remote sensing

MDPI

Article

Building Extraction from Airborne Multi-spectral LiDAR Point Clouds Based on Graph Geometric Moments Convolutional Neural Networks

Dilong Li [1,2], Xin Shen [1,2], Yongtao Yu[3], Haiyan Guan[4], Jonathan Li[5] and Deren Li [1,2]

[1] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China; scholar.dll@whu.edu.cn
[2] Collaborative Innovation Center of Geospatial Technology, Wuhan University, Wuhan 430079, China;
[3] Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, Huaian 223003, China;
[4] School of Remote Sensing and Geomatics Engineering, Nanjing University of Information Science & Technology, Nanjing 210044, China;
[5] Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada;
* Correspondence: xinshen@whu.edu.cn; Tel.: +86-27-68779098.

Received: date; Accepted: date; Published: date

Abstract: Building extraction has been researched for decades as a prerequisite for many applications, and is still a challenging research topic in the field of photogrammetry and remote sensing. Due to the lack of spectral information, massive data processing, and approach universality, building extraction from point clouds is still a thorny and challenging problem. In this paper, a novel deep learning-based framework is proposed for building extraction from point cloud data. In particular, first, a sample generation method is proposed to split the raw preprocessed multi-spectral LiDAR data into numerous samples, the samples, which could be directly fed into convolutional neural networks and cover the original inputs. Then, a graph geometric moments (GGM) convolution is proposed to encode the local geometric structure of point sets. In addition, a hierarchical architecture equipped with GGM convolution, called GGM Convolutional Neural Networks, is proposed to train and recognize building points. Finally, the test scenes with varying sizes can be fed into the framework and obtain a point-wise extraction result. We evaluate the proposed framework and methods on the airborne multi-spectral LiDAR point clouds. Compared with a representative set of previous state-of-the-art networks, our method achieved the best performance with a completeness of 95.0%, a correctness of 87.1%, an F-measure of 90.3%, and an IoU of 82.4% on two test areas. The experimental results confirm the effectiveness and efficiency of the proposed framework and methods.

Keywords: building extraction; airborne multi-spectral LiDAR point clouds; Graph Geometric Moments; Convolutional Neural Networks.

1. Introduction

Building extraction from remote sensing data is a prerequisite for many applications, such as 3D (three-dimensional) building modeling, city planning, disaster assessment, and updating of digital maps and GIS databases [1,2,3,4,5]. Airborne Light Detection and Ranging (LiDAR) data have been extensively used for building extraction as they provide high accuracy, large area coverage, fast acquisition of dense point clouds, and additional information. Due to the lack of rich spectral information of LiDAR data, many studies integrated LiDAR data with high spatial resolution multi-spectral images to improve the performance of building extraction [27,28]. They try to combine the two different data sources in an optimal way so that their weaknesses can be compensated effectively. However, how to accurately register different data sources to the same spatial coordinate system is still an open problem [6].

With the development of sensor technology, some institutes and companies have successively introduced prototypes of multi-spectral and even hyper-spectral LiDAR systems. For example, Teledyne Optech's Titan, the first commercial multi-spectral LiDAR system, was released in Canada in December 2014. Multi-spectral LiDAR data provide relatively complete and consistent spectral information and spatial geometric structure information, which has obvious advantages for building extraction tasks.

At the approach level, although there are recent advances in LiDAR data analysis, several challenges still remain, especially in the areas of massive data processing, approach universality, and process automation. Traditionally, the classical machine learning methods are still considered as a useful tool in this field [7]. The paradigmatic architectures initially transform the raw data into a multi-dimensional feature space, usually called "feature representation", and then optimally estimate by linear or nonlinear associations so as to map the features into desired outputs. Typical techniques, including support vector machines (SVMs) [8], conditional Markov random fields [9], region-growing [10], k-means [11] and graph cut algorithms [12], are quite commonly used. However, the extraction performance of these methods is highly affected by the parameters and adopted features, which are usually content and/or application dependent.

In recent years, the success of deep convolutional neural networks (CNNs) for image processing has motivated the data-driven approaches to extract buildings from airborne LiDAR data. In current studies, CNNs were applied to the existing architectures [13][14], or simply served as a powerful classifier[15]. Nevertheless, due to the unstructured properties of point clouds, these CNN-based methods had to convert the raw point clouds, or the chosen feature representations from the raw point clouds, which still did not completely solve the drawbacks of traditional data-driven methods and did not make full use of the inference ability of CNNs. The key challenges of introducing deep learning methods into building extraction from airborne LiDAR data are still to be resolved.

To address these issues, in this paper, we propose a novel deep learning-based framework for building extraction from point cloud data. With this framework, the LiDAR data or multi-spectral LiDAR data could be directly used for building extraction without transforming them into other data forms, e.g. the multi-view projected images, digital surface model (DSM) or digital terrain model (DTM). Besides, the universality of the framework allows to handle any size of scenes and any shape of buildings without beforehand limitations or assumptions. In addition, the flexibility of the framework allows to replace the model (CNNs) freely.

The main contributions of this paper are listed as follows:

- We propose a deep learning-based framework for building extraction from point cloud data, which only inputs raw point clouds and directly outputs point-wise building extraction results.
- We propose a sample generation method to generate the samples from raw point cloud data, which not only have structured data form to meet the input requirement of CNNs, but also achieve the full coverage of the original input point clouds.
- We propose a novel learn-from-geometric-moments convolution operator, called GGM convolution, which can explicitly encode the local geometric structure of a point set.
- A hierarchical architecture equipped with the GGM convolution, called GGM Convolutional Neural Networks, is proposed. It achieves the best performance on two test areas, compared with a representative set of previous state-of-the-art networks.

The rest of this paper is organized as follows: Section 2 discusses the related work to this subject. Section 3 introduces the study area and the data preprocessing method used in this paper. Section 4 details the methodology. Section 5 presents the experimental results. Section 6 provides the concluding remarks and suggestions for future work.

2. Related Work

To our best knowledge, there are no previous studies about building extraction directly from multi-spectral LiDAR data. Thus, we can only review the previous works with two categories of input data, the raw LiDAR data and the integration of raw LiDAR data and additional remotely sensed data, at the data level. At the approach level, generally, there are two main branches of the methods for building extraction using LiDAR data: model-driven and data-driven approaches. The model-driven approaches estimate the buildings by fitting the input data to a hypothetical model library[10][16], e.g. flat and gable. Thus, the extraction result is always topologically correct and relatively robust as compared to data-driven approaches. However, for a complex building, the respective model may not present in the model library. For instance, [17] interpolate LiDAR raw data into grid digital surface model (DSM) by considering the steep discontinuities of buildings. In contrast, the data-driven approaches have no constraint on the building appearance, and can recognize the buildings with any shapes. Since the deep learning-based methods belong to the data-driven approaches, we will review the most important data-driven methods categorized by their inputs, and discuss the current published deep learning related methods in particular.

2.1. The raw LiDAR input & data-driven methods

Maas and Vosselman [18] presented two techniques for the determination of building models from laser scanner data. Based on invariant moments technique, the parameters of a standard gable roof type building model could be determinated as closed solutions. In addition, the analysis of deviations between the point cloud and the model does allow for modelling asymmetries. Nonparametric buildings with more complex roof types can also be modelled by intersecting planar faces in triangulated point clouds.

Dorninger and Pfeifer [10] proposed a comprehensive approach for automated determination of 3D city models from Airborne Laser Scanning (ALS) data. The approach was based on the assumption that individual buildings can be modeled properly by a composition of a set of planar faces. The approach consisted of a number of steps. The first step was to select the building region by a region-growing algorithm, which resulted in one complete building extracted from the point cloud. Then, the mean shift segmentation algorithm was used to estimate the boundaries of buildings, and the building outline determination was initiated by mean shift segmentation and planar face extraction. Finally, the building outline was regularized by the determination of a 2D-shape, and the building model was generated by the determination of polygonal boundaries of each planar face. The approach can generate the detailed 3D building models with rooftop overhangs, but there are manual interventions required during the preprocessing and post-processing steps. Besides, for the complex building rooftop structures, the interior structure lines cannot be well extracted.

