remote sensing

MDPI

Article

AFGL-Net: Attentive Fusion of Global and Local Deep Features for Building Façades Parsing

Dong Chen [1], Guiqiu Xiang [1], Jiju Peethambaran [2,*], Liqiang Zhang [3], Jing Li [1] and Fan Hu [1]

[1] College of Civil Engineering, Nanjing Forestry University, Nanjing 210037, China; chendong@njfu.edu.cn (D.C.); guiqiuxiang@njfu.edu.cn (G.X.); jingli@njfu.edu.cn (J.L.); fanhu@njfu.edu.cn (F.H.)
[2] Department of Mathematics and Computing Science, Saint Mary's University, Halifax, NS B3P 2M6, Canada
[3] The State Key Laboratory of Remote Sensing Science, Faculty of Geographical Science, Beijing Normal University, Beijing 100875, China; zhanglq@bnu.edu.cn
* Correspondence: jiju.poovvancheri@smu.ca

Abstract: In this paper, we propose a deep learning framework, namely AFGL-Net to achieve building façade parsing, i.e., obtaining the semantics of small components of building façade, such as windows and doors. To this end, we present an autoencoder embedding position and direction encoding for local feature encoding. The autoencoder enhances the local feature aggregation and augments the representation of skeleton features of windows and doors. We also integrate the Transformer into AFGL-Net to infer the geometric shapes and structural arrangements of façade components and capture the global contextual features. These global features can help recognize inapparent windows/doors from the façade points corrupted with noise, outliers, occlusions, and irregularities. The attention-based feature fusion mechanism is finally employed to obtain more informative features by simultaneously considering local geometric details and the global contexts. The proposed AFGL-Net is comprehensively evaluated on Dublin and RueMonge2014 benchmarks, achieving 67.02% and 59.80% mIoU, respectively. We also demonstrate the superiority of the proposed AFGL-Net by comparing with the state-of-the-art methods and various ablation studies.

Keywords: façade parsing; semantic segmentation; MLP; autoencoder; global transformer; attentive feature fusion; Dublin dataset; RueMonge2014 dataset

Citation: Chen, D.; Xiang, G.; Peethambaran, J.; Zhang, L.; Li, J.; Hu, F. AFGL-Net: Attentive Fusion of Global and Local Deep Features for Building Façades Parsing. *Remote Sens.* **2021**, *13*, 5039. https://doi.org/10.3390/rs13245039

Academic Editor: Giuseppe Scarpa

Received: 30 October 2021
Accepted: 8 December 2021
Published: 11 December 2021

1. Introduction

Buildings and architectures represent the most fundamental and important element of cities. Consequently, 3D models of buildings have been widely used in many modern day applications such as indoor and/or outdoor navigation [1,2], building energy modeling [3], 3D visualization and generalization [4], and building abstraction [5], among others. The ability of Light Detection and Ranging (LiDAR) technique to capture complex structures of building rooftop and façade components, e.g., windows, doors, and balconies led to an intensive use of point clouds for creating photo realistic 3D urban scenes. Reconstruction of building models is an active area of research among photogrammetry, computer graphics, computer vision, and remote sensing communities [6]. However, turning the point clouds to semantically rich, geometrically accurate and topologically correct building models for a better understanding and analysis remains an open challenge [7]. Compared with the recognition of rooftop superstructures and ornaments, parsing building façade components such as windows, doors, and balconies is extremely challenging because of the complexities of façade components (diverse windows/doors, irregular arrangements, etc.) and the low quality of point clouds generally caused by outliers, irregularities, and missing data due to occlusions and/or self-occlusions.

Recently, deep learning has shown amazing results for point cloud semantic segmentation. However, because the point cloud is scattered, unordered, and unorganized, it

should be transformed into regular data representation before employing the existing convolutional neural networks. Based on the data representation, the deep learning approaches are roughly divided into three categories: multiview-based methods (MVCNN [8], DeePr3SS [9], SnapNet [10] etc.), volumetric-based methods (VoxNet [11], SparseConvNet [12], Oct-Net [13], Kd-Net [14] etc.), and point-based methods such as the pioneering work PointNet [15] and its successor PointNet++ [16]. To use standard CNN, the point clouds are usually projected into multi-view, volumetric, or other regular representations. However, these conversions will alter the structure of the point clouds, causing problems such as precision loss, high computational cost, and/or large memory consumption. By contrast, the point-based methods try to directly use the operations such as multilayer perceptron (MLP) and convolution to derive pointwise features from unstructured 3D point clouds. Semantic parsing of LiDAR scans has been an active area of research for the last two decades. Many algorithms have been proposed for semantic parsing in the context of building modeling and scene understanding. We review three classes of deep learning techniques, i.e., pointwise MLP methods, point convolution methods, and graph convolution methods for semantic parsing in the subsequent paragraphs.

1.1. Pointwise MLP Methods

This type of method usually uses the shared MLP as the basic unit to extract pointwise features and uses a symmetric function to counter disorder after extracting the features of individual point. The most representative work is PointNet [15] and its successor PointNet++ [16]. PointNet [15] applies deep learning on the disordered point clouds by using a symmetric function to achieve permutation invariance. More specifically, it uses the shared MLP to obtain pointwise features and uses the max-pooling layer to aggregate the global features of all points. PointNet++ [16] as an extension work of the PointNet [15], uses an autoencoder to explore the structures of point clouds hierarchically. To enhance the description of the shape features, Jiang et al. [17], inspired by the two-dimensional shape descriptor SIFT [18], propose a module called PointSIFT. The PointSIFT encodes features in different directions, followed by embedding this module into PointNet++. Based on PointNet++, PointWeb [19] constructs a fully connected web to capture local context and uses an Adaptive Feature Adjustment (AFA) module to enhance the derived features. PointWeb obtains the relationship between all the point pairs in the local neighborhood point set and achieves the information exchange and feature optimization between the local point pairs. To obtain more distinguished features, SO-Net [20] adopts multi-scale and multi-resolution strategies to aggregate hierarchical features on both individual points and Self-Organizing Map (SOM) nodes. Chiang et al. [21] and Geng et al. [22] propose the multi-scale feature aggregation methods, which upsample the features of the encoder and decoder at different scales to the size of the input points. After that, they obtain the global structure of the point clouds by the feature aggregation. Hu et al. [23] propose an efficient and lightweight neural network architecture called RandLA-Net, which adopts a random point sampling strategy for processing point clouds at a large scale and employs a novel local feature aggregation module to preserve the geometric details. Although this type of method can obtain the pointwise features using the shared MLPs and global features using symmetrical pooling functions, they cannot capture the mutual interactions between neighborhood points.

1.2. Point Convolution Methods

This type of method tries to define a suitable convolution operation across the whole neighborhood area of the sampling points for processing the disordered point clouds. A series of publications along this line demonstrates the effectiveness of point convolution. For example, Hua et al. [24] use a 3D convolution kernel to perform convolution on the point cloud to obtain the high-level features of each point, which is simple, robust, but less efficient. Tatarchenko et al. [25] project the local neighborhood point set of each point onto the tangent plane, and then implement convolution on the tangent plane. It can handle

large-scale point clouds, but cannot fully explore the 3D geometric structure of the point clouds. Zhang et al. [26] propose an effective convolution operator called ShellConv, which uses the statistics from concentric spherical shells to define local representative features. PointCNN [27] uses $\mathcal{X}$-Conv operator to transform the input point clouds into a latent space potentially in canonical order and then implements a typical convolution on the transformed features. KPConv [28] operates on point clouds and assigns different levels of weights to the neighbor points using a small set of kernel points. This method has two work modes including rigid KPConv and deformable KPConv to adapt the kernel points to different complexity of local geometry. A-CNN [29] proposes a circular convolution in a specified circular structure and direction to capture the local geometric structure. Although promising results are obtained by this type of method, it should be noted that how to design a reasonable, continuous and discrete convolution operator that progressively assigns the weights to neighborhood points is a challenging step. In addition, it should be observed that the constructed convolution operators using the neighborhood points only reflect the geometric relationships between the center point and its neighbors and ignore the interactions between neighbor points, thereby resulting in insufficient descriptions of local shapes.

