, which resulted in 5 mm ϵ for the whole point cloud dataset. For example, one cross-section shows the evaluated mean loss and accuracy of the model trained by a 50 pts/m^2 point density point cloud with a 5 mm ϵ, tested by other 4 different point densities (with a 5 mm ϵ).

Table 6. The evaluated mean loss and accuracy of each model (with 5 mm ϵ) tested by different point density datasets (with 5 mm ϵ).

Testing Dataset	Output results	50 pts/m^2 with 5 mm ϵ	100 pts/m^2 with 5 mm ϵ	500 pts/m^2 with 5 mm ϵ	1000 pts/m^2 with 5 mm ϵ	5000 pts/m^2 with 5 mm ϵ
50 pts/m^2 with 5 mm ϵ	Evaluated mean loss	0.064441	0.390551	0.319484	0.301836	0.322617
	Evaluated accuracy	0.974841	0.896887	0.902084	0.903745	0.909698
100 pts/m^2 with 5 mm ϵ	Evaluated mean loss	0.204180	0.059853	0.252670	0.241096	0.251216
	Evaluated accuracy	0.939751	0.977121	0.919383	0.919286	0.927504
500 pts/m^2 with 5 mm ϵ	Evaluated mean loss	0.198494	0.335963	0.039649	0.222498	0.224136
	Evaluated accuracy	0.943256	0.911242	0.985640	0.926883	0.936294
1000 pts/m^2 with 5 mm ϵ	Evaluated mean loss	0.197642	0.336379	0.212713	0.041403	0.217927
	Evaluated accuracy	0.943995	0.911218	0.931097	0.985603	0.938086
5000 pts/m^2 with 5 mm ϵ	Evaluated mean loss	0.195393	0.335256	0.207605	0.212298	0.0385530
	Evaluated accuracy	0.945306	0.912575	0.932562	0.929686	0.986497

Table 7~11 show the comparison between 5 paired models. In the paired models, one of the models was trained by a point cloud dataset without error, and the other one was trained by a point cloud dataset with a 5 mm ϵ. Each model was tested by two datasets, one of them was the same point density and the same level of error (no random error or a 5 mm ϵ), and the other one was the same point density but a different level of error. Table 7 shows the evaluated mean loss and the evaluated accuracy of point clouds with point densities of 50 pts/m^2 and 50 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 50 pts/m^2 + 5 mm ϵ and 50 pts/m^2. Table 8 shows the evaluated mean loss and the evaluated accuracy of point clouds with point densities of 100 pts/m^2 and 100 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 100 pts/m^2 +

5 mm ϵ and 100 pts/m^2. Table 9 shows the evaluated mean loss and the evaluated accuracy of point clouds with point densities of 500 pts/m^2 and 500 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 500 pts/m^2 + 5 mm ϵ and 500 pts/m^2. Table 10 shows the evaluated mean loss and the evaluated accuracy of point clouds with point densities of 1000 pts/m^2 and 1000 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 1000 pts/m^2 + 5 mm ϵ and 1000 pts/m^2. Table shows the evaluated mean loss and the evaluated accuracy of point clouds with point densities of 5000 pts/m^2 and 5000 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 5000 pts/m^2 + 5 mm ϵ and 5000 pts/m^2.

Table 7. The evaluated mean loss and the evaluated accuracy of point clouds with point densities of 50 pts/m^2 and 50 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 50 pts/m^2 + 5 mm ϵ and 50 pts/m^2.

Testing Dataset	Output results	50 pts/m^2	50 pts/m^2 + 5 mm ϵ
50 pts/m^2	Evaluated mean loss	0.0621970	0.231289
	Evaluated accuracy	0.974567	0.930419
50 pts/m^2 + 5 mm ϵ	Evaluated mean loss	0.226524	0.0644410
	Evaluated accuracy	0.924967	0.974841

Table 8. The evaluated mean loss and the evaluated accuracy of point clouds with point densities of 100 pts/m^2 and 100 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 100 pts/m^2 + 5 mm ϵ and 100 pts/m^2.

Testing dataset	Output results	100 pts/m^2	100 pts/m^2 + 5 mm ϵ
100 pts/m^2	Evaluated mean loss	0.0561120	0.335085
	Evaluated accuracy	0.977507	0.909702
100 pts/m^2 + 5 mm ϵ	Evaluated mean loss	0.376403	0.0598530
	Evaluated accuracy	0.915965	0.977121

Table 9. The evaluated mean loss and the evaluated accuracy of point clouds with point densities of 500 pts/m^2 and 500 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 500 pts/m^2 + 5 mm ϵ and 500 pts/m^2.

Testing dataset	Output results	500 pts/m^2	500 pts/m^2 + 5 mm ϵ
500 pts/m^2	Evaluated mean loss	0.0404600	0.215862
	Evaluated accuracy	0.985244	0.929615
500 pts/m^2 + 5 mm ϵ	Evaluated mean loss	0.229342	0.039649
	Evaluated accuracy	0.939018	0.985640

Table 10. The evaluated mean loss and the evaluated accuracy of point clouds with point densities of 1000 pts/m^2 and 1000 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 1000 pts/m^2 + 5 mm ϵ and 1000 pts/m^2.

Testing dataset	Output results	1000 pts/m^2	1000 pts/m^2 + 5 mm ϵ
1000 pts/m^2	Evaluated mean loss	0.0386860	0.219362
	Evaluated accuracy	0.985451	0.927673
1000 pts/m^2 + 5 mm ϵ	Evaluated mean loss	0.383003	0.0414030
	Evaluated accuracy	0.923896	0.985603

Table 11. The evaluated mean loss and the evaluated accuracy of point clouds with point densities of 5000 pts/m^2 and 5000 pts/m^2 + 5 mm ϵ, tested by point clouds with point densities of 5000 pts/m^2 + 5 mm ϵ and 5000 pts/m^2.

Testing dataset	Output results	5000 pts/m^2	5000 pts/m^2 + 5 mm ϵ
5000 pts/m^2	Evaluated mean loss	0.0301440	0.212809
	Evaluated accuracy	0.989452	0.939539
5000 pts/m^2 + 5 mm ϵ	Evaluated mean loss	0.328395	0.0385530
	Evaluated accuracy	0.920347	0.986497

4.2 Results analyses

Quantitative analyses were applied on the test results as well as qualitative analyses thru visualization. Analyses were conducted in two sections, including the relation between the results and point density, and the relation between the results and the level of random error. The model stored itself every 10 iterations. This was a pre-test to determine the fixed iteration for all the models, as shown in figure 16. The maximum iteration time was set at 20 iteration for every model, because this study aimed at the accuracy of the models at a certain iteration time (before

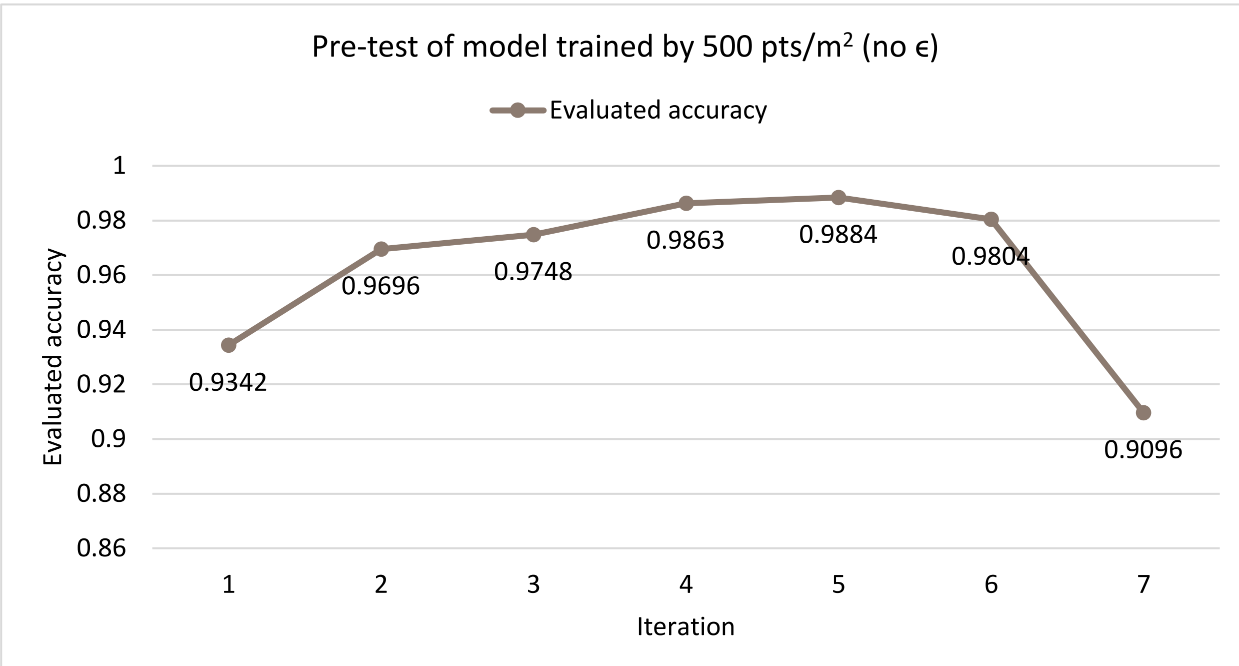

Figure 16. This figure shows the evaluated accuracy of the model after different iteration: 1. 1st iteration, 2. 5th iteration, 3. 10th iteration, 4. 20th iteration, 5. 30th iteration, 6. 40th iteration, 7. 50th iteration.

overfitting) to eliminate the effect of the training time on the performance. There is no need for 10 more iteration once the model reached 20 iterations. Because the increase of the accuracy was not worth of adding 10 more iteration which took much longer time (depending on the computation power the computer has).

Figure 17, Figure 18, Figure 19, Figure 20, Figure 21 display the evaluated accuracy of the point cloud with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000

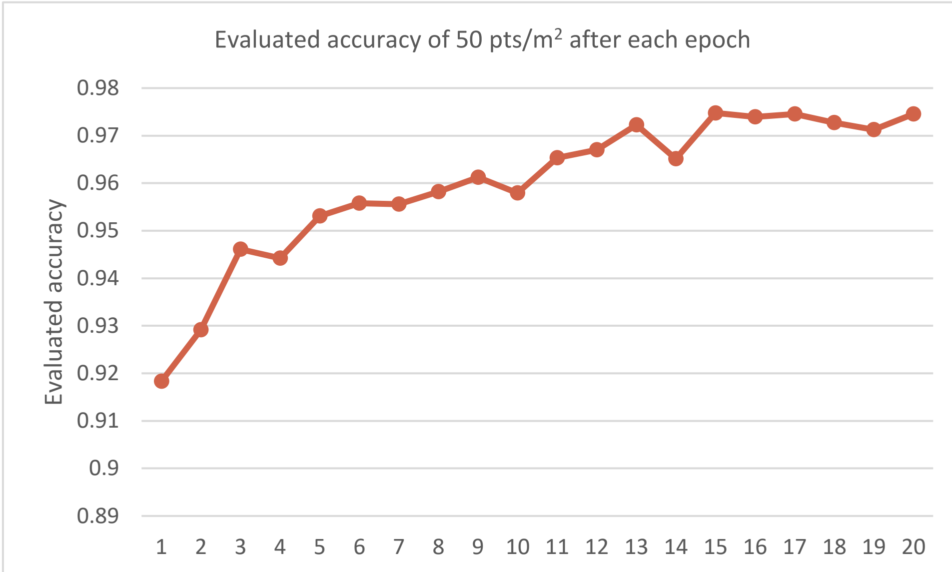

Figure 17. This shows the evaluated accuracy of the point cloud with a point density of 50 pts/m^2 (no random error) after each epoch.

pts/m^2 (all point clouds with no random error) after each epoch. Figure 22, Figure 23, Figure 24, Figure 25, Figure 26 display the evaluated accuracy of the point cloud with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 (all point clouds with a 5 mm ϵ) after each epoch. Figure 27, Figure 28, Figure 29, Figure 30, Figure 31 display the evaluated

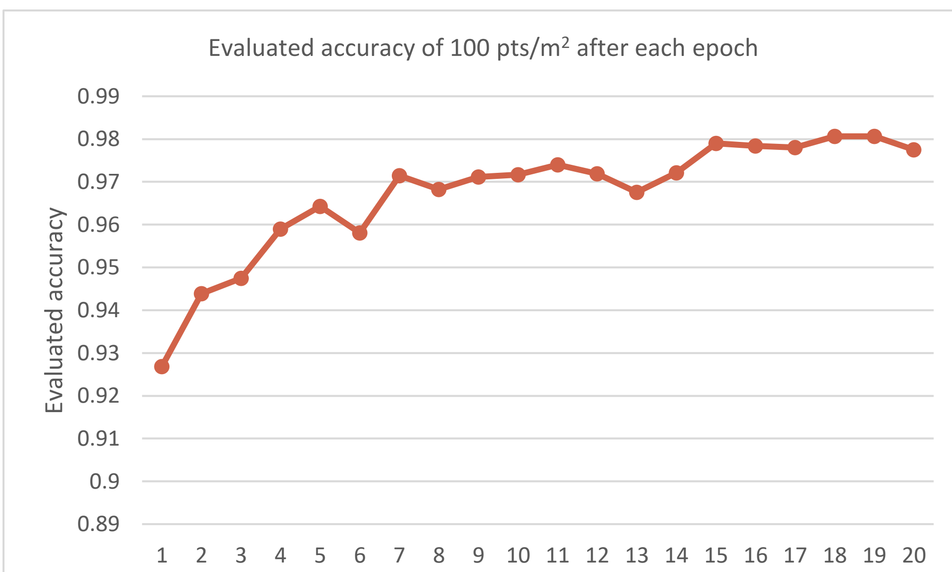

Figure 18. This shows the evaluated accuracy of the point cloud with a point density of 100 pts/m^2 (no random error) after each epoch.

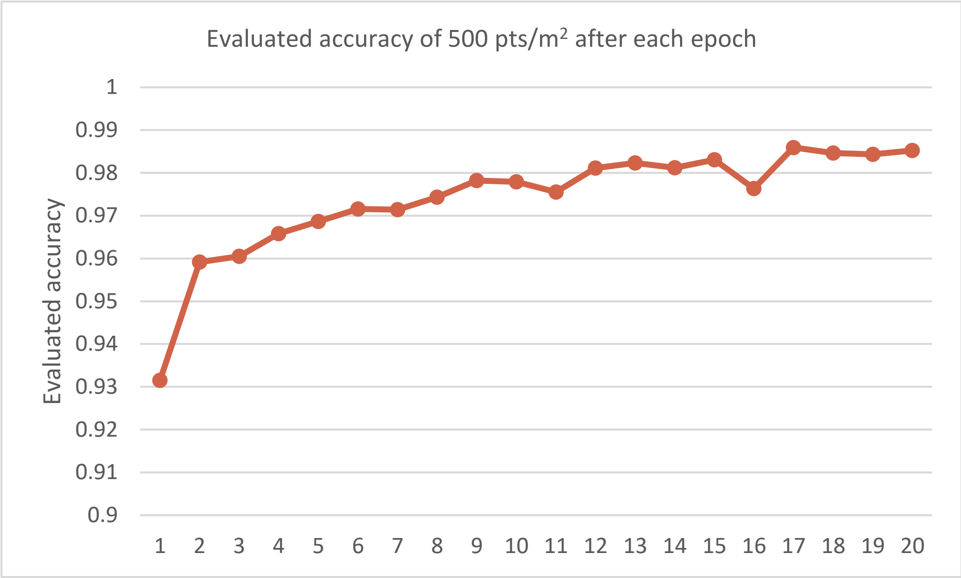

Figure 19. This shows the evaluated accuracy of the point cloud with a point density of 500 pts/m^2 (no random error) after each epoch.

mean loss of the point cloud with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 (all point clouds with no random error) after each epoch. Figure 32, Figure 33, Figure 34, Figure 35, Figure 36 display the evaluated mean loss of the point cloud with the

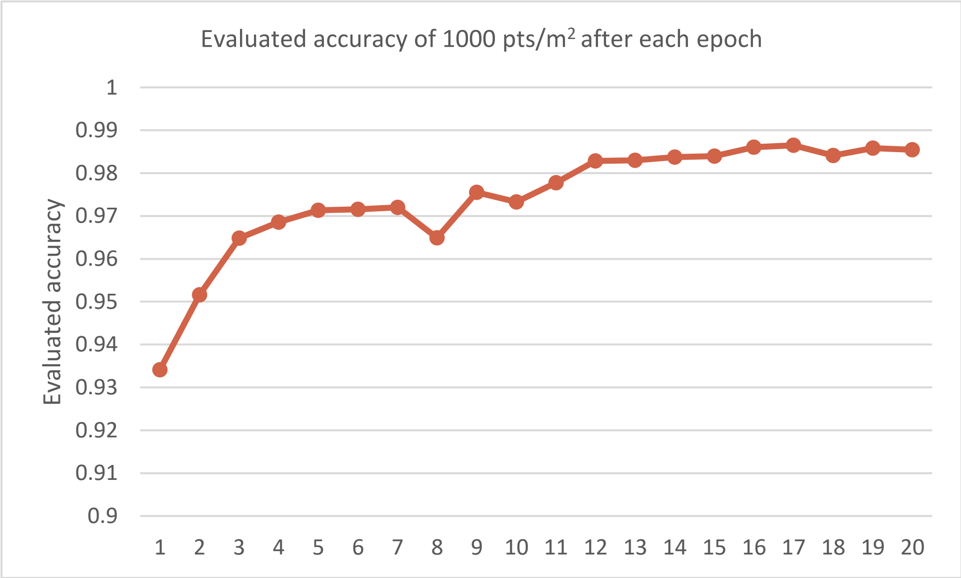

Figure 20. This shows the evaluated accuracy of the point cloud with a point density of 1000 pts/m^2 (no random error) after each epoch.

point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 (all point clouds with a 5 mm ϵ) after each epoch.

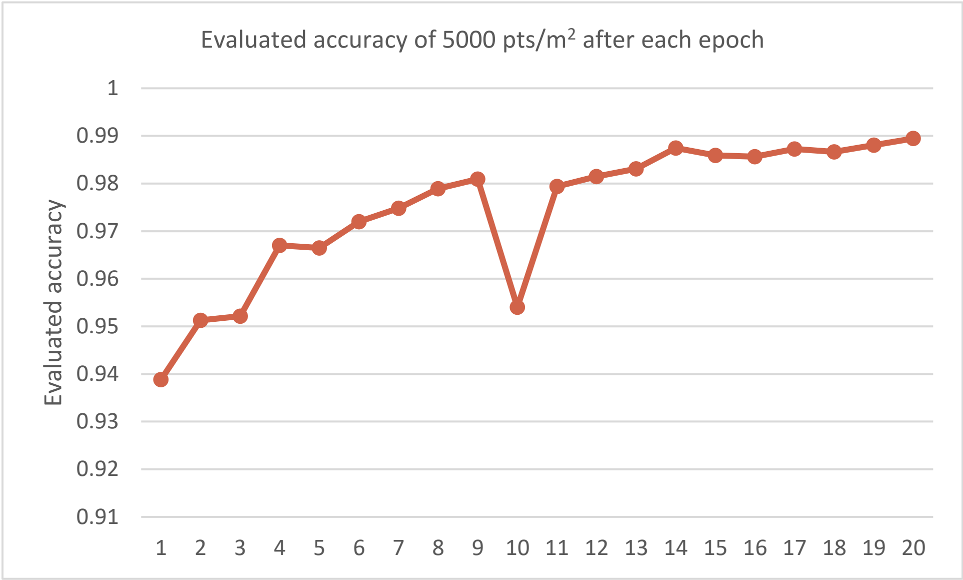

Figure 21. This shows the evaluated accuracy of the point cloud with a point density of 5000 pts/m^2 (no random error) after each epoch.

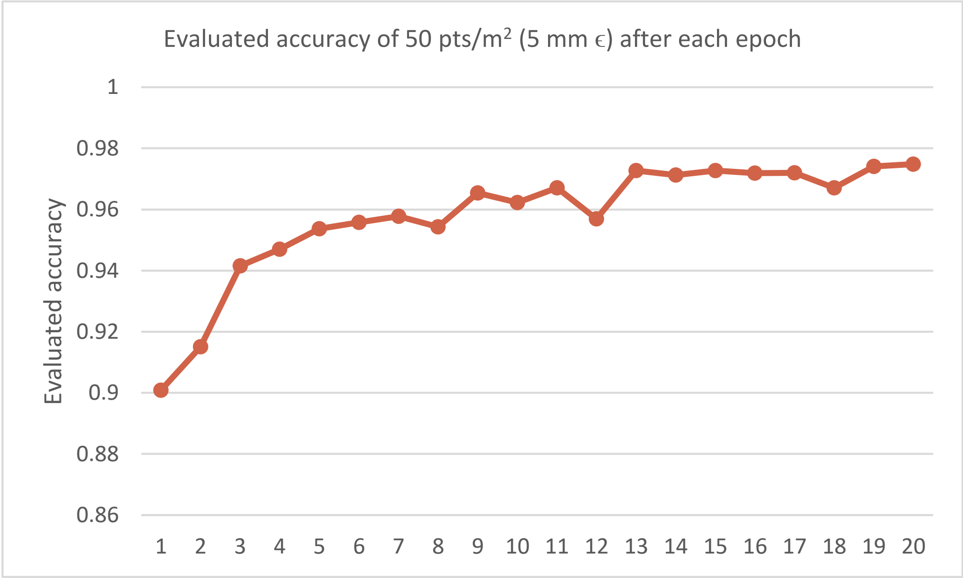

Figure 22. This shows the evaluated accuracy of the point cloud with a point density of 50 pts/m^2 (5 mm ϵ) after each epoch.