Zhou and Neumann [19] proposed an automatic algorithm which reconstructed building models from ALS data of urban areas. There are several major distinct features in their algorithm developed to enhance efficiency and robustness: (1) they designed a novel vegetation detection algorithm based on differential geometry properties and unbalanced SVM; (2) they used a fast boundary extraction method to produce topology-correct water tight boundaries; (3) they proposed a data-driven algorithm which automatically learned the principal directions of roof boundaries and used them in footprint production. However, since each primitive boundary was processed separately, the generated models via this approach cannot guarantee their compactness and watertightness.

Poullis and You [20] proposed a method for the rapid reconstruction of photorealistic large-scale virtual environments. They represented a parameterized geometric primitive for the automatic building identification and reconstruction. They reconstructed buildings with complex roofs containing complex linear and nonlinear surfaces by using a linear polygonal and a nonlinear primitive, respectively. An extension of this work was proposed by Poullis [55], which proposed a complete framework for the automatic modeling of buildings over large areas. Furthermore, the segmentation and boundaries were refined by using a fast energy minimization process in this approach. Nevertheless, because all the building boundaries are regarded as piece-wise linear, the nonlinear boundaries cannot be well processed.

Sampath and Shan [12] presented a solution framework for the segmentation and reconstruction of polyhedral building roofs from ALS data. The proposed segmentation method contained three steps. Firstly, the eigen analysis was carried out for each roof point of a building within its Voronoi neighborhood. Then, the fuzzy k-means method was used to cluster the surface normals of all planar points. Finally, the parallel and coplanar segments were separated based on their distances and connectivity, respectively. Although the feature elements of the most sampled rooftops could be obtained by adjacency matrix, the complex rooftop models, e.g. dutch gable rooftop, would not be generated correctly.

You and Lin [21] presented an approach based on the tensor voting framework for extracting building features from ALS data. They represented geometric features of ALS data by a tensor field, and extracted roof patches by a region-growing method with principal features developed from the properties of eigenvalues and eigenvectors of the tensor field. Additionally, they proposed three new indicators for strengthening, the features to reduce the effect of the number of points on feature identification, and a supervised method to determine the threshold of planar feature strength for the region-growing.

Kim and Shan [22] presented a approach to building roof modeling from ALS data. The rooftop was segmented by minimizing an energy function formulated as a multiphase level set. The roof ridges or step edges were delineated by the union of the zero level contours of the level set functions. Finally, the coplanar and parallel roof segments were separated into individual roof segments based on their connectivity and homogeneity.

Sun and Salvaggio [23] presented an automated method to create 3D watertight building models from ALS data. They used a graph cuts based method to segment vegetative areas from the rest of scene content, and proposed the hierarchical Euclidean clustering technique to extract the ground terrain and building rooftop patches. However, this approach assumed that the boundaries of all parts of a complex rooftop are rectilinear, which affects the extraction accuracy of building models with nonlinear boundary rooftops.

Zou et al. [24] proposed a method for extracting building point sets from ALS data. The method was based on a strip strategy to filter building points and extract the edge point set in large-scale urban building groups. This approach divided the ALS data into small strips and classified each strip of data with an adaptive-weight polynomial in the $x-$ or $y-$direction. Then, the building edge sets were extracted by utilizing the regional clustering relationships between points.

Santos et al. [25] proposed a building roof boundary extraction method from ALS data. The method overcame the limitation of the original alpha-shape algorithm by applying an adaptive strategy. It estimated a local parameter α for each edge based on local point spacing, instead of using a global parameter.

2.2. The fusion of raw LiDAR and additional data input & data-driven methods

In contrast to the aforementioned building extraction approaches, which only use the raw ALS data as the input data, there are vast methods using the additional data, e.g. DSM, DTM, orthoimagery and multi-spectral orthoimagery, to enhance the extraction performance.

Liu et al. [26] applied the Locally Excitatory Globally Inhibitory Oscillator Networks (LEGION) to the segmentation of buildings. They developed a modified LEGION segmentation model to extract buildings from high-quality digital surface models (DSMs). This approach extracted buildings without the assumptions on the underlying structures in the DSM data and without the prior knowledge of the number of regions.

Mohammad et al. [28] proposed a method for automatic 3D roof extraction through an integration of ALS data and multi-spectral orthoimagery. They separated ground points and non-ground points by using the ground height from a DEM. The structural lines were extracted from the grey-scale version of the orthoimage, and classified into several classes such as 'ground', 'tree', 'roof edge', and 'roof ridge' using the ground mask, the NDVI image, and the entropy image. Their further work [29] added the texture information from the orthoimagery for building extraction. The region-growing technique was iteratively applied to segment non-ground points. Finally, they proposed a rule-based procedure to remove planes constructed on trees. Compared with their works [30], [31], which only use ALS data as the input data, this method has further enhanced the building extraction effectiveness.

Gilani et al. [32] proposed a method to extract and regularize the buildings using features from ALS data and orthoimagery. Firstly, the method identified the candidate building regions and segmented them into grids via the building delineation process. Then, the method synthesized the point cloud and image data to eliminate vegetation, detect building and extract their partially occluded parts. Finally, the detected buildings were regularized by exploiting the image lines in the building regularization process.

2.3. The deep-learning related methods

With the success of deep convolutional neural networks for image processing, many researchers try to apply CNNs to extract buildings on ALS data. But it is still a primeval field to research. To our best knowledge, there are few approaches using the deep learning related methods to extract buildings from ALS data.

Bittner et al. [13] proposed a method to automatically generate a building mask out of a DSM using a Fully Convolution Network (FCN) architecture. Firstly, the FCN was trained on a large set of patches consisting of normalized DSM as inputs and ground-truth building masks as target outputs. Then, the trained predictions from the FCN were enabled to create a final binary building mask. Although the method dosed not required any assumptions on the shape and size of buildings, it cannot directly work on raw ALS data, which needs to generate DSM from the ALS data first.

Nahhas et al. [14] proposed a building detection approach based on deep learning using the fusion of ALS data and orthophotos. This approach utilized object-based analysis to create objects and transformed low-level features into compressed features via a feature-level fusion. Then, a convolutional neural network (CNN) was used to transform the compressed features into high-level features, which could be used to differentiate the buildings and the background. However, in this approach, the point clouds were filtered to create DSM, DEM, and nDSM samples, then they were fused with orthophotos feeding into the CNN, which means it also cannot directly work on raw ALS data.

Maltezos et al. [15] proposed a building extraction method from ALS data by applying deep convolutional neural networks. Firstly, they augmented the raw ALS data with seven additional features, e.g. Normalized Height and Entropy. Then, a CNN model was adopted for coding the inputs into structures that were the best for the classification performance. Nevertheless, the method merely considered the CNN as a powerful classifier, extracted the additional features from raw ALS data and then combined with the orthoimage to feed to the classifier to enhance the performance.

3. Study area and data preprocessing

3.1. Study area

As shown in Figure 1, the study area is a small town located in Whitchurch-Stouffville, Ontario, Canada with an area of 2,052m×1,566m and the center position at latitude and longitude of 43°58'00", 79°15'00", respectively[53]. We choose 13 typical scenes as the training and test scenes, which indicates with red boxes (training scenes) and blue boxes (test scenes) in Figure 1. Each selected scene contains a rich variety of objects, such as roads, trees, grass, buildings, and soil, which contribute to our method study in a real-world complex scene. Table 1 shows the size and total number of points in each selected scene.

The experimental data were collected by using an airborne Titan multi-spectral LiDAR system, produced by the Teledyne Optech. The detailed specifications of the multi-spectral LiDAR system are presented in Table 2. Radiometric correction has been applied to the Titan multi-spectural LiDAR data [54] before we test them on building extraction tasks. Since the system parameters and trajectories were unavailable, the three channels of intensities were directly used from the LiDAR outputs without intensity calibration. Iterative closest points (ICP) was used to roughly register these strips. Similarly, without control points or reference points, the geometric quality is not statistically reported. Thus, we selected the study area from the one strip for assessing our building extraction method.

Figure 1. The study area, the general view of the selected scenes and a sample of the corresponding labeled data.

Table 1. The size and total number of points in each selected scene.