1.3. Graph Convolution Methods

Graph convolution methods tend to capture the dependency relationships and underlying connectivity patterns from the unorganized point clouds through representations of complex and diverse graphs. The graph structure is generally regarded as a good solution/representation to encode the object relationships of natural scenes. For example, DGCNN [30] embeds the EdgeConv module into the PointNet to achieve semantic segmentation. EdgeConv establishes a local map and then learns the relationships between the center point and their neighbors. This method clusters similar features in the feature space and the obtained results are promising especially for the tasks of classification and segmentation. RGCNN [31] uses the graph Laplacian matrix to adaptively capture the dynamic graph structure to describe the relationships between the features of different layers. This method is robust to noise and varying density of the point clouds. GACNet [32] proposes a graph attention convolution, which focuses on the most relevant part of the learned features by assigning appropriate attention weights to adjacent points. SPG [33] constructs a new data structure, i.e., a superpoint graph and implements a graph convolution on the superpoint graph to learn the contextual relationships between object parts. SPG can handle large-scale outdoor point clouds with millions of points. Although deep graph tries to leverage the geometry structure of the point clouds to achieve a better shape representation, it usually suffers from shallow learning from the unreasonable graph structures. Therefore, how to build a deep architecture to adaptively explore the deeper structure patterns from the reasonable graph in an open challenge [34].

1.4. Contributions

Unlike point and graph convolution methods, pointwise MLP-based methods do not require any convolution kernels and predefined graphs, it is more flexible and capable of describing local geometry shapes in details. Because of these, we embed MLPs into our autoencoder to encode pointwise features. However, MLPs cannot capture the local geometry of point clouds and the mutual interactions of neighbor points. To alleviate this deficiency, we leverage the Transformer [35] to aggregate global context façade structures. We further use the attentive feature fusion strategy to deeply fuse local and global features to form more augmented features. Given that our work is built on the previous works, we explicitly state our original contributions as follows:

- AFGL-Net Deep Neural Network: We present AFGL-Net a deep neural network for building façade parsing. AFGL-Net uses an attentive-based feature fusion mechanism to aggregate local and global features, derived by an autoencoder and a Transformer

module respectively, thereby learning enhanced and informative features to assist in solving class imbalance problem that typically occurs on building façades.
- Local Spatial Encoder: Based on the classic encoder-decoder neural network architecture, we propose an enhanced local spatial encoder by combining local position encoding and local direction encoding. The enhanced LSE encoder can easily recognize the façade component contours, e.g., window frames.
- Transformer Module: We introduce the existing Transformer module into our AFGL-Net to enhance the global/contextual feature representation. By employing these global features, AFGL-Net can perceive the small unnoticeable windows/doors by context inference from imperfect façade point clouds usually corrupted by density irregularity, outliers, and occlusions.

The remainder of this paper is structured as follows. Section 2 describes the detailed methodology including local feature encoding, global Transform feature extraction, and attentive feature fusion. In Section 3, the experimental dataset, the performance evaluation results of façade parsing accuracy based on Dublin [36] and RueMonge2014 [37] datasets are presented, analyzed, and discussed. Finally, Section 4 concludes the paper along with a few suggestions for future research topics.

2. Methodology

We propose an attentive fusion global and local deep feature network, namely AFGL-Net for building façade parsing. The framework consists of three parts: an autoencoder (Section 2.1.1) to aggregate the local features of 3D façade points using the local spatial encoding (Section 2.1.2), a Transformer (Section 2.2) to employ the global features by referring contextual components of building façades and a self-attention based feature fusion (Section 2.3) to blend the local and global features to form more discriminative façade features, representing the global façade structure and local details of 3D shape more effectively. As shown in Figure 1, AFGL-Net adopts the parallel network architecture including an autoencoder and a global-based Transformer perception module to extract local and global features. More specifically, the original features, such as the coordinates and normal vectors of building façade points are fed to the fully connected layer to map to intermediate features in high-dimensional feature space. These intermediate features are fed into the autoencoder and Transformer to extract the local and global pointwise features. After that, we use an attentional feature fusion module to fuse the local and global features to obtain more augmented fused features. We finally input these fused features into two fully-connected layers and one dropout layer to obtain the semantic label of each 3D façade point.

2.1. Local Feature Aggregation

2.1.1. Autoencoder

As shown in the red dashed box in Figure 1, the autoencoder uses the classical U-Net [38] as a backbone which consists of an encoder and a decoder with skip connection. The input point set is first provided into a fully-connected layer to improve the dimensions of the raw features of each point. In each encoding layer, the input points are randomly subsampled and input features are progressively increased using the random sampling strategy and local spatial encoding (see Section 2.1.2), respectively. In each decoding layer, we use nearest neighbor interpolation to align the sampled points of the corresponding layer of the encoder, and the local features generated by the encoder are concatenated with the corresponding upsampling features through skip connection. After that, the local spatial feature encoding is implemented again to encode the skip concatenated features.

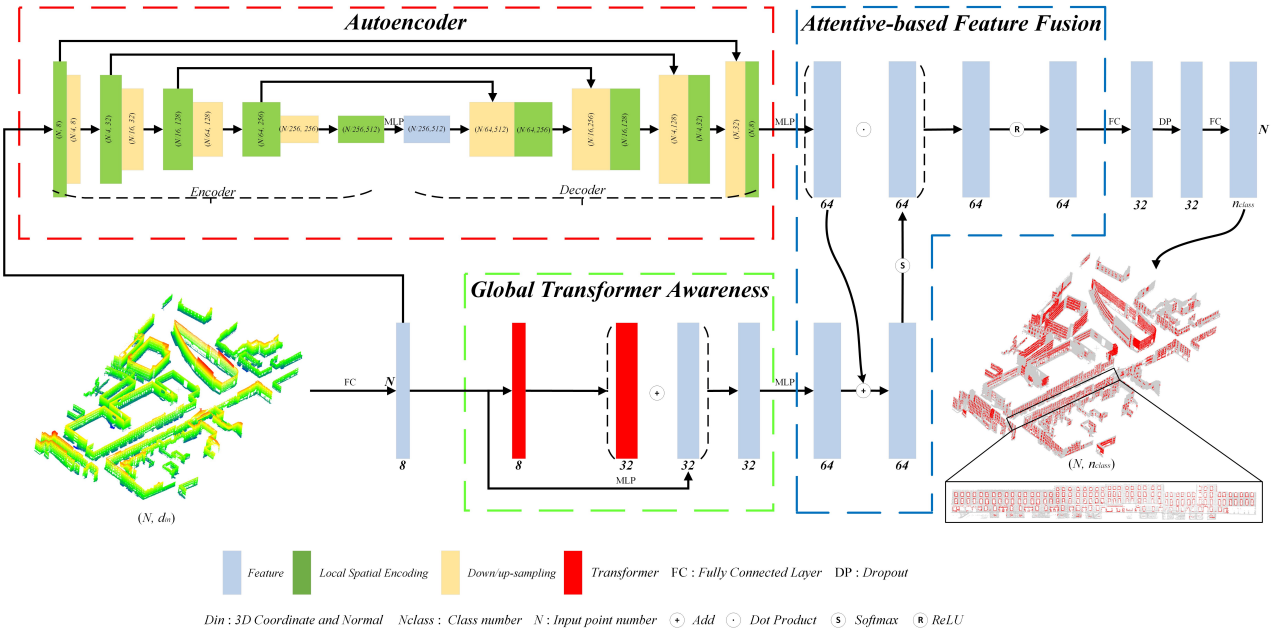

Figure 1. The pipeline of the proposed AFGL-Net. The numbers such as 8, 32 and 64 denote the feature channels.

