

DELFT UNIVERSITY OF TECHNOLOGY

ADDITIONAL THESIS

Semantic Segmentation of 3D Building Facade Point Clouds Using Dynamic Graph CNN

Author name: Zhuo Chen
Student number: 4844068
Start date: January, 2020
Thesis committee: Dr. R. Lindenberg, TU Delft, supervisor
Dr. S. Verhagen, TU Delft



Abstract

The 3D point cloud representation of building holds significant importance in developing a comprehensive smart city model. However, due to the substantial volume of point cloud data and its inherently unstructured nature, accomplishing semantic segmentation of the point cloud poses a formidable challenge. Recent years have witnessed substantial advancements in employing deep learning techniques for point cloud segmentation. Both PointNet and PointNet++ have demonstrated their prowess in managing point cloud data. In the context of this study, we harness the potential of DGCNN for segmenting building point clouds. DGCNN plays a pivotal role in partitioning the input point cloud, thereby influencing the sensor domain's network range. Consequently, this research delves into the role of parameter k in K-NN (K-Nearest Neighbors) and its implications. The findings illustrate that augmenting the 'k' value contributes to enhancing the overall precision of DGCNN. Nonetheless, complete precision in segmenting every module cannot be assured. The research objectives in the study is building façade, and the targets are **windows, walls, balconies and doors**. In this study, the key parameters are the values of 'k' and the 'block size.' Experiments were conducted by varying the values of these two parameters to obtain different segmentation results of building facade point clouds. Notably, setting k to 20, block size to 1m achieves an overall accuracy rate of 89.75%, a mean accuracy rate of 85.03%, an IoU of 76.24%, while elevating k to 30 and block size to 1m results in a heightened accuracy rate of 95.74%, a mean accuracy rate of 89.37%, an IoU of 81.52%.

1. Introduction

In Section 1.1, 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

In recent years, the creation and manipulation of three-dimensional (3D) urban building models have attracted considerable attention from a multitude of researchers. This surge in interest can be attributed to its relevance across diverse domains, including vegetation monitoring [1], aiding navigation for autonomous driving vehicles [2], constructing environmental models [3], facilitating the development and interaction within virtual reality environments [4], and numerous other applications. To tackle these challenges, point clouds have emerged as a pivotal data source. Over the past decade, Light Detection and Ranging (LiDAR), a prominent remote sensing technology, has gained prominence for capturing extensive volumes of 3D point clouds, which subsequently serve as fundamental input datasets for the aforementioned applications. However, the practical implementation of these endeavors encounters substantial computational overheads when processing such vast point cloud datasets for generating corresponding models or conducting associated computational analyses [5]. Concurrently, challenges persist in terms of managing building information during the modeling process [6]. This situation becomes particularly pronounced when the modeled buildings encompass an array of non-rectilinear elements (e.g., curved windows) and intricate geometric components.

1.2 Semantic Segmentation

However, unlike non-semantic segmentation, the concept of semantic segmentation involves the assignment of specific semantic information to each individual point. In the context of the previously mentioned applications, particularly in the reconstruction of three-dimensional urban area models, segmentation, especially in its semantic form, plays a crucial and indispensable role. In the field of computer vision terminology, segmentation entails the classification of point clouds into coherent regions characterized by shared properties [7]. In the context of semantic segmentation, this process goes further, partitioning the input point cloud into distinct sections, each with its unique semantic significance. Consequently, each section receives a label corresponding to one of the predefined semantic classes. Beyond its application in the creation of 3D urban models, semantic segmentation finds utility in various other domains. For example, in the realm of robotics, segmentation can be leveraged to categorize objects in a robot's immediate surroundings. This proves invaluable as these semantic labels enhance the robot's ability to distinguish individual objects within its environment, thereby enabling informed decision-making.

Given the increasing demand for 3D building models in domains such as environmental

modeling and urban planning, the matter of semantic segmentation for building facades has gained heightened importance. This is especially relevant to the intricate segmentation of elements such as windows and doors within Level of Detail 3 (LoD3) building models in the CityGML format [8]. Current approaches to facade segmentation typically rely on grammatical rules or rudimentary computer vision techniques [9]. Although these methods produce reasonably satisfactory results, they still face several challenges.

On one hand, these grammar-based rules are derived from architectural principles that often lack robustness [10]. Moreover, the architectural diversity worldwide means that the existing grammar rules inadequately cover the full range of facade styles and types [11]. Conversely, some basic computer vision methods, such as edge detection and region growing, depend on local gradients or variations in local average grayscale values [10]. Consequently, these methods may lack versatility and be susceptible to noise interference. In general, these conventional methodologies exhibit inherent limitations.

1.3 Deep Learning

Over the past decade, significant strides in comprehending 3D sensed data have been facilitated by the integration of deep learning methodologies. Due to their relevance across a wide spectrum of applications, spanning from indoor robotics navigation to nationwide remote sensing, there is a growing demand for algorithms capable of autonomously comprehending and categorizing 3D sensed data, including point clouds [12]. To tackle the complexities involved in processing 3D point clouds, numerous frameworks grounded in deep learning principles have emerged, notable among them being PointNet [13] and PointNet++ [14]. These frameworks exhibit substantial potential for enhancing performance across various domains, including classification, segmentation, and more. Consequently, deep learning, as a formidable approach, is poised to drive continued progress in 3D point cloud processing.

Historically, traditional techniques for 3D point cloud tasks have often relied on meticulously engineered features, such as normals, in conjunction with specific classifiers. These methods have frequently yielded satisfactory results [15]. The foundation of these handcrafted features is often rooted in the geometric and frequency attributes of point clouds. However, extracting crucial handcrafted features requires a deep understanding of the domain and significant expertise [15]. In contrast, deep learning algorithms inherently possess the capability to autonomously learn what are referred to as computer-generated features. Nevertheless, this typically entails the use of intricate network architectures and a substantial computational investment [15].

1.4 Research question and Sub-questions

This research endeavor revolves around a central inquiry: **How can a deep learning framework be effectively leveraged to achieve both high accuracy and efficiency in point cloud semantic segmentation for building facades?** The research objectives in the study is building façade, and the targets are **windows, walls, balconies and doors**. Additionally, a set of subsidiary questions has been formulated:

- **What is the data distribution of research objectives?**
- **What strategy is best suited for the training process of the deep learning framework?**
- **Which evaluation metrics should be carefully selected to comprehensively assess the performance of the framework?**
- **To what extent does the segmentation outcome align with the established ground truth?**
- **How does the performance of the applied deep learning framework compare to alternative deep or non-deep learning frameworks?**
- **What is the effect of having more classes, like four in the study, instead of two (like windows and walls)?**

These questions collectively form the core of this research, aiming to address critical aspects of point cloud semantic segmentation for building facades using deep learning techniques.

2. Related Work

In this section, we will review previous deep learning 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 segmentation, point cloud semantic segmentation labels each semantic information for each point, which is more flexible compared to clustering-based method, [16].