	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6	Area_7	Area_8	Area_9	Area_10	Area_11	Area_12	Area_13
Size(m^2)	176938	98813	178668	104882	153575	108009	129332	149907	241053	149838	163088	165978	162742
Points	697838	425409	747342	418220	556183	325924	598398	695190	887487	653780	864581	758588	626285

Table 2. Specifications of the Titan Airborne System.

Parameters	Channel 1	Channel 2	Channel 3
Wavelength(nm)	1550 (SWIR)	1064 (NIR)	532 (GREEN)
Deflection Angle(°)	3.5 (forward)	nadir	7 (forward)
Flight Altitude(m)	~1000	~1000	~1000
Point Density(/m^2)	3.6	3.6	3.6

3.2. Data preprocessing

As we can see in Table 2, the original acquired raw Multi-spectral LiDAR data contains three channels of individual spatial coordinates and spectral values. Thus, we have to preprocess the original individual data into the fused data firstly. In this paper, we adopt the same data preprocessing strategy as in [53].

The Titan multi-spectral LiDAR system generates three independent point clouds in three channels, 1550 nm, 1064 nm, and 532 nm. To improve the efficiency of point cloud data preprocessing, especially for the Titan multi-spectral LiDAR data, we merged the three independent point clouds into a single point cloud, where each point contains three spectral wavelengths. Specifically, one of the three single-wavelength point clouds was taken as the reference data, in which each point was processed to find its neighbors in the other two wavelengths of point clouds using a nearest neighbor searching algorithm. Because the average point density of a single wavelength was about 3.6 points/m^2, the searching distance in this study was set to 1.0 m to obtain sufficient points in the two wavelengths of point clouds. To obtain the intensities of the two other wavelengths, an inverse-distance-weighted (IDW) interpolation method was used. If there were no neighboring points in one of the two wavelengths, the intensity value of this wavelength was set to zero. In this way, three wavelengths were merged into a single, multi-spectral point cloud.

4. Methodology

4.1. Framework Overview

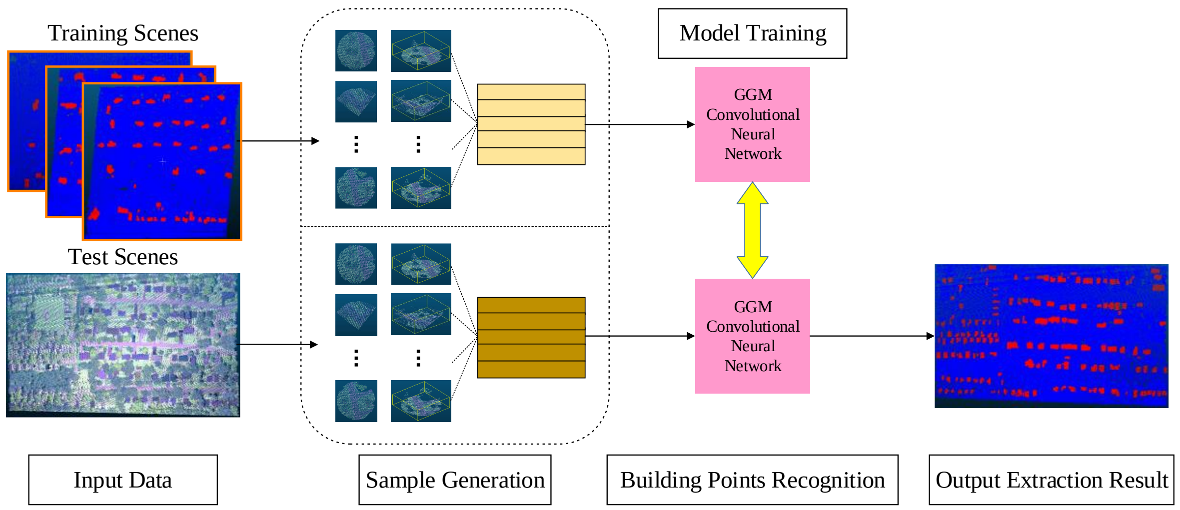


Figure 2. Framework of Building Extraction.

After data preprocessing, we obtain the available multi-spectral LiDAR data. As a supervised method, we have to manually label each of the selected training and test areas before we feed them into the framework.

As shown in Figure 2, our proposed building extraction framework consists of two main stages. Firstly, we feed the labeled training scenes into the GGM Convolutional Neural Networks. Then, we use the trained model to recognize the building points from the input test scenes. Remarkably, the framework requires only point cloud data as input and directly outputs the labels of each point in the test scenes. There are no limitations about the number of training and test scenes, and the size of each input scenes. The framework dose not require any assumptions of the shape and size of the buildings. Furthermore, the model used for training and test is replaceable. That is, any networks, only if they can output the required data form, can be applied in this framework.

During the sample generation stage, the training and test scenes are split into individual samples with a fixed size. Thus, the sampled data could be directly fed into the neural networks. And the input scenes are completely covered by the sampled data at the same time. The details are illustrated in Section 4.2.

For the building points recognition task, we design a convolution operator, called GGM Convolution, which learns local geometric features from geometric moments representation of a local point set. Then, a hierarchical architecture equipped with the GGM Convolution contributes to our model, called GGM Convolutional Neural Networks. The related details are illustrated in Section 4.3.

4.2. Sample Generation

Due to the unstructured properties of point clouds, the characteristics of point clouds in sparsity, permutation invariance, and transformation invariance, are the thorny problems for standard convolution implementations. For building extraction tasks, many researchers transform the point cloud data into multi-view projected images before feeding them to a standard convolutional neural network. And few researchers separate the whole scene into many cuboid regional subsets, and utilize the down-sampling and up-sampling techniques to meet the data form requirement of standard convolutional neural Networks. However, the number of points in unit area is not fixed and the sampling techniques damage the scene integrity, which cannot ensure that every point in the original scene could be labeled.

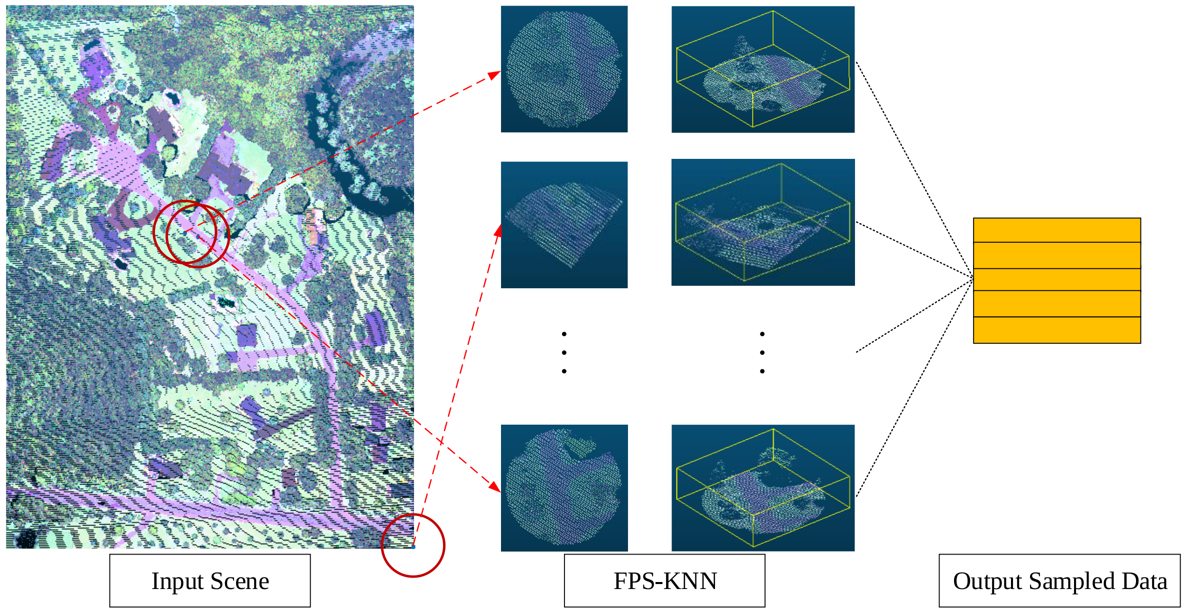


Figure 3. Sample generation workflow with the FPS-KNN method.