2.1.2. Local Spatial Encoding

Local spatial encoding (LSE) aims to extract local features. It is mainly composed of local position encoding (LPE), local direction encoding (LDE) and local feature aggregation (LFA). Figure 2 illustrates the LSE module encoding process:

The overview of the local spatial encoding is described as follows:

- Each building façade point p_i contains three types of features: positions, normal vectors and the intermediate features f_i. We search K nearest neighbors of p_i through KNN algorithm [39]. After that, p_i and its K nearest neighbors are provided as input to achieve encoding of point p_i.
- More specifically, we put p_i and its K nearest neighbor points into LPE and LDE to achieve local position encoding and local direction encoding of point p_i. For local position encoding of point p_i, we put positions of p_i and its K nearest neighbor points into LPE, and obtain a triple $(N, K, 10)$, where N is the number of the processed façade points at the current sampling scale. K is the number of the nearest neighbor points of p_i, and a value of 10 is the dimension of the position feature. This position feature is composed of the x-y-z coordinates of p_i, p_i's neighbor point coordinates p_i^k, the relative point coordinates between p_i and p_i^k, and the Euclidean distance between p_i and p_i^k. For local direction encoding of point p_i, we need to put positions and normal vectors of p_i and its K nearest neighbors into LDE, and obtain a triple $(N, K, 3)$, where N is the number of the processed façade points at the current sampling scale. K is the number of nearest neighbor points of p_i, and the value of 3 is the dimension of the direction feature. The direction feature fully considers the discrepancy between p_i's normal n_i and p_i^k's normal n_i^k.
- After positional and directional encoding of point p_i, we simply concatenate encoded position and direction features together, and input them into the shared MLP to obtain the fused features, i.e., (N, K, d), where d is the dimension of the fused feature channels (see hyperparameter B in Section 3.3). We concatenate the d dimensional fused feature and another d dimensional intermediate features f_i of inputs to form a new triple $(N, K, 2d)$. Through attentive pooling, we finally generate the informative features, denoted by a tuple $(N, 2d)$.

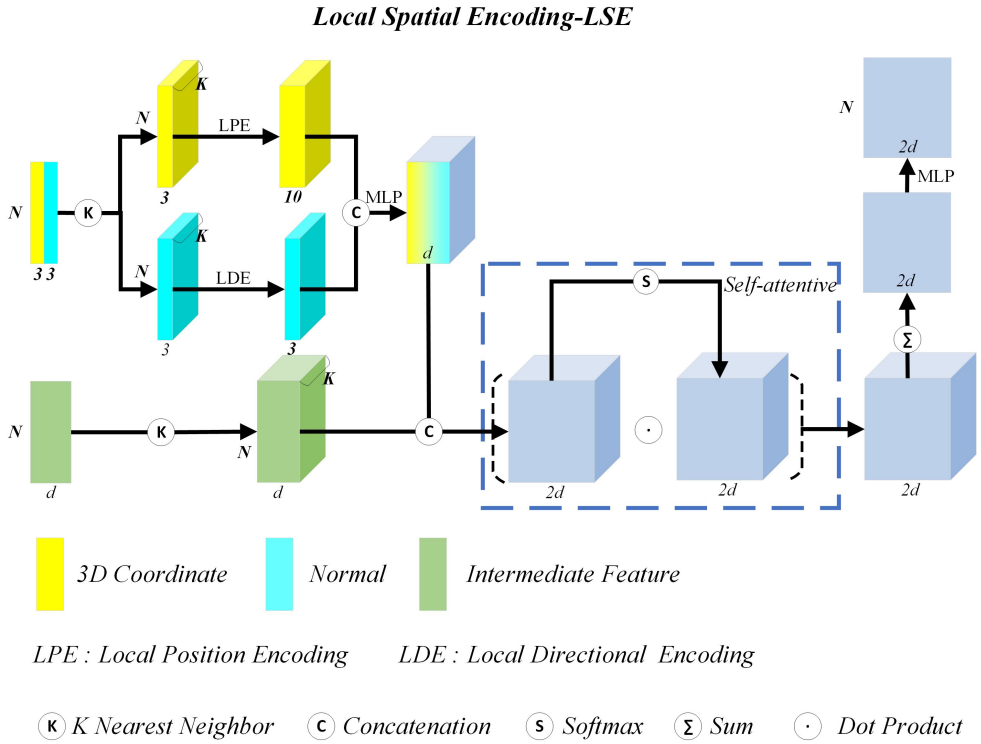

Figure 2. Local spatial encoding.

(1) Local Position Encoding

The local position encoding is proposed in RandLA-Net [23]. Given the façade point set $P = \{p_1, \cdots, p_i, \cdots, p_N\}$, we can retrieve K nearest neighbor point set $P_i = \{p_i^1, \cdots, p_i^k, \cdots, p_i^K\}$ of p_i using the KNN algorithm [39]. Any point p_i arbitrarily selected from a point set P is imposed with positional encoding according to the equation below:

$$l_i^k = \left(p_i © p_i^k © \left(p_i - p_i^k\right) © \left\|p_i - p_i^k\right\|\right) \tag{1}$$

where l_i^k denotes p_i's relative positional encoding with regard to p_i^k. Point p_i denotes the i-th input façade point. p_i^k is the k-th neighbor point of p_i. $\|.\|$ calculates the Euclidean distance between p_i and p_i^k. The symbol "©" stands for concatenation operation.

We can clearly see that Equation (1) concatenates p_i's position, p_i^k's position, the relative positional between p_i and p_i^k, and the Euclidean distance between p_i and p_i^k together to represent the spatial encoding of p_i with regard to its neighbor point p_i^k. The positional encoding l_i^k is an augmented feature, which facilitates AFGL-Net learning complex geometric details from building façades.

(2) Local Direction Encoding

Because normal vectors of windows and doors' edges and frames have an obvious discrepancy with façade points from flat regions, we implement the local direction encoding to represent the structures of the façade windows and doors more effectively. Inspired by the feature PFH [40,41], we first retrieve K nearest neighbor façade points of current point p_i using KNN algorithm. After this, we carry out the directional encoding of p_i with regard to its neighbor point p_i^k.

More specifically, in Figure 3, given a current point p_i and its normal n_i, as well as p_i^k and its normal n_i^k, we define a local U-V-W coordinate system centered at p_i subject to restrictions in Equation (2) [40].