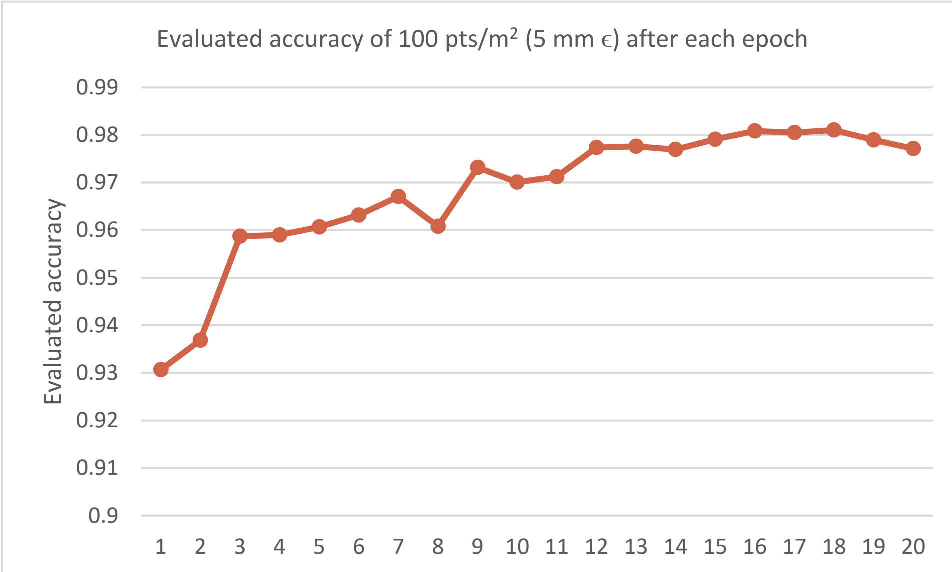

Figure 23. This shows the evaluated accuracy of the point cloud with a point density of 100 pts/m^2 (5 mm ϵ) after each epoch.

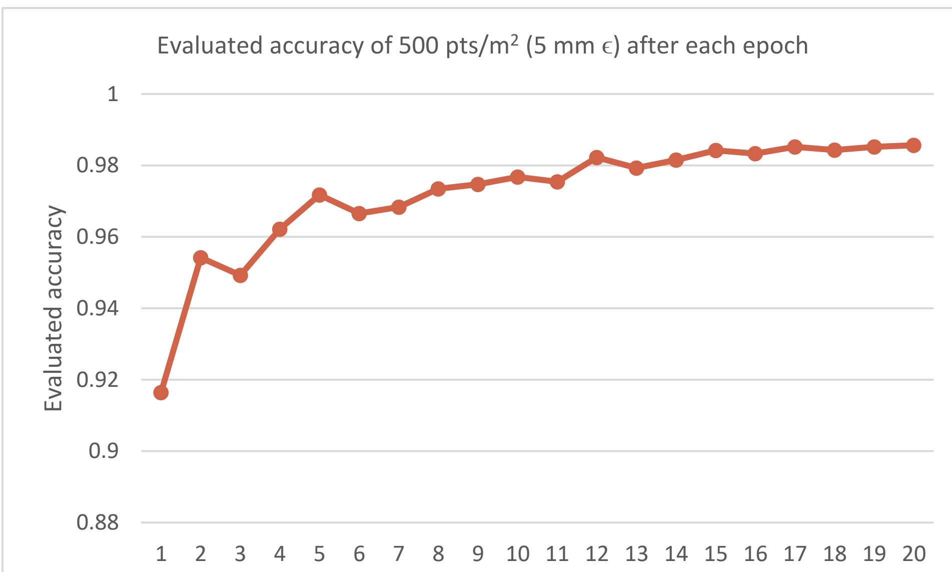

Figure 24. This shows the evaluated accuracy of the point cloud with a point density of 500 pts/m^2 (5 mm ϵ) after each epoch.

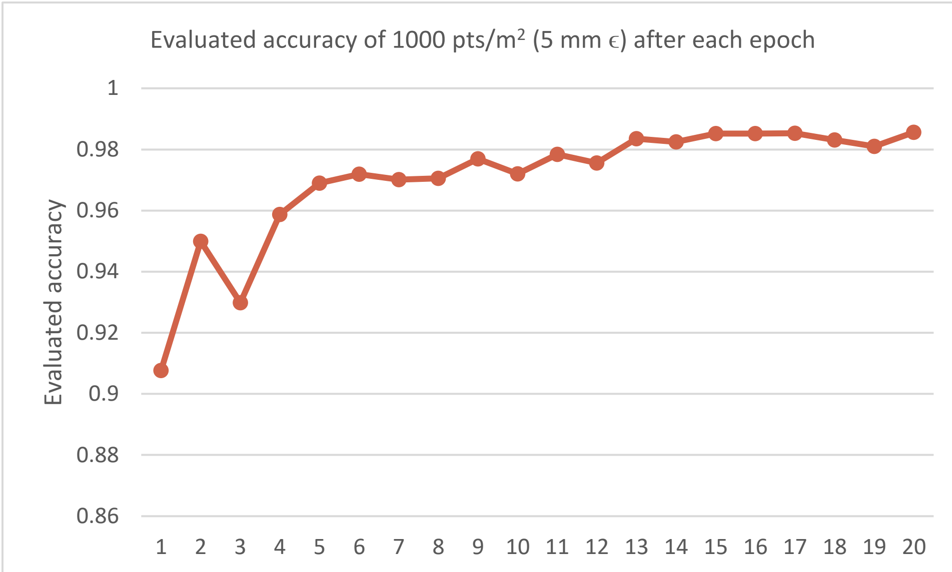

Figure 25. This shows the evaluated accuracy of the point cloud with a point density of 1000 pts/m^2 (5 mm ϵ) after each epoch.

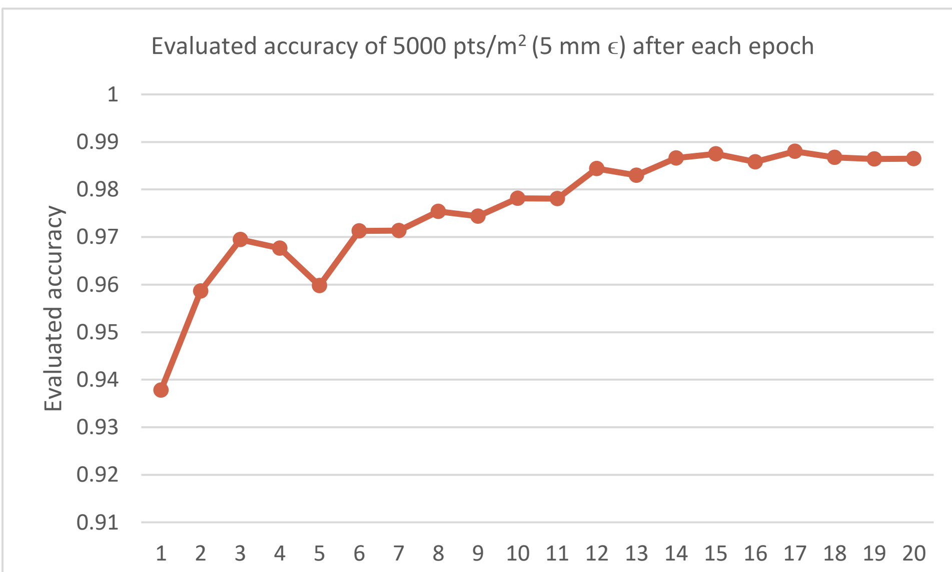

Figure 26. This shows the evaluated accuracy of the point cloud with a point density of 5000 pts/m^2 (5 mm ϵ) after each epoch.

From the figures above indicating the changes of the accuracies of the predictions after each epoch. It shows that the accuracies of the models were gradually increasing from epoch 1 to around epoch 15 and stayed steady from around 15 to epoch 20, despite some drops in the middle. The final models of each training point clouds were chosen to be the models at epoch 20. The final models were then utilized for testing different simulated point clouds and the real-world point clouds. The evaluated mean losses of the machine learning models were to evaluate the predictions made by the models. The changes of the evaluated mean loss were displayed in the following figures. It shows that the evaluated mean losses of the models were gradually decreasing from epoch 1 to around epoch 15 and stayed steady from around 15 to epoch 20, despite some rises in the middle.

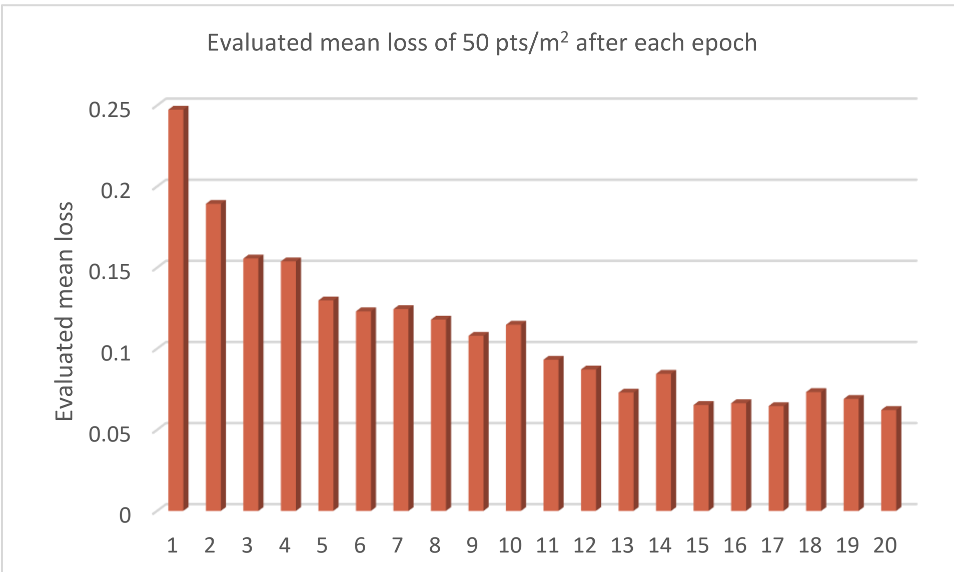

Figure 27. This shows the evaluated mean loss of the point cloud with a point density of 50 pts/m^2 (no random error) after each epoch.

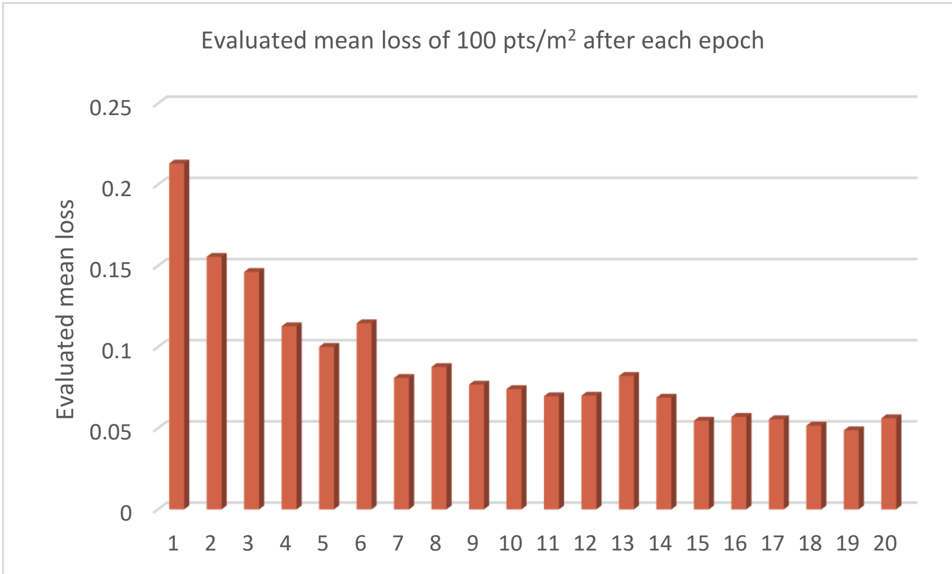

Figure 28. This shows the evaluated mean loss of the point cloud with a point density of 100 pts/m^2 (no random error) after each epoch.

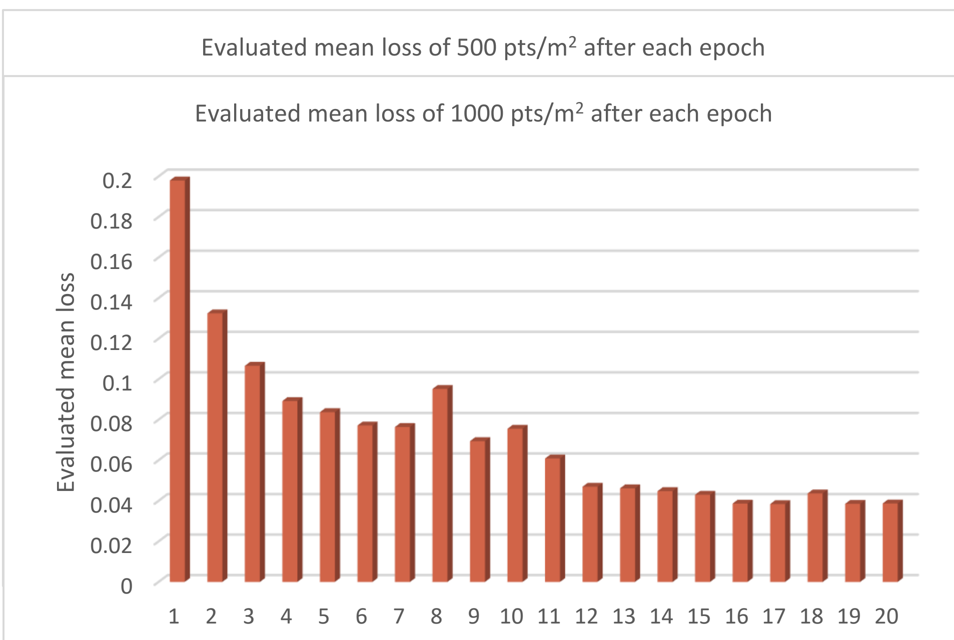

Figure 30. This shows the evaluated mean loss of the point cloud with a point density of 1000 pts/m^2 (no random error) after each epoch.

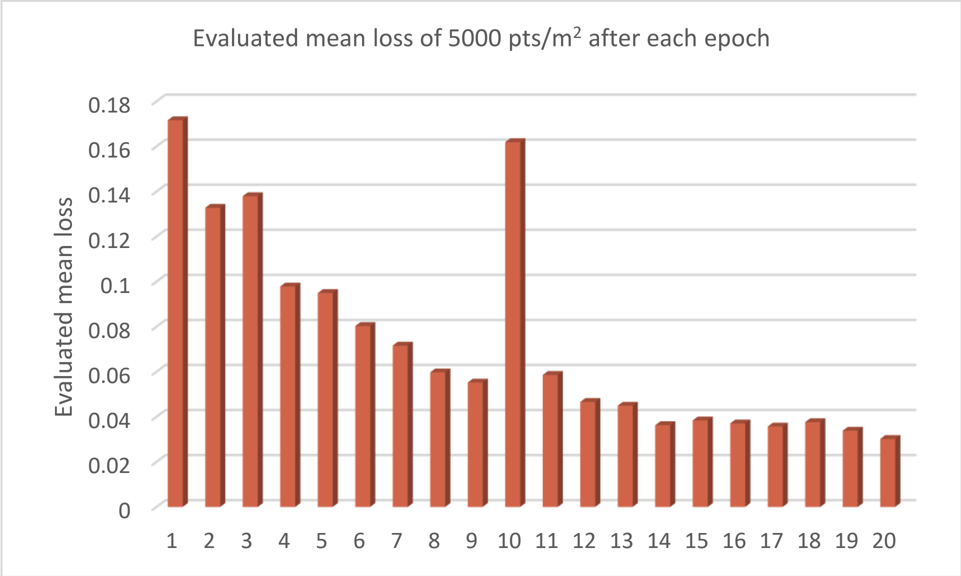

Figure 31. This shows the evaluated mean loss of the point cloud with a point density of 5000 pts/m^2 (no random error) after each epoch.

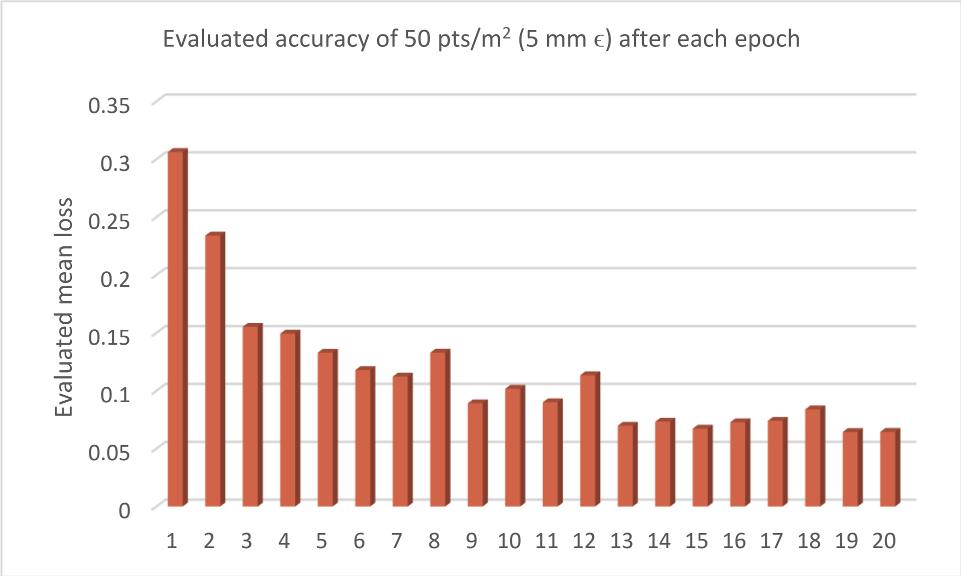

Figure 32. This shows the evaluated mean loss of the point cloud with a point density of 50 pts/m^2 (5 mm ϵ) after each epoch.

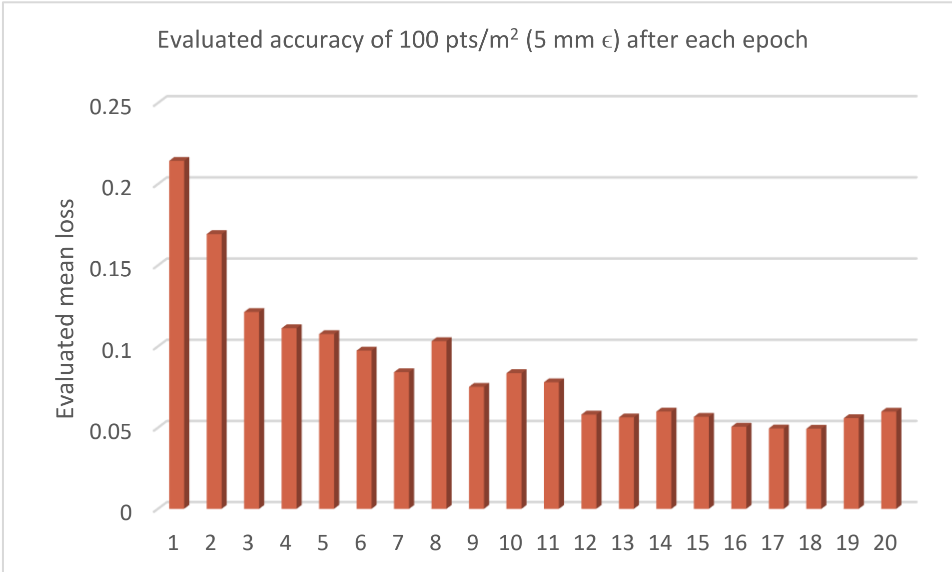

Figure 33. This shows the evaluated mean loss of the point cloud with a point density of 100 pts/m^2 (5 mm ϵ) after each epoch.

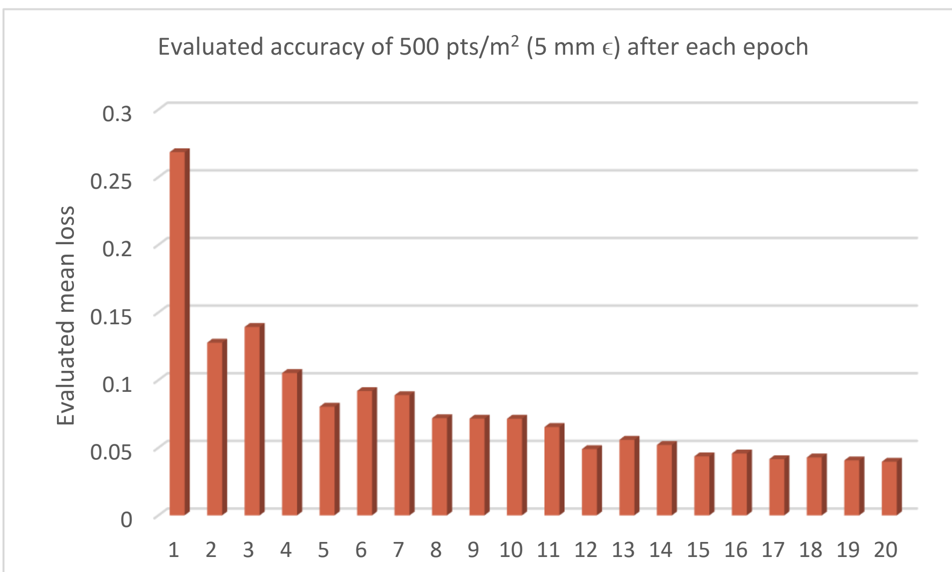

Figure 34. This shows the evaluated mean loss of the point cloud with a point density of 500 pts/m^2 (5 mm ϵ) after each epoch.