Inspired by RandLA-Net[33], we propose an FPS-KNN sample generation method to generate the training and test samples for neural networks. The samples generated by the FPS-KNN not only satisfy the data form requirement of standard convolutional neural Networks, but also achieve the full coverage of the scene. Figure 3 shows the data processing workflow with the FPS-KNN method. The details of the FPS-KNN sample generation method are carried out as follows:

Step 1: For a given scene, we duplicate an identical point set as the evaluation point set. We randomly choose one point in the evaluation point set as the seed point, and search its K nearest neighbors in the original point set, the value of K is set depending on the sample size, e.g. if each sample contains 4096 points, then the value of K is configured as 4096.

Step 2: We calculate the distance from the rest points in the evaluation point set to the seed point and select the most distant point as the next seed point. The seed point and its K nearest neighboring points are saved as one sample, and removed from the evaluation point set.

Step 3: We iteratively find the farthest point as the seed point in the evaluation point set, search its K nearest neighbors in the original point set and remove the sampled points from the evaluation point set, until the evaluation point set is empty.

Thus, we obtain numerous samples with the fixed number of points from the given scene, which can be directly fed into a standard convolutional neural network. At the same time, we can ensure that every point in the scene is contained in some samples, which means the full coverage of the scene. We also notice that some samples are inevitably overlapped. For the points within the overlapped part, we choose the most predicted label as its final predicted label.

In this way, theoretically, for any scene, we can generate samples directly feeding into neural networks by using the FPS-KNN sample generation method and obtain the predicted label for every point in the scene.

4.3. Graph Geometric Moments Convolutional Neural Networks

4.3.1. Geometric Moments

Moments and functions of moments have been widely utilized as pattern features in pattern recognition[34][35][36], edge detection[37][38], image segmentation[39], texture analysis[40] and other domains of image analysis[41][42] and computer vision[43][44].

The general two-dimensional $p+q$ th order moments of a density distribution function $f(x,y)$ is defined as follows:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy\,, \tag{1}$$

where $p,q = 0,1,2,\ldots$. The lower order moments (small values of p and q) have well defined geometric interpretations. For example, m_{00} is the area of the region, $m10/m00$ and $m01/m00$ give the x and y coordinates of the centroid of the region, respectively[38]. Similarly, the three-dimensional geometric moments of $p+q+r$ th order of a 3D object is defined as follows[39]:

$$m_{pqr} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q z^r f(x,y,z)dxdydz\,, \tag{2}$$

where $p,q,r = 0,1,2,\ldots$. The discrete implementation of the moments of a 3D homogeneous object could be defined as follows [38]:

$$m_{pqr} = \sum_{\mathbb{R}^3} x^p y^q z^r f(x,y,z), \tag{3}$$

where $\mathbb{R}^3$ is a 3D region. For the 10 low order 3D moments (order up to 2), we have:

$$\begin{aligned}
m_{000} &= \sum_{\mathbb{R}^3} f(x,y,z) \\
m_{100} &= \sum_{\mathbb{R}^3} x \cdot f(x,y,z) \\
m_{010} &= \sum_{\mathbb{R}^3} y \cdot f(x,y,z) \\
m_{001} &= \sum_{\mathbb{R}^3} z \cdot f(x,y,z) \\
m_{110} &= \sum_{\mathbb{R}^3} x \cdot y \cdot f(x,y,z) \\
m_{101} &= \sum_{\mathbb{R}^3} x \cdot z \cdot f(x,y,z) \\
m_{011} &= \sum_{\mathbb{R}^3} y \cdot z \cdot f(x,y,z) \\
m_{200} &= \sum_{\mathbb{R}^3} x^2 \cdot f(x,y,z) \\
m_{020} &= \sum_{\mathbb{R}^3} y^2 \cdot f(x,y,z) \\
m_{002} &= \sum_{\mathbb{R}^3} z^2 \cdot f(x,y,z)\quad .
\end{aligned} \tag{4}$$

For a raw point cloud, we define its geometric moments representation referring to [45] as follows:

$$M_1 = \begin{bmatrix} x \\ y \\ z \end{bmatrix},\ M_2 = \begin{bmatrix} xy \\ xz \\ yz \\ x^2 \\ y^2 \\ z^2 \end{bmatrix}, \tag{5}$$

M_1 and M_2 are the first and second order geometric moments of the original point cloud data, respectively. The higher order moments give more detailed shape characteristics[40], which means more comprehensive geometric features in deep learning.

The moment-based methods have advantageous qualities like translation and rotation invariance, both of which are important properties for feature descriptors. Translation invariance is obtained by using the central moments for which the origin is at the centroid of the density function[40]. For 3D objects, the translation invariance is obtained by using the central moments μ_{pqr} defined in the same way as for 2D objects[34]. The central moments μ_{pqr} is defined as follows:

$$\mu_{pqr} = \sum_{\mathbb{R}^3} (x-\bar{x})^p (y-\bar{y})^q (z-\bar{z})^r f(x,y,z), \tag{6}$$

where $(\bar{x}, \bar{y}, \bar{z})$ is the centroid of the object, which can be obtained from the first order moments

$$\bar{x} = \frac{m_{100}}{m_{000}} \quad \bar{y} = \frac{m_{010}}{m_{000}} \quad \bar{z} = \frac{m_{001}}{m_{000}}. \tag{7}$$

Mo-Net [45] firstly utilizes the second order geometric moments representation of point clouds as the input features fed into the networks. Compared with PointNet [46], which only considers the first order geometric moments, Mo-Net validates the function of higher order geometric moments. Inspired by that, we design our network to learn features from the geometric moments representation of point clouds.

4.3.2. Graph Generation

Since the Graph Neural Networks(GNNs) proposed by [47], it has been widely used in learning on unstructured data. GNNs apply neural networks for walks on the graph structure, propagating node representations until a fixed point is reached. The resulting node representations are then used as features in classification and regression problems [48]. To apply the graph neural network to the point cloud, first, we need to convert it to a directed graph.

A graph G is a pair (P, E) with $P = \{p_1, \ldots, p_n\}$ denoting the set of vertices and $E \subseteq P \times P$ representing the set of edges. As the consideration of computational complexity, most of the networks would rather construct a k-nearest neighbors(KNN) than a fully connected edges for the whole point cloud.

As shown in Figure 4, we utilize the k-nearest neighbors of each point to construct a local directed graph. In this local directed graph, point p_i is a central node, and e_{ij} are the edges between the central node and its k-nearest neighbors, which are calculated as follows:

$$\begin{aligned} &G = (P, E) \\ &P = \{p_i \mid i = 1,2,\ldots,n\} \\ &E = \{e_{ij} \mid j = 1,2,\ldots,k\}' \\ &e_{ij} = p_{ij} - p_i \end{aligned} \tag{8}$$

where p_{ij} are the neighbors of the central point p_i.

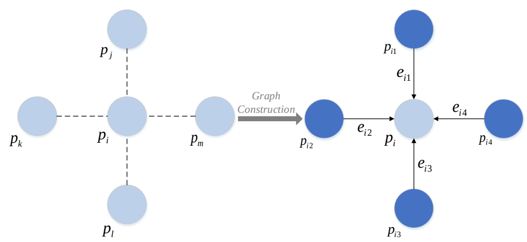


Figure 4. Graph Construction of a point cloud. The p_i, p_j, p_k, p_l, p_m are the points in the point cloud. The p_j, p_k, p_l, p_m on the left and $\{p_{i1},\ldots,p_{i4}\}$ on the right are the nearest neighbors of p_i. The directed edges $\{e_{i1},\ldots,e_{i4}\}$ are the edges from the neighbors to the central point.

