Delft University of Technology

Master Thesis

A deep learning method for 3D point cloud segmentation of building facades

Author name:	Zhuo Chen	
Student number:	4844068	
Start date:	January, 2020	
Thesis committee:	Dr. R. Lindenbergh,	TU Delft, supervisor
	Dr. S. Verhagen,	TU Delft
	Dr. Y. Zang,	Nanjing University of Information Science & Technology

Delft University of Technology

Abstract

Semantic segmentation, a task vital in the creation of 3D point cloud models for buildings, is aimed at assigning meaning to individual points. However, due to the vast volume of unstructured point cloud data, precise semantic segmentation remains challenging. Significant progress has been observed in recent years with the application of deep learning techniques to point cloud segmentation, and the effectiveness of Dynamic Graph Convolutional Neural Network (DGCNN) and K-Nearest Neighbors (K-NN) in handling point cloud data has been recognized.

In this study, Dynamic Graph Convolutional Neural Network (DGCNN) was utilized for semantic segmentation on a building's point cloud scene. We adopted K-Nearest Neighbors (K-NN) as a crucial component of our methodology to optimize the segmentation process. By varying 'k' values in K-NN and exploring different block sizes, we aimed to obtain various segmentation results for comparison. When a block size of 1 meter was employed and 'k' was set to 20, an overall accuracy of 90.32%, mean accuracy of 87.64%, and IoU of 80.71% were achieved. However, the most favorable segmentation outcomes were observed when the block size remained 1 meter, and 'k' was set to 30, resulting in an overall accuracy of 93.86%, mean accuracy of 90.68%, and IoU of 84.97%.

These experiments underscore the significance of parameter selection in optimizing the performance of DGCNN for point cloud segmentation. The findings reveal that adjustments to 'k' values and block sizes can significantly influence segmentation accuracy and quality, emphasizing the importance of parameter optimization in the context of semantic segmentation for building point clouds using deep learning techniques. The utilization of K-NN played a crucial role in achieving these improvements by allowing us to adapt to the inherent variability in point cloud data.

1. Introduction

In Section 1.1, the current situation and difficulty of application of 3D point cloud is introduced. In Section 1.2 and 1.3, semantic segmentation and deep learning on 3D data are introduced. In Section 1.4, the main research question and related sub-questions are proposed.

1.1 Overview

Nowadays, the generation and reconstruction of three-dimensional (3D) urban building models became a hotspot for many researchers, since it is important to many aspects such as vegetation monitoring, [1], navigation of autonomously driving cars, [2], environmental modeling, [3], creation and interaction of virtual reality, [4], and so forth. In order to achieve this, point clouds are one of the most important candidate data sources. Over the last decade, Light Detection and Ranging, which is also known as "LiDAR", is one of the most widespread technologies in the field of remote sensing to acquire massive amounts of 3D point clouds which act as the input data sets for the aforementioned applications. But in practice, it is really computational expensive to process such a huge amount of points clouds for building the corresponding model or perform related computational analysis, [5], while the modeling in terms of building information management, [6], keep proposing prominent challenges. And with no doubt this is the situation we are facing, especially when the buildings in the model contain a variety of non-rectilinear features (e.g. curved windows) and sophisticated geometric items.

1.2 Semantic Segmentation

But different from non-semantic segmentation, semantic segmentation labels each point with semantic information. For the previous applications we describe, i.e. reconstruction of three-dimensional urban area models, segmentation, especially semantic segmentation, plays a fundamental but critical part. As a terminology of computer vision, segmentation refers to the process of classifying point clouds into multiple homogeneous regions, where points in the same segment will have the same properties, [7]. When semantic segmentation is applied, it divides input point cloud into different parts showing distinct semantic meaning, followed by assigning each part a label indicating each part to one of the pre-defined classes semantically. Apart from the reconstruction of 3D urban models, there are many other applications where semantic segmentation also shows its value. For example, in robotics, we can use segmentation to label objects in the surrounding environment of a robot. This is essential since these semantic labels can help a robot to identify each object in its surrounding areas, so that it can make further judgement.

Due to the increasing demand for 3D building models in environmental modeling, urban planning and some other aspects, facades semantic segmentation, especially detailed segmentation for the windows and doors of level of detail (LoD3) building models in CityGML, [8], has been regarded as an important issue in urban reconstruction. The existing facades segmentation approaches are commonly based on grammar rules or basic computer vision methods, [9]. Despite the fact that these approaches can achieve relatively satisfying results, there still exists several problems. On the one hand, these grammar rules originate from weak architectural principles, [10]. Furthermore, there are so many architecture styles in the world, which means the existing grammar rules are not sufficiently comprehensive for processing all kinds of building facades, [11]. On the other hand, some basic computer vision methods, like edge detection and region growing, rely on local gradient or different values of local average grayscale, [10]. Therefore, these methods may lack universality and they are easily affected by noise. Generally, these traditional methods are limited.

1.3 Deep Learning

In the past ten years, progress in the understanding of 3D sensed data has been obtained with the help of deep learning. With a range of important applications from indoor robotics navigation to national scale remote sensing, there is a high demand for algorithms that can learn to automatically understand and classify 3D sensed data, such as point clouds, [12]. For the 3D point cloud processing, many frameworks based on deep learning techniques, such as PointNet, [13], PointNet++, [14], DLA-Net, [47], U-Net, [49], and RandLA-Net, [50], were developed, which also show considerable potential for gaining high performance in different aspects such as classification, segmentation, and so forth. Therefore, deep learning, as one of the most powerful techniques, will continue to boost the development related to 3D point cloud processing.

For 3D point cloud tasks, traditional methods are usually conducted on the basis of specific handcrafted features, such as normals, with a specific classifier, and are often capable of producing satisfactory results, [15]. These handcrafted features can be based on geometry and frequency characteristics of point clouds. However, the extraction of crucial handcrafted features hinges on sufficient knowledge of the field and substantial experience, [15]. In contrast, deep learning algorithms possess the ability to learn so-called computer-designed features automatically, it normally requires complex network architecture and a considerable amount of calculation time, [15].

1.4 Research question and Sub-questions

In this study, the main research question is **How to apply a deep learning framework to perform point cloud semantic segmentation for building facades with high accuracy and efficiency?**

And some sub-questions are proposed below:

- **How to determine the input feature set for the deep learning framework?**
- **How to determine the structure of our deep learning framework?**
- **What is the training strategy?**
- **How to use Domain Adaptation (a technique used to minimize the potential impact of differences in data distribution between training and test datasets.) to minimize possible impact caused by different distribution between training and test datasets?**
- **How to eliminate the influence of measurement geometry, if it’s necessary?**
- **Which evaluation metrics should be chosen to measure the performance?**
- **How does the segmentation outcome fit the ground truth?**

- **How does the deep learning framework we apply perform in comparison to other deep or non-deep learning frameworks?**

2. Related Work

In this section, we will review previous methods for point cloud analysis briefly. Considering the procedure, point cloud semantic segmentation is quite similar to clustering-based point cloud segmentation. But different from non-semantic point cloud seg- mentation, point cloud semantic segmentation labels each semantic information for each point, which is more flexible compared to clustering-based method, [16].

2.1 Projection Networks

In several methods, points are projected to a kind of intermediate grid structure. The point cloud is used to render a series of 2D images at different viewpoint, [17][18][19], which are then processed by image-based networks. We often refer this kind of methods as multi-view. When it comes to scene segmentation, these methods often behave poorly because of point density variation and occluded surfaces. Different from determining a global viewpoint, [20], some people declared that local neighborhood points could be processed with 2D convolutions after projecting them to local tangent planes. However, the results of this method are strongly dependent on the estimation of local tangent planes.

In terms of voxel-based methods, points are usually projected on 3D voxel grids, [21][22][23]. Furthermore, larger grid size can be achieved through sparse structures such as hash-maps or octrees, which often lead to enhanced performance, [24][25]. However, these networks are often constrained by their kernels (usually $3^3 = 27$ or $5^3 = 125$ voxels), which means lack of flexibility. Moreover, for complicated tasks such as scene segmentation, it would make the design of their architectures more straightforward if intermediate structures like 2D images or 3D grid voxels can be avoided.