2.1 PointNet and PointNet++

PointNet, [13], marked a pivotal moment by introducing a novel approach for processing point clouds directly. It employed shared multi-layer perceptrons (MLPs) to capture local patterns and symmetric functions to aggregate point-wise information. PointNet++, [14], an extension of PointNet, addressed the limitations of processing global and local features separately. It introduced a hierarchical network that hierarchically grouped points to capture both local and global context, significantly improving segmentation accuracy and robustness.

2.2 Graph Convolutional Networks (GCNs)

Graph Convolutional Networks (GCNs), [46], harnessed the inherent graph structure of point clouds for segmentation. GCNs captured both local and global information by aggregating features from neighboring points in the graph. This approach proved promising, particularly in capturing long-range dependencies within point clouds.

2.3 Attention Mechanisms

Attention mechanisms, inspired by their success in natural language processing, have found their way into point cloud segmentation. These mechanisms enable models to attend to informative points while filtering out noise. The integration of self-attention mechanisms, akin to Transformer architectures, [47], has notably enhanced segmentation accuracy and robustness, particularly in complex scenes.

2.4 Weakly Supervised Learning

In scenarios with limited labeled data, weakly supervised learning approaches have gained prominence. Methods like pseudo-labeling and self-training, [48], leverage unlabeled data to boost model performance, expanding the applicability of point cloud segmentation in practical, data-scarce environments.

2.5 Transfer Learning and Pretrained Models

Transfer learning has played a pivotal role in point cloud segmentation. Pretrained

models, initially trained on large-scale datasets such as ModelNet, [49], have been fine-tuned for specific tasks. This approach reduces the demand for extensive labeled data and accelerates model convergence, making it highly practical for real-world applications.

2.6 Point Cloud Augmentation

Data augmentation techniques specific to point clouds have also emerged. These methods manipulate point cloud data by applying transformations like rotation, translation, and scaling. Augmentation techniques enhance model robustness by diversifying the training data, allowing models to generalize better across various point cloud scenarios.

2.7 Multimodal Approaches

Some recent efforts have explored the integration of multiple modalities, such as color and intensity data, with 3D point clouds, [14]. These multimodal approaches leverage complementary information to improve segmentation accuracy, especially in scenes with complex texture and color variations.

In summary, deep learning has ushered in a new era for point cloud segmentation. Methods like PointNet, PointNet++, GCNs, attention mechanisms, weakly supervised learning, and transfer learning have significantly improved the accuracy and adaptability of segmentation models. As the field continues to evolve, future research is expected to address challenges related to noisy data, scalability, and real-time applications, pushing the boundaries of what is achievable in point cloud segmentation using deep learning methods.

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

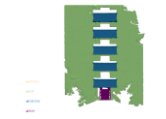
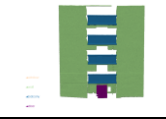
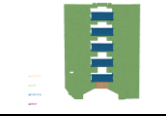

The 3D point cloud dataset related to buildings has been sourced from Nanjing University of Information Science and Technology, encompassing instructional and dormitory structures. The data acquisition process involves the use of the RIEGL VZ 2000i 3D laser scanner, which boasts an impressive scanning accuracy of 3mm and a maximum measurement range spanning 2500m. In conjunction with the Riegl system, a Nikon camera is employed to capture color information simultaneously with the point cloud data, proving particularly suitable for scenarios focused on building scanning. To comprehensively capture the structures of the buildings, 4 to 5 scanning stations are strategically positioned to ensure a comprehensive representation. Furthermore, multiple facades of buildings sharing the same architectural style are also subjected to scanning to provide a thorough dataset.

The study focuses on 9 building facades, from which point cloud data is extracted and filtered. Among these, six facades are allocated for training, one for validation, and the remaining two for testing.

The composition of the dataset utilized in this study is summarized in Table 1. This table presents an overview of the training, validation, and testing data, encapsulating the essential information for each category.

Table 1: Overall introduction of data

	Number of Points	Appearance	Tree Occlusions	Role
F1	1052303		Yes	Training
F2	942594		Yes	Training
F3	1064566		Yes	Training
F4	640378		Yes	Training
F5	751844		Yes	Training

F6	352646		Yes	Training
F7	1200791		No	Validation
F8	547167		No	Testing
F9	1300333		No	Testing

3.2 Denoising and labeling of point cloud data

It is evident that the original building point cloud dataset contains a significant amount of extraneous points. Consequently, it becomes imperative to undergo a denoising process and extract the building's facades. Facade extraction can be achieved through either automated or manual methods. Given the abundance of redundant and interference-laden information within the experimental data, and considering the relatively manageable nature of extracting building facades, manual extraction is the preferred approach.

During the building facade extraction process, the choice of software for visualizing the building point clouds holds significant importance. For this experiment, the primary software used for visualizing point cloud data is Cloud Compare. Furthermore, Cloud Compare also serves as the platform for subsequent point cloud calibration tasks.

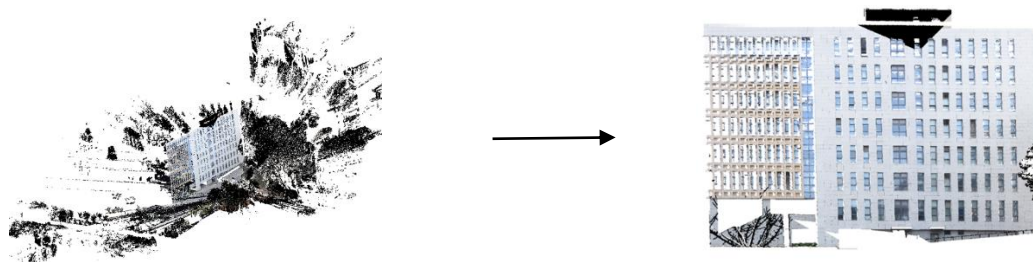


Fig 1: Manual denoising

The architectural point cloud obtained through this experiment contains a plethora of intricate features, including walls, windows, air conditioners, doors, and various other building components. The research primarily concentrates on windows, the exterior walls, the balconies and the doors of the buildings. The extraction process entails manual delineation of targets and the identification of various attribute features present

on the building facades. Furthermore, during the processing phase, distinctive structures that require differentiation are singled out. The ultimate goal of this endeavor is to curate datasets that will serve as the basis for upcoming network training, validation, and testing procedures.

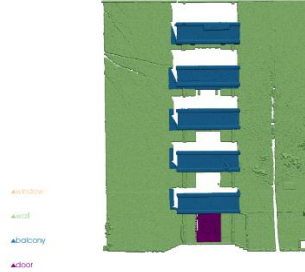


Fig 2: labeled building point cloud

3.3 Down-sampling

Due to the voluminous and unordered nature of point clouds, direct processing involves significant computational overhead when seeking neighboring points. To address this challenge, a common approach is to down-sample the point cloud. This strategy involves converting the operations conducted on the entire point cloud into operations performed solely on the down-sampled points. This effectively reduces the computational workload significantly.

In this study, voxel down-sampling is selected. Voxel down-sampling is a data reduction technique employed in 3D point cloud processing. It entails partitioning the point cloud into discrete, regularly sized cubic volumes known as "voxels." This process serves the purpose of decreasing the data resolution while retaining the inherent structural information within the point cloud.