4.3.3. GGM Convolution

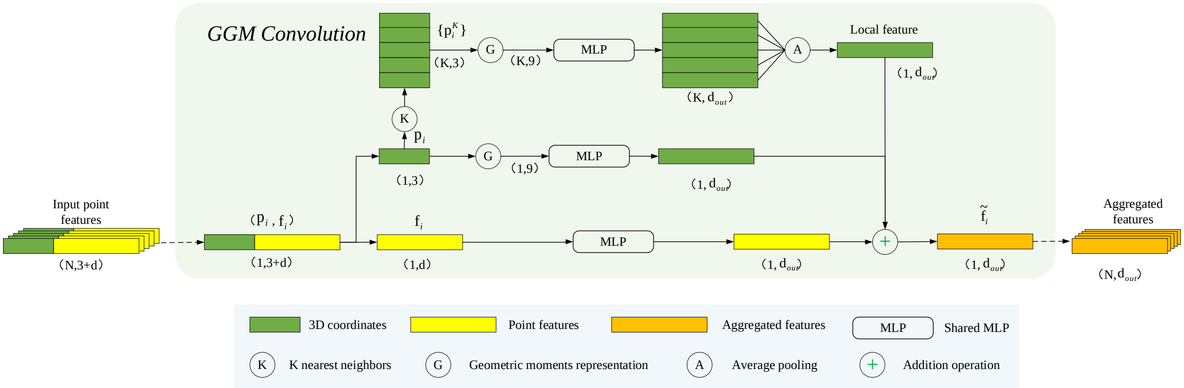


Figure 5. Architecture of the GGM Convolution.

Consider an F-dimensional point cloud with n points, denoted by $X=\{p_1,\ldots,p_n\}\subseteq R^F$. For the simplest setting of $F=3$, each point only contains its 3D coordinates $p_i=(x_i,y_i,z_i)$; it is also possible to contain the additional per-point features, e.g. color, surface normal, and spectral value. In a hierarchical neural network, the subsequent layer operates on the output of the previous layer, so more generally the dimension F represents the feature dimension of a given layer[49], which indicates as the point features in Figure 5.

As show in Figure 5, the point features are combined with its 3D coordinates as the input to the GGM convolution, and the GGM convolution contains two main branches. The bottom branch indicates the input point features directly fed into a Multi-Layer Perceptrons (MLP), through which the dimension of the input features would be raised. The other branch is designed to extract the local features of each point. Firstly, we construct a local directed graph by searching its k-nearest neighbors and calculate the first and second order geometric moments representations of the point and its local directed edges, respectively. Then, they were separately fed into two independent MLPs, and the output of the MLP on the top branch is aggregated by the average-pooling operation. Finally, an addition operation is utilized to fuse all the outputs.

The reason why we use the average-pooling operation instead of the max-pooling operation to aggregate the extracted local features is that we want to obtain the local feature as the compensation of the point feature. The max-pooling operation takes only the max value at each feature channel, which tends to capture the most "special" features and shows less representativeness. To guarantee the extracted compensation feature is sufficiently reliable, the more reasonable local feature should be the average of all local features extracted from the edges.

Although the concatenation and multiplication operations are quite commonly used in related methods. For example, PointNet++ [50] and DGCNN [49] fuse features by using concatenation operation, RS-CNN [51] and GACNet [52] fuse features by using multiplication operation. Here, we choose the addition operation to fuse features. The main reasons are as follows: (1) the concatenation operation is effective to fuse the multiscale features, and the multiplication operation is commonly used in attention mechanism methods. However, we are fusing the features extracted from higher order geometric moments of original coordinates, which contain different forms of underlying geometric information. Thus, we cannot use the concatenation or multiplication operations roughly here. (2) Essentially, the feature space in deep learning is a kind of probability space, the convolution could be viewed as the filter. The value in different channel of the output feature shows the probability that passes the filter with specific parameters. The addition operation could highlight the befitting filters and restrain the improper filters, which effectively refine the point feature.

4.3.4. Network Architecture

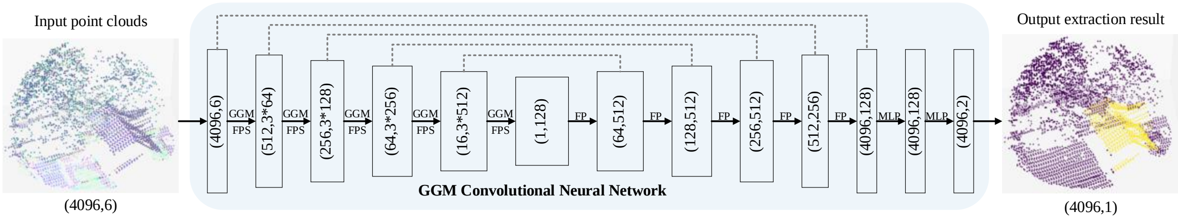


Figure 6. GGM Convolutional Neural Networks architecture. (N, D) represents the number of points and feature dimension respectively. GGM: Graph Geometric Moment Convolution, FPS: Farthest Point Sampling, FP: Feature Propagation, MLP: Multi-Layer Perceptrons.

Figure 6 shows the detailed architecture of the GGM Convolutional Neural Networks. The network follows the widely-used hierarchical structure. After sample generation, the point clouds of each test area are split into many batches, and each batch contains 4096 points. Through the GGM Convolutional Neural Networks, the input points, which contains spatial coordinates and three spectral values, are labeled with their predict labels, e.g. 1 indicates the building point and 0 indicates the background point. The details of GGM Convolutional Neural Networks are as follows:

Hierarchical Structure: Our hierarchical structure is referenced from PointNet++. The hierarchical structure is composed of a number of set abstraction levels. The set abstraction level is made of two key layers: sampling layer and GGM convolution layer. The sampling layer selects a set of points from the input points via the Farthest Point Sampling (FPS) algorithm, which defines the centroids of local regions. The GGM convolution layer is illustrated in Section 4.3.3, which combines local feature extraction and grouping function. A set abstraction level takes an $N \times (d + C)$ matrix as input that is from N points with d -dimensional coordinates and C -dimensional point feature. It outputs an $N' \times (d + C')$ matrix of N' subsampled points with d -dimensional coordinates and new C' -dimensional feature vectors summarizing local features.

Farthest Point Sampling (FPS): In the sampling layer, we utilize iterative farthest point sampling (FPS) to choose a subset of points. Given the input points $\{x_1, x_2, \ldots, x_n\}$, firstly, the FPS randomly picks one point x_i as the seed point, then, calculates the distance from the input points to seed point and selects the most distant point as the next seed point. The selected points will be removed from the input points. Finally, all the selected seed points constitute the subset of input points with a specified size. In this way, the selected subset of input points could have good coverage of the entire input points.

Multi-scale grouping (MSG): Inspired by PointNet++, we implement the MSG strategy to make our model more robust. For every set abstraction level, we apply a GGM convolution with three different scales, e.g. we set the k-nearest neighbors of 16, 32 and 48 for the first set abstraction level. Then, the features at different scales are concatenated to form a multi-scale feature. Thus, as shown in Figure 6, we use 3*D to indicate the number of scales and the dimension of features at different scales, respectively.

Feature Propagation (FP): To predict the labels for all the original points, we need to propagate features from subsampled points to the original points. Here, we choose a hierarchical propagation strategy similar to PointNet++. Firstly, we find one nearest neighboring point for each point, whose point feature set is up-sampled through a nearest-neighbor interpolation. Then, the up-sampled features are concatenated with the intermediate feature produced by set abstraction layers through skip connections, which is indicated by the dotted lines in Figure 6. Finally, we apply a shared MLP and ReLU layer to the concatenated features to update each point's feature vector.

Final Label Prediction: The final label of each point is obtained through two shared MLP with 128 and 2 output dimensions. After a softmax operation, the max value of the two channels indicates the final predicted label.

5. Experimental Results and Discussion

5.1. Implementation details

Our training strategy is the same as in [49]. We used the stochastic gradient descent (SGD) optimizer with 0.1 as the initial learning rate in our network, and the learning rate declined fifty percent after each thirty iterations. Since we applied the MSG strategy in our model, the number of the nearest neighbors k varied from 16 to 64 in different set abstraction levels. The number of input points, batch size, and momentum were 4096, 16, and 0.9, respectively. For every MLP layer, we used the LeakyReLU with 0.2 negative slope as the activation function and applied Batch normalization. After training the whole network, we saved the best performance training variables of the network, and set it as the input in the retraining process. We adjusted the hyper-parameters during the retraining process. Furthermore, we trained our model on a NIVIDIA 2080 TI GPU.