2.2 Graph Convolution Networks

There are several different ways about defining a convolution operator on a graph. A convolution on a graph is focused on the surface indicated by the graph, [26][27][28][29], or it can be computed as a multiplication on its spectral representation, [30][31], both of which learn filters on edge relationships. In other words, the graph convolution groups features on local surface patches, which can recover the missing topological information of a point cloud, thus enriching the representation power of point cloud.

2.3 Pointwise MLP Networks

PointNet is considered to be a milestone in point cloud deep learning, [32]. This network proposed using a shared MLP (Multilayer Perceptron, a class of feedforward artificial neural network, [33].) on each point individually, then a global max-pooling (a sample-based discretization process, the objective is to down-sample an input representation such as image, reducing its dimensionality.) is followed to produce global features. MLP in the network is like a series of learned encodings, and the global feature of the point cloud is obtained as the largest response of all the points in terms of each encoding. However, PointNet is also limited as the local features of point clouds are excluded. From then on, several architectures such as PointNet++ have been designed to take local information into consideration with MLPs, [14][34][35].

2.4 Point Convolution Networks

Recently, some works define the kernel point Convolution, which directly performs operation on the input point cloud, with no need of any intermediate representation.

Pointwise CNN (Convolutional Neural Networks, a class of deep neural networks, [36].), [37] uses voxel bins to determine the kernel weights, which is a little bit similar to the voxel-based network, and thus lacks flexibility. Moreover, expensive computation cost imposed by its normalization operation is also a problem. The kernels are modeled by linear functions in the Flex-convolution network, [38], resulting in its limited power of representation. Furthermore, this network also suffers from the problem of point density variation. In SpiderCNN, [39], a set of polynomial functions are used to model its kernel, with different weights for each neighborhood. However, this network is not spatially consistent, since the weight used for each neighborhood relies on the distance-wise order of the neighborhood. PCNN, [40], proposed the idea that the kernel weights could be carried by the points, and a correlation function is used to model these weights. Nonetheless, this neural network fails to account for neighboring points in its computations. This results in convolution computations being performed in a quadratic fashion with respect to the number of points, rendering the network non-scalable. KPConv, [32], also follows the general idea of PCNN that uses points to carry kernel weights, but differs in many details. For example, unlike PCNN, a simple linear correlation function is used in KPConv to model the weights, which greatly improves the gradient back-propagation operation. Moreover, KPConv also proposed a deformable point convolution for the first time, which can further boost the performance of point convolution for complicated tasks.

2.5 Random Forest-Based Method

Random Forest (RF) is a robust ensemble learning algorithm that leverages the power of decision trees. It has been effectively employed in the segmentation of 3D point clouds due to its ability to handle complex and high-dimensional data. Researchers have explored its application in diverse domains, including urban scene analysis, object detection, and building facade segmentation.

A notable advantage of RF is its capacity to accommodate both geometric and contextual features extracted from point clouds. This versatility allows RF-based models to capture intricate relationships among points, making them well-suited for tasks like distinguishing between different facade components, such as windows, walls, and openings.

The work by [46] represents a significant contribution in the field of point cloud segmentation, as it specifically focuses on the supervised detection of façade openings using thermal attributes.

While not a Random Forest-based approach, this research lays the foundation for exploring alternative methods that incorporate thermal information for improved segmentation accuracy.

3. Data Pre-processing

In Section 3.1, the data used in the study is introduced. In Section 3.2, the denoising and labeling work are introduced. In the Section 3.3, the downsampling methods are introduced.

3.1 Data Source

The 3D point cloud data of buildings come from some teaching buildings and dormitory buildings of Nanjing University of Information Science and Technology. The 3D laser scanner is a RIEGL VZ 2000i with a scanning accuracy of 3mm and a maximum measurement range of 2500m. At the same time, it is a Riegl with a Nikon camera to collect the color information of the point cloud, which is suitable for building scanning scenarios. The main scanning area is the west building of Nanjing University of Information Science and Technology (yellow circle part in figure 1). 4 to 5 stations are set up for the same building to obtain the overall structure of the building, and multiple facades of the same style of building are also scanned. The specific building point cloud data is shown in Figure 2.

Fig 1: Scanning area on Nanjing University of Information Science and Technology, indicated by a yellow circle on the left

Fig 2: Part of the scanned building point cloud

3.2 Denoising and calibration of point cloud data

It can be seen from Figure 2 that the original building point cloud contains a large number of irrelevant points, so it is necessary to denoise the point cloud and extract the facade part of the building. Façade extraction can be realized through automatic or manual extraction. The experimental data has too much redundancy and too much interference information. At the same time, it is not difficult to simply extract building facades, and the workload is not high, so manual extraction is adopted. In the process of building facade extraction, the choice of software for visualizing building point clouds is particularly important. This experiment mainly uses Cloud Compare software to visualize point cloud data, and the subsequent point cloud calibration is also completed on Cloud Compare.

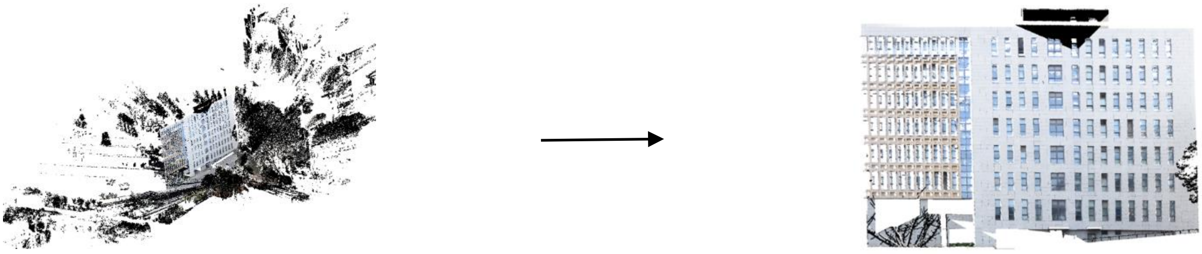


Fig 3: Manual denoising

The architectural point cloud scanned in this experiment contains rich detailed features, such as walls, windows, air conditioners, doors and other detailed building components. The research objects of this paper are mainly windows and building exterior walls. Manually extract windows and distinguish different attribute features on building facades. In the process of processing, special structures that need to be distinguished are extracted. The purpose is to prepare data sets for subsequent network training, verification, and testing.

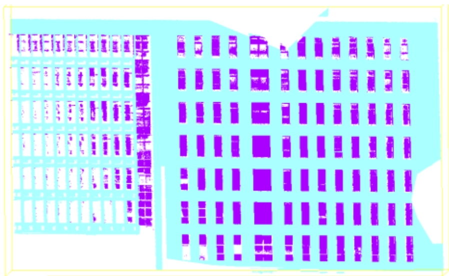


Fig 4: Ground truth: manually classified building point cloud

3.3 Downsampling

Due to the massive and disordered nature of point clouds, direct processing requires high computational costs when searching for neighborhoods. A commonly used solution is to downsample the point cloud, and convert the operation of the entire point cloud to the points obtained by downsampling to reduce the amount of calculation.

3.3.1 Uniform Downsampling

There are many different sampling methods for uniform downsampling, [44], of which the farthest point sampling is the most commonly used method. Select any point in the point cloud as the seed point, and set an interior point set. Place the seed point into the interior point set. Each time, find the point farthest from the interior point set from the point cloud and place it in the interior point set. The point placed in the interior point set is deleted from the point cloud. The distance between a point in a point cloud and an interior point set is the minimum distance between that point and all points in the interior point set.

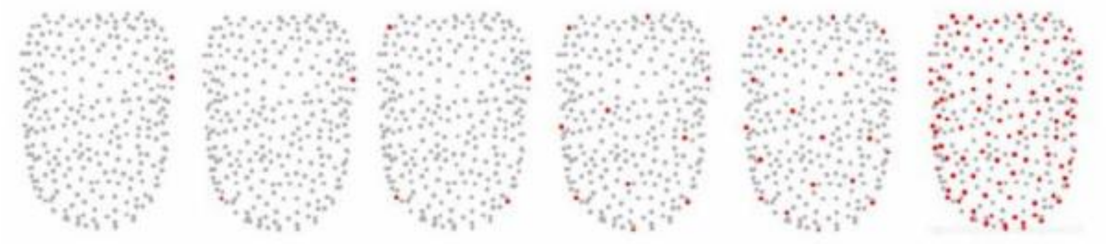


Fig 5: Uniform downsampling

3.3.2 Curvature Downsampling

Principle of curvature downsampling, [45]: In traditional surveying and mapping, in order to describe an object in detail, more piecemeal measurements are set in places with obvious changes, so that the fitting results are closer to the real situation. In areas with significant changes, the curvature of the point cloud also increases, so it is necessary to increase the density of the point cloud in that area.