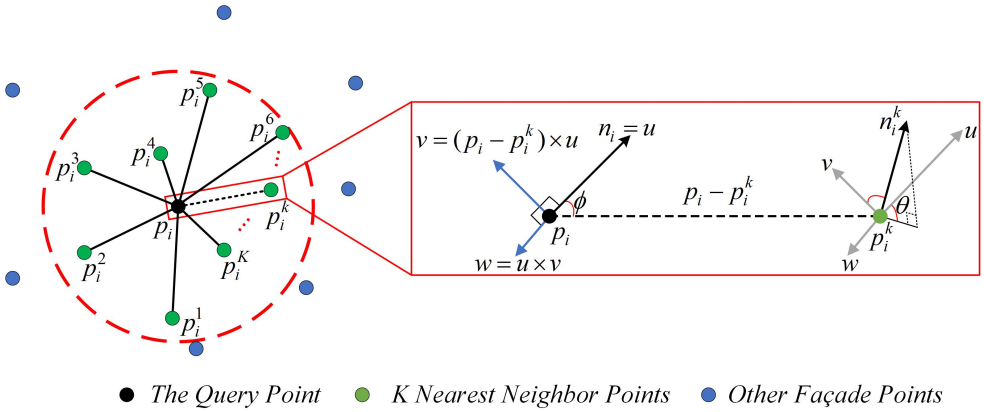

Figure 3. The position and normal relationships between center point p_i and its neighbor point p_i^k.

$$\begin{cases} u = n_i^k \\ v = u \times \frac{(p_i - p_i^k)}{\|p_i - p_i^k\|_2} \\ w = u \times v \end{cases} \tag{2}$$

where $\left\|p_i - p_i^k\right\|_2$ denotes the Euclidean distance between two points with "$\times$" being the symbol of cross product. Based on the predefined local U-V-W coordinate system, the discrepancies between n_i and n_i^k can be represented by three angles, i.e., α, θ and ϕ, which are defined in Equation (3) [40]:

$$\begin{cases} \alpha = v \cdot n_i^k \\ \phi = u \cdot \frac{(p_i - p_i^k)}{\|p_i - p_i^k\|_2} \\ \theta = \arctan\left(w \cdot n_i^k, u \cdot n_i^k\right) \end{cases} \tag{3}$$

where the symbol "$\cdot$" is a dot product, and U-V-W is defined in Equation (2). It should be noted that the current three angles have capability to fully represent the geometric relationships between p_i and p_i^k, which generally requires 12 parameters, i.e., two pairs of positions and two pairs of normal vectors. The three angles fully consider various interactions between n_i and n_i^k, thereby enhancing the representations of façade shapes.

Based on the above analysis, the directional encoding p_i with regard to its neighbor point p_i^k is given in Equation (4):

$$d_i^k = (\alpha_i^k \copyright \phi_i^k \copyright \theta_i^k) \tag{4}$$

Although we obtain the inspirations of directional encoding from the construction of the hand-crafted PFH feature [41], we only consider the interactions between the current point p_i and its neighbor point set, deliberately ignoring the interactions between neighbor points. In this way, we make the direction encoding more reasonable and computationally efficient.

(3) Local Feature Aggregation

After obtaining the local positional encoding l_i^k and directional encoding d_i^k of p_i and p_i^k, they are concatenated with the corresponding intermediate feature f_i^k to obtain the fused feature $\hat{f}_i^k$ as below:

$$\hat{f}_i^k = f_i^k \copyright MLP(l_i^k \copyright d_i^k) \tag{5}$$

After this, we further sum the K weighted neighbor features of p_i using attentive pooling strategy [23] to aggregate the useful information. The final aggregated feature of p_i is given below:

$$\bar{f}_i = MLP(\sum_{k=1}^{K} (\delta(\hat{f}_i^k) \cdot \hat{f}_i^k)) \tag{6}$$

where $\bar{f}_i$ denotes the local aggregated feature of p_i. δ is the softmax function, and the symbol "$\cdot$" denotes the dot product.

Inspired by the idea of dilated residual block [23], in practice, we stack two LSEs with a skip link concatenation to increase the receptive field of each point, as shown in Figure 4.

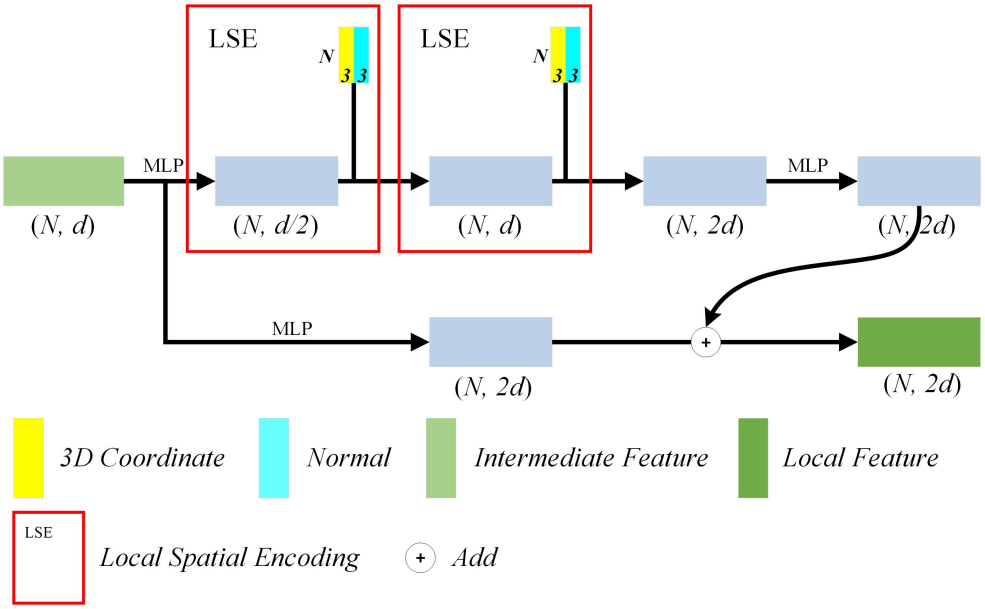

Figure 4. Residual concatenation of local spatial encoding.

2.1.3. Relationship with Prior Works

Autoencoder in AFGL-Net is inspired from RandLA-Net [23] and PFH [40]. On the contrary to RandLA-Net's decoder where the upsampled point set is directly fed into the shared MLP, we replace MLPs with LSEs in the multi-level decoding process, as evident in the red dashed box in Figure 1. In this way, the receptive field of the upsampled points is increased accompanied by an increase in the depth of the encoder-decoder network. Unlike RandLA-Net which has local position encoding, AFGL-Net has both LPE (Local Position Encoding) and LDE (Local Direction Encoding) modules. Therefore, it can capture geometric irregularity of building façades and enhance contour feature representation of windows and doors. We borrow the concept of the hand-crafted feature detector PFH during the encoding of LDE. On the contrary to PFH, AFGL-Net only considers the interactions between the current point and its K nearest neighbor point set, intentionally omitting the direction interactions between neighbor points, thereby improving the encoding efficiency.

2.2. Global Transformer

Transformer adopts the mechanism of self-attention and differentially weights the significance of each part of the input point features. To obtain the global contextual features between windows and doors, we introduce the Transformer [35] (see the green dashed box in Figure 1) into AFGL-Net to describe p_i at the global level. The process is as follows: we feed each point p_i into three independent fully connected layers of the Transformer to

acquire pointwise feature vectors q_i, k_i and v_i, which are input into Equation (7) to calculate the global features of p_i.

$$\tilde{f}_i = MLP(\sum_{k=1}^{K} \delta(\gamma(q_i - k_i + l_i^k)) \cdot \sum_{k=1}^{K} (v_i + l_i^k)) \tag{7}$$

where γ denotes the two consecutive fully connected layers. δ denotes the normalized function softmax, and the symbol "$\cdot$" represents dot product. l_i^k denotes the local position encoding for p_i defined in Equation (1). Note that we use l_i^k as the positional encoding in Equation (7), instead of coordinate difference as originally defined by the Transformer. We further encode p_i through two consecutive Transformer layers according to Equation (7), and then combine the output with the corresponding intermediate features by residual connection to obtain the final global features $\tilde{f}_i$. By stacking multiple Transformer encoding modules, the receptive field of the façade points can be progressively increased, therefore the façade components' position and their arrangements can be accurately perceived. Section 3.5 proves the effectiveness of the residual connection strategy and the connection of several Transformer layers in tandem.