5.2. Accuracy evaluation metrics

To assess the quality of the proposed methodology, we used some metrics commonly used for semantic segmentation and useful for binary classification task. Let TP, FP, FN denote the total numbers of true positives, false positives, and false negatives, respectively. Then we calculate precision/correctness, recall/completeness as following:

$$Precision = \frac{TP}{TP + FP}, \tag{9}$$

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

where the $Precesion$ is the proportion of the true positives over the extracted building points, the $Recall$ is the proportion of true positives with regard to the labeled ground-truth building points. The higher these metrics, the better the performance of the method.

Besides, we employed the $F-measure$ derived from the precision and recall values for the point-wise overall evaluation, which is defined as follows:

$$Fmeasure = \frac{(1+\beta^2)TP}{(1+\beta^2)TP + \beta^2 FN + FP}. \tag{11}$$

For simplicity, we set β=1.

Another useful metric is Intersection over Union (IoU), which is an average value of the intersection of the prediction and ground truth regions over the union of them. Here we adapted this metric to the binary case, because in our data there are many more points which belong to the background than those belonging to the building rooftops. Therefore, in our case, IoU is defined as the number of points labeled as building in on both the ground truth and predicted result, divided by the total number of points labeled as building in each of them. We calculate it as follows:

$$IoU = \frac{TP}{n_{pred} + n_{gt}}, \tag{12}$$

where n_{pred} is the number of points labeled as buildings in the predicted result and n_{gt} is the one in the ground truth.

5.3. Parameter Sensitivity

5.3.1. Spectral information

To investigate the effect of the input feature selection, e.g. spatial and/or spectral information, we trained our model based on two sets of input data. Since the main characteristic of our model is learning local features from geometric moments, we considered the spatial coordinates as the essential feature. The first model was trained using 3D coordinates only. The second model was trained using both 3D coordinates and spectral information (three channels) for each point.

We evaluated our model on area_6 and area_7. After sample generation, these two test scenes were split into 257 and 474 samples, respectively. As we mentioned in Section 4.2, for the overlapped part between samples, we counted the predicted labels from different samples of the same point, and chose the most predicted label as its final predicted label. After we obtained the predicted label for each point in the test scenes, we calculated a point-based evaluation result for each test scene by the four metrics mentioned above. Here, we defined the point-based evaluation result of the combination of the test scenes as the comprehensiveness result, instead of the commonly used average result.

As shown in Table 3, the second model achieved better performance on Area_6, Area_7 and comprehensiveness for each metric. This suggests that combining both features could improve the accuracy of the results. It also validates the powerful geometric feature learning ability of our model. The results are quite promising even by only using 3D coordinates as input.

Table 3. A comparison between training with the different input feature.

Input	Area	Precision	Recall	Fmeasure	IoU
Coordinates	Area_6	86.0	85.0	85.5	74.7
	Area_7	85.0	85.3	85.1	74.1
	comprehensiveness	86.6	86.1	86.3	76.0
Coordinates and spectral values	Area_6	92.0	88.1	90.0	81.9
	Area_7	95.0	86.3	90.4	82.5
	comprehensiveness	93.9	87.4	90.5	82.7

5.3.2 Sample size

Furthermore, we investigated the effect of sample size by training our model based on three different sample sizes. As we mentioned in Section 4.2, during the sample generation stage, we can set the number of points each sample contained. Considering the limitation of GPU memory, we set the maximum number of points as 4096, and the other two were set as 2048 and 1024. All the models were trained using the same input features (coordinates and spectral values).

In Table 4, "#points" indicates the number of points in each sample. As we can see, the larger scale performed better than the smaller scale. For deep learning methods, the larger scale input sample provides the more comprehensive information and the better geometric continuity of objects in the scene, which decides "how good" feature the model can learn from. And that is the reason why the larger scale performed better. The results also confirmed our speculation.

Table 4. A comparison between training with different sample sizes.

#points	Area	Precision	Recall	Fmeasure	IoU
1024	Area_6	85.6	85.7	85.6	74.9
	Area_7	89.2	85.4	87.2	77.4
	comprehensiveness	88.2	86.3	87.2	77.3
2048	Area_6	87.2	85.4	86.3	75.9
	Area_7	92.3	86.0	89.1	80.3
	comprehensiveness	90.6	84.9	87.6	78.0
4096	Area_6	92.0	88.1	90.0	81.9
	Area_7	95.0	86.3	90.4	82.5
	comprehensiveness	93.9	87.4	90.5	82.7

5.4. Results and Comparisons

Since there is no previous method proposed for building extraction from ALS data fitting for our framework, to better evaluate our method, we compared our model with a representative set of previous state-of-the-art networks designed for semantic segmentation on point clouds. The compared networks include PointNet[46], KCNet[56], DGCNN[49], and RS-CNN[51].

Table 5 shows the point-based evaluation comparison results for the two test scenes. All experiments used the same input data size (4096 points) and features (coordinates and three spectral values), and the training iteration was configured as 200 for all. As shown in Table 5, our model, GGM Convolutional Neural Networks, achieved significantly better performance than the other networks, especially on Recall and IoU metrics. The KCNet achieved higher precision in area_6, but the other three metrics were observably below ours. Hence, for the overall extraction quality, our model achieved a better performance, which was also demonstrated by the following visualization of results.

Figure 7 shows the visualization of the comparison results. For each model, we selected the same test area to show its overall extraction result (left part) and chose three kinds of typical buildings in the scene for detailed inspections (right part). As reflected by the overall results, most of models recognized all buildings in object-level regardless of the building size, even the small-size buildings (less than 5 m^2) could be recognized a part points. This demonstrated the powerful inference capability of deep learning methods. Our model achieved a more complete building extraction result with less misrecognition points. For example, the PointNet and RS-CNN misrecognized some powerline points as the building points, because they have the similar altitudes, which was indicated by the black circle in Figures 7 (a) and (d).

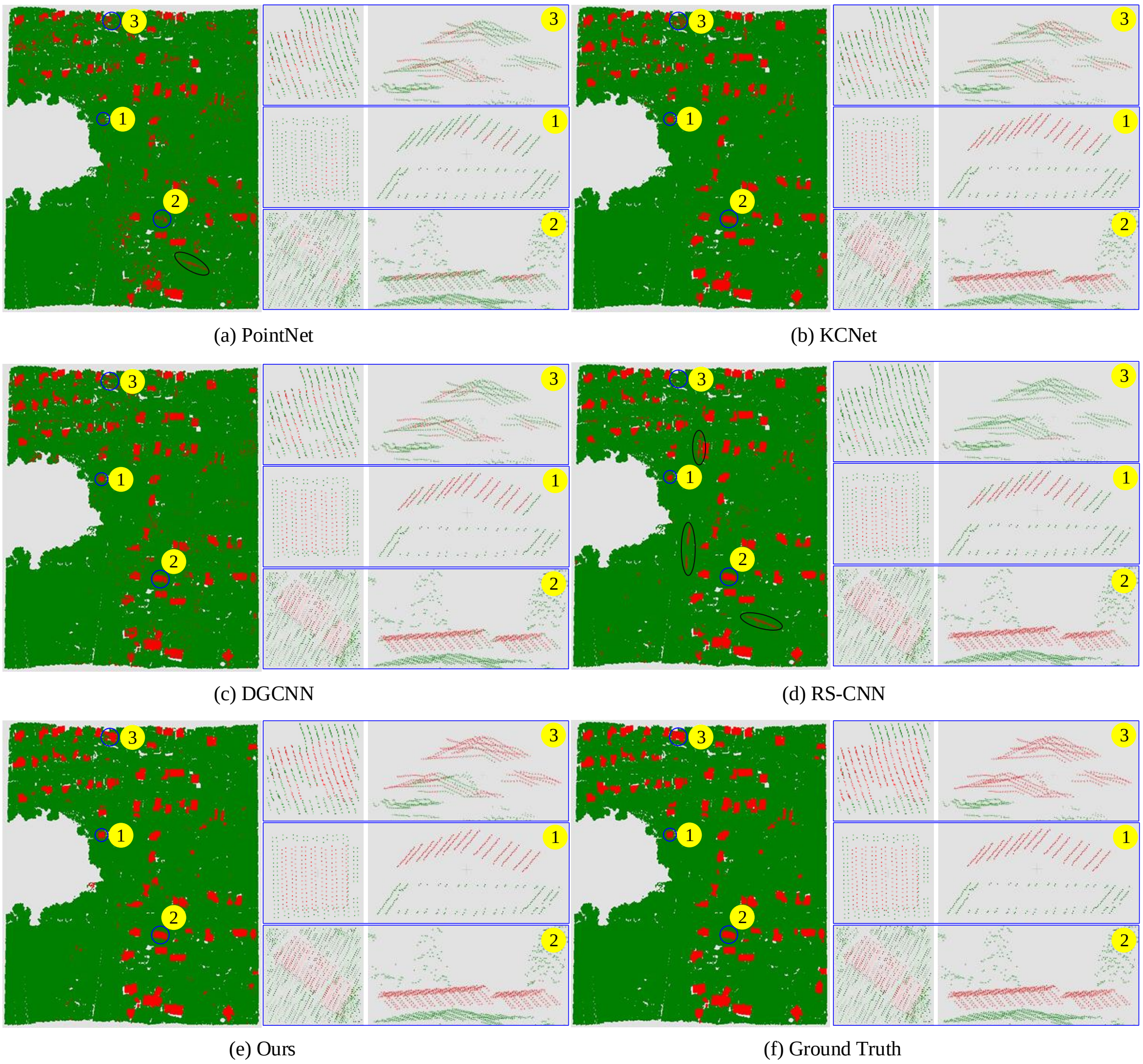
To compare the extraction results of these models in detail, we chose three typical buildings to represent the extraction difficulty in three levels. In Figure 7, the details are showed in the right blue bounding rectangles, where the two images are, respectively, the vertical view and side view of a building, and the numbers "1", "2", and "3" with yellow background indicate the easy, normal and hard levels, respectively. In the easy case, the building structure is simple, and surrounding environment is clear (only flat grass). Our model completely recognized all the building points and separated them from the grass points clearly. The other models failed to recognize part of the building points. In the normal case, two buildings with different sizes and heights are combined, and they are surrounded by tall trees. Although it is much harder than the easy case, our model also completely recognized all the building points, but misrecognized three tree points as the building points. Similarly, the performance of our model is obviously better than the others. In the hard case, the building is a multi-story building with irregular rooftops, which has more complex structure than the former two cases. Our model relatively completely recognized the main rooftop and one side rooftop, but only few building points of the other side rooftop with chimney were recognized. As for the other models, only some cracked pieces were recognized.