The characteristics of this sampling method are: in areas with significant changes, the higher the density of the point cloud in the obtained results, which can fully reflect the changing characteristics of that part; Classifying and sampling high and low curvature can ensure uniform sampling of specific terrain features.

3.3.3 Voxel downsampling

Voxel downsampling, [44], is the process of voxelizing a three-dimensional space and then sampling a point within each voxel. Typically, the center point or the point closest to the center can be used as the sampling point. The specific method is as follows:

1. Create voxels: Calculate the bounding box of the point cloud, and then discretize the bounding box into small voxels
2. Each voxel contains several points, and the center point or the point closest to the center point is set as the sampling point

The characteristics of voxel sampling are: very high efficiency; The distribution of sampling points is relatively uniform, but the uniformity is not as high as uniform sampling; The distance between points can be controlled by the size of voxels; The number of sampling points cannot be precisely controlled. With reasonable parameters, voxel downsampling will produce sufficiently accurate results to reduce CPU power consumption; The voxelated point cloud data will be stored in an orderly manner in memory, which is beneficial for reducing random memory access and increasing data processing efficiency; Benefiting from the orderly storage and downsampling of data brought by voxelization, this method can process point cloud data with large order of magnitude.

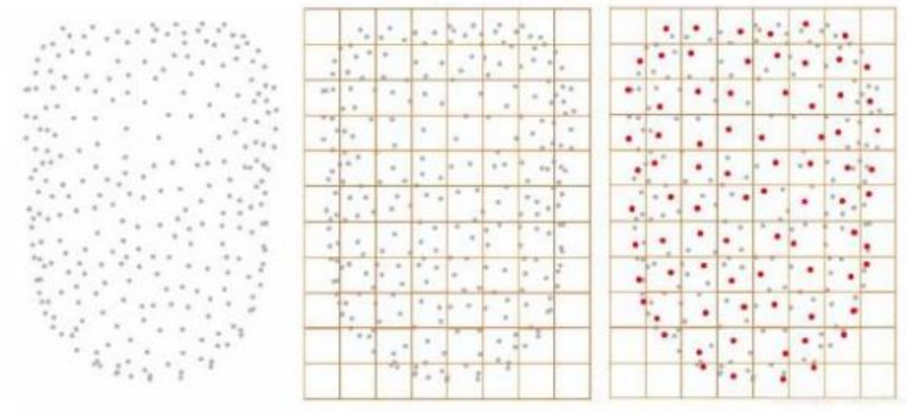


Fig 6: Voxel downsampling

This experiment needs a sampling method that can retain detail information, extract local features at multiple levels, and has high computational efficiency, so this experiment uses voxel downsampling method for downsampling. The uniform downsampling algorithm has a large amount of computation, a long time, and low efficiency. Curvature downsampling needs to calculate curvature information, which takes a long time, and is not suitable for this article; Building facades are mostly regular geometry, so surface downsampling is not used.

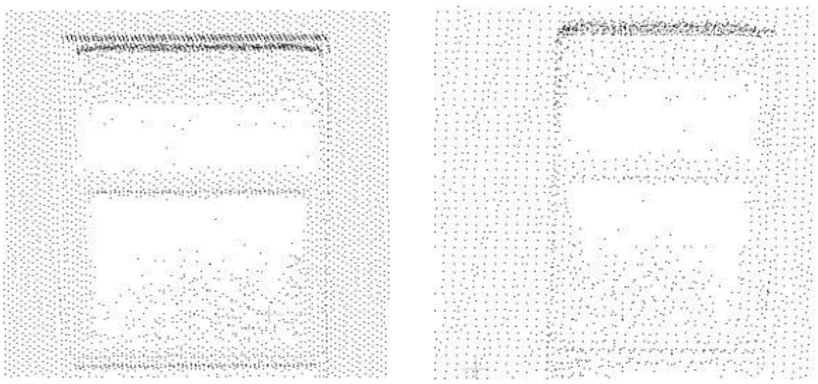


Fig 7: Comparison before and after voxel downsampling

The point cloud before down-sampling consists of 3,011,839 points, and after down-sampling of 911,095 points.

4. Method

In Section 4.1, a data augmentation method is applied on training data to produce more façade data. In Section 4.2, the principle of the DGCNN model is introduced and the role of the loss function is discussed. In Section 4.3, the geometric method is introduced, which could also be used to complete the segmentation task on simple facades.

4.1 Point Cloud Data Augmentation

Data enhancement, [43] is a very effective method to solve the problem of overfitting. It is a common data preprocessing method of convolutional neural networks. For example, when the amount of data on hand is too small, some meaningful data can be generated manually for training. The prominent advantages of this data acquisition method are: low cost, good effect. In addition, when the data set used for classification has data skew, that is, one type of samples is much more than the other, data augmentation can be performed on a class with fewer samples. It assumes that more information can be extracted from the original dataset through augmentation, making the enhanced dataset represent a more comprehensive dataset, thereby narrowing the gap between the training set and the validation set.

Due to the limited sample data collected in this experiment, it is necessary to increase the point cloud data required for training through data augmentation. At the same time, considering the main segmentation of walls and window construction, neural network learning mainly focuses on depth features. Therefore, the augmentation method of this experiment consists of performing a 90 transformation on the same wall around the Z-axis to generate four walls.

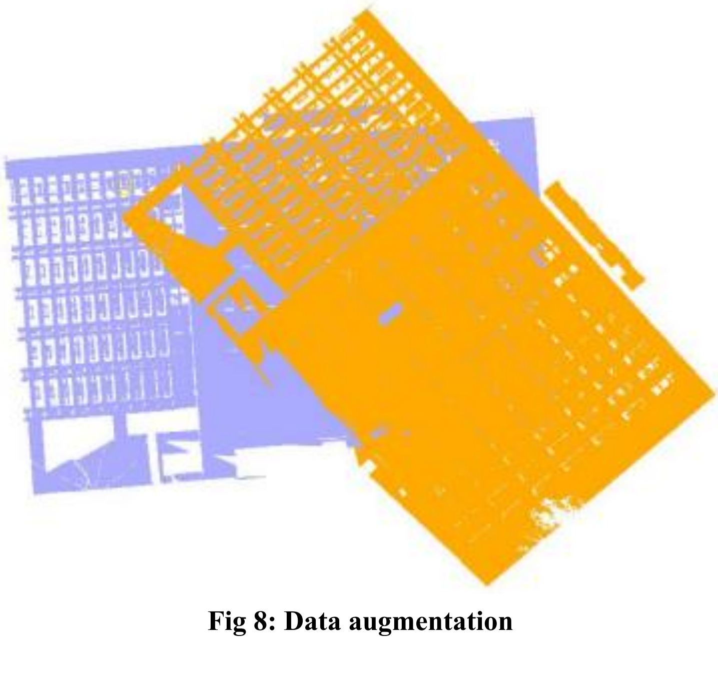


Fig 8: Data augmentation

4.2 Network Design

4.2.1 DGCNN Network Architecture

The main structure of DGCNN, [41], refers to PointNet, [13], which inputs the entire point cloud data and directly outputs the segmentation results, all of which are end-to-end neural network structures.