2.3. Attention Mechanism

The attentive mechanism can fuse local features and global features to obtain more discriminate features of building façades. The fusion is illustrated in the blue dashed box in Figure 1 and the process is as follows:

- Local and global feature generation: Given the input point set (N, d_{in}), we can learn local geometric features (N, d_{local}) from autoencoder (see Section 2.1.1) and the global features (N, d_{global}) (see Section 2.2) from two consecutive Transformers through a residual connection.
- Attention matrix construction: The local and global features (N, d_{local}) and (N, d_{global}) are respectively mapped onto the size $(N, 64)$ by the shared MLP. After this, these two outputs are summed up to obtain the attention matrix, followed by the normalization using softmax function.
- Feature fusion via attention mechanism: We implement dot product between the normalized attention matrix and local feature matrix (N, d_{local}) to compute attention scores. Afterwards, the output vector is nonlinearly mapped to obtain the fused feature with attention scores. The whole fusion process is defined as below:

$$f_i = MLP(\delta(\tilde{f}_i + \bar{f}_i) \cdot \bar{f}_i)) \tag{8}$$

  where f_i denotes the output of the attention feature. $\bar{f}_i$ and $\tilde{f}_i$ are local and global features. p_i is the 3D façade point, δ is the normalized function softmax, and the symbol "$\cdot$" denotes the dot product.

3. Performance Evaluation

In this section, we use the annotated dataset of Dublin [36] and RueMonge2014 [37] to evaluate the performance of façade parsing using the proposed AFGL-Net. The labeling accuracy, effectiveness, and robustness of AFGL-Net framework are fully verified by various experiments.

3.1. Dataset Description

(1) The annotated Dublin façade point clouds

The Dublin urban dataset, acquired in 2015 by Laefer et al. [42], contains 41 air flights (see the orange and green flights in Figure 5) covering an area of about 5.6 km^2. The air platform carries the TopEye System S/N 443 LiDAR sensor and the Phase One camera. The flight altitude is about 300 m and the mean density of the point clouds is around 250 to 348 points/m^2. Due to the low flying height, Dublin dataset captures abundant points

of building façades. In 2019, Zolanvarid et al. [36] select some regions (see red boxes in Figure 5) having higher density, larger image coverage, and abundant rooftop shapes for semantic point cloud labeling. These annotated point clouds are expected to be fed into the deep learning network to train the convolutional neural networks. To further explore Dublin benchmark and train our the neural network sufficiently, we manually annotate semantic labeling for another B_2 block, as shown in Figure 5, which extends the scope of the areas annotated by Zolanvarid et al. [36]. These annotated Dublin points are used to train and predict semantic labels of the point clouds by the proposed AFGL-Net. Note that we have removed the roof point cloud and hence, only remaining ∼36 million façade points are used in our experiments to facilitate the recognition of façades, windows, doors, and other components. The statistics of the component class from B_1 to B_7 is listed in Table 1. It is clearly shown that the class ratio of "Window & Door" is far lower than the class ratio of the wall, causing a typically imbalanced parsing problem when there is an unequal distribution of classes in the training dataset. The imbalance parsing problem poses a great challenge to the proposed AFGL-Net.

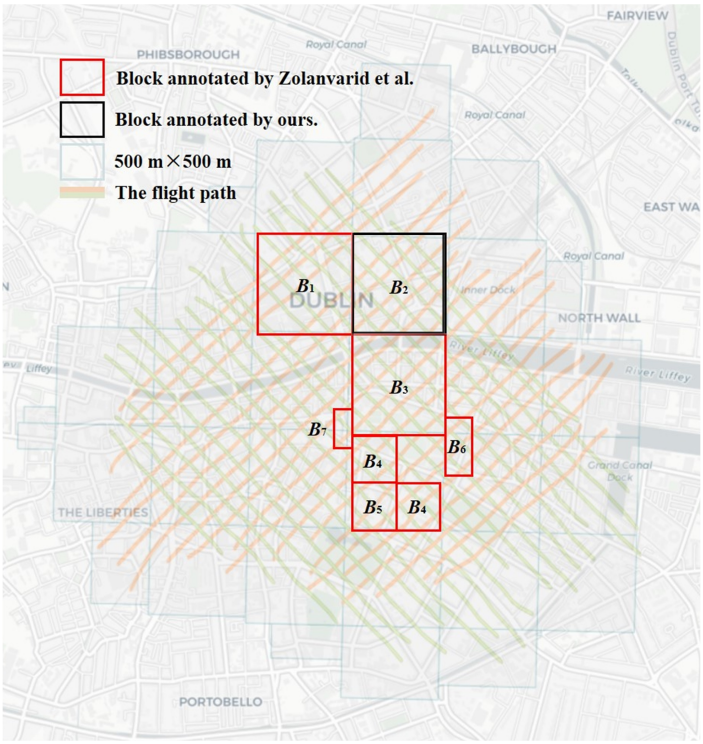

Figure 5. The distribution of the annotated dataset in Dublin urban area. The point clouds within the red boxes are annotated by Zolanvarid et al. [36], while the point clouds within the black box are manually labeled by us. The orange and green lines are the 41 air flights.

Table 1. Class proportions of Dublin façade dataset.

Building Façade	Window & Door	Wall
#Pts	5,375,755	30,749,496
Percentage	14.88%	85.12%

Note that "Percentage" is calculated by dividing the number of points per component to the number of all annotated façade points.

To facilitate the following training and testing, we divide the annotated façade point clouds into seven blocks, i.e., B_1~B_7 shown in Figure 5. The statistics for the components of the façade from B_1 to B_7 is given in Figure 6.

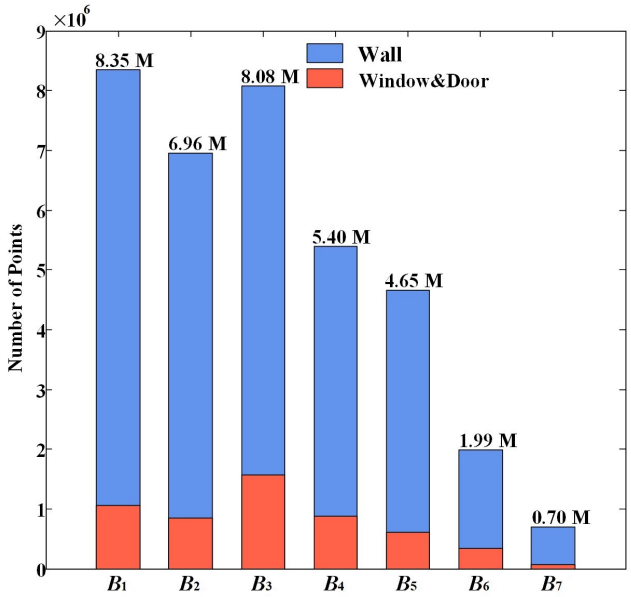

Figure 6. The statistics for the components from block B_1 to B_7.

(2) The annotated RueMonge2014 façade points

RueMonge2014 [37] dataset covers 700 meters along Rue Monge street in Paris. Using 428 high-resolution multi-view images of building façades, the geometric shapes of building façades represented by the triangular meshes are obtained using SFM/MVS algorithms. The dense 3D point clouds of the façades are also simultaneously generated by employing the geometric relationships between the images and the triangular meshes, Riemenschneider et al. [37] select the most suitable images for semantic labeling, and then transfer the semantic labels to the corresponding 3D point clouds. The annotated point clouds include seven classes: wall, shop, door, window, balcony, roof, and sky. Each point has 10-dimensional features, i.e., position, color, normal and relative height. The annotated point clouds of this dataset are divided into two equal-sized parts for training and testing the tasks such as mesh labeling and point cloud labeling. The statistics of the training and testing points for each class are listed in Figure 7.