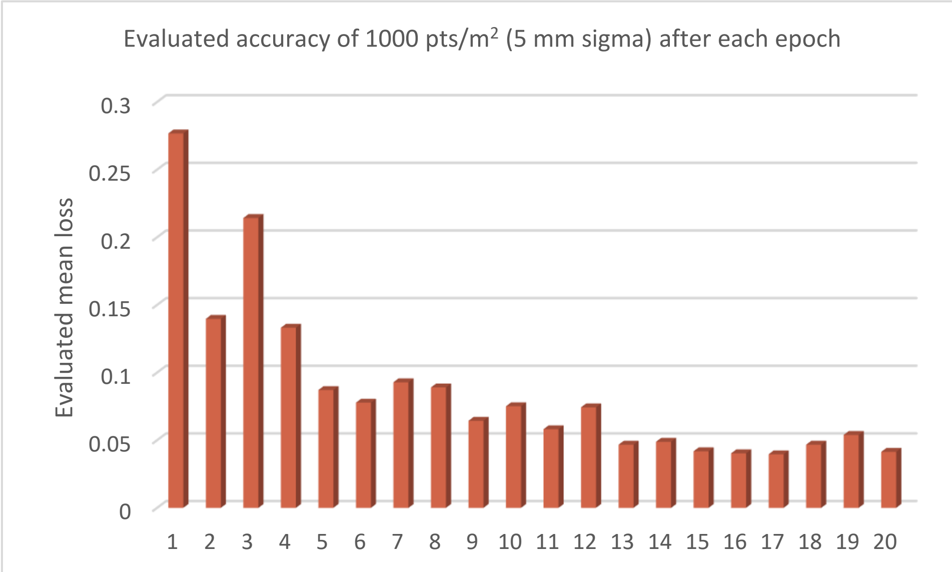

Figure 35. This shows the evaluated mean loss of the point cloud with a point density of 1000 pts/m^2 (5 mm sigma) after each epoch.

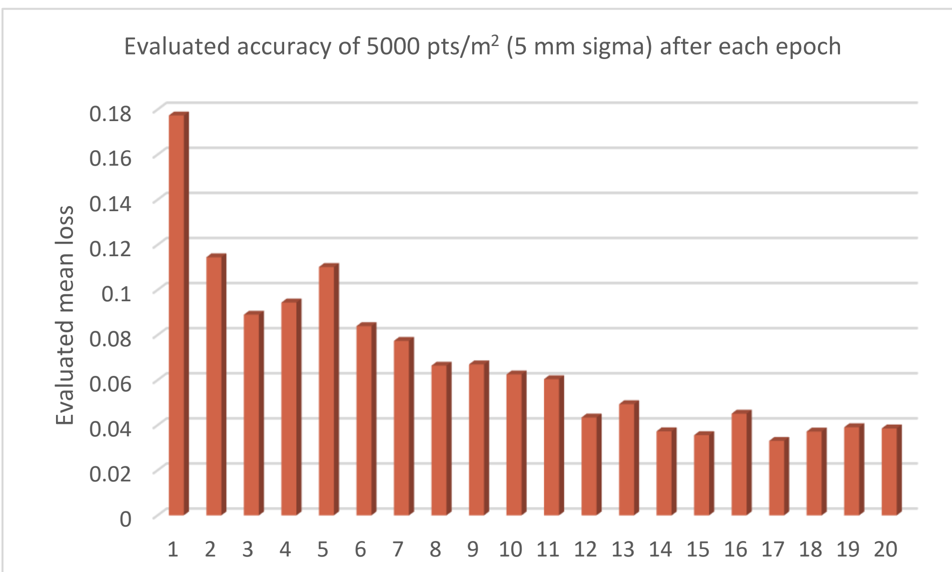

Figure 36. This shows the evaluated mean loss of the point cloud with a point density of 5000 pts/m^2 (5 mm sigma) after each epoch.

4.2.1 Point density

4.2.1.1 Point density without random error

In the field of point cloud processing, most of the works (Rabbani et.al., 2006; Jagannathan and Miller, 2007; Rottensteiner et.al., 2014; Weinmann et.al., 2015; Charles et.al., 2017) regarding the accuracy have been analyzed practically, instead of statistically. Normally, there are over thousands of sample points in a typical point cloud. It is inappropriate to use statistical models for the analyses.

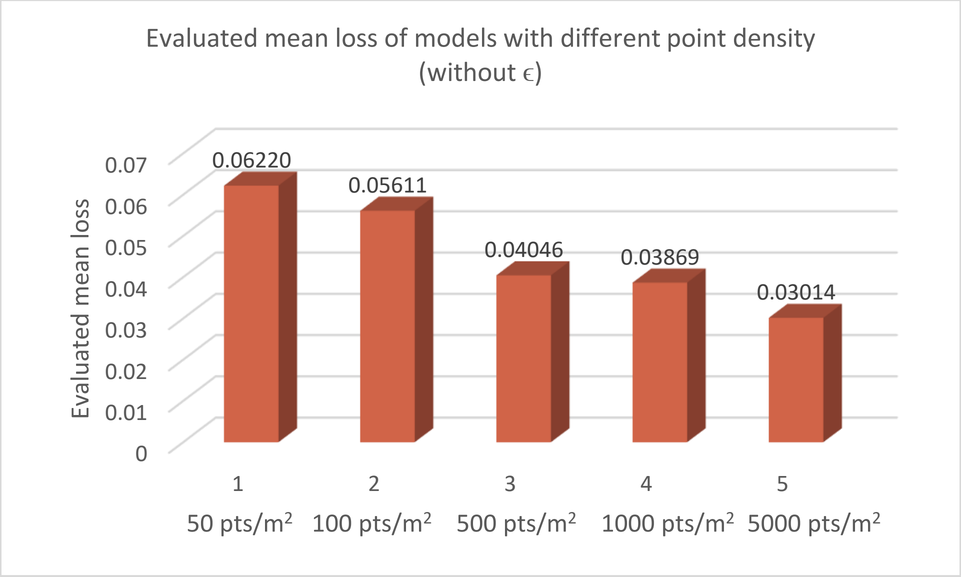

Figure 37. This shows the evaluated mean loss of the point cloud with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 (no random error).

This section analyzes how point density affects the results in the point clouds without random error. In Figure 37, the evaluated mean loss of the models was displayed in an order of 50 pts/m^2 (no random error), 100 pts/m^2 (no random error), 500 pts/m^2 (no random error), 1000 pts/m^2 (no random error), and 5000 pts/m^2 (no random error). These point clouds were simulated by scattering points on the objects' surfaces without adding random shifts in order to decouple point density from random error. From Figure 37, it clearly shows that with the increasing of point density, the evaluated mean loss of the models decreased. Similarly, the evaluated mean loss of 50-point density (pts/m^2) point cloud was approximately twice the evaluated mean loss the 5000-point density point cloud. This indicates that the machine learning model makes fewer mistakes in classifying simulated points of higher point density. However, this argument is based on the condition that the testing point clouds' point density is the same as the training point clouds' point density.

From Figure 38, it clearly shows that with the increasing of point density, the evaluated accuracy of the models increased. However, this was true under the condition that the testing point clouds' point density was the same as the training point clouds' point density. Therefore, later in the experiment, the trained machine learning models were tested by other point clouds with different point clouds' point density. This figure indicates that, when applied on the point clouds with the same point density, the higher point density it is, the higher the accuracy the model provides. But the difference of the accuracies between the point cloud (50 pts/m^2, no random error) and the point cloud (5000 pts/m^2, no random error) was only about 1.5%. However, the data size of the point cloud (5000 pts/m^2, no random error) was nearly 1000 times bigger than the data size of the point cloud (50 pts/m^2, no random error). Moreover, the baking time of these points during the point cloud simulation was displayed in Table 12. It took the system (CPU: Intel i7-8700K 3.70 GHz, RAM: 32 GB, GPU: GTX1070) about 5 minutes to generate 374131 points for the point cloud (50 pts/m^2, no random error), while it took the system about 133 minutes to generate 37414252 points for the point cloud (5000 pts/m^2, no random

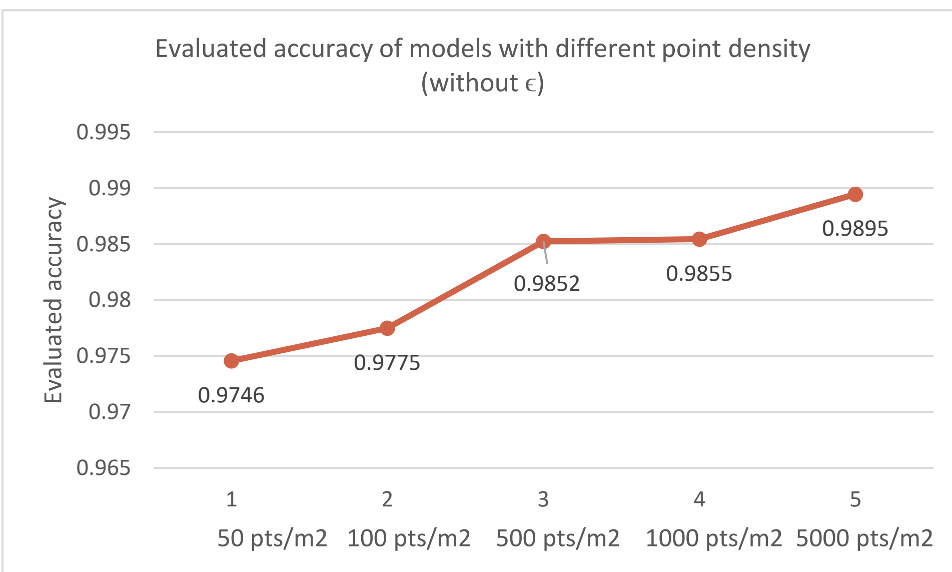

Figure 38. This shows the evaluated accuracy of the point cloud with the point density of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2 (no random error).

error). Therefore, in this study, it is not worth of spending a lot more time and computation resources for a 1.5% accuracy gain.

Table 12. The number of points in each testing dataset (Area 4).

Point density (pts/m^2)	Number of points	Generation time (minute)
50	374131	5
100	748100	8
500	3741416	20
1000	7482860	36
5000	37414252	133

Table 13. The data size of each testing dataset (Area 4). The TXT file is the original format of the point cloud data and the NPY is the file format for testing and implementation.

File type	50	100	500	1000	5000
	pts/m^2, no ϵ	pts/m^2, no ϵ	pts/m^2, no ϵ	pts/m^2, no ϵ	pts/m^2, no ϵ
TXT	0.113 GB	0.227 GB	1.130 GB	2.170 GB	11.300 GB
NPY	0.019 GB	0.039 GB	0.194 GB	0.391 GB	1.950 GB

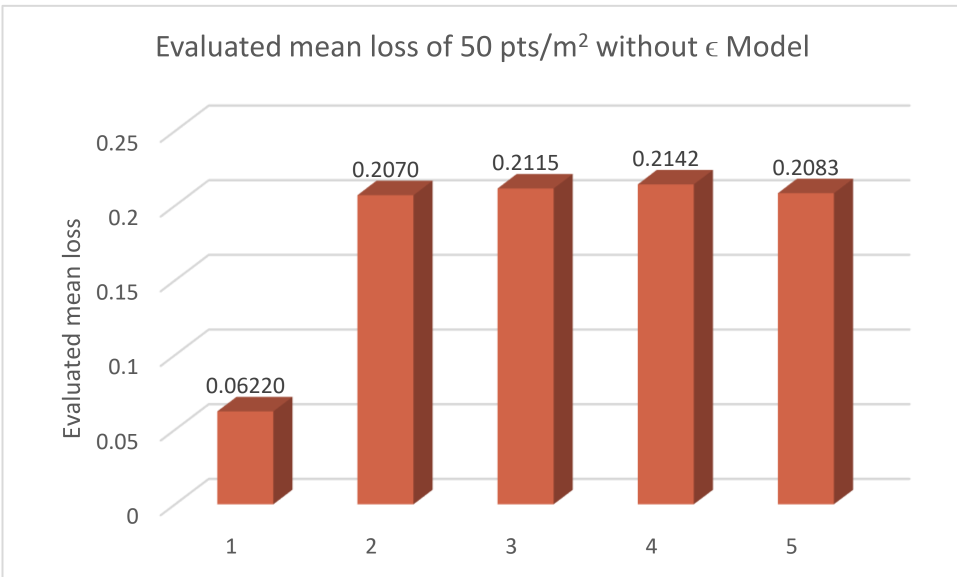

Figure 39. This shows the evaluated mean loss of the point cloud with the point density of 50 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

Figure 39 shows the evaluated mean loss of the point cloud with the point density of 50 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with point density of 50 pts/m^2 than the other testing data with different point densities. 50 pts/m^2 is also the point density of the training data. This indicates that the machine learning model makes fewer mistakes in classifying simulated points in the situation that the trained model is implemented on the point clouds with the same point cloud's density as the training point cloud. There was not much difference between the evaluated mean loss for the point clouds' point densities other than 50 pts/m^2. For instance, the least evaluated mean loss difference between the testing data with a point density of 50 pts/m^2 and other testing data with different point densities was 0.1447. The largest evaluated mean loss difference between other testing data with different point densities was 0.0072, which was approximately 0.05 times the least difference between the testing data with a point density of 50 pts/m^2 and other testing data with different point densities.

Figure 40 shows the evaluated accuracy of the point cloud with the point density of 50 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy

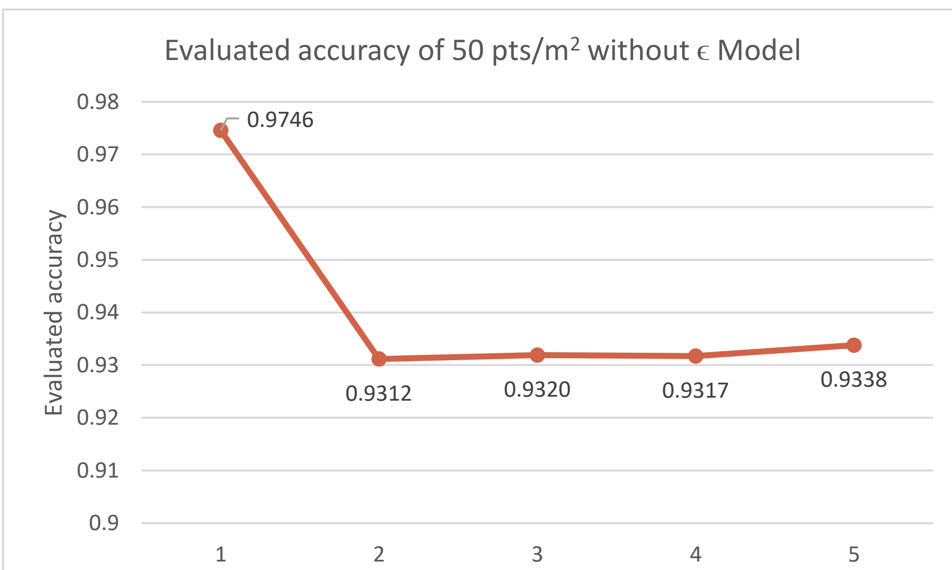

Figure 40. This shows the evaluated mean accuracy of the point cloud with the point density of 50 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

of the model was higher when implemented on the testing data with the same point density of 50 pts/m^2 than the other testing data with different point densities. This indicates that the machine learning model performs better in classifying simulated points in the situation that the trained model is implemented on the point clouds with the same point density as the training point cloud. There was not much difference between the evaluated accuracy for the point clouds' point densities other than 50 pts/m^2. For instance, the least evaluated accuracy difference between the testing data with the point density of 50 pts/m^2 and other testing data with different point densities was 4.08%. The largest evaluated mean loss difference between other point density testing data was 0.26%, which was approximately 0.05 times the least difference between the testing data with the point density of 50 pts/m^2 and other testing data with different point densities.

Figure 41 shows the evaluated mean loss of the point cloud with the point density of 100 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when it was implemented on the point clouds with the same point clouds' density

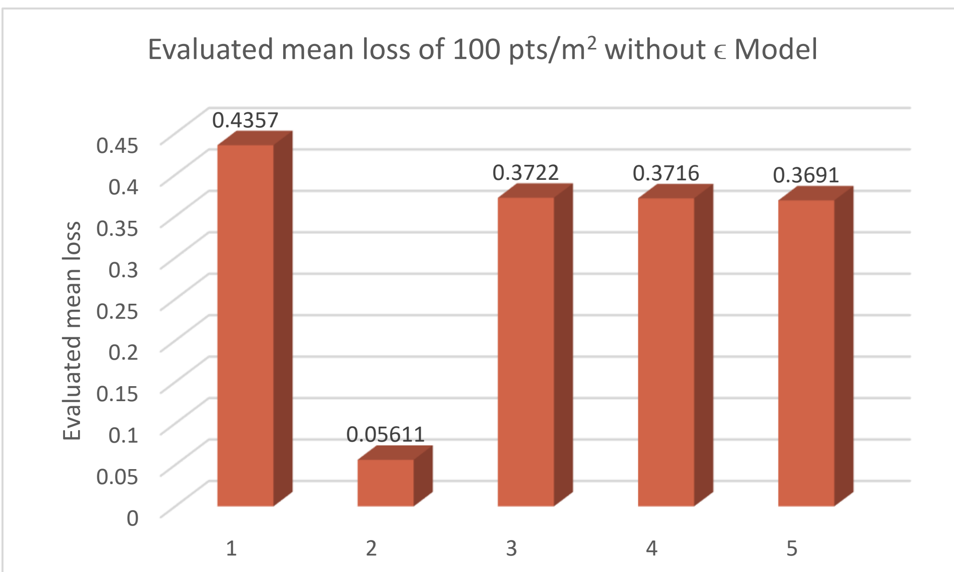

Figure 41. This shows the evaluated mean loss of the point cloud with the point density of 100 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

as the training point clouds. There was not much difference between the evaluated mean loss for the point clouds' point densities larger than 100 pts/m^2. For instance, the least evaluated mean loss difference between the testing data with a point density of 100 pts/m^2 and other testing data with larger point densities was 0.31299. The largest evaluated mean loss difference between other point density testing data with larger point densities was 0.0031, which was approximately 0.01 times the least difference between the testing data with a point density of 100 pts/m^2 and other testing data with larger point densities than 100 pts/m^2. However, the evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 100 pts/m^2. The least evaluated mean loss difference between the testing data with a point density of 50 pts/m^2 and other testing data with larger point densities that were larger than 100 pts/m^2 was 0.0635, which was approximately 20 times the largest evaluated mean loss difference between the testing point clouds with point densities that were larger than 100 pts/m^2. This indicates that, if the testing point clouds' point density is different from the training point clouds' point density, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density. Nevertheless, when the testing point clouds' point density was larger than the training point cloud's point density, there was not much difference in the evaluated mean loss.

Figure 42 shows the evaluated accuracy of the point cloud with the point density of 100 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density than it was with the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point

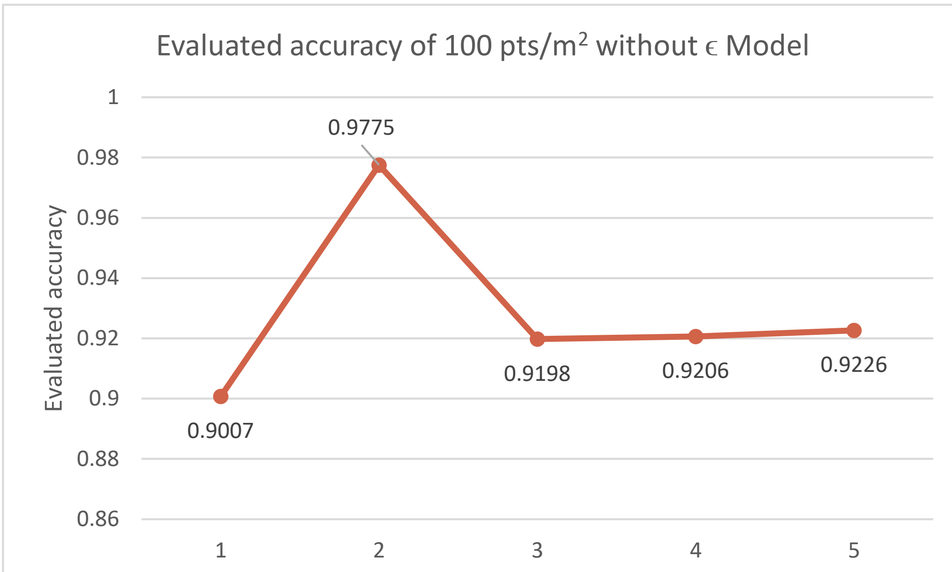

Figure 42. This shows the evaluated accuracy of the point cloud with the point density of 100 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

clouds' density as the training point clouds. There was not much difference between the evaluated accuracy for the point clouds' point densities larger than 100 pts/m^2. For instance, the least evaluated accuracy difference between the testing data with a point density of 100 pts/m^2 and other testing data with larger point densities was 5.49%. The largest evaluated accuracy difference between other point density testing data with larger point densities was 0.28%, which was approximately 0.05 times the least accuracy difference between the testing data with a point density of 100 pts/m^2 and other point density testing data with larger point densities than 100 pts/m^2. However, the evaluated accuracy of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably lower than the evaluated accuracy of the model for the testing point clouds with point densities that were larger than 100 pts/m^2. The least evaluated accuracy difference between the testing data with a point density of 50 pts/m^2 and other testing data with larger point densities that were larger than 100 pts/m^2 was 1.91%, which was approximately 6.6 times the largest evaluated accuracy difference between the testing point clouds with point densities that were larger than 100 pts/m^2. This indicates that, if the testing point clouds' point density is different from the training point clouds' point density, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Nevertheless, when the testing point clouds' point density was larger than the training point cloud's point density, there was not much difference in the evaluated accuracy.