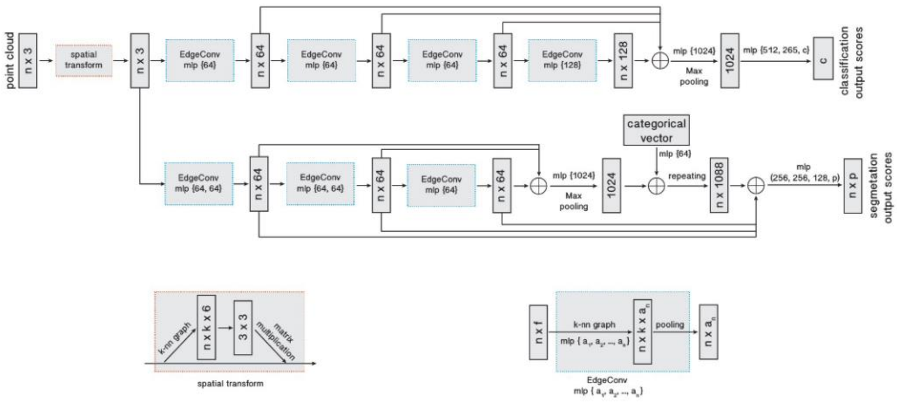


Fig 9: DGCNN Network Architecture

Overview of DGCNN architecture, with each input point containing 3 features (such as x, y, and z). The segmentation model extends the classification model by combining global feature vectors with all local features generated by EdgeConv, [41]. The spatial transformation block is implemented through a 3 × 3 matrix to align the input point cloud. This matrix is expected to be an orthogonal matrix and is estimated during training. In EdgeConv operation, the dimension f of edge features of each point is calculated by applying a multi-layer perceptron. The number of layer neurons of a multi-layer perceptron is defined as $\{a_1,a_2,...,\}$. And after pooling the edge features, a tensor of shape $(n \times a_n)$ will be finally generated.

In this study, DGCNN was used for semantic segmentation of point clouds, which takes the entire point set as input and outputs a semantic label for each point. DGCNN uses a basic version of PointNet as the backbone network.

Based on the architecture of PointNet, DGCNN integrates local features by replacing MLP with edge convolution operations (EdgeConv). DGCNN first constructs a directed graph $G = (V,\ E)$ representing the internal local structure of the point cloud, where $V = \{1, \dots, N\}$denote vertices and $E \in V \times V$denote edges. In the simplest case, G is a k-nn adjacency graph. DGCNN does not directly convolution point features, but first calculates the K edge features related to its nearest neighboring points for each point, which can obtain local features of the point cloud.

4.2.2 Classification of Building Components Based on DGCNN

Point cloud data is massive and uneven, so it cannot be directly calculated and needs to be divided into blocks. According to the requirements of neural network parameter settings, the number of points in each block is 4096. Point cloud blocks with less than 4096 points are upsampled and integrated into the point cloud block to increase the number of point clouds to the standard of 4096; Resampling dense point cloud blocks is used to reduce the number of point operations to the standard of 4096.

In the feature dimension, the coordinates of the original point cloud after blocking are first centralized and normalized. All information is merged according to the original coordinates, centralized coordinates and normalized coordinates, added to the attribute information of the point cloud, and the Semantic information of the point cloud is merged together. The point cloud block is used as the processing unit, and the center of the point cloud block is set as the origin to establish a coordinate system, and the operation of recalculating each point cloud coordinate to the new coordinate system is centralization; Normalization involves linearly transforming the original coordinates into 0-1 to obtain the new coordinates. In this experiment, the annotation information of point clouds includes two types: walls and windows. For each point cloud, probability values are calculated based on the above three types for prediction. We take the category with the highest probability value as the category of the point. Therefore, in the training process of the neural network, the maximum probability value and the annotation are jointly used for loss calculation, and the parameters in the network are improved through backpropagation. During network training, in order to reduce the difference in contours of the same category, the point cloud blocks of each round of training are input in batches.

In order to study the feasibility of DGCNN on aerial point clouds and the effect of different effective ranges, we conducted experiments using five block sizes (10m, 3m, 1m, 0.5m, and 0.1m) and five k values (20, 25, 30, 35, and 40). To compare the performance of different settings, we first used k=30 as the default neighborhood size and changed the block size. Then, we used the block size that achieved the best segmentation result to explore the impact of different k values.

4.2.3 Loss Function Design

The loss function, [42], is an index used to measure the performance of the model. The larger the value of the loss function, the worse the performance of the model. The role of the neural network is to optimize the parameters to reduce the value of the loss function and find the best weight value for the performance of the entire neural network. The loss layer brings the predicted value and the real value into the loss function to obtain the current loss function and backpropagates it into the first layer of the neural network to continue training, improve the performance of the neural network, and achieve the minimum loss. The essence of classification and segmentation tasks in this paper is to classify point clouds. In neural networks, cross entropy loss function is usually used for classification tasks. When the performance of the model is poor, the cross-entropy loss function can speed up the optimization of the model, improve the optimization efficiency and reduce the training time by greatly modifying the parameters. The cross-entropy function is mainly concerned with the difference between the real value and the predicted value, so it is the best choice for classification tasks. The formula of cross entropy loss function is as follows:

$$C = -\frac{1}{n}\sum_{x}[y \ln a + (1-y)\ln(1-a)]$$

Among them, C is the loss value, n is the number of samples, x is the prediction vector dimension, y is the true value, and a is the predicted value.

This experiment evaluates the performance of all experiments using several indicators. The overall accuracy (Acc) is calculated by dividing the number of correctly classified points in all categories by the total number of predictions, and is determined. The formula to calculate Acc is as follow:

$$Acc = \frac{Number\ of\ correctly\ classified\ points}{Total\ number\ of\ points}$$

Due to the uneven number of points in different categories, such as points from windows being much less than those from walls, the average accuracy of each category (mAcc) is also calculated. The formula to calculate mAcc is as follow:

$$mAcc = \frac{1}{N}\sum_{i=1}^{N}\frac{C_{ii}}{t_i}$$

Where: N is the total number of classes; C_{ii} represents the number of points correctly classified in class i; t_i denotes the total number of points in class i.

In addition, we also evaluate the average of intersection over union (IoU) for each class. This is a commonly used indicator in semantic segmentation tasks. For point cloud data, the IoU from a class is calculated as:

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

In this paper, TP represents the number of correctly predicted window point clouds; FP represents the number of point clouds that are wrongly predicted as windows; FN represents the number of point clouds that are incorrectly predicted as non-windows; TN represents the number of point clouds that are correctly predicted as non-windows. In this In this work, we also determined the confusion matrix, which contains information for all classes of TP, FP, FN and TN.

4.3 Geometric Method

Apart from the deep learning method, a relatively simple method could also be adopted to carry out the segmentation task.

The general principle is that a plane is fitted to the point cloud. Then the distances between points and best plane could be calculated, which could be called ‘façade distance’.

The ‘façade distance’ could be a criterion to separate walls and windows, because the ‘façade distance’ of window points are usually different from the ‘façade distance’ of wall points. Therefore, a threshold could be adopted to separate two classes.

In this study, the geometric method is also applied on the test data, and the results are shown in the result chapter. The plane fitting and distance calculations are done in CloudCompare.

5. Results

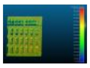
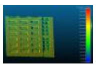
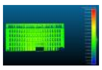
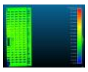
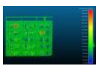
In Section 5.1, training settings would be generally introduced, including training data, validation data and test data, plus other parameters such as batch_size and epoch. In Section 5.2, visualized predicted results would be shown and be compared with the ground truth, and some accuracy indices (Acc, mAcc, IoU) with different parameters settings would be displayed. In Section 5.3, the results of geometric method are also shown. In Section 5.4, how different training settings lead to better / worse results would be discussed. In Section 5.5, performance of deep learning model in this paper would be compared with the expected performance of other methods.

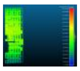
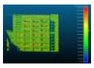
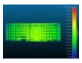
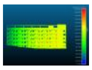
5.1 Training Settings

To compare the influence of different parameter settings on the point cloud segmentation accuracy, this paper sets block size and k (number of nearby points) respectively, and the block size values are 1 and 10. The value of k is 10, 20, 30.

In this study, the extracted point clouds of 15 building facades are filtered, and finally the point cloud data of 8 facades with complete information are selected. The data of these eight facades contains two types of facades of different types, and four of them are used as training set, two sides are used as the validation set, and the last two sides are used as the test set. Among them, considering the limited training data, this paper enhances the data of the four facades for training. In order to better enable the network to learn the features of windows and walls, this paper rotates the facades according to the size of 90°, and finally obtains 8 facades of training data. Table 1 shows the general introduction of the data used in this study, including training, validation and testing data.