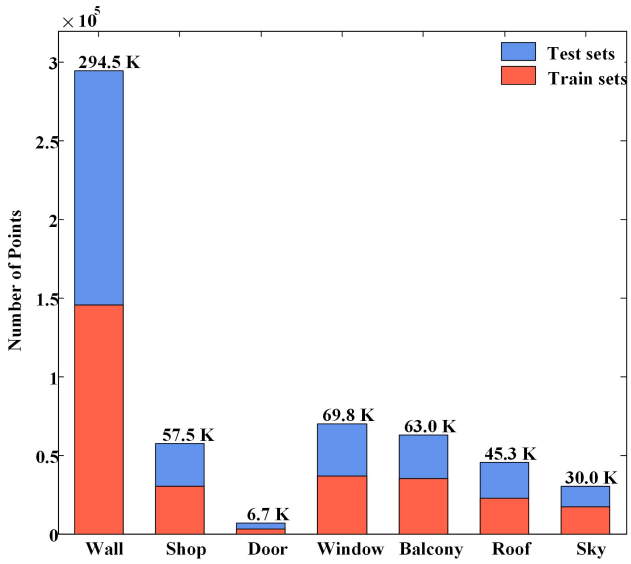

Figure 7. The statistics of the training and testing points for each class.

3.2. Evaluation Metrics

In this section, the performance of AFGL-Net is evaluated by three evaluation metrics: IoU, mIoU, and OA. The definitions of metrics mIoU and OA are shown in Equations (9) and (10). In these equations, k is the number of classes in the dataset, and p_{ii} denotes the number of ground truth point clouds with class label i that are correctly predicted into the same class i; Parameters p_{ij} and p_{ji} denotes the number of points that have class label i/j but incorrectly predicted as class j/i. Using the intersection and union between the prediction set and ground truth, IoU denotes the ratio of intersection to the union, while mIoU denotes the mean IoU scores over all classes. OA denotes the overall accuracy of all classes.

$$mIoU = \sum_{i=1}^{k} IoU_i \tag{9}$$

$$s.t. \quad IoU_i = \frac{p_{ii}}{\sum_{j=1}^{k} p_{ij} + \sum_{j=1}^{k} p_{ji} - p_{ii}}$$

$$OA = \frac{\sum_{i=1}^{k} p_{ii}}{\sum_{i=1}^{k} \sum_{j=1}^{k} p_{ij}} \tag{10}$$

3.3. Hyperparameter Setting

To fully exploit the potentials of the proposed AFGL-Net, we need to find the optimal hyperparameters. We mainly consider the following 6 hyperparameters: the number of façade point N, the number of adjacent points K of each point p_i, the number of autoencoder layers A and the corresponding feature dimension B, the number of Transformer layers C and the corresponding feature dimension D.

For Dublin dataset, we give 4 options, i.e., Schemes 1 to 4 for each hyperparameter (Table 2). We change only one hyperparameter at a time, with the rest retaining baseline setting in Scheme 2. In this way, we can obtain 24 hyperparameter combinations (Figure 8) based on 6 hyperparameters. It should be noted that for obtaining an optimal hyperparameter combination, blocks B_1~B_3 are used as the testing data, and the rest of the blocks are set as the training data. Figure 8 shows the performance comparisons between the baseline setting, i.e., Scheme 2 and other 18 hyperparameter combinations.

Similarly, for searching an optimal hyperparameter combination for RueMonge2014, we set 3 optional values for each hyperparameter (see Table 3), thereby obtaining 18 different combinations (Figure 9). To compare the the performance of these combinations with baseline, i.e., Scheme 2 in Table 3, we use the default training and testing dataset in RueMonge2014 [37]. The mIoU differences of AFGL-Net between the baseline setting and other hyperparamter combinations for these two datasets are listed in Figures 8 and 9.

Table 2. The hyperparameter combinations for Dublin dataset.

Hyperparameters	Scheme 1	Scheme 2 (Baseline Parameter)	Scheme 3	Scheme 4
N	4096	8192	16,384	32,768
K	8	16	24	32
A	3	4	5	6
B	(8-32-64-128)	(16-64-128-256)	(32-128-256-512)	(64-256-512-1024)
C	1	2	3	4
D	(4-16)	(8-32)	(16-64)	(32-128)

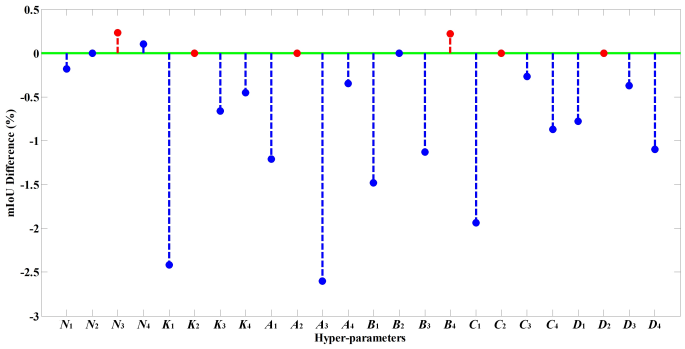

Figure 8. The mIoU differences between the baseline setting and other 18 combinations of hyperparameters for Dublin dataset. Note that the point above/below the green line represents the mIoU values of the corresponding hyperparameter setting is higher/lower than the mIoU obtained under the baseline parameter setting.

Table 3. The hyperparameter combinations for RueMonge2014 dataset.

Hyperparameters	Scheme 1	Scheme 2 (Baseline Parameter)	Scheme 3
N	1024	2048	4096
K	8	16	24
A	3	4	5
B	(8-32-64-128)	(16-64-128-256)	(32-128-256-512)
C	1	2	3
D	(4-16)	(8-32)	(16-64)

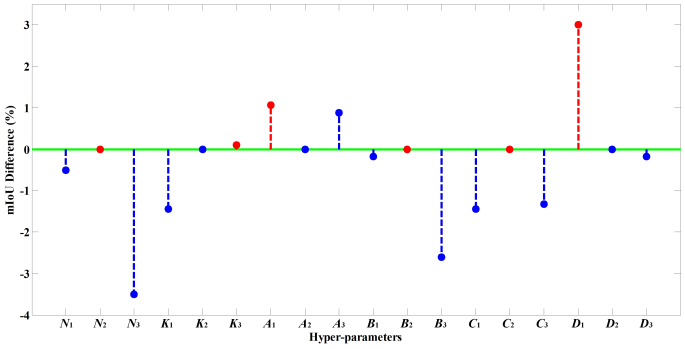

Figure 9. The mIoU differences between the baseline setting and other 12 combinations of hyperparameters for RueMonge2014 dataset. Note that the point above/below the green line represents the mIoU values of the corresponding hyperparameter setting is higher/lower than the mIoU obtained under the baseline parameter setting.

To fully take into account a tradeoff between accuracy and the memory consumption of GPU, in Dublin dataset we set the hyperparameters of AFGL-Net as: N_3 = 16,384, K_2 = 16, A_2 = 4, B_2 = (16-64-128-256), C_2 = 2, D_2 = (8-32). For RueMonge2014 dataset, the hyperparameters are set as: N_2 = 2048, K_3 = 24, A_1 = 3, B_2 = (16-64-128), C_2 = 2, D_1 = (4-16). In the training process, AFGL-Net's training epoch is set to 100 for both Dublin and RueMonge2014 datasets. The learning rate is set to 0.01. The batch_size is set to 4 and the batch_number is set to 500, therefore the number of point blocks per epoch is set to 2000 ($batch\_size \times batch\_number$). Each block in Dublin dataset has 16,384 points, while the number is 2048 for RueMonge2014 dataset. The proposed AFGL-Net was trained and tested on deep learning server with a 3.00 GHz Intel Core i9-10980XE CPU, 64GB main memory and Intel Nvidia 24G GPU. More precisely, we implement the AFGL-Net based on Tensorflow 1.15 framework in Python language and Nvidia GTX 3090 24GB GPU powered by CUDA 11.1.