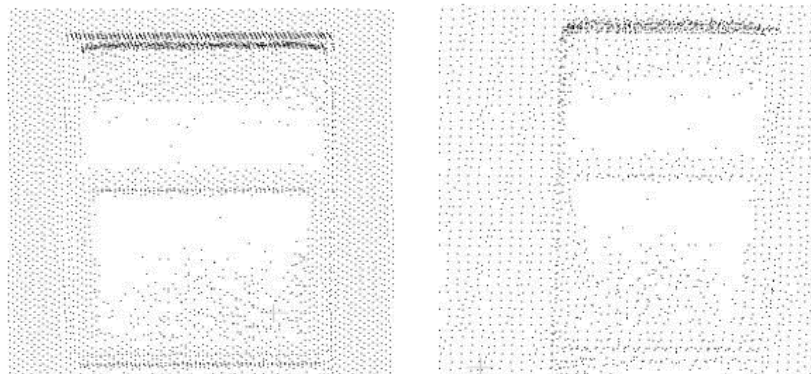


Fig 5: Comparison before and after voxel down-sampling

The original point cloud dataset, before the application of down-sampling, contains 3,011,839 points. After the down-sampling process, the point cloud dataset has been reduced to 911,095 points.

4. Method

In Section 4.1, the principle of DGCNN model is introduced with the loss function. In Section 4.2, the geometric method is introduced, which could also be used to complete the segmentation task on simple facades.

4.1 Network Design

4.1.1 DGCNN Network Architecture

DGCNN (Dynamic Graph Convolutional Neural Network) [41] is primarily built upon the foundation of PointNet [13]. In this architecture, the entire point cloud dataset serves as input, and the model directly produces segmentation results. Both DGCNN and PointNet are end-to-end neural network structures designed to seamlessly process point cloud data from input to output.

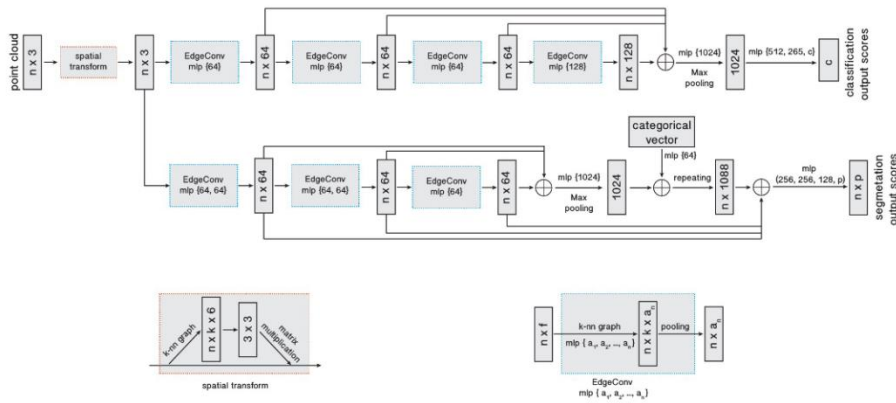


Fig 6: DGCNN Network Architecture

The DGCNN (Dynamic Graph Convolutional Neural Network) architecture is specifically designed for semantic segmentation tasks on point clouds. Each input point in this architecture typically consists of three features representing coordinates such as x, y, and z. DGCNN extends the PointNet classification model by combining global feature vectors with local features generated using the EdgeConv operation [41].

In the segmentation model, a spatial transformation block is used to align the input point cloud by applying a 3x3 matrix. This matrix, expected to be orthogonal, is estimated during the training process. The EdgeConv operation plays a pivotal role in integrating local features. The dimension 'f' of edge features for each point is determined by applying a multi-layer perceptron (MLP), where the number of neurons in each layer of the MLP is denoted as {a₁, a₂, ...}. After pooling the edge features, a tensor of

shape ($n \times a_n$) is generated.

For the purpose of this study, DGCNN is employed for the semantic segmentation of point clouds, wherein the entire set of points is input into the network to produce a semantic label for each individual point. DGCNN builds upon a foundational version of PointNet by integrating local features using edge convolution operations (EdgeConv) instead of traditional MLPs. In constructing the architecture, DGCNN establishes a directed graph $G = (V, E)$ that represents the internal local structure of the point cloud, where $V = \{1, \dots, N\}$ represents vertices, and $E \in V \times V$ represents edges. This graph can be as simple as a k -nearest neighbor (k -NN) adjacency graph. Rather than directly convoluting point features, DGCNN calculates K edge features related to the nearest neighboring points for each point. This process facilitates the extraction of local features from the point cloud, making it suitable for semantic segmentation tasks.

4.1.2 Classification of Building Components Based on DGCNN

The point cloud data is characterized by its massive and uneven nature, necessitating a process of division into blocks for efficient computation. Each block is designed to contain 4096 points to meet the neural network's parameter requirements. Blocks with fewer than 4096 points undergo upsampling and integration to reach this standard, while dense blocks are resampled to reduce point operations to 4096 points.

Concerning feature dimensions, the coordinates of the original point cloud, post-blocking, undergo centering and normalization. All information derived from original, centered, and normalized coordinates is merged and incorporated into the point cloud's attribute data, including semantic information. Point cloud blocks serve as processing units, with their centers established as origins for establishing a coordinate system. Centralization involves recalculating point cloud coordinates with respect to the new coordinate system. Meanwhile, normalization entails linearly transforming original coordinates to a range between 0 and 1.

The point cloud's annotation information in this study encompasses four categories: windows, walls, balconies and doors. For each point cloud, probability values are computed for each of the three aforementioned types to facilitate predictions. The category with the highest probability value determines the point's classification. During neural network training, the maximum probability value and the annotation work together to calculate the loss, refining network parameters through backpropagation. To ensure the uniformity of contours within the same category, point cloud blocks are batched for each training round.

To explore the viability of DGCNN with aerial point clouds and to assess the effects of different effective ranges, experiments were conducted using five block sizes (10m, 3m, 1m, 0.5m, and 0.1m) and five k values (20, 25, 30, 35, and 40). After comparing the

performance of distinct configurations, the default neighborhood size $k=30$ was retained, while block sizes were altered. Subsequently, the block size that yielded the best segmentation results was used to investigate the influence of varying k values.

4.1.3 Loss Function Design

The loss function [42] serves as a metric to assess the performance of a model. A higher loss value indicates poorer model performance. In the context of neural networks, the primary objective is to optimize parameters to minimize the loss function, thereby identifying the optimal weight values that improve overall network performance. The loss layer is responsible for aligning predicted values with actual values within the loss function, resulting in the current loss value. This loss value is then propagated backward through the network layers for further training, ultimately leading to improved neural network performance and the pursuit of minimal loss.

Both the classification and segmentation tasks in this study involve the classification of point clouds. In neural networks, the cross-entropy loss function is commonly used for classification tasks. This function accelerates model optimization and enhances efficiency, especially when the model's performance is suboptimal. By significantly adjusting parameters, the cross-entropy loss function expedites training and reduces training time. The essence of the cross-entropy function lies in assessing the discrepancy between actual values and predicted values, making it a suitable choice for classification tasks.

The formula for the cross-entropy loss function is as follows:

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

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

In addition, due to the uneven distribution of number of point cloud of four classes, relevant weight coefficients will be added to the loss function to try out the impact of data skew.