Figure 43 shows the evaluated mean loss of the point cloud with the point density of 500 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. There was not much difference between the evaluated mean loss for the point clouds' point densities larger than 500 pts/m^2. However, the evaluated mean loss of the

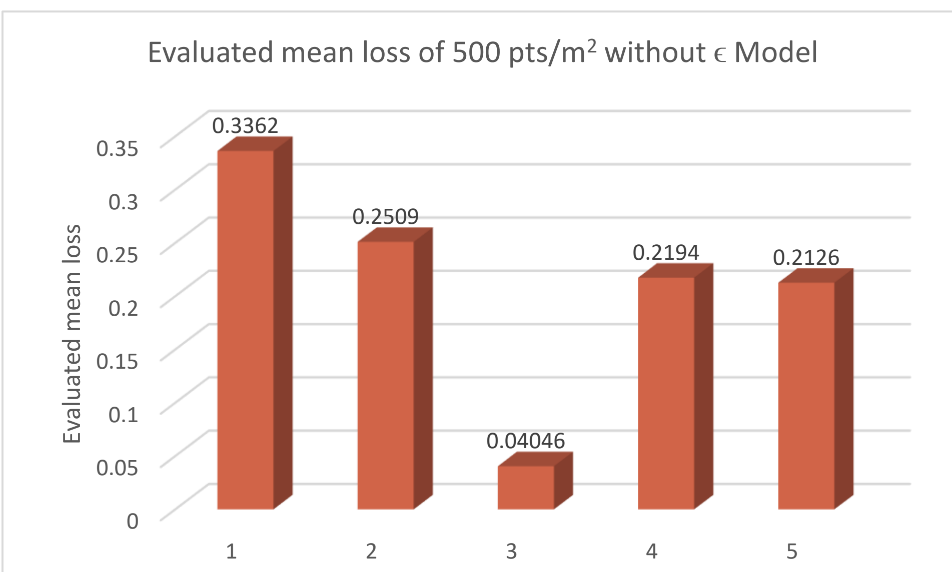

Figure 43. This shows the evaluated mean loss of the point cloud with the point density of 500 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. Moreover, the evaluated mean loss of the model for the testing point cloud with a point density of 100 pts/m^2 was also larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. This indicates that, if the testing point cloud's point density is different from the training point clouds' point density, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, if the testing point clouds' point density is smaller than the training point clouds' point density, the smaller it is, the more mistakes the model is going to make. Finally, the evaluated mean loss for the testing point cloud with a point density of 5000 pts/m^2 was slightly smaller than the evaluated mean loss for the testing point cloud with a point density of 1000 pts/m^2. This indicates that, if the testing point clouds' point density is larger than the training point clouds' point density, the larger it is, the fewer mistakes the model is going to make.

Figure 44 shows the evaluated accuracy of the point cloud with the point density of 500 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy

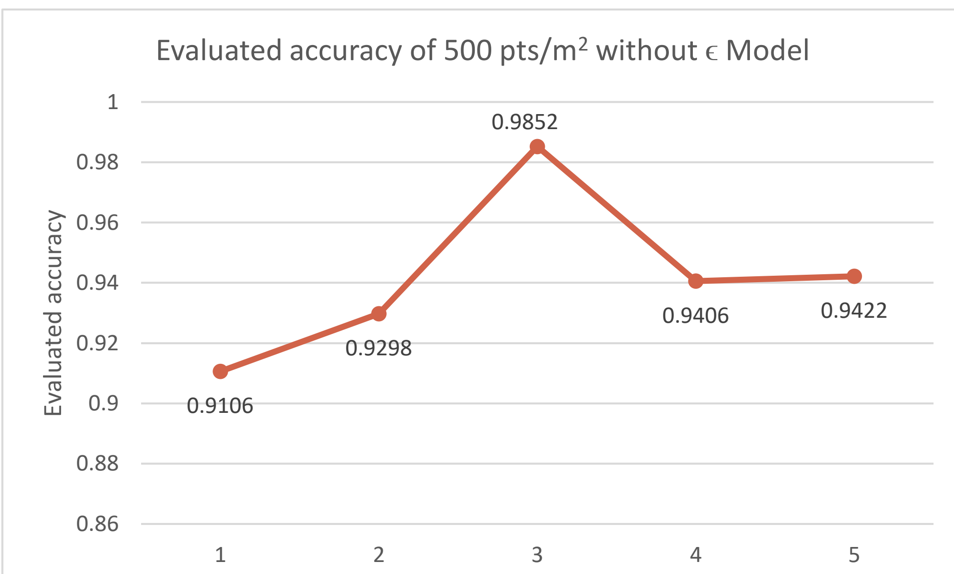

Figure 44. This shows the evaluated accuracy of the point cloud with the point density of 500 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

of the model was higher when implemented on the testing data with the same point density than the other testing point clouds with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. There was not much difference between the evaluated accuracy for the point clouds' point densities larger than 500 pts/m^2. However, the evaluated accuracies of the model for the testing point clouds with a point density of 50 pts/m^2 and a point density of 100 pts/m^2 were noticeably lower than the evaluated accuracies of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. Moreover, the evaluated accuracy of the model for the testing point cloud with a point density of 100 pts/m^2 was also lower than the evaluated accuracy of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. If the testing point clouds' point density is smaller than the training point clouds' point density, the smaller it is, the worse the performance of the model is going to provide. Finally, the evaluated accuracy for the testing point cloud with a point density of 5000 pts/m^2 was slightly higher than the evaluated accuracy for the testing point cloud with a point density of 1000 pts/m^2. This indicates that, if the testing point clouds' point density is larger than the training point clouds' point density, the larger it is, the better performance the model is going to provide.

Figure 45 shows the evaluated mean loss of the point cloud with the point density of 1000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And the evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 50 pts/m^2. And with the increasing of point density, the evaluated mean loss decreased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. The evaluated mean loss difference between the testing data with a point density of 50 pts/m^2 and the testing data with a point density of 500 pts/m^2 was 0.1487. The evaluated mean loss difference between the testing data with a point density of 500 pts/m^2 and the testing data with a point density of 5000 pts/m^2 was 0.0120. In the first pair, including the point clouds with point densities 50 pts/m^2 and 500 pts/m^2, and the

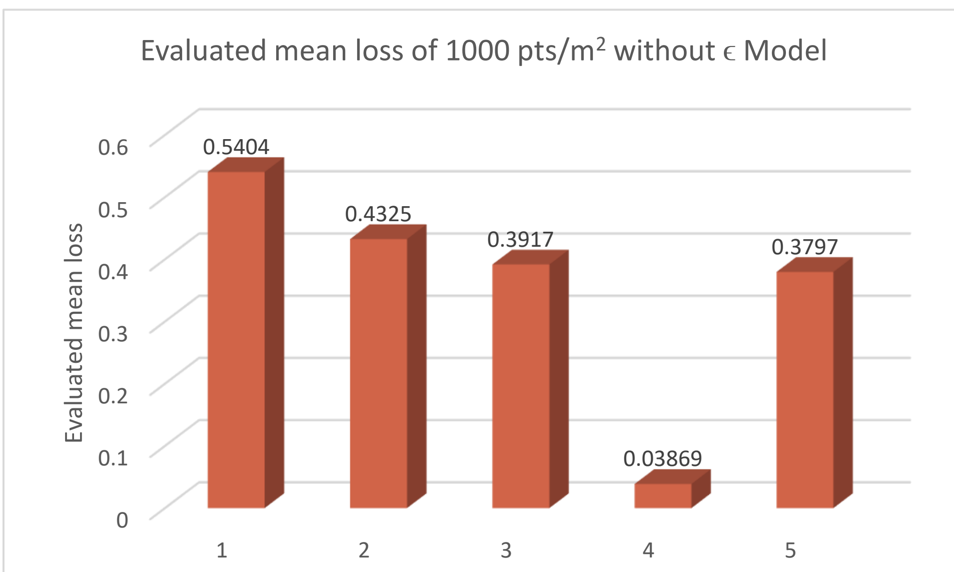

Figure 45. This shows the evaluated mean loss of the point cloud with the point density of 1000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

second pair, including the point clouds with point densities 500 pts/m^2 and 5000 pts/m^2, the larger point clouds' densities were both 10 times the smaller ones. However, the evaluated mean loss difference of the first pair was about 12.4 times the one of the second pair. This indicates that, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the mistakes that the machine learning model makes in classifying simulated point clouds decreases.

Figure 46 shows the evaluated accuracy of the point cloud with the point density of 1000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density than the other testing point clouds with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And with the increasing of point density, the

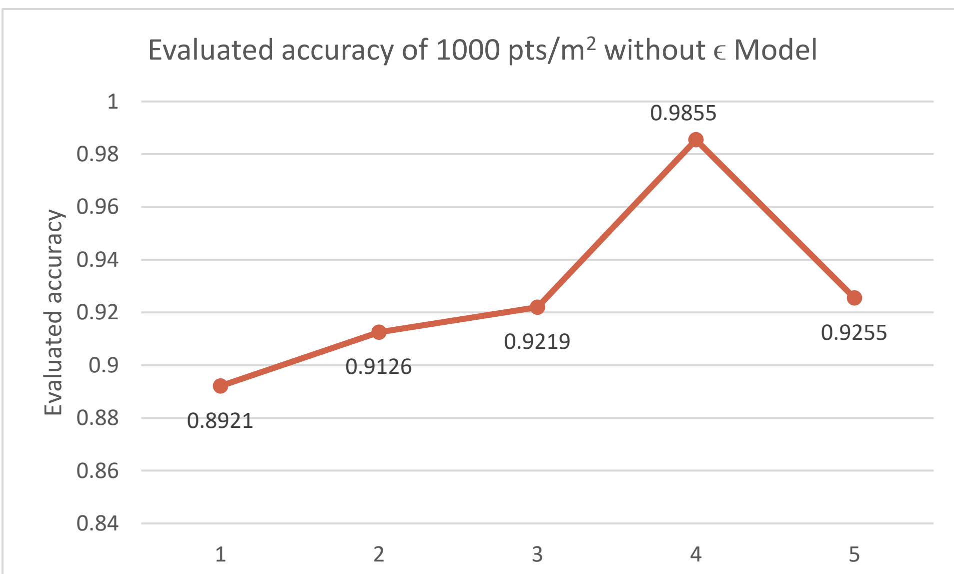

Figure 46. This shows the evaluated accuracy of the point cloud with the point density of 1000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

evaluated accuracy also increased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, as described in the analysis of the evaluated mean loss, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the accuracies of the machine learning models decreases.

Figure 47 shows the evaluated mean loss of the point cloud with the point density of 5000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And the evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for

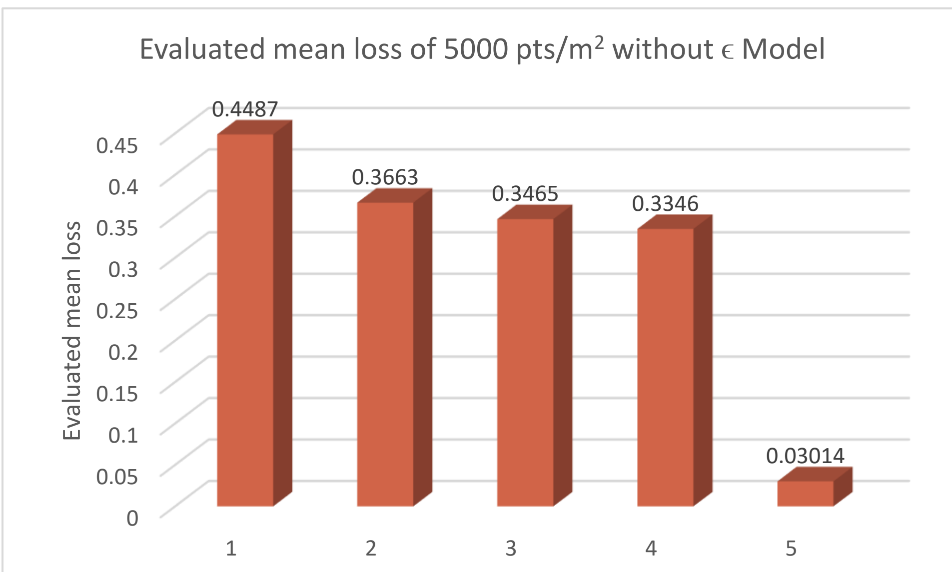

Figure 47. This shows the evaluated mean loss of the point cloud with the point density of 5000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

the testing point clouds with point densities that were larger than 50 pts/m^2. And with the increasing of point density, the evaluated mean loss decreased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Similar to the result of the model trained by the point cloud with a point density of 1000 pts/m^2, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the mistakes that the machine learning model makes in classifying simulated point clouds decreases.

Figure 48 shows the evaluated accuracy of the point cloud with the point density of 5000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the

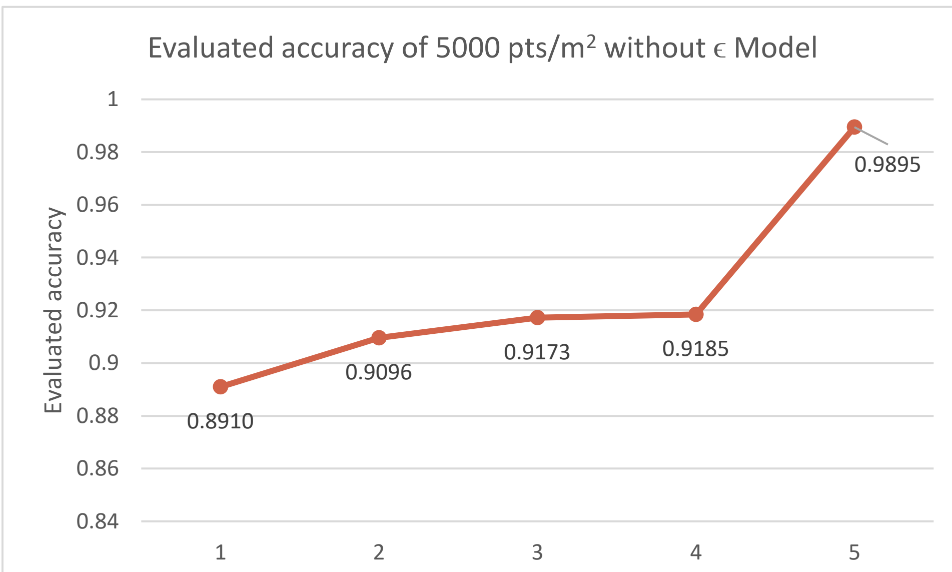

Figure 48. This shows the evaluated accuracy of the point cloud with the point density of 5000 pts/m^2 (no random error), tested by 5 different point clouds (no random error): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

training point clouds. And the evaluated accuracy of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably lower than the evaluated accuracy of the model for the testing point clouds with point densities that were larger than 50 pts/m^2. And with the increasing of point density, the evaluated accuracy also increased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Similar to the result of the model trained by the point cloud with a point density of 1000 pts/m^2, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the accuracy of the machine learning model decreases.

4.2.1.2 Point cloud with 5 mm ϵ

This section analyzes how point density affect the results in the point clouds with random error (5 mm ϵ). In Figure 49, the evaluated mean loss of models were displayed in an order of 50 pts/m^2 (5 mm ϵ), 100 pts/m^2 (5 mm ϵ), 500 pts/m^2 (5 mm ϵ), 1000 pts/m^2 (5 mm ϵ), and 5000 pts/m^2 (5 mm ϵ). These point clouds were simulated by scattering points on the objects' surfaces with adding random shifts while keeping the overall ϵ the same, in order to decouple point density from random error. From Figure 49, it shows that with the increasing of point density, overall, the evaluated mean loss of the models decreased. However, the evaluated mean loss of 1000 point density (pts/m^2) point cloud was larger than the evaluated mean loss of the point cloud with a point density of 500 pts/m^2. Moreover, the difference of the evaluated mean losses when point density was larger than 500 pts/m^2 was small. This is different from the point clouds without random errors. This indicates that, in general, the machine learning model makes fewer mistakes in classifying simulated points of higher point density, but the difference is minor when the point density is relatively large (500 pts/m^2 in this experiment). This argument is based on the

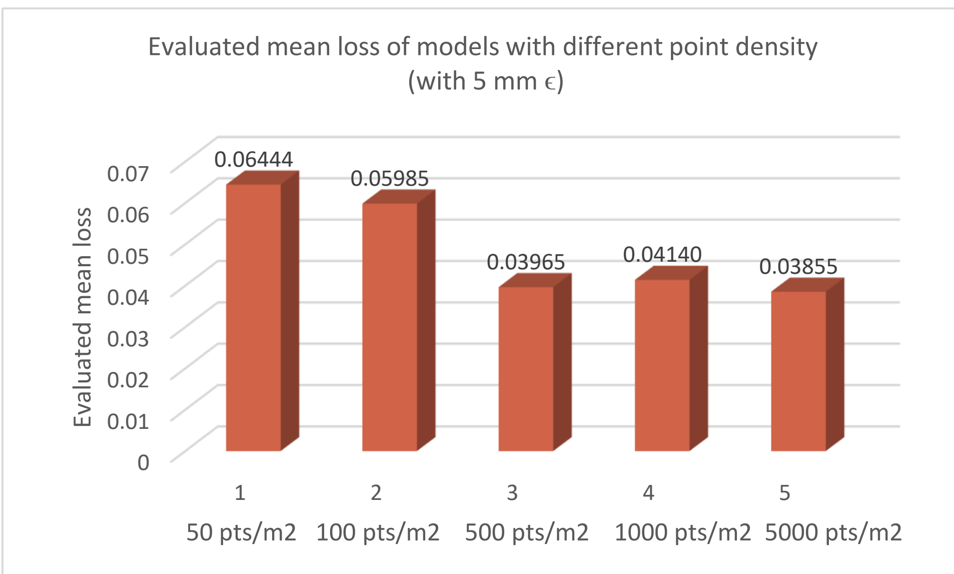

Figure 49. This shows the evaluated mean loss of the point cloud with the point density (5 mm ϵ) of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2.

condition that the testing point clouds' point density is the same as the training point clouds' point density.

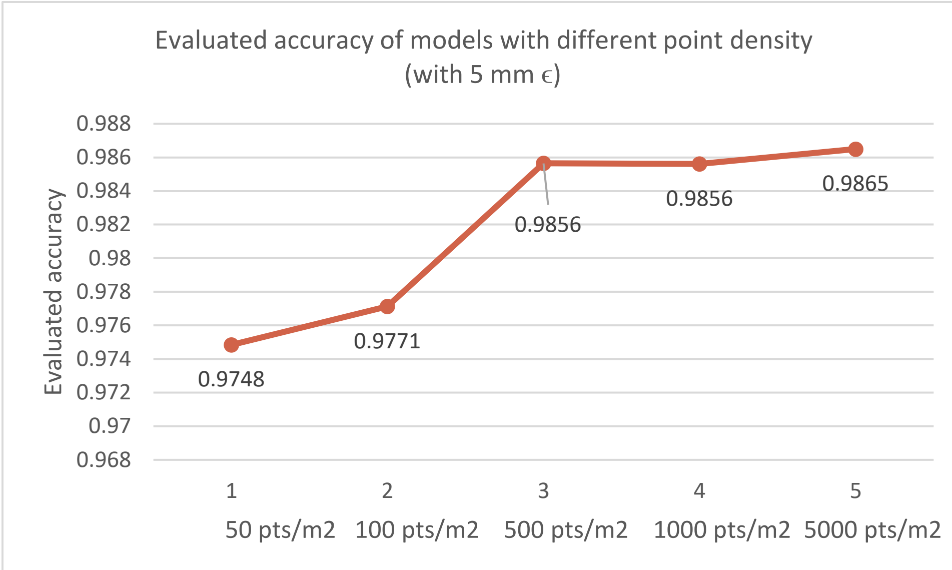

Figure 50. This shows the evaluated accuracy of the point cloud with the point density (5 mm ϵ) of 50 pts/m^2, 100 pts/m^2, 500 pts/m^2, 1000 pts/m^2, 5000 pts/m^2.

From Figure 50, it shows that with the increasing of point density, the evaluated accuracy of the models increased overall, despite the evaluated accuracy of the point cloud with a point density of 1000 pts/m^2 dropped. This also indicates that the machine learning model performs better in classifying simulated points with higher point density, but the difference is minor when the point density is relatively large (500 pts/m^2 in this experiment). This argument is based on the condition that the testing point clouds' point density is the same as the training point clouds' point density. Therefore, later in the experiment, the trained machine learning models were tested by other point clouds with different point clouds' point density.