3.4. Training Size Determination

To strike a balance between the semantic labeling precision and the efficiency of the proposed AFGL-Net, we strive to learn the geometric shapes of façades features effectively by using the least training data. To this end, we progressively increase the number of the training datasets to train multiple versions of AFGL-Net. By testing AFGL-Net on the same dataset, we can evaluate the semantic labeling accuracy of these versions and determine the most appropriate volume of training data.

Training datasets taking from 12.88% to 80.74% of the entire annotated Dublin dataset are selected to verify how much training volume do we really need. The statistics is shown in Table 4. Based on this table, we draw a red polyline shown in Figure 10, which reflects the relationship between the training volume and corresponding mIoU performance. We can see that when the training volume increases, mIoU has basically an increasing trend. However, we should be aware that there exists an elbow point where the training set takes up to 35.27% and achieves 68.44% mIoU. After that, the polyline is saturated and mIoU only has a small increase, i.e., less than 0.3%, which implies that the training volume of 35.27% can make a good balance between the façade parsing and the training efficiency.

Table 4. The network performance on Dublin B_2 block under 6 scales of training set.

Training Set	Window & Door IoU (%)	Wall IoU (%)	mIoU (%)	OA (%)
B5 (12.88%)	32.37	80.39	56.38	82.08
B1 (23.11%)	42.34	88.25	65.29	89.19
B4 ∼B7 (35.27%)	**47.46**	**89.42**	**68.44**	**90.30**
B3∼B7 (57.63%)	48.49	88.86	68.67	89.92
B1; B3∼B4; B6∼B7 (67.86%)	48.57	88.91	68.74	89.96
B1; B3∼B7 (80.74%)	48.55	88.88	68.72	89.94

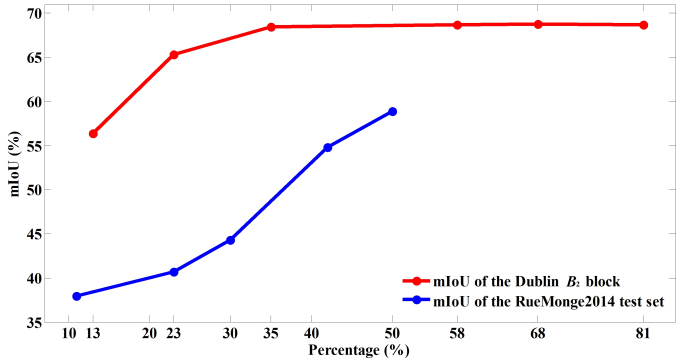

Figure 10. The mIoU performance under different sizes of training set. The red polyline denotes the mIoU performance regarding Dublin B_2 block. The blue polyline denotes the mIoU performance of the RueMonge2014 test set.

For the RueMonge2014 dataset, the original size of the training set account for 50% of the total. To obtain the best volume for training, we use five scales of volume of training data taking up to 11%, 22%, 30%, 42%, and 50% of RueMonge2014 to train AFGL-Net. Meanwhile, the rest 50% of RueMonge2014 dataset is used as the test set. The relation between semantic accuracy and the scale of the training data is illustrated by the blue polyline in Figure 10. It can be found that with an increase of the training set size, the semantic accuracy is monotonically increasing and mIoU reaches the maximum value of 59.80% when the training set account for the maximum of 50% of RueMonge2014 dataset. The trend of a blue polyline indicates that it does not reach a saturated condition soon. This is probably because AFGL-Net is not sufficiently trained even though all the training dataset of RueMonge2014 is used. As the largest training set account for 50% of RueMonge2014, we still use this volume of training dataset to train our network and carry out robustness experiment (see Section 3.7).

3.5. Ablation Study

We verify the effectiveness of the modules embedded in the proposed AFGL-Net through six group ablation experiments: (a) solely using the proposed autoencoder to verify the effect of the local encoding, (b) replacing LSE with the shared MLP in the decoder to verify the effect of the LSE in the decoder, (c) removing LDE from the proposed autoencoder to demonstrate the effect of the LDE, (d) only using GTA to prove its effectiveness for learning global features from the contaminated façade point clouds, (e) retaining an autoencoder and a GTA, and fusing their derived local and global features through simple feature concatenation, and (f) adopting a strategy of AFF, instead of directly fusing features. Note that the configuration in the group (f) is the proposed AFGL-Net. During the ablation study, we used blocks B_1~B_3 as the testing set, and the rest of the annotated blocks are used as the training set. The hyperparameters are chosen based on the strategy in Section 3.3.

From the comparison results in Figure 11a–d, we can see that the autoencoder can learn the local geometric features of the façade more accurately, while GTA focuses on the global features and contextual semantic components. In Table 5, IoU of "Window & Door" and mIoU in group (a) are increased by 19.63% and 14.13% compared with group (d), which proves the significant role of the autoencoder. The mIoU in groups (b) and (c) are smaller than that of the group (a), which demonstrates the effectiveness of LDE and LSE embedding in the decoder. Essentially, LSE has the capability to recognize façade's doors and windows, thereby resulting in the segmented façade components with more regular geometry, as evident in the blue ovals in Figure 11a,b. LDE has the capability to recognize the contours of the windows and doors, thereby making the descriptions of boundaries and frames of the windows and doors accurate, as demonstrated by the purple ovals in Figure 11a–c.

However, it should be noted that some windows and doors in façade are not prominent (see green ovals in Figure 11) because of the complexity of façade point clouds, i.e., corrupted by noise, outliers, missing data, and irregularities. In this case, autoencoder based on the local features is extremely hard to recognize these weak components. Fortunately, GTA can recognize these weak components to a certain extent by context inference and global learning façade components' position and arrangement. However, GTA often results in high commission error, i.e., mistakenly classify wall points into windows and/or doors. Fortunately, AFF can balance weights between local and global attention through the attention mechanism to suppress the side effects of the commission error produced by GTA. By comparing the parsing results between Figure 11e,f, it is obvious that after AFF feature aggregation, façade parsing results are more homogeneous with only a fewer commission errors.

Table 5. Different groups of ablation experiment results. Note that the autoencoder module is shown in Figure 1 denoted by the red dash line. LSE is the local spatial encoding module. GTA denotes the global Transformer awareness module, and AFF represents the attention feature fusion module.

	Configurations	**Window & Door IoU (%)**	**Wall IoU (%)**	**mIoU (%)**	**OA (%)**
(a)	Autoencoder	45.85	86.12	65.98	87.58
(b)	Autoencoder (without LSE in decoder)	43.76	84.47	64.12	86.15
(c)	Autoencoder (without LDE in LSE)	44.31	86.48	65.40	87.79
(d)	GTA	26.22	77.48	51.85	79.15
(e)	Autoencoder + GTA	46.17	85.59	65.88	87.17
(f)	Autoencoder + GTA + AFF	46.80	87.12	66.96	88.43