In this experiment, the evaluation of all conducted experiments relies on several performance metrics. The overall accuracy is determined by dividing the count of correctly classified points across all categories by the total number of predictions. Additionally, due to the varying number of points within different categories (e.g., significantly fewer points from windows compared to walls), the average accuracy for each category is also computed.

Furthermore, the evaluation includes the calculation of the average Intersection over Union (IoU) for each class. IoU is a widely used metric in semantic segmentation tasks.

For point cloud data, IoU for a particular class is determined through the following formula:

$$IoU_{class} = \frac{\text{Number of intersection points between predicted and true class}}{\text{Number of union points between predicted and true class}}$$

In essence, IoU measures the overlap between the predicted and true class by considering the common points (intersection) and the total points (union) associated with that class. This metric provides insights into the segmentation accuracy for each class, serving as an important benchmark in evaluating the model's performance.

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

In this paper, the following terms are of specific significance:

- TP (True Positives) represents the count of correctly predicted window point clouds.
- FP (False Positives) denotes the number of point clouds inaccurately predicted as windows.
- FN (False Negatives) indicates the count of point clouds incorrectly predicted as non-windows.
- TN (True Negatives) pertains to the number of point clouds accurately predicted as non-windows.

To comprehensively evaluate the model's performance, a confusion matrix is established. This matrix provides a detailed breakdown of information for all classes, including TP, FP, FN, and TN values. It serves as a valuable tool for assessing the model's predictive capabilities across various categories.

4.2 Geometric Method

In addition to the deep learning approach, another method could be employed to accomplish the segmentation task.

In this study, a segmentation approach is proposed for point clouds representing building facades. The primary objective of our work is to categorize points into four classes: wall, door, window, and balcony. The method leverages two key features: the distance of each point to the best fitting plane and the number of connected points.

The initial binary segmentation, which distinguishes between "wall" and "non-wall" points based on their distances to the best fitting plane, serves as the foundation of our approach. This binary categorization is particularly effective for simple facades, where a large distance corresponds to "non-wall" and a small distance corresponds to "wall." However, with the introduction of more complex building elements, such as doors and balconies, further refinement is necessary.

To address this challenge, a multi-step classification process is proposed. For points classified as "non-wall" due to their significant distances from the best fitting plane, we further classify them into "door," "window," or "balcony" based on the number of connected points and their respective distances to the facade.

Large Number of Connected Points:

When a specific point is surrounded by a substantial number of connected points, it typically indicates the presence of a large, contiguous structure in the vicinity. This configuration is often associated with balconies on building facades. Balconies are prominent architectural features protruding from the facade, and they are typically represented by multiple closely adjacent points. The abundance of connected points in this context can be attributed to the relatively extensive spatial extent of balconies, necessitating a multitude of points to accurately capture their geometric characteristics.

Moderate Number of Connected Points:

A moderate number of connected points suggests the presence of some adjacent points in the vicinity of a specific point, although not as numerous as observed for balconies. This configuration is commonly encountered in the context of doors. Doors on building facades typically represent relatively smaller openings, and the surrounding points are indicative of the door's boundary and shape. The intermediate quantity of connected points aligns with the characteristic geometric properties of doors.

Few Connected Points:

A small number of connected points indicates that only a limited subset of points in the vicinity is adjacent to a specific point, in contrast to the more extensive connectivity observed for balconies or doors. This scenario is often associated with windows. Windows on building facades typically manifest as smaller openings, and the relatively sparse distribution of connected points suffices to capture their geometric attributes accurately. The limited number of connected points corresponds with the characteristic features of windows.

This categorization methodology, based on the quantity of connected points, serves as a rudimentary segmentation strategy, aimed at distinguishing different architectural elements such as balconies, doors, and windows based on point density. The effectiveness of this approach hinges on the density and quality of point cloud data, as well as the geometric characteristics of the architectural elements in question. Typically, balconies exhibit the highest number of connected points, followed by doors, while windows display the lowest connectivity. This represents a common geometric trait that can aid in the automated segmentation of various components of building facades.

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, 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, the performance of deep learning model in this paper would be compared with the expected performance of other methods.

5.1 Training Settings

In order to assess the impact of different parameter configurations on point cloud segmentation accuracy, this study explores variations in block size and the number of nearby points, denoted as k . Specifically, block sizes of 1 and 10 are examined, and k values are set at 10, 20, and 30.

For the training process, a NVIDIA GeForce RTX 3090 GPU with 24GB of graphics memory was used. During training, the batch size was set at 8, which means that the input point cloud data is divided into 8 segments for simultaneous training. The training process consists of 100 epochs, indicating a total of 100 iterations. Network optimization was performed using the Adam optimizer, with an initial learning rate set at 0.001.

5.2 Results and analysis (deep learning method)

Table 2 compiles the quantitative outcomes of point cloud semantic segmentation using distinct parameter configurations during testing. Notably, with a default k value of 20, employing a block size of 1 meter yields the highest segmentation accuracy: Accuracy (Acc) at 89.75%, Mean Accuracy (mAcc) at 85.03%, and Intersection over Union (IoU) at 76.24%. This result underscores that a smaller block size setting is more conducive to the extraction of window and door point clouds. A larger block size might lead the network to focus on the overall wall and balcony features while potentially disregarding intricate structural details.

With the block size set to 1 meter, a k value of 30 attains the optimal segmentation accuracy: Acc of 95.74%, mAcc of 89.37%, and IoU of 81.52%. This observation indicates that greater k values correspond to heightened segmentation accuracy. Larger k values allow the edge convolution (EdgeConv) to capture a richer array of edge features, which in turn reflects an increased capacity to extract local features. However, excessively large k values may result in mixed features of different class, such as balconies and walls, undermining point cloud segmentation. Additionally, extremely large k values could impede program execution due to limitations in computer graphics card performance.