Figure 51 shows the evaluated mean loss of the point cloud with the a point density of 50 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with point density of 50 pts/m^2 than the other testing data with different point densities. 50 pts/m^2 is also the point density of the training data. This indicates that the machine learning model performs better in classifying simulated points in the situation that the trained model is implemented on the point clouds with the same point cloud's density as the training point cloud. There was not much difference between the evaluated mean loss for the point clouds' point densities other than 50 pts/m^2. Despite the minor altitudes, the evaluated mean loss of the point clouds decreased when the point densities increased, which was different from the tests of point clouds without random errors.

Figure 52 shows the evaluated accuracy of the point cloud with a point density of 50 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density of 50 pts/m^2 than the other testing data with different point densities. This indicates that the machine learning model performs better in classifying simulated points in the situation that the trained model is

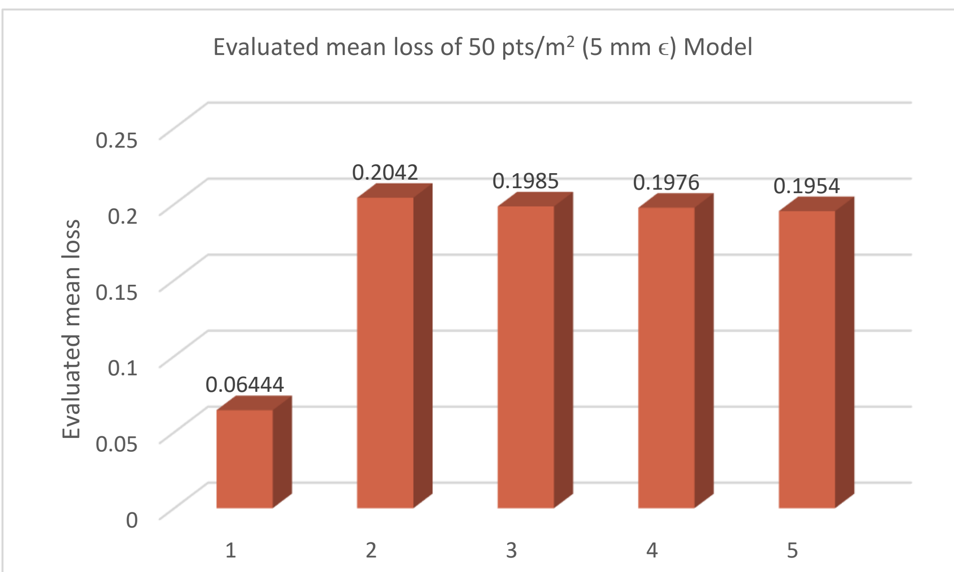

Figure 51. This shows the evaluated mean loss of the point cloud with the point density of 50 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

implemented on the point clouds with the same point cloud's density as the training point cloud. And there was not much of a difference between the evaluated accuracy for the point clouds' point densities other than 50 pts/m^2. Despite the minor altitudes, the evaluated accuracy of the point clouds increased when the point densities increased, which was different from the tests of point clouds without random errors.

Figure 53 shows the evaluated mean loss of the point cloud with a point density of 100 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500

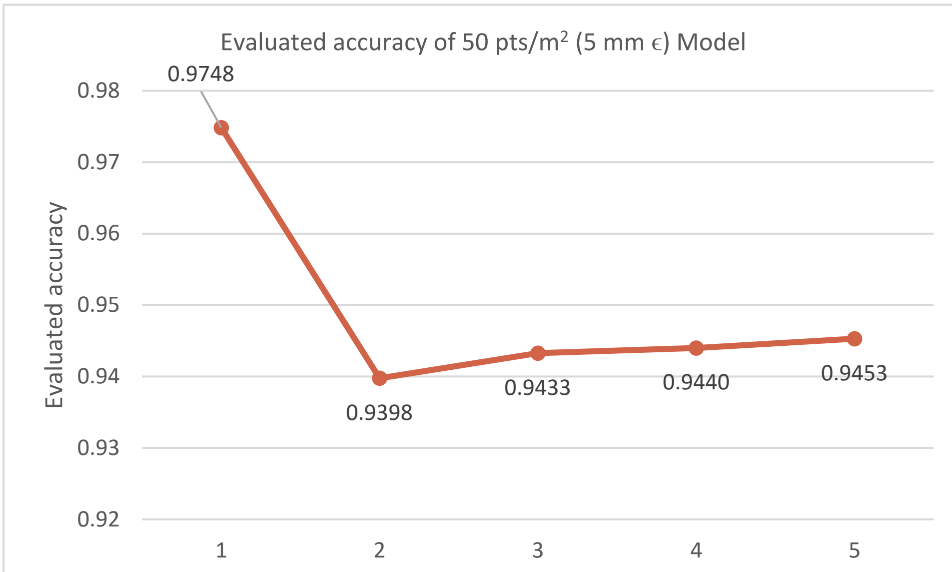

Figure 52. This shows the evaluated accuracy of the point cloud with the point density of 50 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs the better when implemented on the point clouds with the same point clouds' density as the training point clouds. And there was not much of a difference between the evaluated mean loss for the point clouds' point densities larger than 100 pts/m^2. However, the evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 100 pts/m^2. This indicates that, if the testing point clouds' point density is different from the training point clouds' point density, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density, with respect to the point density of the training point cloud. Nevertheless, when the testing point clouds' point density was larger than the training point cloud's point density, there was not much of a difference in the evaluated mean loss.

Figure 54 shows the evaluated accuracy of the point cloud with a point density of 100 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was

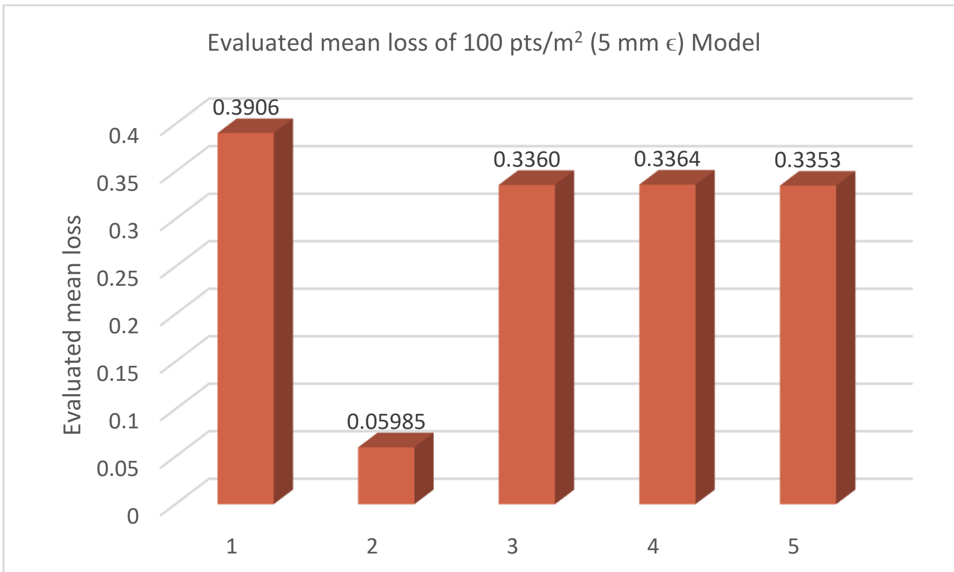

Figure 53. This shows the evaluated mean loss of the point cloud with the point density of 100 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

higher when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And there was not much of a difference between the evaluated accuracy for the point clouds' point densities larger than 100 pts/m^2. However, the evaluated accuracy of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably lower than the evaluated accuracy of the model for the testing point clouds with point densities that were larger than 100 pts/m^2. This indicates that, if the testing point clouds' point density is different from the training point clouds' point density, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density with respect to the point density of the training point cloud. Nevertheless, when the testing point clouds' point density was larger than the training point cloud's point density, there was not much of a difference in the evaluated accuracy.

Figure 55 shows the evaluated mean loss of the point cloud with a point density of 500 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing

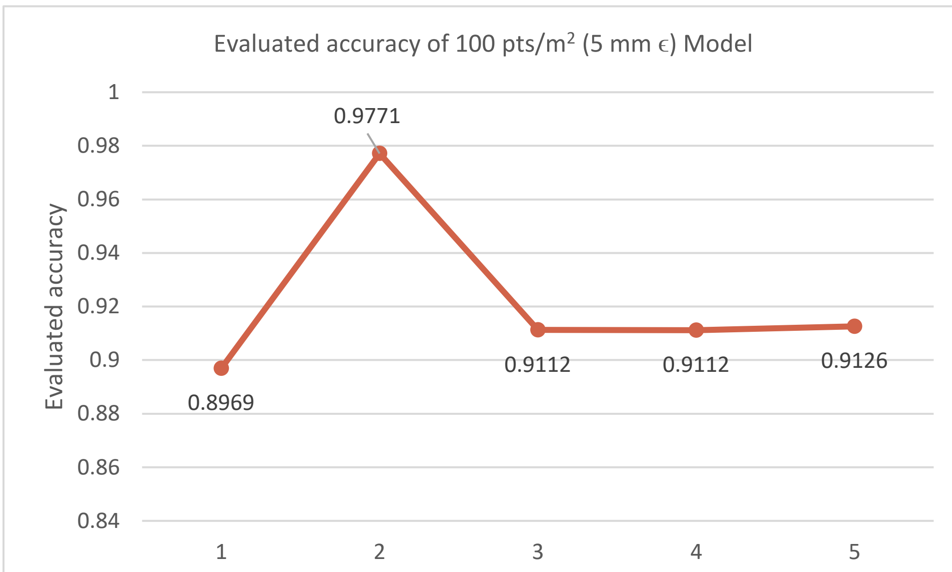

Figure 54. This shows the evaluated mean loss of the point cloud with the point density of 100 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And there was not much of a difference between the evaluated mean loss for the point clouds' point densities larger than 500 pts/m^2. However, the evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. Moreover, the evaluated mean loss of the model for the testing point cloud with a point density of 100 pts/m^2 was also larger than the evaluated mean loss of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density with respect to the point density of the training point cloud. Furthermore, if the testing point clouds' point density is smaller than the training point clouds' point density, the smaller it is, the more mistakes the model is going to make. Finally, the evaluated mean loss for the testing point cloud with a point density of 5000 pts/m^2 was slightly smaller than the evaluated mean loss for the testing point cloud with a point density of 1000 pts/m^2. This indicates that, if the testing point clouds' point density is larger than

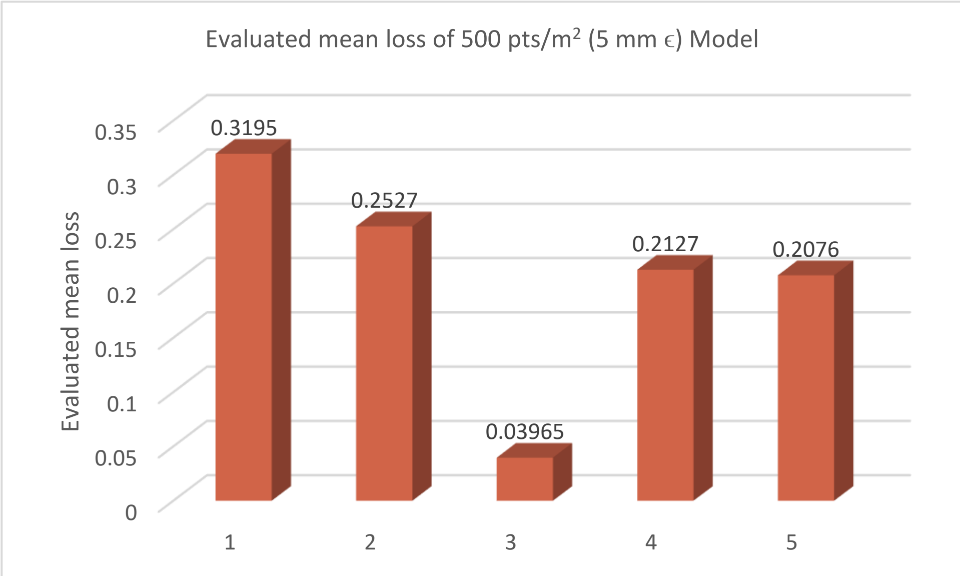

Figure 55. This shows the evaluated mean loss of the point cloud with the point density of 500 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

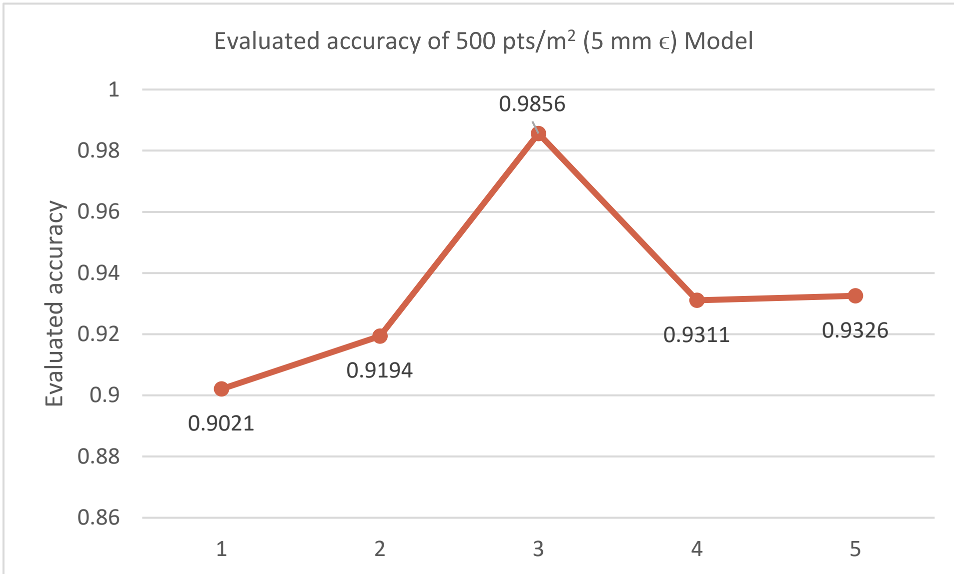

Figure 56. This shows the evaluated accuracy of the point cloud with the point density of 500 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

the training point clouds' point density, the larger it is, the fewer mistakes the model is going to make. Nevertheless, there was not much difference in the evaluated mean loss.

Figure 56 shows the evaluated accuracy of the point cloud with the point density of 500 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density than the other testing point clouds with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. There was not much difference between the evaluated accuracy for the point clouds' point densities larger than 500 pts/m^2. However, the evaluated accuracies of the model for the testing point clouds with a point density of 50 pts/m^2 and a point density of 100 pts/m^2 were noticeably lower than the evaluated accuracies of the model for the testing point clouds with point densities that were larger than 500 pts/m^2. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density with respect to the point density of the training point cloud. And, if the testing point clouds' point density is smaller than the training point clouds' point density, the smaller it is, the worse the performance of the model is going to provide. Finally, the evaluated accuracy for the testing point cloud with a point density of 5000 pts/m^2 was slightly higher than the evaluated accuracy for the testing point cloud with a point density of 1000 pts/m^2. This indicates that, if the testing point clouds' point density is larger than the training point clouds' point density, the larger it is, the better performance the model is going to provide. Nevertheless, there was not much difference in the evaluated accuracy.

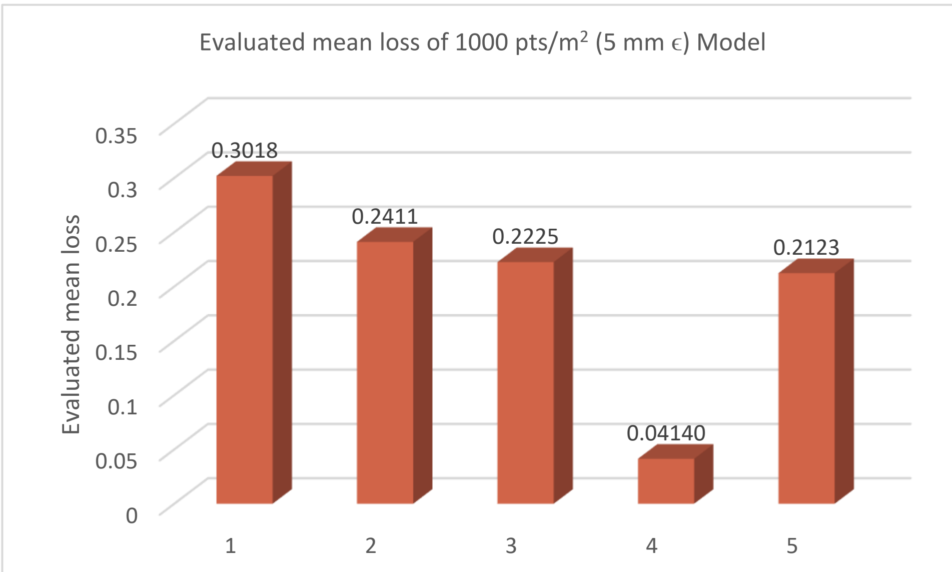

Figure 57. This shows the evaluated mean loss of the point cloud with the point density of 1000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

Figure 57 shows the evaluated mean loss of the point cloud with the point density of 1000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. The evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities larger than 50 pts/m^2. And with the increasing of point density, the evaluated mean loss decreased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model makes fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the mistakes that the machine learning model makes in classifying simulated point clouds decreases.

Figure 58 shows the evaluated accuracy of the point cloud with a point density of 1000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was higher when implemented on the testing data with the same point density than the other testing point clouds with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. And with the increasing of point density, the evaluated accuracy is also increasing.

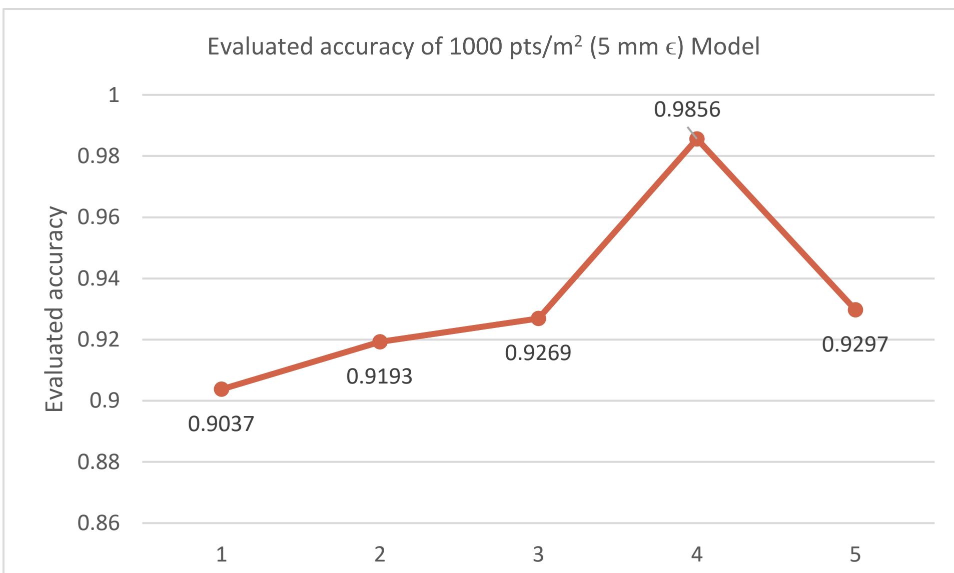

Figure 58. This shows the evaluated accuracy of the point cloud with the point density of 1000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the accuracies of the machine learning models decreases.

Figure 59 shows the evaluated mean loss of the point cloud with a point density of 5000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated mean loss of the model was

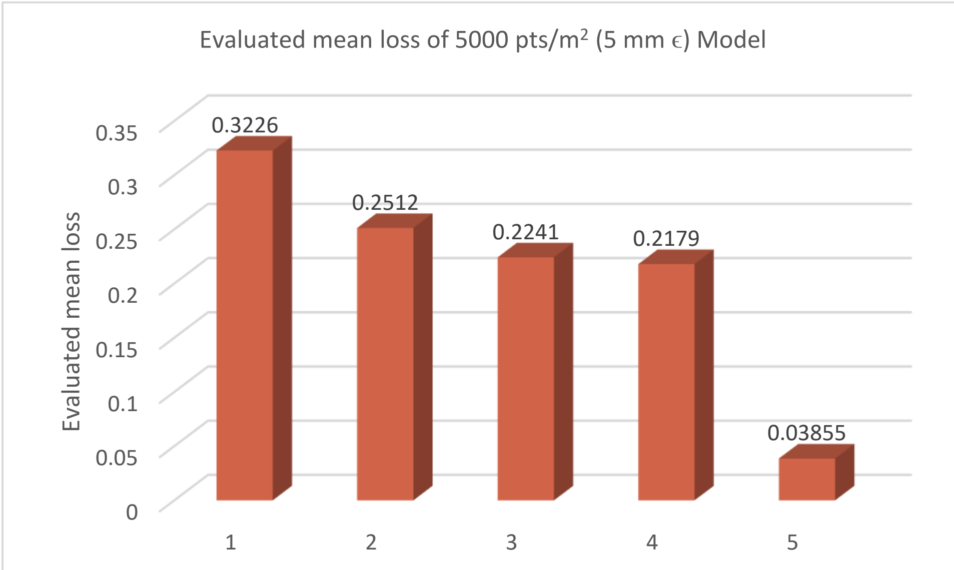

Figure 59. This shows the evaluated mean loss of the point cloud with the point density of 5000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

smaller when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. The evaluated mean loss of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably larger than the evaluated mean loss of the model for the testing point clouds with point densities larger than 50 pts/m^2. And with the increasing of point density, the evaluated mean loss decreased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model will make fewer mistakes in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the mistakes that the machine learning model makes in classifying simulated point clouds decreases.