Table 1: Overall introduction of data

	Number of Points	Point Density Variation	Tree Occlusions	Complicated Windows	Data Augmentation	Role
F1	123029		No	No	No	Training
F2	123029		No	No	Yes	Training
F3	617559		Yes	No	No	Training
F4	617559		Yes	No	Yes	Training
F5	208770		No	No	No	Training
F6	208770		No	No	Yes	Training
F7	629647		Yes	No	No	Training

F8	629647		Yes	No	Yes	Training
F9	170382		Yes	No	No	Validation
F10	755509		No	No	No	Validation
F11	471130		Yes	No	No	Testing
F12	382300		No	No	No	Testing

We used an NVIDIA GeoForce RTX 3090 GPU with 24G graphics memory. During the training process, the batch_size is set to 8, which means that the input point cloud data can be cut into 8 pieces for training at a time. The epoch is set to 100, that is, a total of 100 iterations . In addition, Adam is used to optimize the network, and the initial learning rate is set to 0.001.

5.2 Results and analysis (deep learning method)

Table 2 summarizes the quantitative results of point cloud semantic segmentation with different parameter settings in the test. When k is the default value of 20, a block_size of 1m obtains the best segmentation accuracy: Acc: 90.32%, mAcc: 87.64%; IoU: 80.71 %. This result is also showing that a smaller block_size setting is more conducive to the extraction of window point clouds. The possible reason is that the range of the wall is small. When the block_size is set too large, the network may learn the overall features of the wall, while ignoring the features of the detailed structure.

When the block_size is set to 1m, the value of k is 30 to obtain the best segmentation accuracy: Acc: 93.86%, mAcc: 90.68%; IoU: 84.97%. This shows that the larger the value of k, the higher the accuracy of the segmentation, because when the value of k is larger, the edge convolution (EdgeConv) can obtain more edge features, which means more local features. However, if the k value is set too large, it is possible to obtain mixed features of windows and walls, which is not conducive to the segmentation of point clouds. Moreover, due to the performance of the computer graphics card, too large a k value will cause the program fail to run.

Table 2: Segmentation accuracy for different parameter settings

Block_size(m)	k	Acc(%)	mAcc(%)	IoU(%)
1	20	90.32	87.64	80.71
10	20	87.53	80.65	75.68
1	10	85.67	80.26	72.89
1	20	89.56	83.72	79.28

1	**30**	**93.86**	**90.68**	**84.97**

Figure 10 shows the results after segmentation. Compared with the ground truth, the network is not fully segmented in some details, which is the area that needs to be improved in the future.

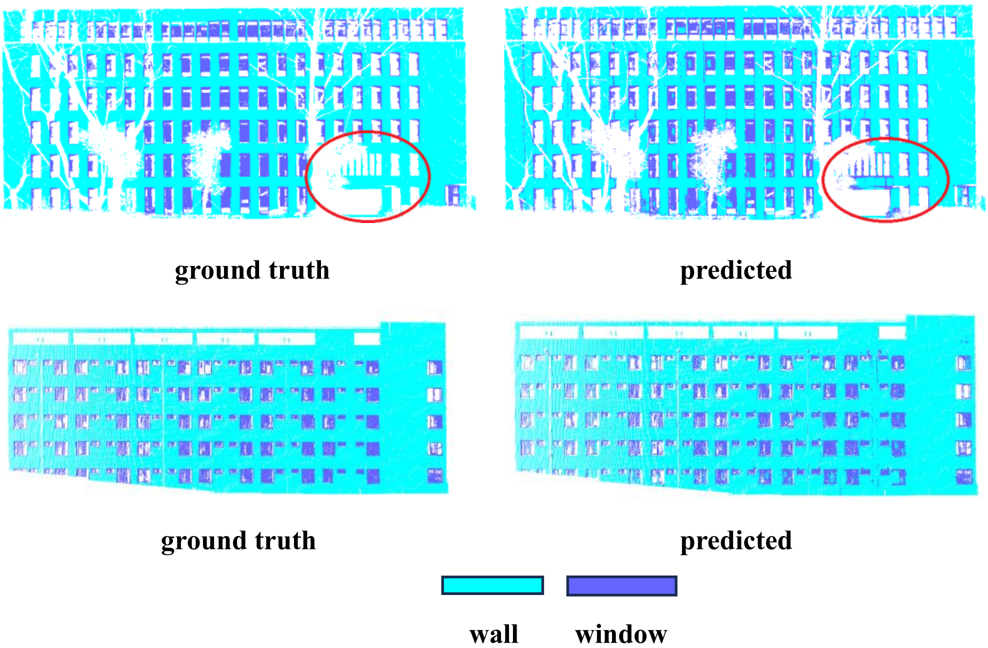
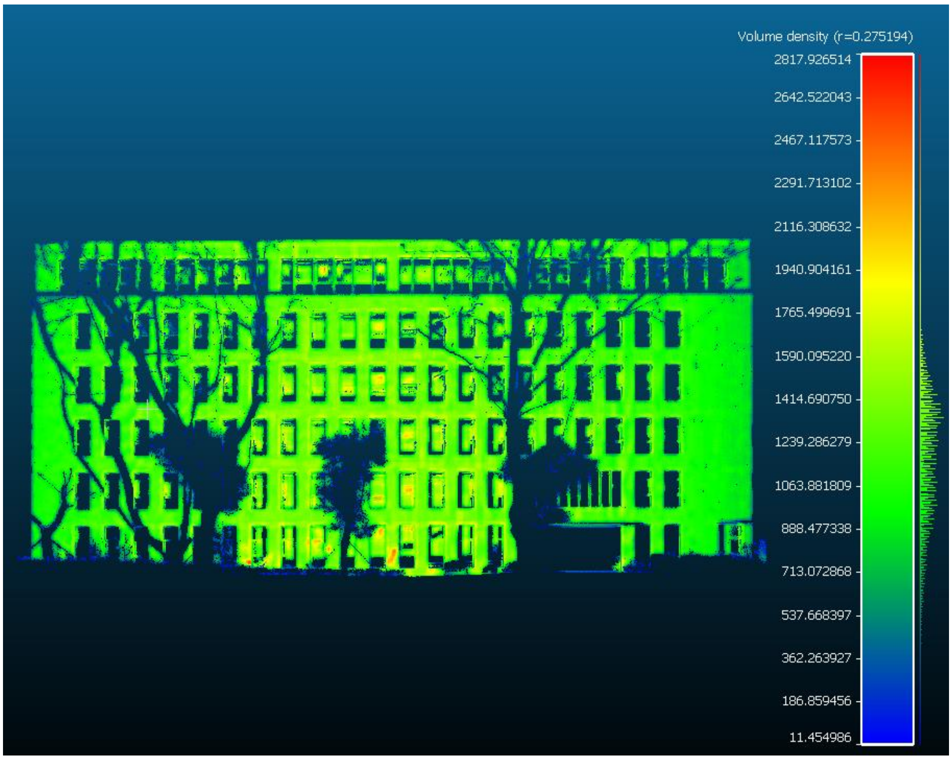


Fig 10: Segmentation results of F11 (above) and F12 (bottom)

Firstly, the red circles in the figure show that when the façade point cloud is under occlusion, performance of network is not that well, the possible reason is that the occlusion causes damage of original structures of windows and walls, which increase the difficulty of segmentation.