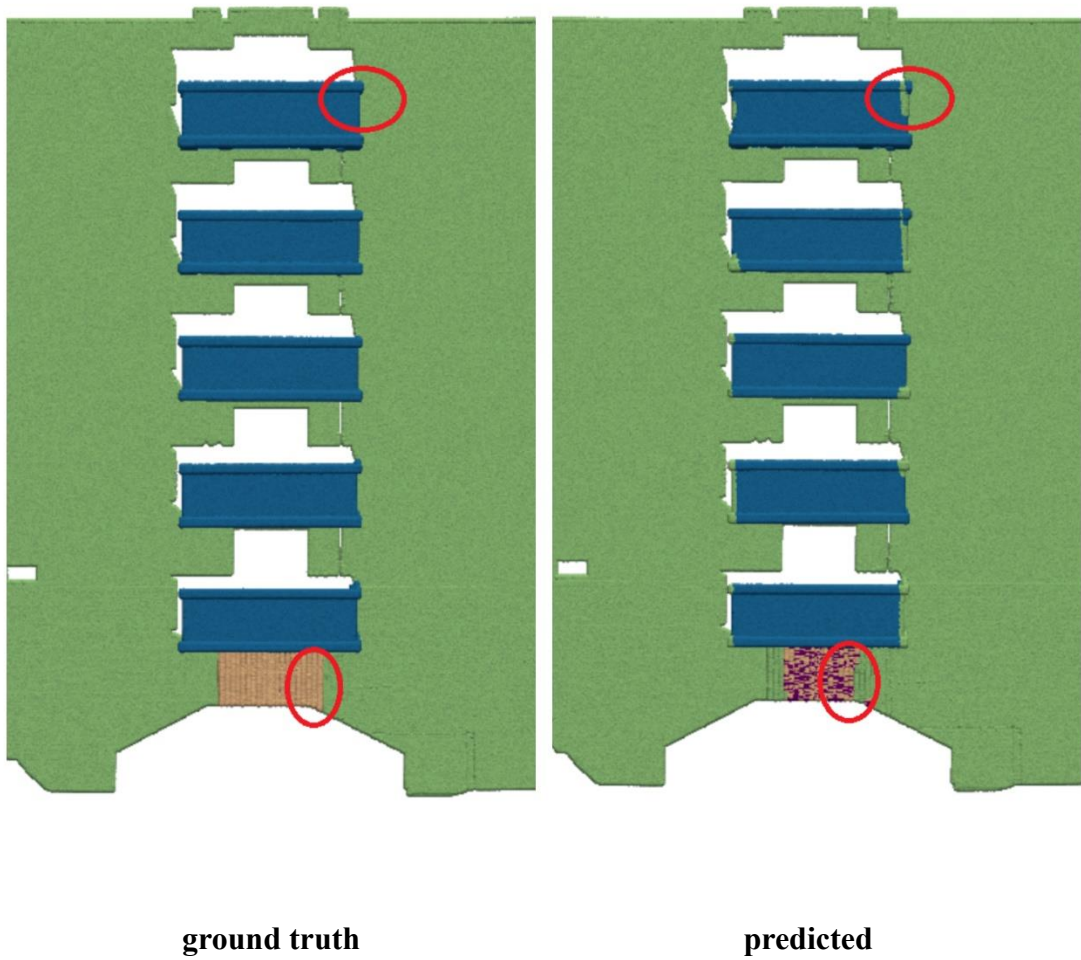
Table 2: Segmentation accuracy for different parameter settings

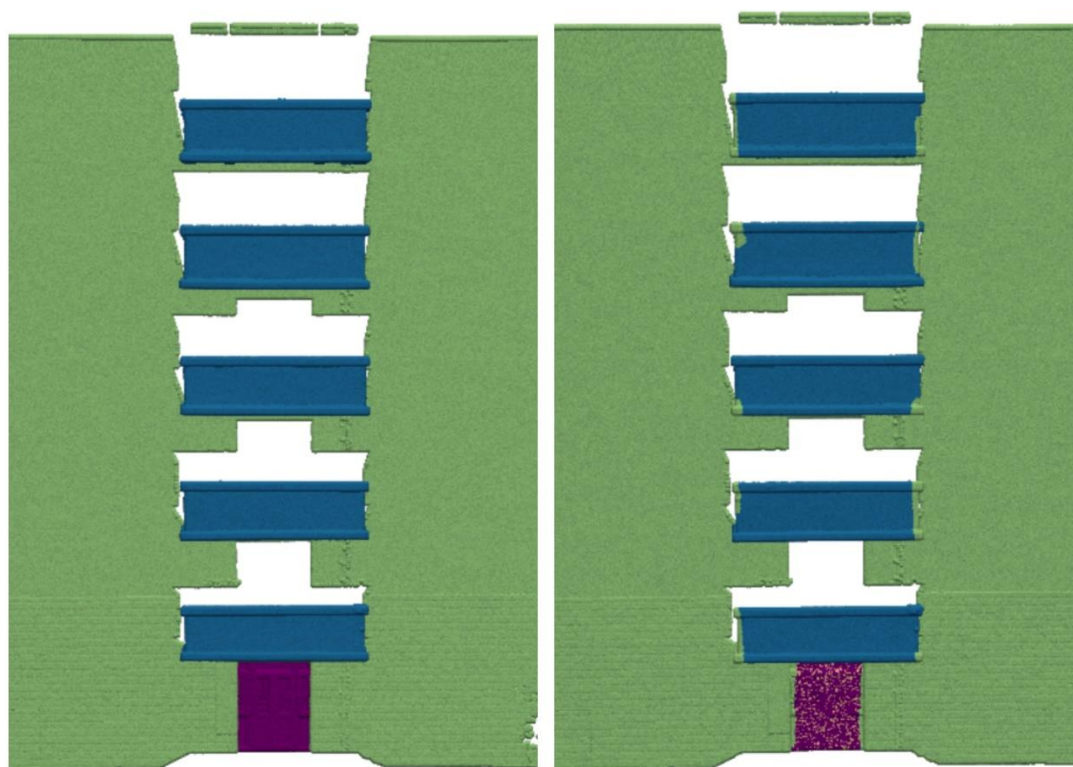
Block_size(m)	k	Acc(%)	mAcc(%)	IoU(%)
1	20	89.75	85.03	76.24
10	20	82.14	77.62	70.23
1	10	86.01	80.26	73.57
1	30	95.74	89.37	81.52

Table 3: Distribution of training data of four classes

window	wall	balcony	door
0.11%	75.83%	22.55%	1.51%

Figure 7 visually presents the segmentation results.





ground truth

predicted

- ▲window
- ▲wall
- ▲balcony
- ▲door

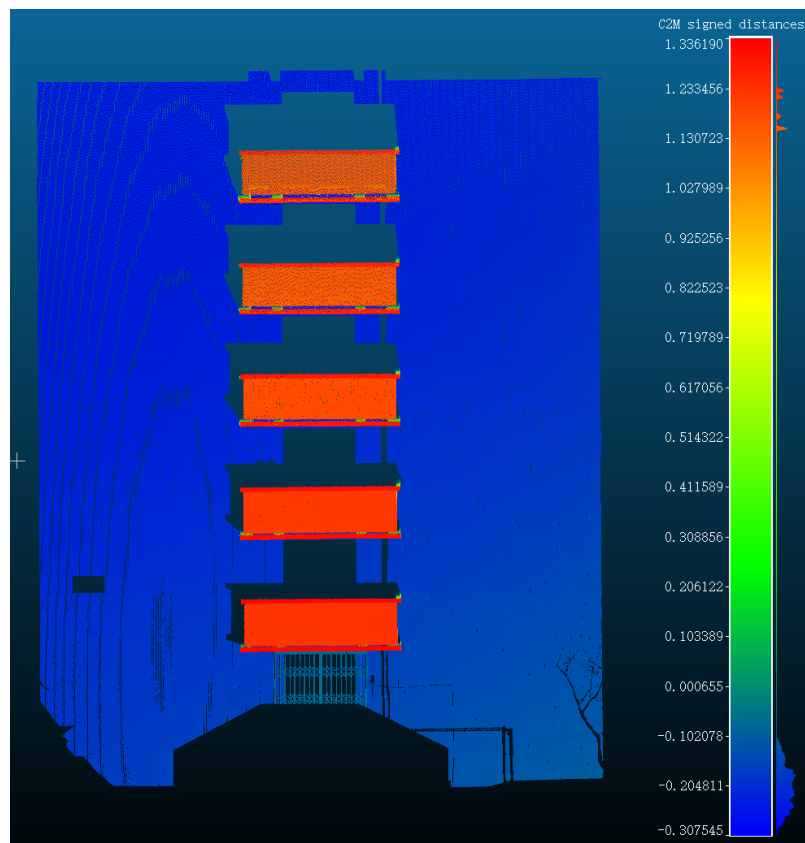
Fig 7: Segmentation results of F8 (above) and F9 (bottom)

Firstly, it is readily apparent from the highlighted regions in the figure 7, denoted by the red circles, that for other types of point clouds in proximity to the wall area, they are prone to being misclassified into the wall category. Whether it is a balcony, door, or window, the situation appears quite similar. One possible explanation is revealed through Table 3, where it is evident that the point cloud data for the wall category overwhelmingly dominates the data among the four categories. For these other categories of points in the vicinity of the wall area, when computing features, a significant portion of the points selected are likely to belong to the wall category,

resulting in them being erroneously assigned wall-like features.