Figure 60 shows the evaluated accuracy of the point cloud with a point density of 5000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2. It is clear that the evaluated accuracy of the model was

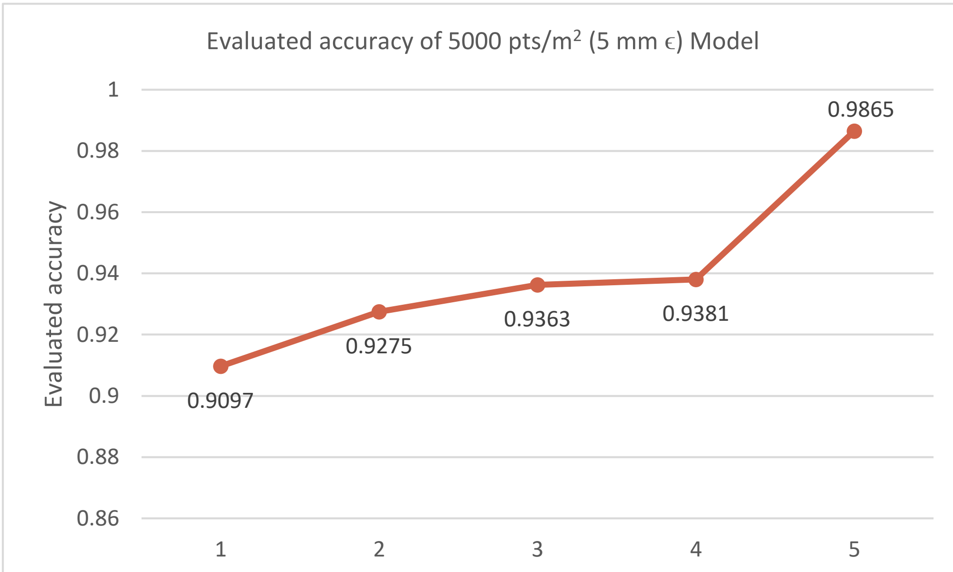

Figure 60. This shows the evaluated accuracy of the point cloud with the point density of 5000 pts/m^2 (5 mm ϵ), tested by 5 different point clouds (5 mm ϵ): 1. 50 pts/m^2, 2. 100 pts/m^2, 3. 500 pts/m^2, 4. 1000 pts/m^2, 5. 5000 pts/m^2.

higher when implemented on the testing data with the same point density than the other testing data with different point densities. This indicates that a machine learning model performs better when implemented on the point clouds with the same point clouds' density as the training point clouds. The evaluated accuracy of the model for the testing point cloud with a point density of 50 pts/m^2 was noticeably lower than the evaluated accuracy of the model for the testing point clouds with point densities that were larger than 50 pts/m^2. And with the increasing of point density, the evaluated accuracy also increased. This indicates that, if the testing point cloud's point density is different from the training point cloud, the machine learning model performs better in classifying simulated point clouds with larger point density than simulated point clouds with smaller point density. Moreover, if the testing point cloud's point density is different from the training point clouds' point density, with the increasing of point density, the difference of the accuracy of the machine learning model decreases.

4.1.2.2 The level of random error

This section analyzes how random error affects the segmentation and classification performance of the point clouds. In the following paragraphs, 5 pairs of point clouds were compared. In each pair, the point clouds had the same point density but different levels of random error. One of them was generated without random errors and the other one was added with random shifts to the point cloud with a ϵ of 5 mm.

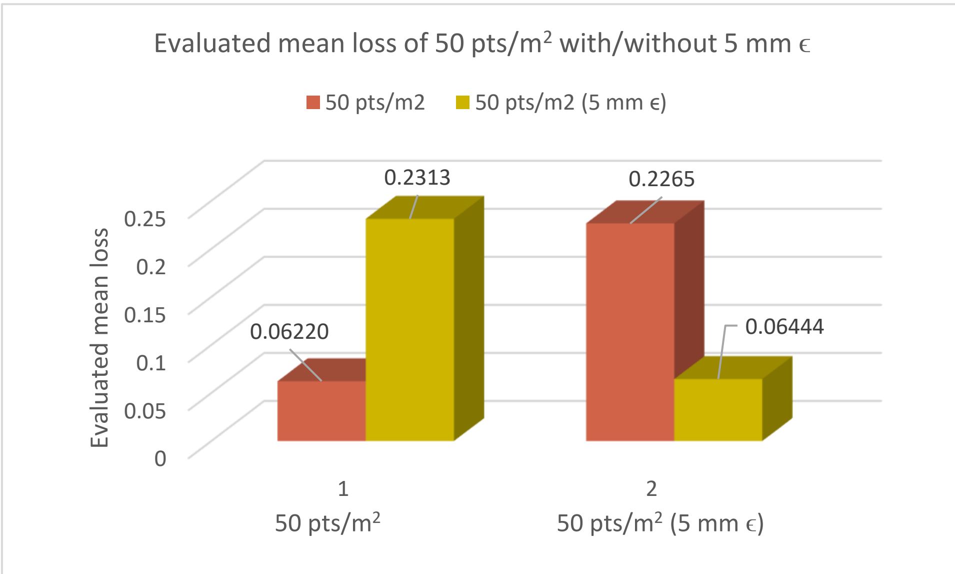

Figure 61. This shows the evaluated mean loss of two point clouds with the point density of 50 pts/m^2 (no random error) and 50 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 50 pts/m^2 (no random error) 2. 50 pts/m^2 (5 mm ϵ)

Figure 61 shows the evaluated mean loss of two point clouds with a point density of 50 pts/m^2 (no random error) and 50 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 50 pts/m^2 (no random error) 2. 50 pts/m^2 (5 mm ϵ). It is clear that the machine learning models made fewer mistakes when implemented on the testing point clouds with the same level of random error. And when the point density was 50 pts/m^2, the two models, the one with random errors and the other one without random errors, made almost the same altitude of mistakes in each situation.

Figure 62 shows the evaluated accuracy of two point clouds with a point density of 50 pts/m^2 (no random error) and 50 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 50 pts/m^2 (no

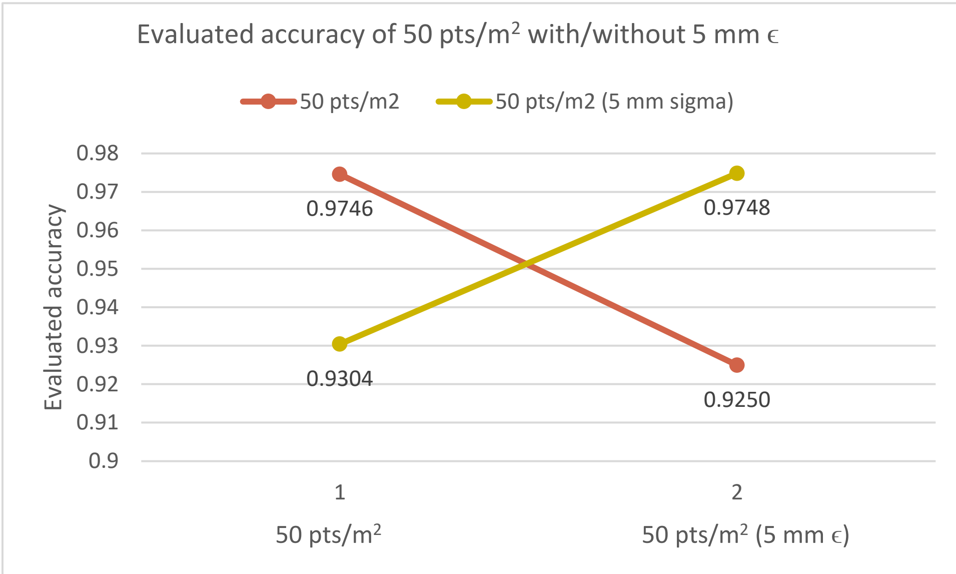

Figure 63. This shows the evaluated mean loss of two point clouds with the point density of 50 pts/m^2 (no random error) and 50 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 50 pts/m^2 (no random error) 2. 50 pts/m^2 (5 mm ϵ)

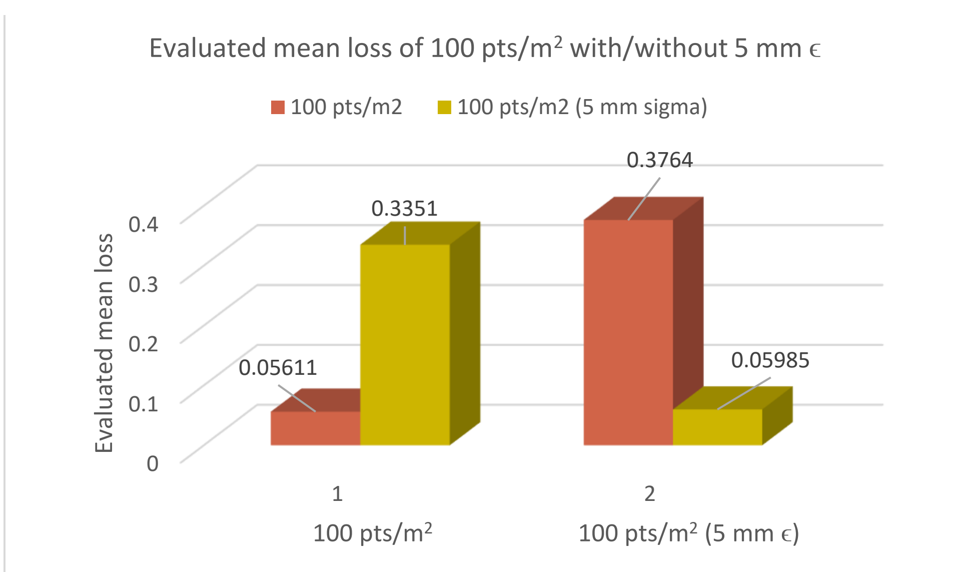

Figure 62. This shows the evaluated mean loss of two point clouds with the point density of 100 pts/m^2 (no random error) and 100 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 100 pts/m^2 (no random error) 2. 100 pts/m^2 (5 mm ϵ)

random error) 2. 50 pts/m^2 (5 mm ϵ). It is clear that the evaluated accuracy of the machine learning models was higher when implemented on the testing point clouds with the same level of random error. This indicates that a machine learning model performs better when implemented on the point clouds with the same level of random error as the training point clouds. And when the point density was 50 pts/m^2, the point cloud with random errors performed a little bit better.

Figure 63 shows the evaluated mean loss of two point clouds with a point density of 100 pts/m^2 (no random error) and 100 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 100 pts/m^2 (no random error) 2. 100 pts/m^2 (5 mm ϵ). It is clear that the machine learning models made fewer mistakes when implemented on the testing point clouds with the same level of random error. And when the point density was 100 pts/m^2, the evaluated mean losses were almost the same when implemented on the point clouds with the same level of error as the training point cloud. However, when implemented on the point cloud with different level of random error, the model trained by the point cloud with a 5 mm ϵ made fewer mistakes than the model trained by errorless point cloud.

Figure 64 shows the evaluated accuracy of two point clouds with a point density of 100 pts/m^2 (no random error) and 100 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 100 pts/m^2 (no random error) 2. 100 pts/m^2 (5 mm ϵ). It is clear that the evaluated accuracy of the machine

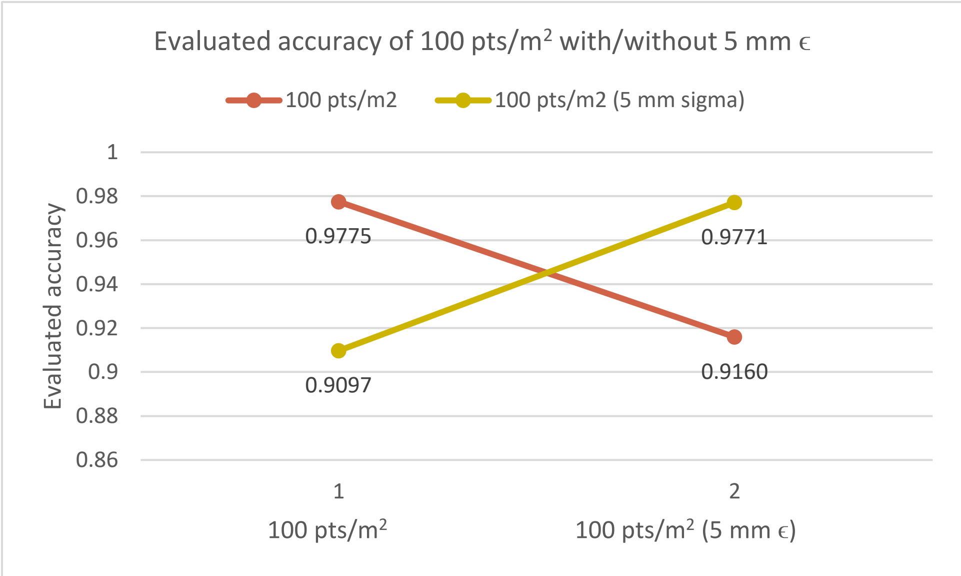

Figure 64. This shows the evaluated accuracy of two point clouds with the point density of 100 pts/m^2 (no random error) and 100 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 100 pts/m^2 (no random error) 2. 100 pts/m^2 (5 mm ϵ)

learning models was higher when implemented on the testing point clouds with the same level of random error. This indicates that a machine learning model performs better when implemented on the point clouds with the same level of random error as the training point clouds. And when the point density was 100 pts/m^2, the point cloud with random errors performed a little bit better when implemented on the point cloud with the same level of error, although the difference was very small. However, when implemented on the point cloud with a different level of error, the figure indicates that the model trained by the point cloud without random errors performs better than the model trained by the point clouds with the same point density but with a 5 mm ϵ.

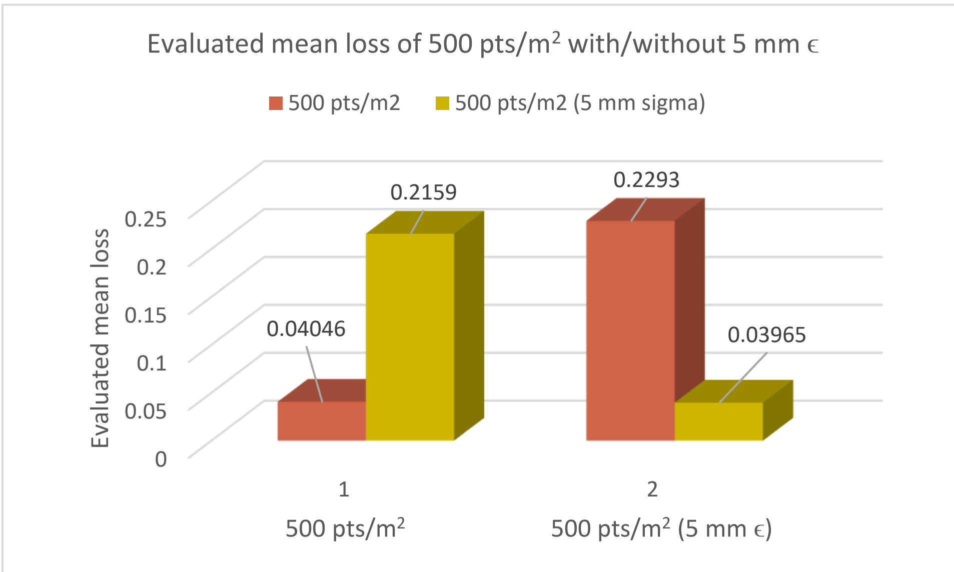

Figure 65. This shows the evaluated mean loss of two point clouds with the point density of 500 pts/m^2 (no random error) and 500 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 500 pts/m^2 (no random error) 2. 500 pts/m^2 (5 mm ϵ)

Figure 65 shows the evaluated mean loss of two point clouds with a point density of 500 pts/m^2 (no random error) and 500 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 500 pts/m^2 (no random error) 2. 500 pts/m^2 (5 mm ϵ). It is clear that the machine learning models made fewer mistakes when implemented on the testing point clouds with the same level of random error. And when the point density was 500 pts/m^2, the evaluated mean losses were almost the same when implemented on the point clouds with the same level of error as the training point cloud. However, when implemented on the point cloud with different level of random error, the model trained by the point cloud with a 5 mm ϵ made fewer mistakes than the model trained by the errorless point cloud.

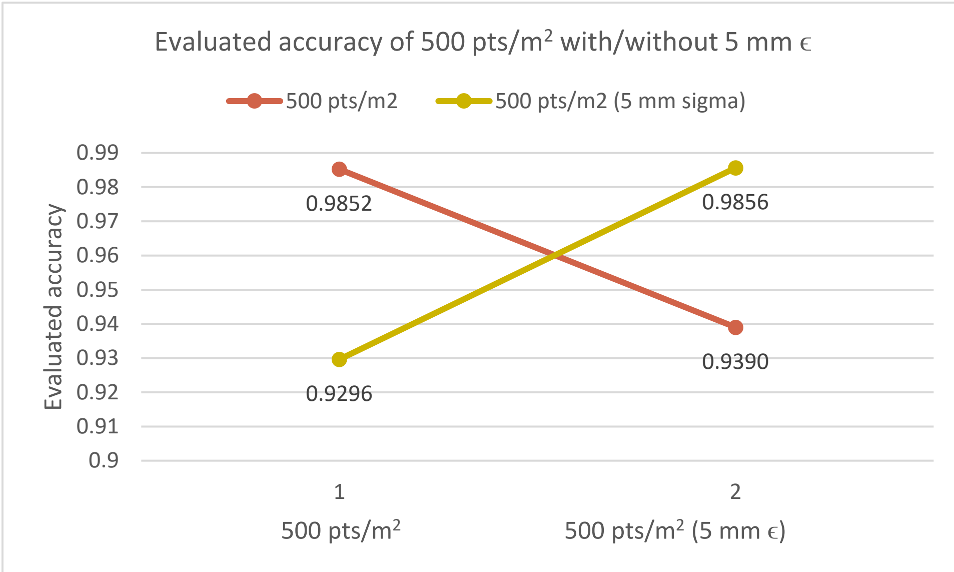

Figure 66. This shows the evaluated accuracy of two point clouds with the point density of 500 pts/m^2 (no random error) and 500 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 500 pts/m^2 (no random error) 2. 500 pts/m^2 (5 mm ϵ)

Figure 66 shows the evaluated accuracy of two point clouds with a point density of 500 pts/m^2 (no random error) and 500 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 500 pts/m^2 (no random error) 2. 500 pts/m^2 (5 mm ϵ). It is clear that the evaluated accuracy of the machine learning models was higher when implemented on the testing point clouds with the same level of random error. This indicates that a machine learning model performs better when implemented on the point clouds with the same level of random error as the training point clouds. And when the point density was 500 pts/m^2, the point cloud with random errors performed a little bit better when implemented on the point clouds with the same level of random error as the training point clouds. However, when implemented on the point cloud with a different level of random error, the model trained by the point cloud with a 5 mm ϵ made fewer mistakes than the model trained by the errorless point cloud. This indicates that a model trained by point cloud with a 5 mm ϵ performs better than a model trained by point clouds with the same point density but no random error when they are both implemented on the point cloud with a different level of random error.

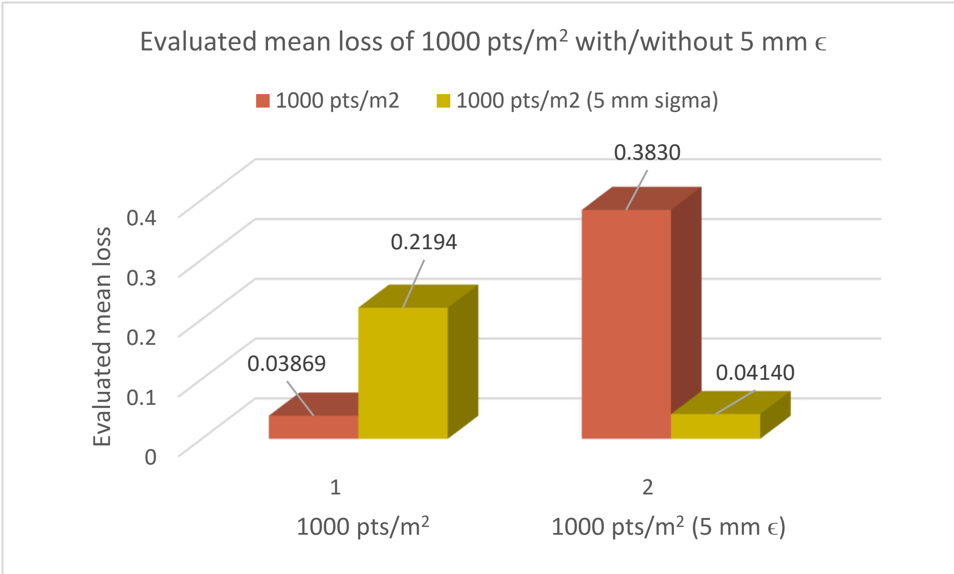

Figure 67. This shows the evaluated mean loss of two point clouds with the point density of 1000 pts/m^2 (no random error) and 1000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 1000 pts/m^2 (no random error) 2. 1000 pts/m^2 (5 mm ϵ)

Figure 67 shows the evaluated mean loss of two point clouds with a point density of 1000 pts/m^2 (no random error) and 1000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 1000 pts/m^2 (no random error) 2. 1000 pts/m^2 (5 mm ϵ). It is clear that the machine learning models made fewer mistakes when implemented on the testing point clouds with the same level of random error. And when the point density was 1000 pts/m^2, the evaluated mean losses were almost the same when implemented on the point clouds with the same level of error as the training point cloud. However, when implemented on the point cloud with different level of random error, the model trained by the point cloud with a 5 mm ϵ made fewer mistakes than the model trained by the errorless point cloud. And the difference of the evaluated mean losses was larger than the difference when the point density was 50 pts/m^2, 100 pts/m^2, and 500 pts/m^2.