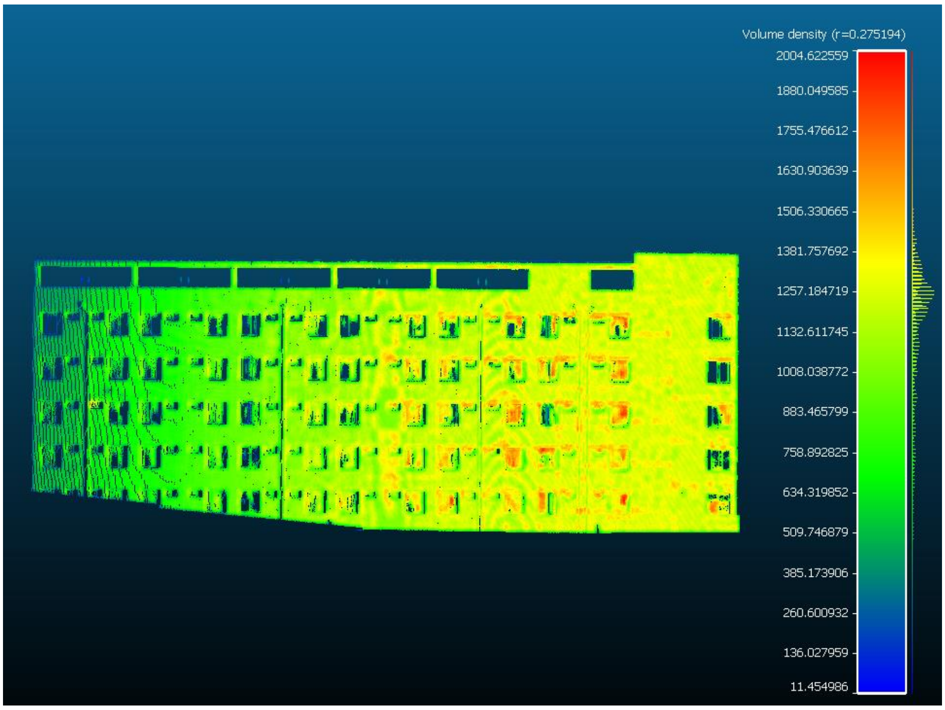


Fig 11: Point density distribution of F11 (above) and F12 (bottom)

Another one is that when it comes to the areas with low point density (figure 11 shows the point density distribution of tested façades), the accuracy of segmentation also goes down. The possible reason is that when the point density is low, the difference of features of windows and walls might be smaller, which makes it more difficult to distinguish them.

5.3 Results and analysis (geometric method)

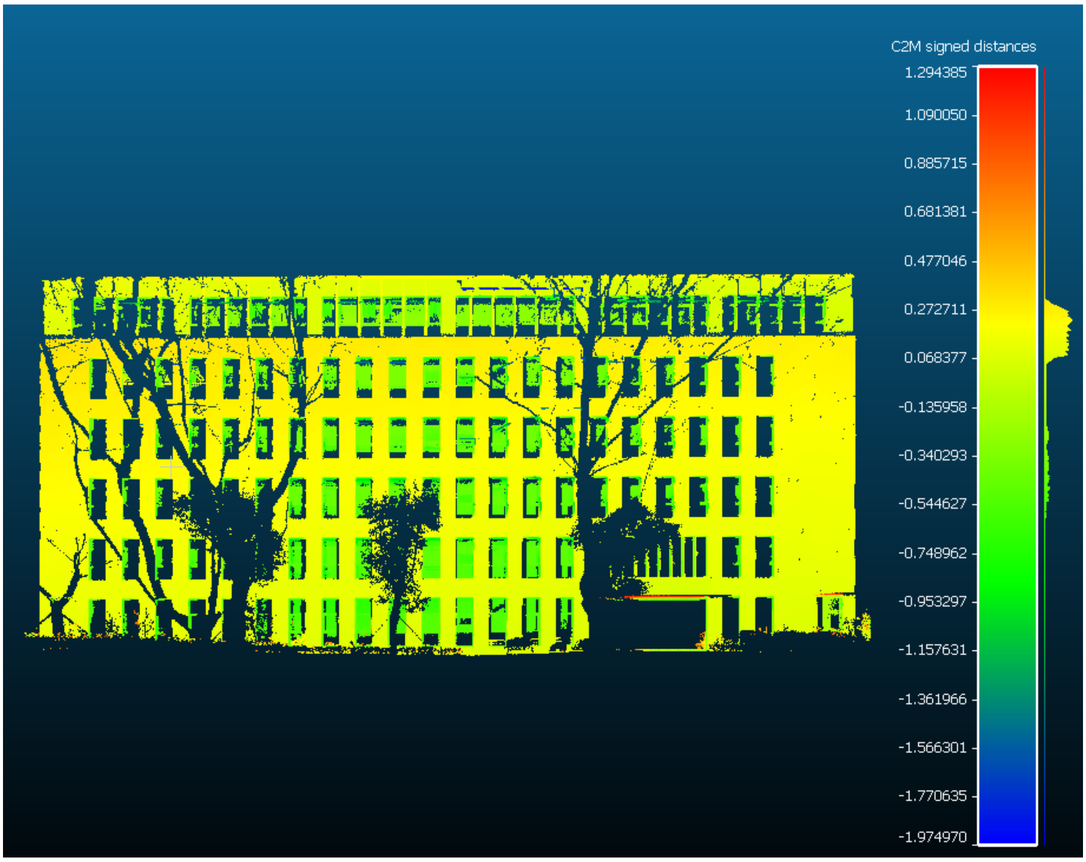


Fig 12: Outcomes of geometric method on F11

Fig 12 shows the outcome of geometric method described in method chapter. The colors of facade in the figure denote the distance between points and fitted plane. From the figure, the difference between the color of wall points and that of window points is relatively obvious, and these two classes could be separated under a distance threshold. For F11, the threshold could be set to 0.06 meter.

However, this method will only work for simple facades, since the feature is too simple and single, which is difficult to handle the complicated situations. For the facades with complicated structure, the deep learning method will work better.

5.4 Settings discussion

5.4.1 k

The 'k' value, representing the number of nearest neighbors considered in the K-Nearest Neighbors (K-NN) scheme, plays a pivotal role in point cloud segmentation. Starting with 'block size' set at 1 meter and 'k' at 10, the model exhibits an OA of 85.67%, MA of 80.26%, and IoU of 72.89%. In this scenario, the smaller 'k' value limits the contextual understanding of each point, as it considers fewer neighboring points during feature aggregation. Consequently, the model may overlook important contextual information, leading to a decrease in segmentation accuracy, especially for complex structures with intricate geometries.

Conversely, when 'k' is increased to 30 under the same 'block size' conditions, a remarkable improvement in segmentation accuracy is observed. The model achieves an OA of 93.86%, MA of 90.68%, and IoU of 84.97%. This improvement can be attributed to the larger 'k' value enabling the model to capture a more extensive contextual range by considering a greater number of nearest neighbors during feature aggregation. This broader contextual awareness is especially beneficial when dealing with complex and interconnected object structures.

However, it's worth noting that an excessively large 'k' value may introduce computational complexity and memory resource challenges. Therefore, the choice of 'k' should strike a balance between obtaining a sufficiently broad contextual understanding and ensuring computational efficiency.

5.4.2 Block size

The impact of 'block size,' which refers to the spatial size of the units into which the point cloud is divided, on the point cloud segmentation outcomes is of paramount importance. When we set 'block size' to 1 meter while keeping 'k' constant at 20, the model demonstrates an impressive performance, achieving an Overall Accuracy (OA) of 90.32%, Mean Accuracy (MA) of 87.64%, and Intersection over Union (IoU) of 80.71%. This configuration effectively captures fine-grained details within smaller spatial units, contributing to higher segmentation accuracy.

However, as we explore 'block size' variations, reducing it to 10 meters under the same 'k' conditions results in a noticeable decline in all three metrics. The OA decreases to 87.53%, MA to 80.65%, and IoU to 75.68%. This decline can be attributed to the larger 'block size,' which encompasses a more extensive spatial range, potentially introducing noise into the point cloud representation. In essence, the larger 'block size' dilutes the localized context within each block, making it challenging for the model to accurately classify points, especially along the boundaries of object features.

Conversely, when 'block size' is excessively reduced, for instance, below 1 meter, it can lead to increased sensitivity to noise and the uneven representation of object surfaces. This sensitivity to noise may result in segmentation artifacts and inaccuracies. Therefore, while smaller 'block sizes' offer the advantage of capturing intricate details, they must be chosen judiciously to strike the right balance between capturing fine-grained features and managing potential noise.

5.4.3 Rotation

In the study, data augmentation is also adopted to enrich existing training data, and the augmentation method used in the study is rotation. Considering the relatively regular structure of windows and walls, the rotation angle is set to 90 degrees in this paper. After rotation, “new” façade data is obtained and can be used in the training process.

a. Expected Impact of Data Augmentation:
In building facade point cloud segmentation, the implementation of data augmentation techniques, such as 90-degree rotations about the Z-axis, is expected to have a substantial influence on model performance. These augmentations are anticipated to provide several benefits:

Increased Diversity: Data augmentation introduces additional viewpoint variations, allowing the model to be exposed to a more diverse set of training examples. This, in turn, should improve the model's ability to handle facade objects with different orientations and shapes.