The accuracies and visualization results demonstrated the effectiveness and efficiency of the proposed framework and methods. Furthermore, the test scenes we used are more complicated than the commonly used urban areas, which dramatically increase the difficulty for building extraction tasks. In addition, the point-based evaluation we used has higher resolution, which means the stricter evaluation way, compared with pixel-based and object-based evaluations.

Table 5. Point-based building extraction comparison results on test scenes.

Model	Area	Precision	Recall	Fmeasure	IoU
PointNet	Area_6	74.4	55.1	63.3	46.3
	Area_7	72.3	56.5	63.4	46.4
	comprehensiveness	72.6	56.1	63.3	46.3
KCNet	Area_6	96.0	77.8	85.9	75.3
	Area_7	92.8	78.6	85.1	73.9
	comprehensiveness	93.6	78.0	85.3	74.1
DGCNN	Area_6	79.5	76.2	77.8	63.7

	Area_7	79.5	73.4	76.4	61.8
	comprehensiveness	79.3	73.6	76.4	61.8
RS-CNN	Area_6	80.9	77.8	79.3	65.8
	Area_7	87.3	78.6	82.7	70.6
	comprehensiveness	85.0	79.2	82.0	69.5
Ours	Area_6	92.0	88.1	90.0	81.9
	Area_7	95.0	86.3	90.4	82.5
	comprehensiveness	93.9	87.4	90.5	82.7



(a) PointNet (b) KCNet (c) DGCNN (d) RS-CNN (e) Ours (f) Ground Truth

Figure 7. The visualization of comparison results. The green colored points are the background (non-building) points, and the red colored points are the recognized or labeled building points. The blue circles in the left images indicate the selected three kinds of typical buildings, and the black circles in (a) and (d) indicate the misrecognized building points from powerline points. The three blue bounding rectangles on the right contain the corresponding detailed visualization in the left images.

6. Conclusions

In this paper, we proposed a novel deep learning-based framework for building extraction from multi-spectral point cloud data. Meanwhile, a sample generation method, a convolution operator and a convolutional neural network implemented in the framework were proposed. The proposed framework provided a novel architecture for the better application of deep learning methods in this research field. Besides, with the characteristic of good universality, theoretically, the proposed framework could handle any point sets and be implemented in any networks, which could greatly promote the practical applications of the proposed framework. As for the point-based evaluation we used in this paper, obviously, it is more difficult to achieve the same accuracy, compared with the traditional used pixel-based and object-based evaluation. But it has higher resolution and reflects the direct connection with the real world, which is of greater practical significance. Compared with the other state-of-the-art networks, our method achieved the best comprehensive performance with regard to the four metrics. In addition, the corresponding visualization showed the strong capacity of our model, especially for the difficult cases such as the buildings surrounded by tall trees and the multi-storey buildings with complex structure rooftops, our model still achieved outstanding performance than the others. In future work, we will test the influence of adding the other additional features to our method, and try to process the larger area scenes by using our method in our framework.

Author Contributions: Methodology, D.L. (first author); Validation, D.L. (first author) and X.S.; Writing-original draft preparation, D.L. (first author); Writing-review and editing, Y.Y. and D.L.; resources, H.G. and J.L.; All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the National Natural Science Foundation of China under Grants 41671454 and 41971414, and in part by the Six Talent Peaks Project in Jiangsu Province under Grant XYDXX-098.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Khoshelham, K., Nardinocchi, C., Frontoni, E., Mancini, A., Zingaretti, P., 2010. Performance evaluation of automated approaches to building detection in multi-source aerial data. ISPRS J. Photogramm. Rem. Sens. 65 (1), 123–133.
2. Tomljenovic, I., Höfle, B., Tiede, D., Blaschke, T., 2015. Building extraction from airborne laser scanning data: an analysis of the state of the art. Rem. Sens. 7 (4), 3826–3862.
3. G. Groger and L. Plumer, “CityGML—Interoperable semantic 3D city models,” ISPRS J. Photogramm. Remote Sens., vol. 71, pp. 12–33, 2012.
4. Khoshelham K , Oude Elberink S , Xu S . Segment-Based Classification of Damaged Building Roofs in Aerial Laser Scanning Data[J]. IEEE Geoence & Remote Sensing Letters, 2013, 10(5):1258-1262.
5. B. Sirmacek, H. Taubenboeck, P. Reinartz, and M. Ehlers, “Performance evaluation for 3-D city model generation of six different DSMs from air- and spaceborne sensors,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 5, no. 1, pp. 59–70, Feb. 2012.
6. Pan S , Guan H , Yu Y , et al. A Comparative Land-Cover Classification Feature Study of Learning Algorithms: DBM, PCA, and RF Using Multispectral LiDAR Data[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2019, 12(4):1314-1326.
7. E. Maltezos, A. Doulamis, N. Doulamis and C. Ioannidis, "Building Extraction From LiDAR Data Applying Deep Convolutional Neural Networks," in IEEE Geoscience and Remote Sensing Letters, vol. 16, no. 1, pp. 155-159, Jan. 2019, doi: 10.1109/LGRS.2018.2867736.
8. J. Zhang, X. Lin, and X. Ning, “SVM-based classification of segmented airborne LiDAR point clouds in urban areas,” Remote Sens., vol. 5, no. 8, pp. 3749–3775, Jul. 2013
9. Y. Verdie, F. Lafarge, and P. Alliez, “LOD generation for urban scenes,” ACM Trans. Graph., vol. 34, no. 3, May 2015, Art. no. 30.
10. Peter D , Norbert P . A Comprehensive Automated 3D Approach for Building Extraction, Reconstruction, and Regularization from Airborne Laser Scanning Point Clouds[J]. Sensors, 2008, 8(11):7323-7343.
11. Sampath A , Shan J . Segmentation and Reconstruction of Polyhedral Building Roofs From Aerial Lidar Point Clouds[J]. IEEE Transactions on Geoence and Remote Sensing, 2010, 48(3):1554-1567.
12. S. Ural and J. Shan, “Min-cut based filter for airborne Lidar data,” in Proc. Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci. (PRSSIS), 2016, pp. 395–401.
13. Bittner K , Cui S , Reinartz P . Building Extraction from Remote Sensing Data using Fully Convolutional Networks[C]// Isprs Hannover Workshop: Hrigi. 2017.