Figure 68 shows the evaluated accuracy of two point clouds with a point density of 1000 pts/m^2 (no random error) and 1000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 1000 pts/m^2 (no random error) 2. 1000 pts/m^2 (5 mm ϵ). It is clear that the evaluated accuracy of the machine learning models was higher when implemented on the testing point clouds with the same level of random error. This indicates that a machine learning model performs better when implemented on the point clouds with the same level of random error as the training point clouds. And when the point density was 1000 pts/m^2, the two models, the one with random errors and the other one without random errors, the point cloud with random errors performed a little bit better when implemented on the point clouds with the same level of random error as the training point clouds. However, when implemented on the point cloud with different levels of random error, the model trained by the point cloud without random errors had higher accuracy than the model trained by the point cloud with a 5 mm ϵ. This indicates that a model trained by the point cloud with a 5 mm ϵ performs better than a model trained by the point clouds with the same point density but no random error when implemented on the point cloud with different levels of

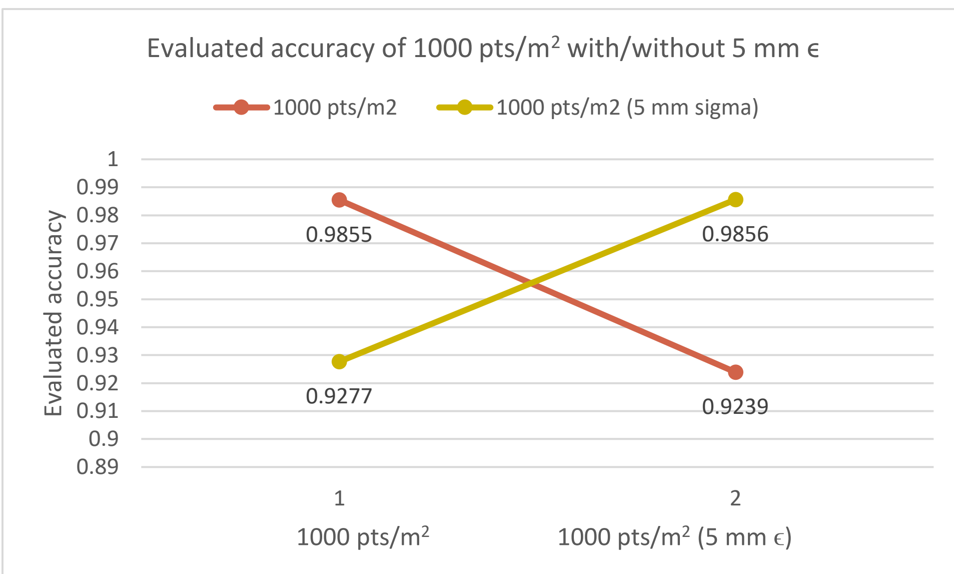

Figure 68. This shows the evaluated accuracy of two point clouds with the point density of 1000 pts/m^2 (no random error) and 1000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 1000 pts/m^2 (no random error) 2. 1000 pts/m^2 (5 mm ϵ)

random error. This result is different from the results of point clouds with a point density of 100 pts/m^2 or 500 pts/m^2, but the same as the results of point clouds with a point density of 50 pts/m^2.

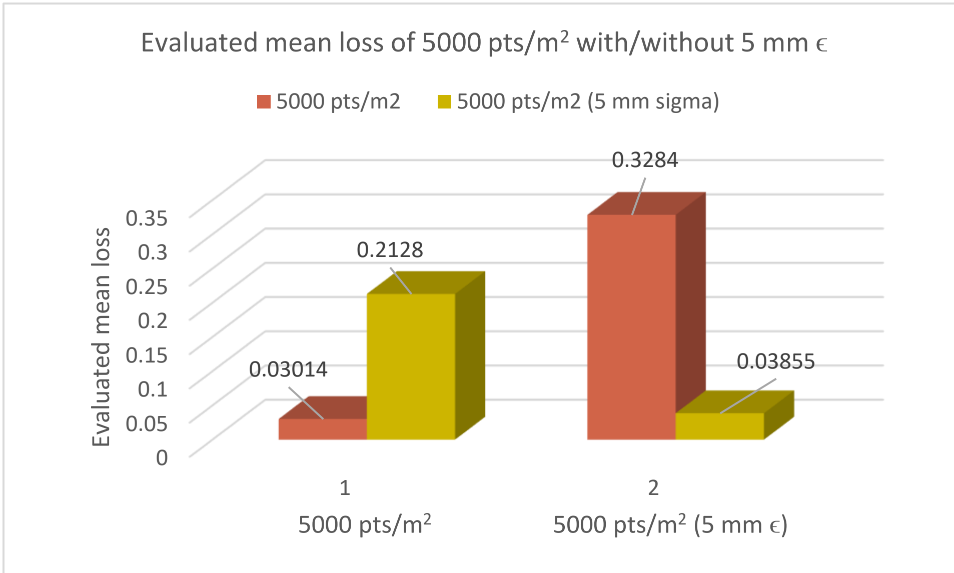

Figure 69. This shows the evaluated mean loss of two point clouds with the point density of 5000 pts/m^2 (no random error) and 5000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 5000 pts/m^2 (no random error) 2. 5000 pts/m^2 (5 mm ϵ)

Figure 69 shows the evaluated mean loss of two point clouds with a point density of 5000 pts/m^2 (no random error) and 5000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 5000 pts/m^2 (no random error) 2. 5000 pts/m^2 (5 mm ϵ). It is clear that the machine learning models made fewer mistakes when implemented on the testing point clouds with the same level of random error. And when the point density was 5000 pts/m^2, the evaluated mean losses were almost the same when implemented on the point clouds with the same level of error as the training point cloud. However, when implemented on the point cloud with different levels of random error, the model trained by the point cloud with a 5 mm ϵ made fewer mistakes than the model trained by the errorless point cloud. And the difference of the evaluated mean losses was larger than the difference when the point densities were 50 pts/m^2, 100 pts/m^2, and 500 pts/m^2, but smaller than the difference when the point density was 1000 pts/m^2.

Figure 70 shows the evaluated accuracy of two point clouds with a point density of 5000 pts/m^2 (no random error) and 5000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 5000 pts/m^2 (no random error) 2. 5000 pts/m^2 (5 mm ϵ). It is clear that the evaluated accuracy of the machine learning models was higher when implemented on the testing point clouds with the same level of random error. This indicates that a machine learning model performs better when implemented on the point clouds with the same level of random error as the training point clouds. And when the point density was 5000 pts/m^2, the two models, the one with random errors and the other one without random errors, the point cloud without random errors performed a little bit better when implemented on the point clouds with the same level of random error as the training point clouds. However, when implemented on the point cloud with different levels of random error, the model trained by the point cloud with a 5 mm ϵ had higher accuracy than the model trained by the point cloud without random errors. This indicates that a model trained by the point cloud with a 5 mm ϵ performs better than a model trained by the point clouds with the same point density but no random error when implemented on the point cloud with different levels of random error. This result is different from the result of point clouds with a point density of 100 pts/m^2 or 500 pts/m^2, but the same with the result of point clouds with a point density of 50 pts/m^2.

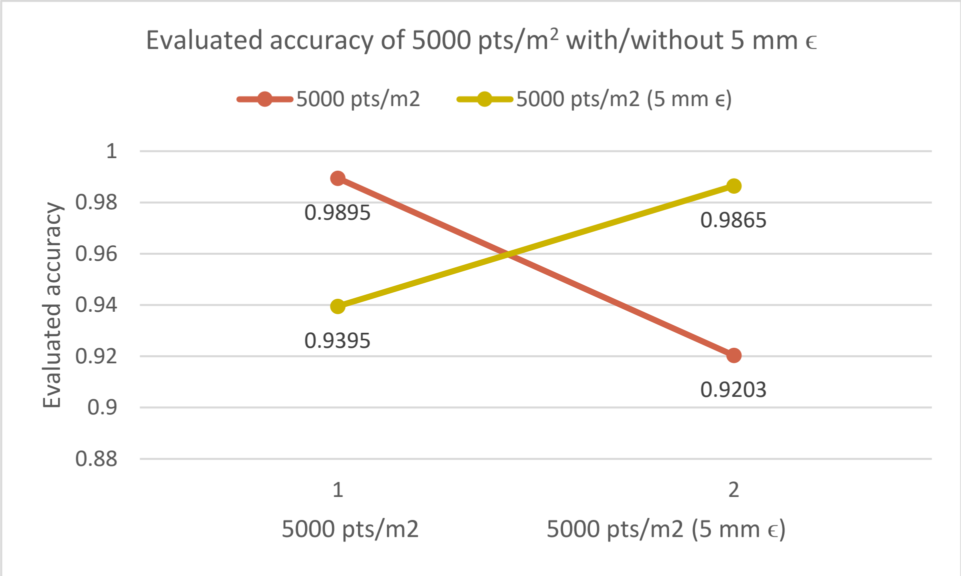

Figure 70. This shows the evaluated accuracy of two point clouds with the point density of 5000 pts/m^2 (no random error) and 5000 pts/m^2 (5 mm ϵ), tested by 2 point clouds: 1. 5000 pts/m^2 (no random error) 2. 5000 pts/m^2 (5 mm ϵ)

After 5 sets of paired experiments, it is clear that the machine learning models performed better when the models were implemented on the testing point clouds with the same level of random errors. Despite that there were differences in the evaluated accuracies, there was no pattern that revealed which models were better because the differences were too small and random. For example, when the models were implemented on the testing point clouds with the same level of random errors, the models without random errors performed a little bit better when the point densities were 100 pts/m^2, 5000 pts/m^2. When the models were implemented on the testing point clouds with the different level of random errors, the models without random errors performed a little bit better when the point densities were 100 pts/m^2, 500 pts/m^2.

4.2.3 Segmentation and classification visualization

Qualitative analyses were conducted in the way of visualization. The color coding of the points was based on Table.1. Some of the classification results are displayed in the figures below.

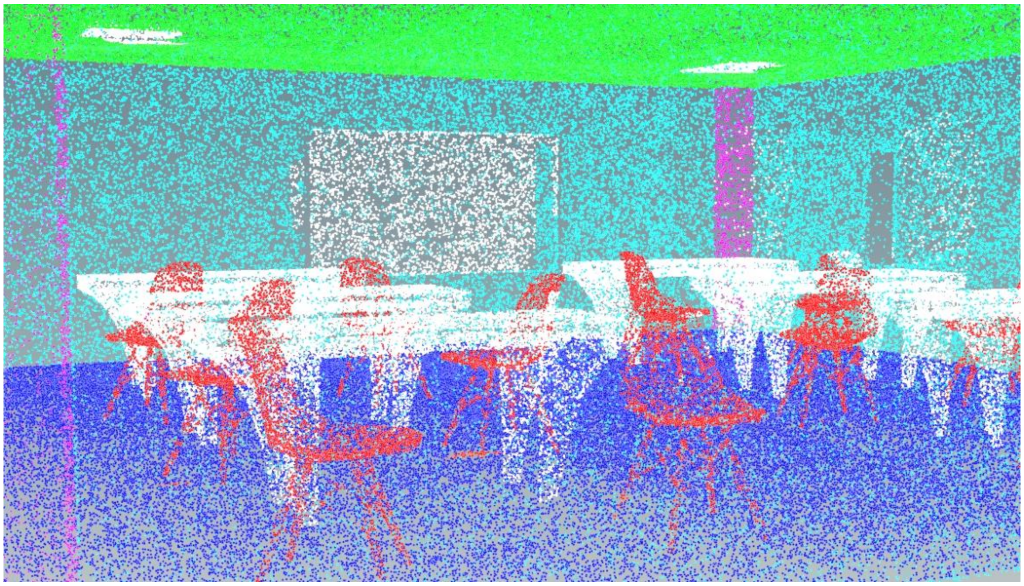

Figure 71. The visualization of the classification results of the machine learning model trained by a point cloud with a point density of 50 pts/m^2 (without random error), implemented on a point cloud with a point density of 5000 pts/m^2 (without random error).

Figure 71 and Figure 72 visually and directly show the performance of two models trained by different training data. The two models both did a great job on classifying tables and chairs. However, it was difficult for the model trained by the 5000 pts/m^2 density point cloud to identify planar objects, e.g. board. Thus, the evaluated accuracy of the results in Figure 71 was 93.38% while the evaluated accuracy of the results in Figure 72 was 89.10%. This visualization is representative of the difference in evaluated accuracy of these two models.

Figure 73 and Figure 74 are the visualizations of models with different levels of random error. In Figure 73, the right side of the figure is the visualization of the classification results of the machine learning model trained by a point cloud with a point density of 500 pts/m^2 (without random error), implemented on a point cloud with a point density of 500 pts/m^2 (5 mm ϵ). The

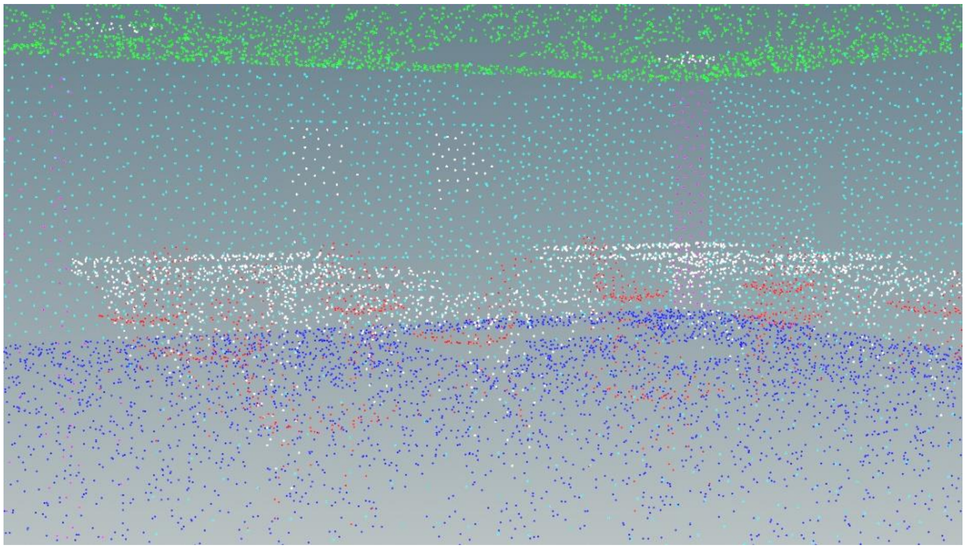

Figure 72. The visualization of the classification results of the machine learning model trained by a point cloud with a point density of 5000 pts/m^2 (without random error), implemented on a point cloud with a point density of 50 pts/m^2 (without random error).

left side of the figure is the ground truth. This also indicates that the model performs poorly on planar objects, e.g. the door, the wall with a door. In Figure 74, the right side of the figure is the

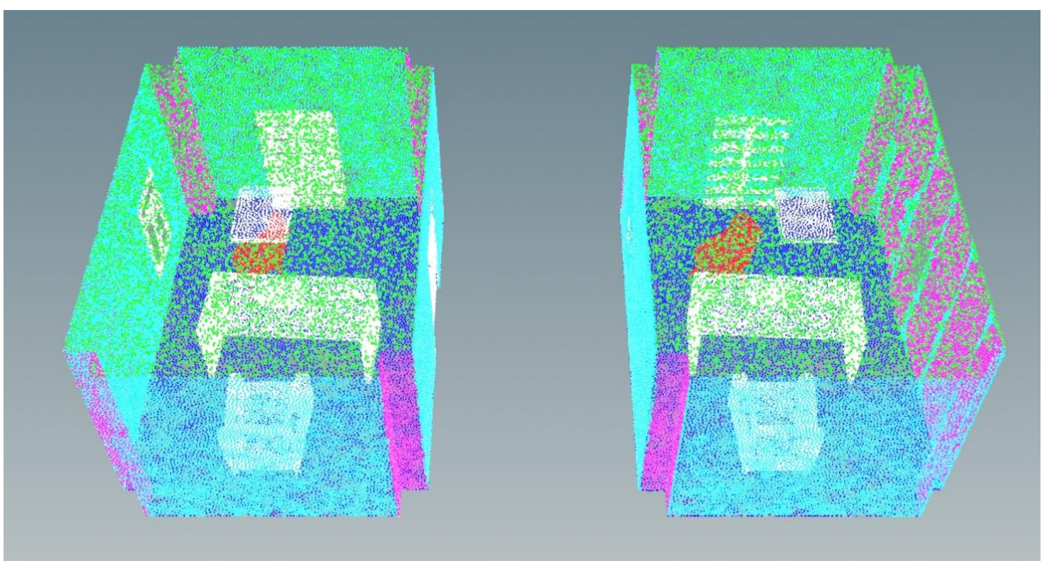

Figure 73. The right side of the figure is the visualization of the classification results of the machine learning model trained by a point cloud with a point density of 500 pts/m^2 (without random error), implemented on a point cloud with a point density of 500 pts/m^2 (5 mm sigma). The left side of the figure is the ground truth.

visualization of the classification results of the machine learning model trained by a point cloud with a point density of 500 pts/m^2 (without random error), implemented on a point cloud with a point density of 500 pts/m^2 (5 mm ϵ). The left side is the visualization of the classification results of the machine learning model trained by a point cloud with a point density of 500 pts/m^2 (5 mm ϵ), implemented on a point cloud with a point density of 500 pts/m^2 (without random error). The figure infers that the two models perform equally poor on planes.

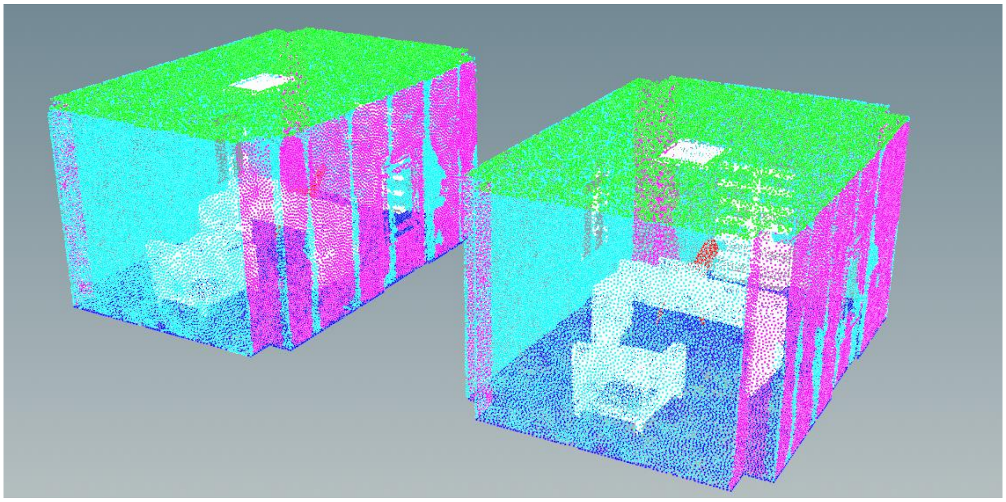

Figure 74. The right side of the figure is the visualization of the classification results of the machine learning model trained by a point cloud with a point density of 500 pts/m^2 (without random error), implemented on a point cloud with a point density of 500 pts/m^2 (5 mm sigma). The left side is the opposite.

4.3 Summary

The mean loss of the model is used for measuring how poor the predictions are, but accuracy is the essential measurement of the performance of a model. The performance of different models and how point cloud attributes affect these different models are summarized as follows.

Firstly, all of the 10 models' performances were gradually getting better before over-fitting. The evaluated accuracies of the models became steady after 15 to 20 epochs. All the comparisons were based on the final models, which were at epoch 20.

Secondly, the models performed well overall except for thin, planar objects, e.g., walls, doors, boards, and windows.

Thirdly, all the models performed the best when the testing point clouds' point density was the same as the training point clouds' point density. And the larger the point density was, the better performance the model provided. However, when the storage and computation consumption were taken into consideration, it is not worth of spending a lot more time and computation resources for a little bit higher accuracy, with respect to this study. When the models were implemented on other point clouds with different point densities, the accuracy dropped. When the models were implemented on point clouds with larger point density, the accuracy did not vary a lot in testing data. The testing point cloud with larger point density had only a slightly higher accuracy, but when the models were implemented on point clouds with smaller point density, the difference in accuracies were relatively big. The smaller the point density was, the larger the drop was in accuracy. Nevertheless it seemed that, when a model trained with smaller point density performed on a point cloud with a relatively larger point density, and a model trained with the same larger point density performed on a point cloud with the same smaller point density, the former performed better than the latter. But further experiments need to be conducted with smaller point density difference between testing groups.

Fourth, all the models performed the best when the testing point clouds' level of random error was the same as the training point clouds' level of error. The random error did affect the performance of the machine learning models, but there was not a direct relation between them. Some of the models with random error performed better, but some of them did not. The values of random errors were only set to 2, so this experiment only indicates that adding a 5 mm sigma to the point clouds does not increase nor decrease the accuracy of the classification. Therefore, more values and intervals are needed to be set for further research.