Improved Generalization: By training on augmented data, the model is likely to develop a more robust understanding of windows and walls. It should be better equipped to generalize its knowledge to real-world scenarios where facade objects may exhibit various configurations.

Enhanced Accuracy: The augmented dataset is anticipated to enable the model to segment windows and walls more accurately, even in cases where these objects are presented at non-standard angles or orientations.

b. Potential Challenges without Data Augmentation:
While data augmentation is expected to provide significant benefits, the absence of data augmentation may introduce certain challenges in building facade point cloud segmentation:

Limited Training Data: Without data augmentation, the training dataset may remain relatively small, potentially hindering the model's ability to generalize effectively. The model's performance might suffer when confronted with facade objects not well-represented in the limited training data.

Orientation Sensitivity: A model trained without data augmentation could be more sensitive to the orientation of facade objects. It may struggle to accurately segment objects that deviate from standard orientations, leading to lower segmentation accuracy.

Overfitting Risk: With a smaller and less diverse training dataset, there is an increased risk of overfitting. The model might become overly specialized in recognizing specific orientations or configurations present in the limited training data, which may not align with real-world scenarios.

c. Discussion:
In the absence of actual baseline results, it's important to recognize that not using data augmentation can pose challenges in terms of limited training data and potential sensitivity to object orientations. While the specific segmentation performance metrics cannot be quantified without a baseline, it is reasonable to anticipate that data augmentation would likely improve the accuracy and generalization capabilities of the model, as discussed in section a.

Furthermore, the model's ability to handle diverse facade configurations, especially those deviating from standard orientations, is expected to be a significant benefit of data augmentation.

Future work should consider conducting experiments to validate these expectations and quantify the actual improvements in segmentation accuracy when data augmentation is employed. Such experiments would provide concrete evidence of the benefits of data augmentation in building facade point cloud segmentation.

5.5 Discussion

The performance of deep learning model in this paper would be compared with the expected performance of other methods at table 3.

Table 3: Comparison of (expected) performance of different methods.

	Input Features	Pre-processing	Feature Generation	Training Effort	Computational Cost	Accuracy
PointNet	x, y, z coordinates	Yes	Automatic	high	medium	medium
PCNN	x, y, z coordinates	Yes	Automatic	high	high	high
Random Forest	x, y, z coordinates, hand-crafted features	Yes	Manual and Automatic	medium	low	low
Geometric Method	x, y, z coordinates	Yes	Manual	low	low	low
Ours	x, y, z coordinates	Yes	Automatic	high	medium	high

For input features, only random forest method requires extra hand-crafted features. All methods require pre-processing. For feature generation, random forest method needs manually computed input features and geometric needs distance, which is also calculated manually. For training effort, random forest method usually needs to prepare lots of input features. For computational cost, the deep learning methods are usually more expensive than non-deep learning methods. For accuracy, the deep learning methods usually could obtain higher accuracy.

6. Conclusion and Recommendations

In Section 6.1, the research question and corresponding sub-questions are answered in order. In Section 6.2, several advices are proposed to improve the performance of deep learning model in the study.

6.1 Conclusion

At the beginning of this paper, the research question and some sub-questions are proposed. Here, these questions are going to be answered.

How to apply a deep learning framework to perform point cloud semantic segmentation for building facades with high accuracy and efficiency?

This work mainly studies the performance of the DGCNN network in point cloud segmentation of building facades, and sets different parameters to find out which parameter input results in the highest segmentation accuracy. Experiments have proved that DGCNN can be used for the segmentation of building facade point clouds, and the effect is good. Two block_sizes (1,10) and three k values (10,20,30) are set. When block_size is 1 and k is 30, the best segmentation accuracy is achieved: Acc: 93.86%, mAcc: 90.68%; IoU: 84.97%. Increasing the k value is beneficial to improve the overall segmentation accuracy. This paper only considers the two categories of walls and windows, and does not segment the complex structures of other categories such as air conditioners, balconies, and door posts. Subsequent experiments could consider adding other structures, and at the same time attach color information to the point cloud for segmentation.

The main outstanding part of DGCNN is the network module dubbed EdgeConv. Point clouds inherently lack topological information so designing a model to recover topology can enrich the representation power of point clouds. To this end, EdgeConv is proposed, and it's suitable for CNN-based high-level tasks on point clouds including classification and segmentation. EdgeConv acts on graphs dynamically computed in each layer of the network. It is differentiable and can be plugged into existing architectures. Compared to existing modules operating in extrinsic space or treating each point independently, EdgeConv has several appealing properties: It incorporates local neighborhood information; it can be stacked applied to learn global shape properties; and in multi-layer systems affinity in feature space captures semantic characteristics over potentially long distances in the original embedding, [41].

How to determine the input feature set for the deep learning framework?
In this paper, the input feature set consists of the x, y, z coordinates of the façade point cloud.

How to determine the structure of our deep learning framework?
In this paper, the overall structure of deep learning framework is Dynamic Graph CNN, which is also called DGCNN.

What is the training strategy?

In this paper, the training strategy is to change the value of k and block_size, then compare the accuracy of segmentation of different parameters settings. Meanwhile, data augmentation method is also adopted on training data in order to obtain more training data.

How to use Domain Adaptation (a technique used to minimize the potential impact of differences in data distribution between training and test datasets.) to minimize possible impact caused by different distribution between training and test datasets?
In order to use domain adaptation to minimize possible impact caused by different distribution between training and test datasets, different types of facades should be contained in training. In this study, several different facades make up the training data.

How to eliminate the influence of measurement geometry, if it's necessary?
Generally, measurement geometry would affect the point density of the point cloud. According to the previous analysis, a too small point density would affect the segmentation accuracy. In order to avoid the influence of measurement geometry, the point density should not be too small.

Which evaluation metrics should be chosen to measure the performance?
In this paper, three evaluation metrics are chosen to measure the performance. The **overall accuracy** is calculated by dividing the number of correctly classified points in all categories by the total number of predictions, and is determined. Due to the uneven number of points in different categories, such as points from windows being much less than those from walls, the **average accuracy of each category** is also calculated. In addition, we also evaluate the **average of intersection over union (IoU)** for each class. This is a commonly used indicator in semantic segmentation tasks.

How does the segmentation outcome fit the ground truth?
According to the result chapter, when the block_size is set to 1m, the k is set to 30, the best segmentation accuracy is obtained: Acc: 93.86%, mAcc: 90.68%; IoU: 84.97%.

How does the deep learning framework we apply perform in comparison to other deep or non-deep learning frameworks?
In the chapter 5.5, several deep or non-deep learning methods are compared with the method applied in this study. Generally, the method applied in this study usually require pre-processing, can automatically generate features with medium training effort and computational cost, and the accuracy of segmentation is relatively high.

6.2 Recommendations

In this paper, the training set is small, the amount of data is not enough, and there is still room for improvement in the accuracy of the point cloud automatic semantic segmentation algorithm; the study also found that the accuracy is low when classifying the point cloud under occlusion and the point cloud with low density. Follow-up research improvements are as follows:

(1) Improve the automatic segmentation algorithm and improve the accuracy of the segmentation algorithm. EdgeConv considers the distance between the coordinates of the point and the neighboring points, and ignores the vector direction between adjacent points (the local neighborhood graph is an undirected graph, and it is not known who points to the central point and the neighboring points), and eventually loses part of the local geometric information. Moreover, some other details of implementation could be revised, e.g. incorporating fast data structures rather than computing pairwise distances to evaluate k-nearest neighbors queries. We could also consider higher-order relationships between larger tuples of points, rather than considering them pairwise. Another possible extension is to design a non-shared transformer network that works on each local patch differently, adding flexibility to the model, [41].

(2) Collect data from other buildings, divide buildings of different periods and styles, and import more and different types of windows into the training set so that there will be no data skew in the training set; Other feature elements such as air conditioners, and finally realize rapid and automatic modeling relying on 3D laser point cloud data.

(3) In the follow-up research, the DGCNN network can be improved, such as changing the number of EdgeConv layers, and bringing the residual network idea into the neural network.

References

[1] Höfle B, Hollaus M, Hagenauer J. Urban vegetation detection using radiometrically calibrated small-footprint full-waveform airborne LiDAR data[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2012, 67: 134-147.