14. Hamed N F , Shafri H Z M , Ibrahim S M , et al. Deep Learning Approach for Building Detection Using LiDAR-Orthophoto Fusion[J]. Journal of Sensors, 2018, 2018:1-12.
15. E. Maltezos, A. Doulamis, N. Doulamis and C. Ioannidis, "Building Extraction From LiDAR Data Applying Deep Convolutional Neural Networks," in IEEE Geoscience and Remote Sensing Letters, vol. 16, no. 1, pp. 155-159, Jan. 2019, doi: 10.1109/LGRS.2018.2867736.
16. Sohn G , Huang X , Tao V . Using a Binary Space Partitioning Tree for Reconstructing Polyhedral Building Models from Airborne Lidar Data[J]. Photogrammetric Engineering & Remote Sensing, 2008, 74(11):1425-1440.
17. Zhou G , Song C , Simmers J , et al. Urban 3D GIS From LiDAR and digital aerial images[J]. Computers & Geoences, 2004, 30(4):345-353.
18. Maas H G , Vosselman G . Two algorithms for extracting building models from raw laser altimetry data[J]. Isprs Journal of Photogrammetry & Remote Sensing, 1999, 54(2-3):153-163.
19. Qian-Yi Zhou, Ulrich Neumann. Fast and extensible building modeling from airborne LiDAR data[C]// Acm Sigspatial International Conference on Advances in Geographic Information Systems. ACM, 2008.
20. Poullis C , You S . Photorealistic Large-Scale Urban City Model Reconstruction[J]. IEEE Transactions on Visualization & Computer Graphics, 2009, 15(4):654-669.
21. You R J , Lin B C . Building Feature Extraction from Airborne LiDAR Data Based on Tensor Voting Algorithm[J]. Photogrammetric Engineering & Remote Sensing, 2011, 77(12):1221-1231.
22. KyoHyouk Kim, Jie Shan. Building roof modeling from airborne laser scanning data based on level set approach. ISPRS Journal of Photogrammetry and Remote Sensing, Volume 66, Issue 4, 2011, Pages 484-497, ISSN 0924-2716.
23. Sun S , Salvaggio C . Aerial 3D Building Detection and Modeling From Airborne LiDAR Point Clouds[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2013, 6(3):1440-1449.
24. Zou X , Feng Y , Li H , et al. An Adaptive Strips Method for Extraction Buildings From Light Detection and Ranging Data[J]. IEEE Geoence and Remote Sensing Letters, 2017, PP(10):1-5.
25. Santos R C D , Galo M , Carrilho A C . Extraction of Building Roof Boundaries From LiDAR Data Using an Adaptive Alpha-Shape Algorithm[J]. IEEE Geoence and Remote Sensing Letters, 2019:1-5.
26. Liu, C, Shi, et al. Automatic Buildings Extraction From LiDAR Data in Urban Area by Neural Oscillator Network of Visual Cortex[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing, 2013, 6(4):2008-2019.
27. Awrangjeb M , Ravanbakhsh M , Fraser C S . Automatic Building Detection Using LIDAR Data and Multispectral Imagery[C]// International Conference on Digital Image Computing: Techniques & Applications. IEEE, 2011.
28. Awrangjeb, Mohammad & Zhang, Cissce & Fraser, Clive. (2012). AUTOMATIC RECONSTRUCTION OF BUILDING ROOFS THROUGH EFFECTIVE INTEGRATION OF LIDAR AND MULTISPECTRAL IMAGERY. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. I-3. 203-208. 10.5194/isprsannals-I-3-203-2012.
29. Awrangjeb M , Zhang C , Fraser C S . Automatic extraction of building roofs using LIDAR data and multispectral imagery[J]. Isprs Journal of Photogrammetry & Remote Sensing, 2013, 83:1-18.
30. Awrangjeb, M. and Fraser, C. S.: Rule-based segmentation of LIDAR point cloud for automatic extraction of building roof planes, ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., II-3/W3, 1–6, https://doi.org/10.5194/isprsannals-II-3-W3-1-2013, 2013.
31. Mohammad A , Clive F . Automatic Segmentation of Raw LIDAR Data for Extraction of Building Roofs[J]. Remote Sensing, 2014, 6(5):3716-3751.
32. Syed G , Mohammad A , Guojun L . An Automatic Building Extraction and Regularisation Technique Using LiDAR Point Cloud Data and Orthoimage[J]. Remote Sensing, 2016, 8(3):27.
33. Hu Q , Yang B , Xie L , et al. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds[J]. 2019.
34. M.K. Hu, Visual pattern recognition by moment invariants, IRE Transactions on Information Theory 8 (2) (1962) 179–187.
35. S.O. Belkasim, M. Shridhar, M. Ahmadi, Pattern recognition with moment invariants: a comparative study and new results, Pattern Recognition 24 (12) (1991) 1117–1138.
36. J. Flusser, T. Suk, Pattern recognition by affine moment invariants, Pattern Recognition 26 (1) (1993) 167–174.

37. L.M. Luo, C. Hamitouche, J.L. Dillenseger, J.L. Coatrieux, A momentbased three-dimensional edge operator, IEEE Transactions on Biomedical Engineering 40 (7) (1993) 693–703.
38. S.T. Liu, W.H. Tsai, Moment-preserving corner detection, Pattern Recognition 23 (5) (1990) 441–446.
39. N. Yokoya, M.D. Levine, Range image segmentation based on differential geometry: a hybrid approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (6) (1989) 643–649.
40. M. Tuceryan, Moment-based texture segmentation, Pattern Recognition Letters 15 (7) (1994) 659–668.
41. C.H. Teh, R.T. Chin, On image analysis by the methods of moments, IEEE Transactions on Pattern Analysis and Machine Intelligence 10 (4) (1988) 496–513.
42. M.R. Teague, Image analysis via the general theory of moments, Journal of the Optical Society of America 70 (1980) 920–930.
43. C.H. Lo, H.S. Don, 3-D moment forms: their construction and application to object identification and positioning, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (10) (1989) 1053–1064.
44. Y.S. Abu-Mostafa, D. Psaltis, Recognitive aspects of moment invariants, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (6) (1984) 698–706.
45. M. Joseph-Rivlin, A. Zvirin, and R. Kimmel, "Mo-Net: Flavor the moments in learning to classify shapes," in ICCVW, 2018.
46. C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in CVPR, 2017, pp. 652–660.
47. Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. IEEE Transactions on Neural Networks, 20(1):61–80, 2009.
48. Niepert M , Ahmed M , Kutzkov K . Learning Convolutional Neural Networks for Graphs[J]. 2016.
49. Wang, Yue, Sun, Yongbin, Liu, Ziwei, Sarma, Sanjay E, Bronstein, Michael M, & Solomon, Justin M. (2018). Dynamic graph cnn for learning on point clouds.
50. Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Advances in Neural Information Processing Systems, pages 5105–5114, 2017.
51. Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis" in CVPR, 2019, pp. 1–10.
52. L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in CVPR, 2019, pp. 10 296–10 305.
53. Pan S , Guan H , Chen Y , et al. Land-cover classification of multispectral LiDAR data using CNN with optimized hyper-parameters[J]. ISPRS Journal of Photogrammetry and Remote Sensing, 2020, 166:241-254.
54. Briese C , Pfennigbauer M , Lehner H , et al. RADIOMETRIC CALIBRATION OF MULTI-WAVELENGTH AIRBORNE LASER SCANNING DATA[J]. 2012.
55. Poullis C . A Framework for Automatic Modeling from Point Cloud Data[J]. IEEE Trans Pattern Anal Mach Intell, 2013, 35(11):2563-2575.
56. Shen Y , Feng C , Yang Y , et al. Mining Point Cloud Local Structures by Kernel Correlation and Graph Pooling[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2018.