Furthermore, it is evident that points belonging to the "door" and "window" categories are prone to mutual misclassification. There are three potential reasons for this. Firstly, the data volume for these two categories of point clouds is relatively small, making it challenging for deep learning models to effectively generate meaningful semantic features from this limited data. Secondly, among all the training data, the relative positions of doors and windows on building facades are highly similar, which can easily lead to confusion. Lastly, it's important to note that among the selected facades for this experiment, there were no facades containing both "door" and "window" category points simultaneously. This absence of combined points from these two categories further complicates the deep learning model's ability to distinguish between them.

5.3 Results and analysis (geometric method)



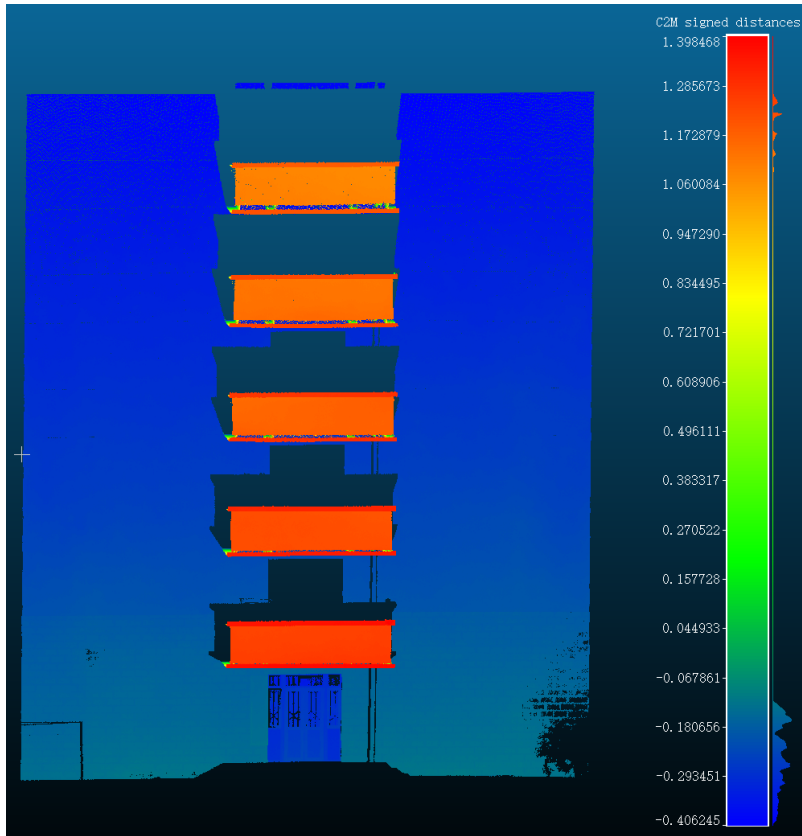
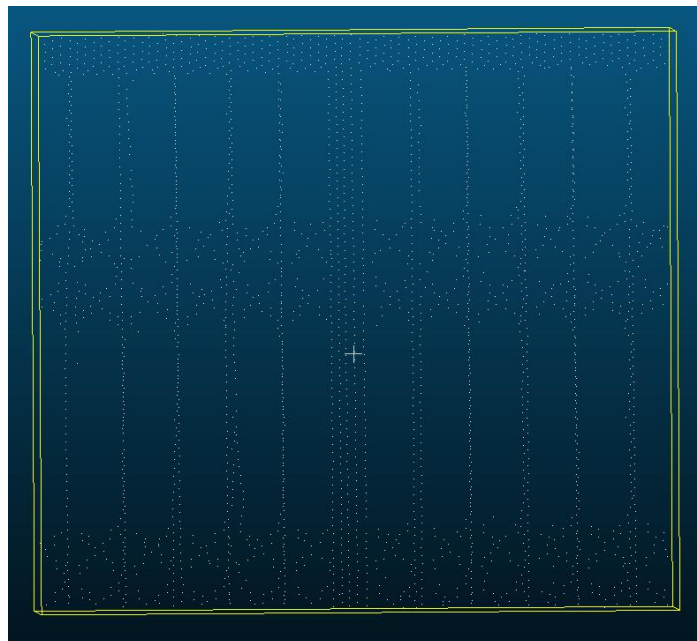


Fig 8: distances between points to best fitting plane of F8 (above) and F9 (bottom)



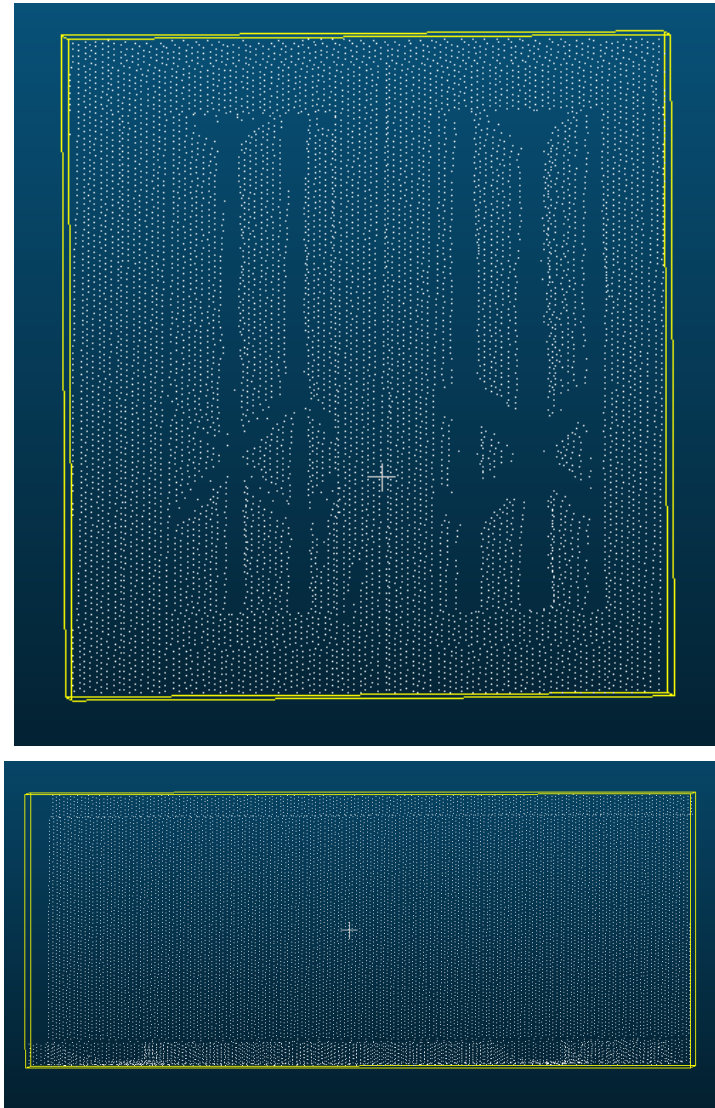


Fig 9: connected points of window(above), door(middle) and balcony(bottom)

The analysis of the results reveals interesting findings that align with the theoretical framework discussed earlier.