[2] Wang L, Huang Y, Hou Y, et al. Graph attention convolution for point cloud semantic segmentation[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 10296-10305.

[3] Singh M, Laefer D F. Recent trends and remaining limitations in urban microclimate models[J]. Open Urban Studies and Demography Journal, 2015, 1(1).

[4] Bui G, Morago B, Le T, et al. Integrating videos with LIDAR scans for virtual reality[C]//2016 IEEE Virtual Reality (VR). IEEE, 2016: 161-162.

[5] Lehmann R, Lösler M. Multiple outlier detection: hypothesis tests versus model selection by information criteria[J]. Journal of surveying engineering, 2016, 142(4): 04016017.

[6] Laefer D F, Truong-Hong L. Toward automatic generation of 3D steel structures for building information modelling[J]. Automation in Construction, 2017, 74: 66-77.

[7] Nguyen A, Le B. 3D point cloud segmentation: A survey[C]//2013 6th IEEE conference on robotics, automation and mechatronics (RAM). IEEE, 2013: 225-230.

[8] Gröger G, Plümer L. CityGML–Interoperable semantic 3D city models[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2012, 71: 12-33.

[9] Gadde R, Marlet R, Paragios N. Learning grammars for architecture-specific facade parsing[J]. International Journal of Computer Vision, 2016, 117: 290-316.

[10] Kong G, Fan H. Enhanced facade parsing for street-level images using convolutional neural networks[J]. IEEE Transactions on Geoscience and Remote Sensing, 2020, 59(12): 10519-10531.

[11] Fathalla R, Vogiatzis G. A deep learning pipeline for semantic facade segmentation[J]. 2017.

[12] Griffiths D, Boehm J. A review on deep learning techniques for 3D sensed data classification[J]. Remote Sensing, 2019, 11(12): 1499.

[13] Qi C R, Su H, Mo K, et al. Pointnet: Deep learning on point sets for 3d classification and segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 652-660.

[14] Qi C R, Yi L, Su H, et al. Pointnet++: Deep hierarchical feature learning on point sets in a metric space[J]. Advances in neural information processing systems, 2017, 30.

[15] Hsu P H, Zhuang Z Y. Incorporating handcrafted features into deep learning for point cloud classification[J]. Remote Sensing, 2020, 12(22): 3713.

[16] Wang L, Huang Y, Hou Y, et al. Graph attention convolution for point cloud semantic segmentation[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 10296-10305.

[17] Boulch A, Le Saux B, Audebert N. Unstructured point cloud semantic labeling using deep segmentation networks[J]. 3dor@ eurographics, 2017, 3: 17-24.

[18] Lawin F J, Danelljan M, Tosteberg P, et al. Deep projective 3D semantic segmentation[C]//Computer Analysis of Images and Patterns: 17th International Conference, CAIP 2017, Ystad, Sweden, August 22-24, 2017, Proceedings, Part I 17.

Springer International Publishing, 2017: 95-107.
[19] Su H, Maji S, Kalogerakis E, et al. Multi-view convolutional neural networks for 3d shape recognition[C]//Proceedings of the IEEE international conference on computer vision. 2015: 945-953.
[20] Tatarchenko M, Park J, Koltun V, et al. Tangent convolutions for dense prediction in 3d[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 3887-3896.
[21] Ben-Shabat Y, Lindenbaum M, Fischer A. 3dmfv: Three-dimensional point cloud classification in real-time using convolutional neural networks[J]. IEEE Robotics and Automation Letters, 2018, 3(4): 3145-3152.
[22] Maturana D, Scherer S. Voxnet: A 3d convolutional neural network for real-time object recognition[C]//2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2015: 922-928.
[23] Roynard X, Deschaud J E, Goulette F. Classification of point cloud scenes with multiscale voxel deep network[J]. arXiv preprint arXiv:1804.03583, 2018.
[24] Graham B, Engelcke M, Van Der Maaten L. 3d semantic segmentation with submanifold sparse convolutional networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 9224-9232.
[25] Riegler G, Osman Ulusoy A, Geiger A. Octnet: Learning deep 3d representations at high resolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 3577-3586.
[26] Bronstein M M, Bruna J, LeCun Y, et al. Geometric deep learning: going beyond euclidean data[J]. IEEE Signal Processing Magazine, 2017, 34(4): 18-42.
[27] Masci J, Boscaini D, Bronstein M, et al. Geodesic convolutional neural networks on riemannian manifolds[C]//Proceedings of the IEEE international conference on computer vision workshops. 2015: 37-45.
[28] Monti F, Boscaini D, Masci J, et al. Geometric deep learning on graphs and manifolds using mixture model cnns[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 5115-5124.
[29] Simonovsky M, Komodakis N. Dynamic edge-conditioned filters in convolutional neural networks on graphs[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 3693-3702.
[30] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering[J]. Advances in neural information processing systems, 2016, 29.
[31] Wang L, Huang Y, Hou Y, et al. Graph attention convolution for point cloud semantic segmentation[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 10296-10305.
[32] Thomas H, Qi C R, Deschaud J E, et al. Kpconv: Flexible and deformable convolution for point clouds[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 6411-6420.
[33] LLC, W. (2020). Multilayer perceptron. Retrieved 2020-12-06, from https://en.wikipedia.org/wiki/Multilayer_percept
[34] Li J, Chen B M, Lee G H. So-net: Self-organizing network for point cloud

analysis[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 9397-9406.

[35] Liu X, Han Z, Liu Y S, et al. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network[C]//Proceedings of the AAAI conference on artificial intelligence. 2019, 33(01): 8778-8785.

[36] LLC, W. (2021). Convolutional neural network. Retrieved 2021-01-17, from https://en.wikipedia.org/wiki/Convolutional_neural_n

[37] Hua B S, Tran M K, Yeung S K. Pointwise convolutional neural networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 984-993.

[38] Groh F, Wieschollek P, Lensch H P A. Flex-Convolution: Million-scale point-cloud learning beyond grid-worlds[C]//Asian Conference on Computer Vision. Cham: Springer International Publishing, 2018: 105-122.

[39] Wang L, Huang Y, Hou Y, et al. Graph attention convolution for point cloud semantic segmentation[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 10296-10305.

[40] Atzmon M, Maron H, Lipman Y. Point convolutional neural networks by extension operators[J]. arXiv preprint arXiv:1803.10091, 2018.

[41] Wang Y, Sun Y, Liu Z, et al. Dynamic graph cnn for learning on point clouds[J]. ACM Transactions on Graphics (tog), 2019, 38(5): 1-12.

[42] Christoffersen P, Jacobs K. The importance of the loss function in option valuation[J]. Journal of Financial Economics, 2004, 72(2): 291-318.

[43] Van Dyk D A, Meng X L. The art of data augmentation[J]. Journal of Computational and Graphical Statistics, 2001, 10(1): 1-50.

[44] Wang X, Zhang D, Niu H, et al. Segmentation Can Aid Detection: Segmentation-Guided Single Stage Detection for 3D Point Cloud[J]. Electronics, 2023, 12(8): 1783.

[45] Ziwen C, Wu W, Qi Z, et al. Visualizing point cloud classifiers by curvature smoothing[J]. arXiv preprint arXiv:1911.10415, 2019.

[46] Jarząbek-Rychard M, Lin D, Maas H G. Supervised detection of façade openings in 3D point clouds with thermal attributes[J]. Remote Sensing, 2020, 12(3): 543.

[47] Su Y, Liu W, Yuan Z, et al. DLA-Net: Learning dual local attention features for semantic segmentation of large-scale building facade point clouds[J]. Pattern Recognition, 2022, 123: 108372.

[48] Yu B, Hu J, Dong X, et al. A Robust Automatic Method to Extract Building Facade Maps from 3D Point Cloud Data[J]. Remote Sensing, 2022, 14(16): 3848.

[49] Zhang P, He H, Wang Y, et al. 3D urban buildings extraction based on airborne lidar and photogrammetric point cloud fusion according to U-Net deep learning model segmentation[J]. IEEE Access, 2022, 10: 20889-20897.

[50] Maru M B, Wang Y, Kim H, et al. Improved building facade segmentation through digital twin-enabled RandLA-Net with empirical intensity correction model[J]. Journal of Building Engineering, 2023, 78: 107520.