Finally, the results of simulated models on real-world data and real-world models on simulated data were displayed. Figure 75 displays the evaluated accuracies of the original model trained by real-world data with a point density of about 11067 pts/m^2 (non-uniform through the point clouds, with random errors), implemented on the 10 simulated point clouds. For the testing point clouds, the larger the point density was, the higher accuracy the model had. But the

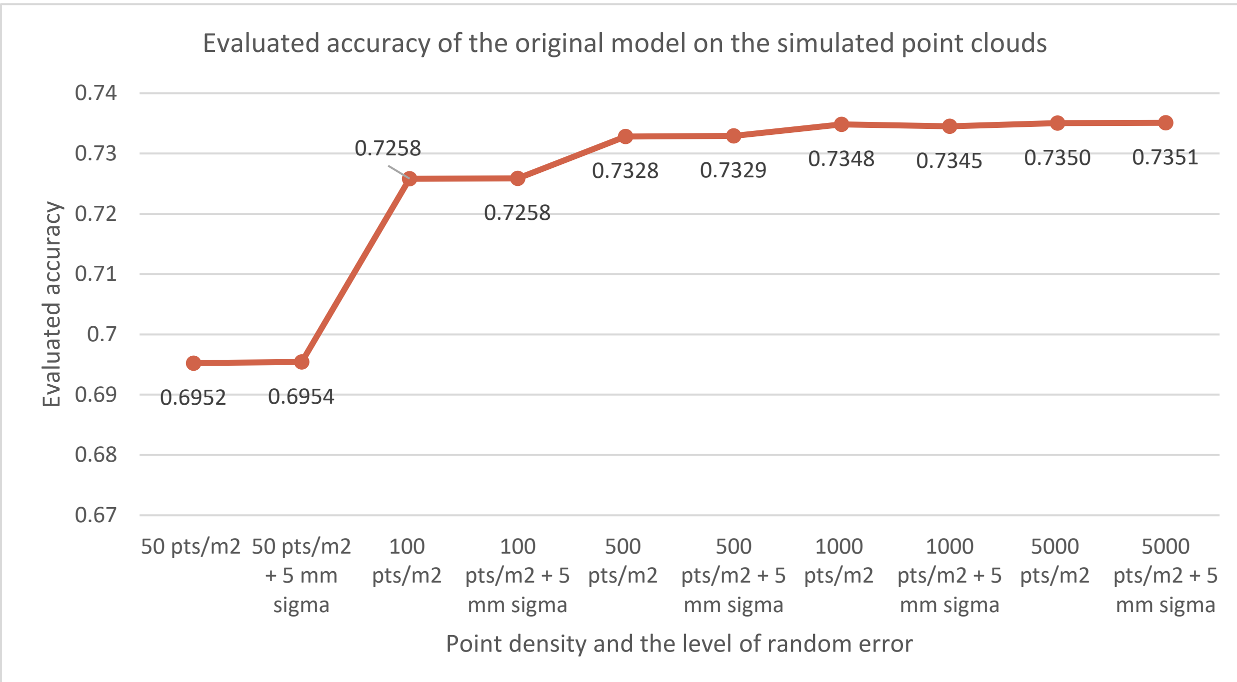

Figure 75. This shows the evaluated accuracy of the original model on the simulated point clouds.

difference was small when the point density was over 500 pts/m^2. As shown in the paired point clouds (with or without random errors), random errors in testing data did not significantly affect the performance. Figure 76 displays the evaluated accuracies of the 10 models trained by simulated point clouds, implemented on the real-world point clouds. The evaluated accuracies varied from one model to another one. For this experiment, the point density and the level of random error did not affect the models' performance on real-world point clouds, and the accuracies were relatively low compared to the evaluated accuracies of the model trained by the real-world data, as shown in Figure 75.

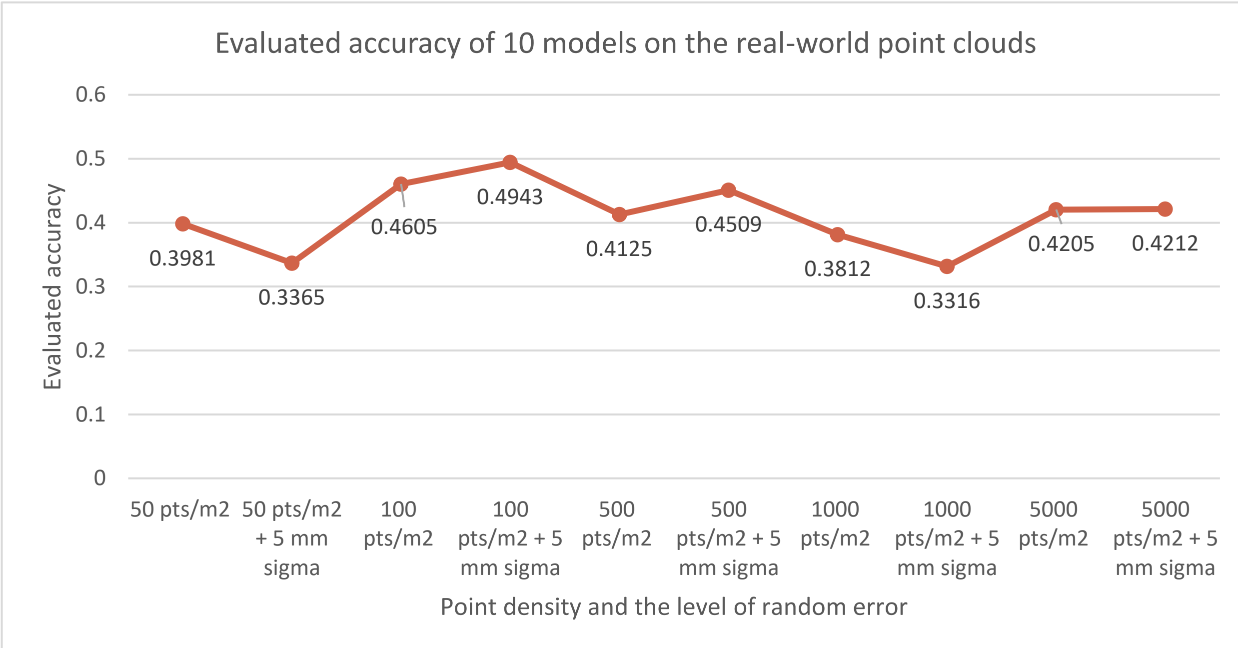

Figure 76. This shows the evaluated accuracy of 10 models on the real-world point clouds.

5. CONCLUSIONS

Semantic building reconstruction is an essential element for building maintenance and management in current society. LiDAR and photogrammetry are two main techniques to acquire information from existing buildings. However, the outcome of these techniques is raw without semantic information. Therefore, further point cloud processing is as important as the acquisition. Segmentation and classification are two key steps in semantic building modeling using point clouds. But these two procedures have been mostly done manually and semi-automatically. Rapid construction requires automation in the field, and researchers have produced a lot of automatic segmentation and classification approaches in the latest decade. Most of these approaches were machine learning approaches. The performance of machine learning segmentation and classification methods depend on many aspects, including the algorithm and point cloud attributes, e.g. point density, the level of random error, point distribution, dataset size, etc. Nevertheless, in real-world environments, it is difficult and almost impossible to control all of the aspects when acquiring the point cloud data. Therefore, it is difficult to decouple all the point cloud attributes that affect the performance of the segmentation and classification methods. This study proposed a method of simulating point clouds in virtual environments using a scattering algorithm in Houdini. 10 different point clouds were generated to find out the relation between the point cloud attributes, including point density and the level of random errors, and the performance of a machine learning segmentation and classification method, PointNet (Charles et.al., 2017).

Two measurements were used in the experiment. The mean loss was used to evaluate how poor the prediction was during each epoch, and the accuracy was used as the measurement for evaluating the performance of the models. The study found that all the models were better for non-planar objects, e.g., chairs, tables, lights, etc. But the models had a hard time distinguishing between planar objects, e.g., a door, a window, from other planar objects that were intersecting with it, e.g., a wall. In terms of point density, the machine learning model performed better when the point density of the testing point clouds was the same as the point density of the training point clouds. Therefore, it is important to know the project's point clouds point density before the training of the model and to adjust the point density of the training point clouds to match with the target point clouds that need semantic processing. The larger the point density is, the better the model will perform. However, when the storage and computation consumption are taken into consideration, it is maybe not worth of spending a lot more time and computation resources for a little bit higher accuracy. Therefore, all factors need to be considered in order to produce a cost-effective model. When the models were not applied to point clouds with the same density, the performance was affected negatively. And based on the limited number of tests, it is better to apply a model trained by point clouds with smaller point density on the target point clouds with larger point density rather than the opposite. But more experiments need to be conducted to make a solid conclusion regarding this. If the point density of the target point cloud is larger than the point density of the training point cloud, the increase of the point density does not significantly affect the performance. But if the point density of the target point cloud is smaller than the point density of the training point cloud, the decrease of the point density negatively affects the performance substantially. In terms of the level of random errors, all the models performed the best when the level of random errors of the testing point clouds were the same as the training point clouds. This finding indicates that, in real-world environments, the point cloud acquisition tool for training data should be the same acquisition tool for the target point cloud data, because different tools produce different levels of random error. Based on the limited number of tests, there was no direct relation found between the level of random error and the performance of the model.

The performance of the models by simulated data was relatively poor when applied to real-world data. Future works will be conducted to refine the machine learning algorithm and the simulated point clouds, in order to provide better performance for real-world environments. Specially, other than optimizing the original algorithm, more machine learning algorithms will be tested in future works. In terms of the simulated points, this study only used positional values of the points. Future works will add weights to RGB values of the points. More intervals need to be added to the studies point cloud attributes to convey a more general conclusion, and more point cloud attributes need to be taken into consideration. The simulated indoor environments need more complexity and variety, e.g., the logic of the generation, the structure of the room, the types of the room, the sizes of the room, the number of objects, etc. The ideal product of this study is a system that creates segmentation and classification models based on simulated point clouds that adjust automatically according to the point cloud attributes of the target real-world point clouds in order to provide better performance than models derived from real-world point cloud.

REFERENCES

Adán, A., Quintana, B., Prieto, S. A., & Bosché, F. (2018). Scan-to-BIM for 'secondary' building components. Advanced Engineering Informatics, 37(January), 119–138.

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., & Savarese, S. (2016). 3D semantic parsing of large-scale indoor spaces. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December, 1534–1543. https://doi.org/10.1109/CVPR.2016.170

Barazzetti, L., Banfi, F., Brumana, R., Gusmeroli, G., Previtali, M., & Schiantarelli, G. (2015). Cloud-to-BIM-to-FEM: Structural simulation with accurate historic BIM from laser scans. Simulation Modelling Practice and Theory, 57, 71–87.

Barazzetti, L. (2016). Parametric as-built model generation of complex shapes from point clouds. Advanced Engineering Informatics, 30(3), 298–311.

Barazzetti, L. (2018). Point cloud occlusion recovery with shallow feedforward neural networks. Advanced Engineering Informatics, 38(September), 605–619.

Bueno, M., Bosché, F., González-Jorge, H., Martínez-Sánchez, J., & Arias, P. (2018). 4-Plane congruent sets for automatic registration of as-is 3D point clouds with 3D BIM models. Automation in Construction, 89(January), 120–134.

Bechtold, S., & Höfle, B. (2016). HELIOS: A multi-purpose LiDAR simulation framework for research, planning and training with airborne, ground-based mobile and stationary platforms. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 3(June), 161–168. https://doi.org/10.5194/isprs-annals-III-3-161-2016

Biswas, H. K., Bosché, F., & Sun, M. (2015). Planning for scanning using building information models: A novel approach with occlusion handling. 32nd International Symposium on Automation and Robotics in Construction and Mining: Connected to the Future, Proceedings, 1–8.

Bosche, F., & Haas, C. T. (2008). Automated retrieval of 3D CAD model objects in construction range images. Automation in Construction, 17(4), 499–512. https://doi.org/10.1016/j.autcon.2007.09.001

Bosché, F. (2010). Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction.

Advanced Engineering Informatics, 24(1), 107–118. https://doi.org/10.1016/j.aei.2009.08.006

Bosché, F., Ahmed, M., Turkan, Y., Haas, C. T., & Haas, R. (2015). The value of integrating Scan-to-BIM and Scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components. Automation in Construction, 49, 201–213.

Boulch, A., Gorce, M. de La, & Marlet, R. (2014). Piecewise-planar 3D reconstruction with edge and corner regularization. Eurographics Symposium on Geometry Processing, 33(5), 55–64. https://doi.org/10.1111/cgf.12431

Boulch, A., Guerry, J., Le Saux, B., & Audebert, N. (2018). SnapNet: 3D point cloud semantic labelling with 2D deep segmentation networks. Computers and Graphics (Pergamon), 71, 189–198. https://doi.org/10.1016/j.cag.2017.11.010

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.

Chabat, F., Yang, G.Z., Hansell, D.M., ()1999. A corner orientation detector. Image Vis. Comput. 17, 761–769.

Charles R. Qi*, Hao Su*, Kaichun Mo, Leonidas J. Guibas, Stanford University. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, Conference on Computer Vision and Pattern Recognition (CVPR) 2017.

C. Dore, M. Murphy, Semi-automatic generation of as-built BIM façade geometry from laser and image data, J. Inform. Technol. Constr. 19 (2014) 20–46.

C.M. Eastman, BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors, Wiley, 2011.

C. Eastman, P. Teicholz, R. Sacks, K. Liston, BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Architects, Engineers, Contractors, and Fabricators, Wiley, Hoboken, NJ, 2011.

Dimitrov, A., & Golparvar-Fard, M. (2015). Segmentation of building point cloud models including detailed architectural/structural features and MEP systems. Automation in Construction, 51(C), 32–45. https://doi.org/10.1016/j.autcon.2014.12.015

E.A. Pärn, D.J. Edwards, M.C.P. Sing, The building information modelling trajectory in facilities management: a review, Autom. Constr. 75 (2017) 45–55.

F. Bosché, C.T. Haas, Automated retrieval of 3D CAD model objects in construction range images, Autom. Constr. 17 (2008) 499–512.

F. Bosché, Automated recognition of3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction, Adv. Eng. Inform. 24 (1) (2010) 107–118.

F. Bosché, A. Guillemet, Y. Turkan, C.T. Haas, R.G. Haas, Tracking the Built Status of MEP Works: Assessing the Value of a Scan-vs.-BIM System, ASCE J. Comput. Civ. Eng. (2014), http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000343.

F.N. Bosche, S. O'Keeffe, The need for convergence of BIM and 3D imaging in the open world, Proceedings in the CitA BIM Gathering Conference, The Construction IT Alliance, Dublin, Ireland, 2015, pp. 109–116.

Grilli, E., Menna, F., & Remondino, F. (2017). A review of point clouds segmentation and classification algorithms. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, 42(2W3), 339–344. https://doi.org/10.5194/isprs-archives-XLII-2-W3-339-2017

G. Carbonari, S. Stravoravdis, C. Gausden, Building information model implementation for existing buildings for facilities management: a framework and two case studies, WIT Trans. Built Environ. 149 (2015) 395–406.

Hall, D., & Giglio, N. (2016). Architectural Graphic Standards, 12th Edition. The American Institute of Architects

Heipke, C., Mayer, H., Wiedemann, C., Jamet, O. (1997). Evaluation of automatic road extraction. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Part 3-4W2. pp. 47–56.

Jagannathan, A. and Miller, E. L. (2007). Three-dimensional surface mesh segmentation using curvedness-based region growing approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29(12), pp. 2195-2204.

Jung, J., Hong, S., Jeong, S., Kim, S., Cho, H., Hong, S., & Heo, J. (2014). Productive modeling for development of as-built BIM of existing indoor structures. Automation in Construction, 42, 68–77.

Jung, J., Jwa, Y., & Sohn, G. (2017). Implicit regularization for reconstructing 3D building rooftop models using airborne LiDAR data. Sensors (Switzerland), 17(3). https://doi.org/10.3390/s17030621

Jung, J., Stachniss, C., Ju, S., & Heo, J. (2018). Automated 3D volumetric reconstruction of multiple-room building interiors for as-built BIM. Advanced Engineering Informatics, 38(February), 811–825.

Jung, J., & Sohn, G. (2019). A line-based progressive refinement of 3D rooftop models using airborne LiDAR data with single view imagery. ISPRS Journal of Photogrammetry and Remote Sensing, 149(July 2017), 157–175. https://doi.org/10.1016/j.isprsjprs.2019.01.003

Kovesi, P.D., (2011). MATLAB and Octave Functions for Computer Vision and Image Processing. Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia.

Kukko, A. and Hyypp¨a, J. (2009). Small-footprint Laser Scanning Simulator for System Validation, Error Assessment, and Algorithm Development. Photogrammetric Engineering and Remote Sensing 75(10), pp. 1177–1189.

Lari, Z. (2014). Adaptive Processing of Laser Scanning Data and Texturing of the Segmentation Outcome. https://doi.org/http://hdl.handle.net/11023/1672

Li, D., Liu, J., Feng, L., Zhou, Y., Liu, P., & Chen, Y. F. (2020). Terrestrial laser scanning assisted flatness quality assessment for two different types of concrete surfaces. Measurement, 154, 107436. https://doi.org/10.1016/j.measurement.2019.107436

Lohani, B. and Mishra, R. K., (2007). Generating lidar data in laboratory: Lidar simulator. In: International Archive of Photogrammetry and Remote Sensing XXXVI(3)W52 of Laser Scanning 2007 and SilviLaser 2007, p. 6.

M. Ahmed, C.T. Haas, R. Haas, Autonomous modeling of pipes within point clouds, Proceedings of the 30th ISARC, Montréal, Canada, 2013, pp. 1093–1100.

M. Ahmed, C.T. Haas, R. Haas, Automatic detection of cylindrical objects in built facilities, ASCE J. Comput. Civ. Eng. 28 (3) (2014), http://dx.doi.org/10.1061/ (ASCE)CP.1943-5487.0000329

Ng, R. T. and Han, J. (1994). Efficient and effective clustering methods for spatial data mining. Proc. 20th Int. Conference on Very Large Databases.

P. Pishdad-Bozorgi, X. Gao, C. Eastman, A.P. Self, Planning and developing facility management-enabled building information model (FM-enabled BIM), Autom. Constr. 87 (2018) 22–38.

Park, Y., & Guldmann, J. M. (2019). Creating 3D city models with building footprints and LIDAR point cloud classification: A machine learning approach. Computers, Environment and Urban Systems, 75(November 2018), 76–89. https://doi.org/10.1016/j.compenvurbsys.2019.01.004

Quintana, B., Prieto, S. A., Adán, A., & Bosché, F. (2018). Door detection in 3D coloured point clouds of indoor environments. Automation in Construction, 85(October 2017), 146–166.

Rabbani, T., Van Den Heuvel, F., & Vosselmann, G. (2006). Segmentation of point clouds using smoothness constraint. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 36(5), pp. 248-253.

Rebolj, D., Pučko, Z., Babič, N. Č., Bizjak, M., & Mongus, D. (2017). Point cloud quality requirements for Scan-vs-BIM based automated construction progress monitoring. Automation in Construction, 84(August), 323–334. https://doi.org/10.1016/j.autcon.2017.09.021

Rottensteiner, F., Sohn, G., Gerke, M., Wegner, J. D., Breitkopf, U., & Jung, J. (2014). Results of the ISPRS benchmark on urban object detection and 3D building reconstruction. ISPRS Journal of Photogrammetry and Remote Sensing, 93, 256–271. https://doi.org/10.1016/J.ISPRSJPRS.2013.10.004

Silverman, B.W. (1986). Density Estimation for Statistics and Data Analysis. New York: Chapman and Hall.

V. Pătrăucean, I. Armeni, M. Nahangi, J. Yeung, I. Brilakis, C. Haas, State of research in automatic as-built modelling, Adv. Eng. Inf. 29 (2015) 162–171.

Valero, E., Bosché, F., & Forster, A. (2018). Automatic segmentation of 3D point clouds of rubble masonry walls, and its application to building surveying, repair and maintenance. Automation in Construction, 96(September), 29–39.

Wang, Y. b., Xie, D. b., Yan, G. b., Zhang, W. b. and Mu, X. b. (2013). Analysis on the inversion accuracy of LAI based on simulated point clouds of terrestrial LiDAR of tree by ray tracing algorithm. International Geoscience and Remote Sensing Symposium (IGARSS) pp. 532–535.

Wang, C., Cho, Y. K., & Kim, C. (2015). Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. Automation in Construction, 56, 1–13.

Weinmann, M., Schmidt A., Mallet C., Hinz S., Rottensteiner F., Jutzi B. (2015). Contextual classification of point cloud data by exploiting individual 3D neighbourhoods. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. II(3-W4).

X. Liu, M. Eybpoosh, B. Akinci, Developing as-built building information model using construction process history captured by a laser scanner and a camera, Proceedings in the Construction Research Congress, ASCE, Indiana, USA, 2012, pp. 1232-1241.

Yousefhussien, M., Kelbe, D. J., Ientilucci, E. J., & Salvaggio, C. (2018). A multi-scale fully convolutional network for semantic labelling of 3D point clouds. ISPRS Journal of Photogrammetry and Remote Sensing, 143, 191–204. https://doi.org/10.1016/j.isprsjprs.2018.03.018

Zhang, L., Li, Z., Li, A., & Liu, F. (2018). Large-scale urban point cloud labeling and reconstruction. ISPRS Journal of Photogrammetry and Remote Sensing, 138, 86–100. https://doi.org/10.1016/j.isprsjprs.2018.02.008

Zhu, L. (2015). A Pipeline of 3D Scene Reconstruction from Point Clouds. https://doi.org/10.3390/rs3071406