Distance Analysis:

Figure 8 shows the distances between points to best fitting plane of F8 and F9.

Balcony Points: The balcony points exhibit an average distance of approximately 0.1 to 0.2 meter from the best fitting plane. This is consistent with the theoretical expectation, where balconies were hypothesized to have a relatively small distance from the plane, indicating they are positioned relatively close to the building facade.

Window Points: Window points show an average distance of around -0.1 meter from

the best fitting plane. This negative distance suggests that window points are slightly recessed compared to the facade, which is in line with expectations. Windows are typically embedded within the building structure and would have a negative distance.

Wall Points: Wall points demonstrate an average distance of approximately -0.1 to -0.3 meter from the best fitting plane. This negative distance is consistent with walls being positioned behind the facade, further validating the theoretical framework. The variation in distance may be due to variations in wall thickness or surface irregularities.

Door Points: Door points exhibit an average distance of around -0.3 to -0.4 meter from the best fitting plane. This considerable negative distance indicates that doors are typically set back significantly from the building facade, as expected. Doors are often deeply embedded within the building structure.

Connected Points Analysis:

Figure 9 shows the connected points of window, door and balcony. The window and the balcony come from F8, and the door comes from F9. For the distances between points to the best fitting plane and the size of them, figure 8 is a good reference. The analysis of connected points within equally sized regions for different architectural features provides additional insights:

Window Points: The window point cloud has 2603 connected points within the defined region. This relatively lower number of connected points aligns with the theoretical framework, as windows are relatively small openings compared to doors and balconies.

Door Points: Door point cloud exhibits a significantly higher count of 9105 connected points within the region. This is in line with the expectation that doors, being larger and more substantial architectural elements, would have a greater number of connected points.

Balcony Points: The balcony point cloud shows the highest count of connected points, totaling 15086. This is consistent with the theoretical prediction that balconies, being larger and protruding structures, would have the most extensive set of connected points.

The geometric method for architectural facade segmentation, although simple and interpretable, presents limitations when contrasted with deep learning techniques. It relies on manual feature engineering, making it less adaptable to complex architectural designs. Additionally, it may struggle with noisy or incomplete data and lacks the semantic understanding that deep learning models offer. While suitable for straightforward cases, this method falls short in handling the diversity, robustness, and scalability demands of more intricate architectural segmentation tasks, where deep learning-based approaches excel through data-driven feature learning and adaptability. The choice between these methods should consider the specific complexity and

requirements of the architectural analysis at hand.

5.4 Settings discussion

5.4.1 k

a. Smoothness of Boundaries:

Higher k-Value: Increasing the k-value tends to result in smoother boundaries between segmented regions. In the context of building facades, this means that the boundaries between different architectural elements (windows, walls, balconies, doors) are more likely to be less fragmented, leading to a cleaner segmentation output.

Lower k-Value: Conversely, reducing the k-value can lead to sharper boundaries. While this might seem desirable for precisely separating different elements, it may also introduce noise and result in fragmented segments, potentially misclassifying small architectural features as separate entities.

b. Over-Segmentation and Under-Segmentation:

Higher k-Value: A high k-value can cause over-segmentation, where closely spaced elements (e.g., windows on a wall) are segmented as separate entities. In building facades, this might lead to an excessive number of segments, including sub-segments of windows or doors within walls.

Lower k-Value: Conversely, a low k-value may lead to under-segmentation, where connected components are merged into a single segment. In this case, smaller architectural features like balconies or doors may be incorrectly merged with walls.

c. Segmentation Accuracy:

Optimal k-Value: Achieving optimal segmentation accuracy requires a balance between over-segmentation and under-segmentation. For building facade analysis, the optimal k-value should ensure that each class (windows, walls, balconies, doors) is well-distinguished while maintaining the coherence of architectural elements.

d. Computational Efficiency

Higher k-Value: A higher k-value can demand more computational resources, especially for large point clouds, as it involves processing a larger neighborhood for each point. However, it may lead to more efficient memory usage as fewer segments are generated.

Lower k-Value: Lower k-values are computationally more efficient but might require post-processing steps to address over-segmentation issues, which can add complexity to the workflow.

In summary, the choice of the k-value in kNN for building facade point cloud segmentation should consider the specific characteristics of the dataset, such as point density, noise level, and the size of architectural elements. An optimal k-value aims to achieve accurate segmentation while maintaining computational efficiency.

Additionally, post-processing techniques can be employed to refine the segmentation results, ensuring that architectural elements like windows, walls, balconies, and doors are correctly identified and distinguished from each other, which is crucial for precise building facade analysis.

5.4.2 Block size

The parameter "block_size" in this paper pertains to the unit size of the training data. The impact of block size variation would be analyzed from the following aspects:

a. Block Size and Local Context:

Smaller Block Size: When using a smaller block size, the local context considered by the network is limited. This means that the network may have difficulty capturing long-range dependencies between points. In the context of building facades, where architectural elements like windows, balconies, and doors can span multiple points, smaller block sizes might not adequately capture the entire structure.

Larger Block Size: Conversely, a larger block size allows the network to capture a more extensive local context, potentially aiding in recognizing larger architectural features. However, excessively large block sizes can introduce computational overhead and increase the risk of over-smoothing, where fine details are lost.

b. Detail Preservation:

Smaller Block Size: A smaller block size is more likely to preserve fine details in the point cloud. In building facades, this might be beneficial for capturing intricate architectural features such as ornate window frames or decorative elements on doors and balconies.

Larger Block Size: A larger block size may tend to over-smooth the data, potentially causing the loss of fine details and reducing the ability of the network to distinguish between different architectural elements.

c. Computational Efficiency:

Smaller Block Size: Smaller block sizes are computationally more efficient as they involve processing fewer points in each block. This can be advantageous when dealing with large-scale point cloud datasets, where processing speed is a concern.

Larger Block Size: Larger block sizes require processing more points in each block, which can increase computational demands. However, they might reduce the need for extensive post-processing steps to refine segmentation results.

d. Segmentation Accuracy:

Optimal Block Size: The optimal block size strikes a balance between capturing local details and maintaining computational efficiency. For building facade point cloud segmentation, it should be chosen to ensure that architectural elements like windows, walls, balconies, and doors are accurately recognized and differentiated while avoiding over-smoothing or loss of fine features.

e. Application-Specific Considerations:

The choice of block size should consider the specific goals of the analysis. For applications that require detailed modeling and preservation efforts, a smaller block size may be preferred. Conversely, when efficiency and processing speed are critical, a larger block size might be more suitable.

In summary, the block size parameter in Dynamic Graph CNN (DGCNN) for building facade point cloud segmentation should be carefully selected based on the specific characteristics of the dataset, the size and complexity of architectural elements, and the desired level of detail in the segmentation results. An optimal block size balances detail preservation, segmentation accuracy, and computational efficiency. It ensures that architectural features such as windows, walls, balconies, and doors are accurately segmented, making it a critical parameter in building facade analysis.

5.5 Discussion

The performance of the deep learning model in this paper will be compared with the anticipated performance of other methods, as illustrated in Table 4.

Table 4: 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

When it comes to input features, the only approach requiring additional handcrafted features is the random forest method. Nevertheless, all these methods entail preprocessing steps. Concerning feature generation, the random forest method necessitates manually computed input features, while the geometric method relies on manually calculated distances. Regarding training efforts, the random forest method often demands a substantial number of prepared input features. In terms of computational costs, deep learning methods typically incur higher expenses compared

to non-deep learning methods. Regarding accuracy, deep learning methods tend to achieve superior levels of 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 outset of this paper, the research question and its sub-questions are introduced, and now we'll address them comprehensively.

How can a deep learning framework be effectively leveraged to achieve both high accuracy and efficiency in point cloud semantic segmentation for building facades?

This paper primarily delves into assessing the efficacy of the DGCNN network for the segmentation of building facade point clouds, systematically exploring various parameter configurations to identify the optimal settings for achieving the highest segmentation accuracy. The empirical results substantiate the effectiveness of DGCNN in accurately segmenting building facade point clouds, showcasing promising outcomes. The investigation incorporates two block sizes (1 and 10) and three k values (10, 20, 30). Notably, the most favorable segmentation accuracy surfaces when `block_size` is set to 1 and k is set to 30, yielding impressive results: Accuracy (Acc): 95.74%, mean accuracy (mAcc): 89.37%; Intersection over Union (IoU): 81.52%. It's observed that elevating the k value contributes positively to enhancing overall segmentation accuracy. However due to large skew of data distribution between four classes, some research objectives such as window and door are not recognized properly.

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].

What is the data distribution of research objectives?

As described in table 3, the distribution of training data of four classes is 0.11% for window points, 75.83% for wall points, 22.55% for balcony points, 1.51% for door points.

What strategy is best suited for the training process of the deep learning framework?

This paper employs a training strategy that involves varying the values of parameters such as `k` and `block_size`. The aim is to assess the segmentation accuracy across different parameter settings.

Which evaluation metrics should be carefully selected to comprehensively assess the performance of the framework?

This paper employs three evaluation metrics to assess the performance of the segmentation approach. The first metric is overall accuracy, which is calculated by dividing the number of correctly classified points across all categories by the total number of predictions, providing a comprehensive measure of accuracy. Given the uneven distribution of points among different categories, such as the prevalence of wall points compared to window points, the paper also calculates the average accuracy for each category to provide a more balanced assessment.

Additionally, the paper utilizes the average of the intersection over union (IoU) metric for each class. IoU is a widely used indicator in semantic segmentation tasks. It quantifies the degree of overlap between predicted and ground truth segments, offering insights into the precision and recall of the segmentation model for each class.

To what extent does the segmentation outcome align with the established ground truth?

As outlined in the results chapter, the optimal segmentation accuracy is achieved when utilizing a block size of 1 meter and a `k` value of 30. In this configuration, the accuracy metrics are as follows: Overall Accuracy (Acc): 95.74%, Mean Class Accuracy (mAcc): 89.37%, and Intersection over Union (IoU): 81.52%. This combination of parameters yields the highest level of accuracy in segmenting the building facade point clouds.

How does the performance of the applied deep learning framework compare to alternative deep or non-deep learning frameworks?

In the Section 5.5 of the paper, a comparison is presented between the method employed in this study and several other deep or non-deep learning methods. It is evident that the approach adopted in this study typically necessitates pre-processing steps, but it excels in automatically generating features with a moderate training effort and computational cost. Furthermore, the accuracy of segmentation achieved by this method is notably high in comparison to the alternative techniques evaluated.

What is the effect of having more classes, like four in the study, instead of two (like windows and walls)?

the performance of the two-class and four-class segmentation objectives in terms of the following aspects:

a. **Accuracy:** The accuracy of segmenting windows and walls in the two-class objective is typically higher due to the simpler classification task. In contrast, the four-class objective may exhibit lower accuracy, as distinguishing between balconies and doors

can be challenging due to their similar geometries.

b. Complexity: The four-class objective is more complex in terms of both algorithmic implementation and computational demands. It requires handling additional classes and dealing with intricate geometric features, which can be computationally intensive.

c. Application Suitability: The choice between the two-class and four-class objectives depends on the specific application. For applications primarily concerned with window-to-wall ratios or energy efficiency assessments, a two-class objective may suffice. However, tasks involving detailed building facade modeling, such as architectural design or heritage preservation, benefit from the granularity of a four-class objective.

6.2 Recommendations

The paper acknowledges certain limitations and suggests potential avenues for future research and improvements:

1. Enhancing Segmentation Algorithm Accuracy: The current study identifies areas for improvement in the automatic point cloud semantic segmentation algorithm. For instance, the employed EdgeConv approach focuses on the distances between point coordinates and their neighbors but overlooks the directional information between adjacent points. This might lead to a loss of local geometric details. Addressing this by incorporating directional information and revising implementation details, such as utilizing fast data structures for k-nearest neighbor queries, could result in improved accuracy. Exploring higher-order relationships between larger sets of points, instead of pairwise considerations, could also lead to enhanced performance. Introducing non-shared transformer networks that adapt to specific local patches could add further flexibility to the model.

2. Data Collection and Diversity: To mitigate the limitations posed by a small training set, collecting data from various buildings, including those from different periods and architectural styles, could be pursued. Expanding the training dataset with diverse window types and other architectural elements like air conditioners could balance the dataset and improve overall segmentation performance. This comprehensive dataset could facilitate rapid and automatic 3D laser point cloud-based modeling for various structures.

3. DGCNN Network Enhancements: Future research could involve refining the DGCNN network architecture. This might encompass experimenting with different numbers of EdgeConv layers, and incorporating concepts from residual networks (ResNets) to enhance the model's feature representation capabilities. Such adjustments could lead to improved segmentation results and more robust feature learning.

By addressing these avenues for improvement, future research could contribute to the refinement and advancement of automatic point cloud semantic segmentation techniques, fostering more accurate and efficient analysis of 3D point cloud data for various applications.

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] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.
- [47] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
- [48] Oliver A, Odena A, Raffel C A, et al. Realistic evaluation of deep semi-supervised learning algorithms[J]. Advances in neural information processing systems, 2018, 31.
- [49] Wu Z, Song S, Khosla A, et al. 3d shapenets: A deep representation for volumetric shapes[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1912-1920.