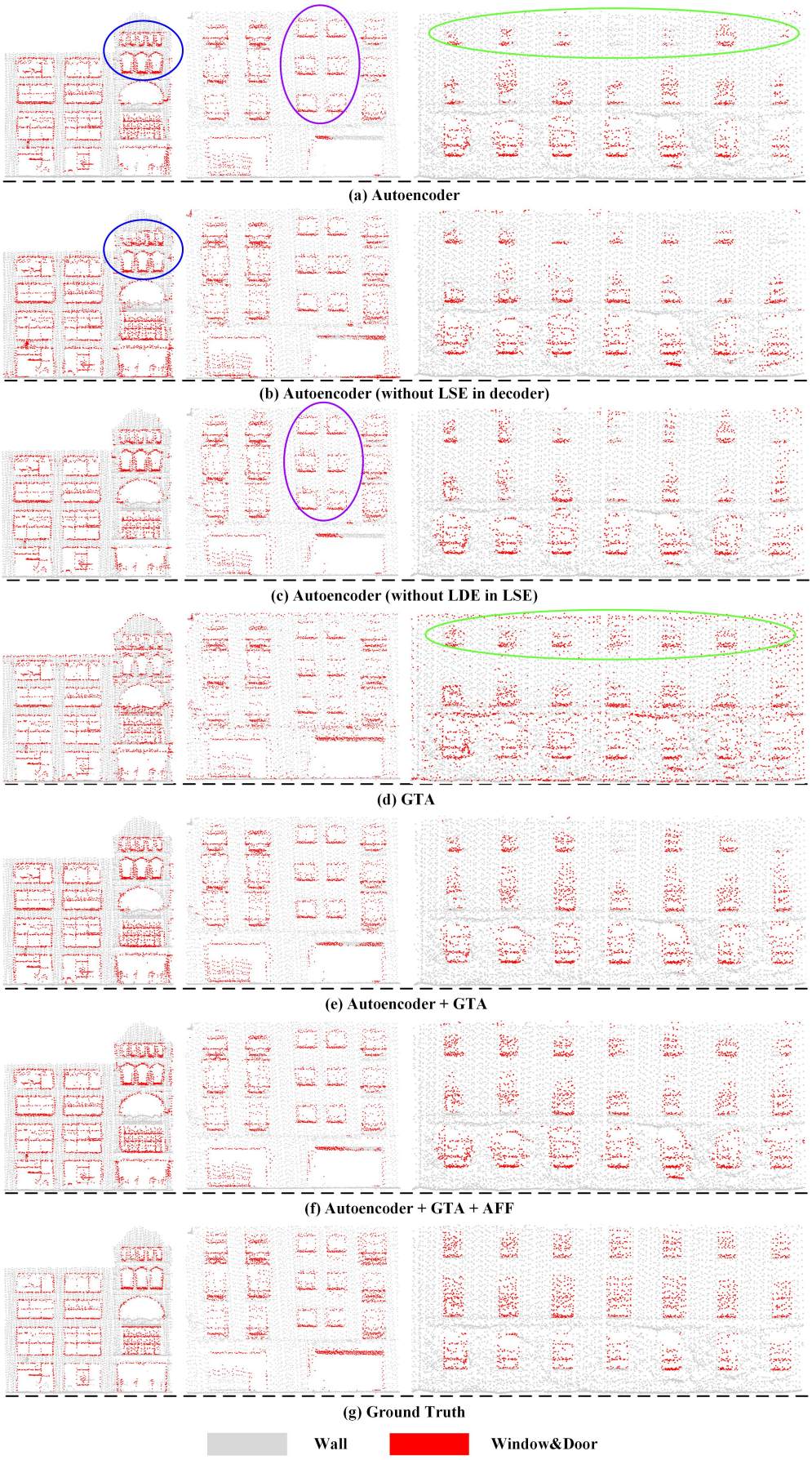

Figure 11. The comparisons of ablation experiments.

3.6. Comparison

To evaluate the performance of the proposed AFGL-Net on Dublin dataset, we compare AFGL-Net with the state-of-the-art methods, i.e., pointwise MLP methods (PointNet [15], PointNet++ [16] and RandLA-Net [23]), graph convolution method (DGCNN [30]) and point convolution (KPConv [28]). We select annotated blocks B_4~B_7 as training data, and the remaining blocks B_1~B_3 as testing data. The hyperparameters are determined according to the principle under Section 3.3. The comparisons of semantic façade parsing results are shown in Table 6, as well as Figures 12 and 13.

It should be noted that PointNet, PointNet++, and DGCNN are limited to process extremely small-sized point clouds, i.e., 1×1 m block with around 4 k points. However, too many small block partition will destroy the completeness of the windows and doors, thereby weakening the feature learning of the building façades. Meanwhile, as the building density and building height are varying within the Dublin dataset, the number of the point clouds within 1×1 m block varies significantly. These two flaws will directly weaken the predicted results of PointNet, PointNet++, and DGCNN. To make the comparison unbiased, we use the KNN algorithm to generate blocks as inputs of PointNet, PointNet++, and DGCNN to avoid any pre-processing steps such as block partition.

Table 6. The comparison results of Dublin dataset.

Methods	Window & Door IoU (%)	Wall IoU (%)	mIoU (%)	OA (%)	Training Time (min)	Testing Time (min)
PointNet	20.14	78.20	49.17	79.34	86.75	2.23
PointNet++	33.85	79.76	56.81	81.66	231.72	5.35
RandLA-Net	43.64	84.32	63.98	86.02	161.67	3.93
DGCNN	28.28	79.41	53.85	80.97	170.90	3.83
KPConv	42.79	**88.86**	65.83	**89.72**	194.44	6.02
Ours	**47.06**	86.97	**67.02**	88.32	253.95	4.42

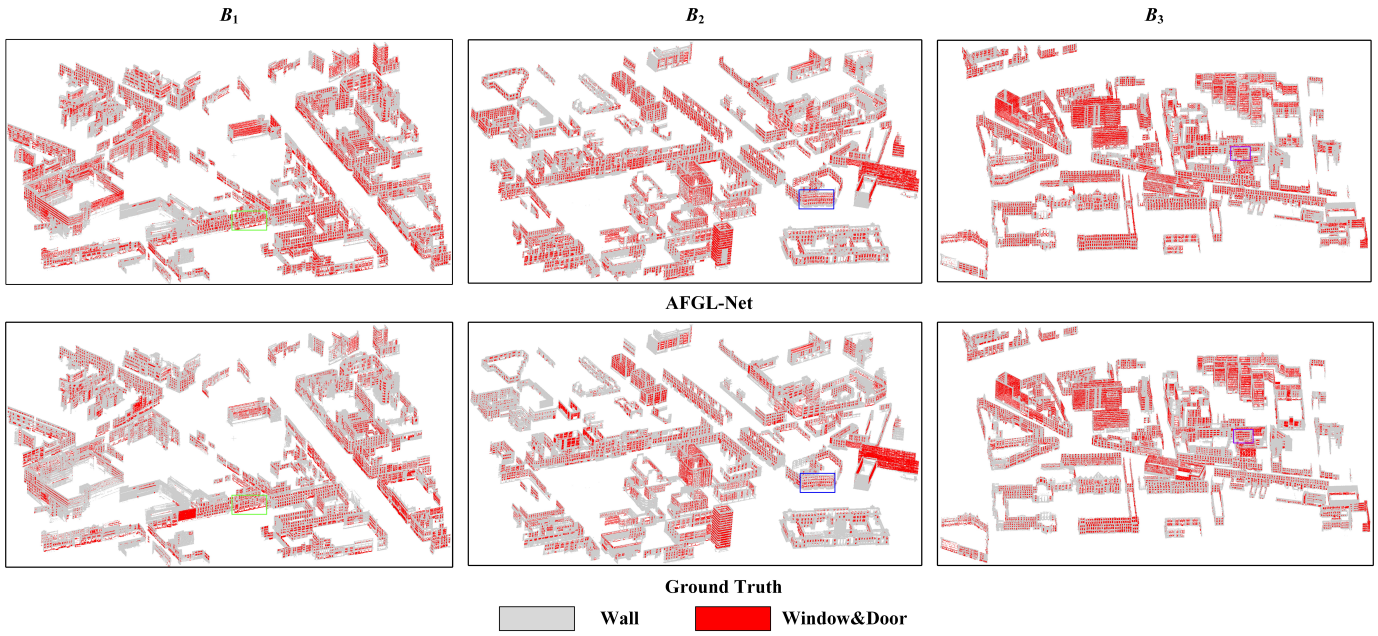

Figure 12. The semantic labeling results for B_1~B_3 using the proposed AFGL-Net. Our results are given in the top line, while the ground truths are provided at the bottom line. The enlarged views for the rectangles denoted by the green, blue and the purple are provided in Figure 13.

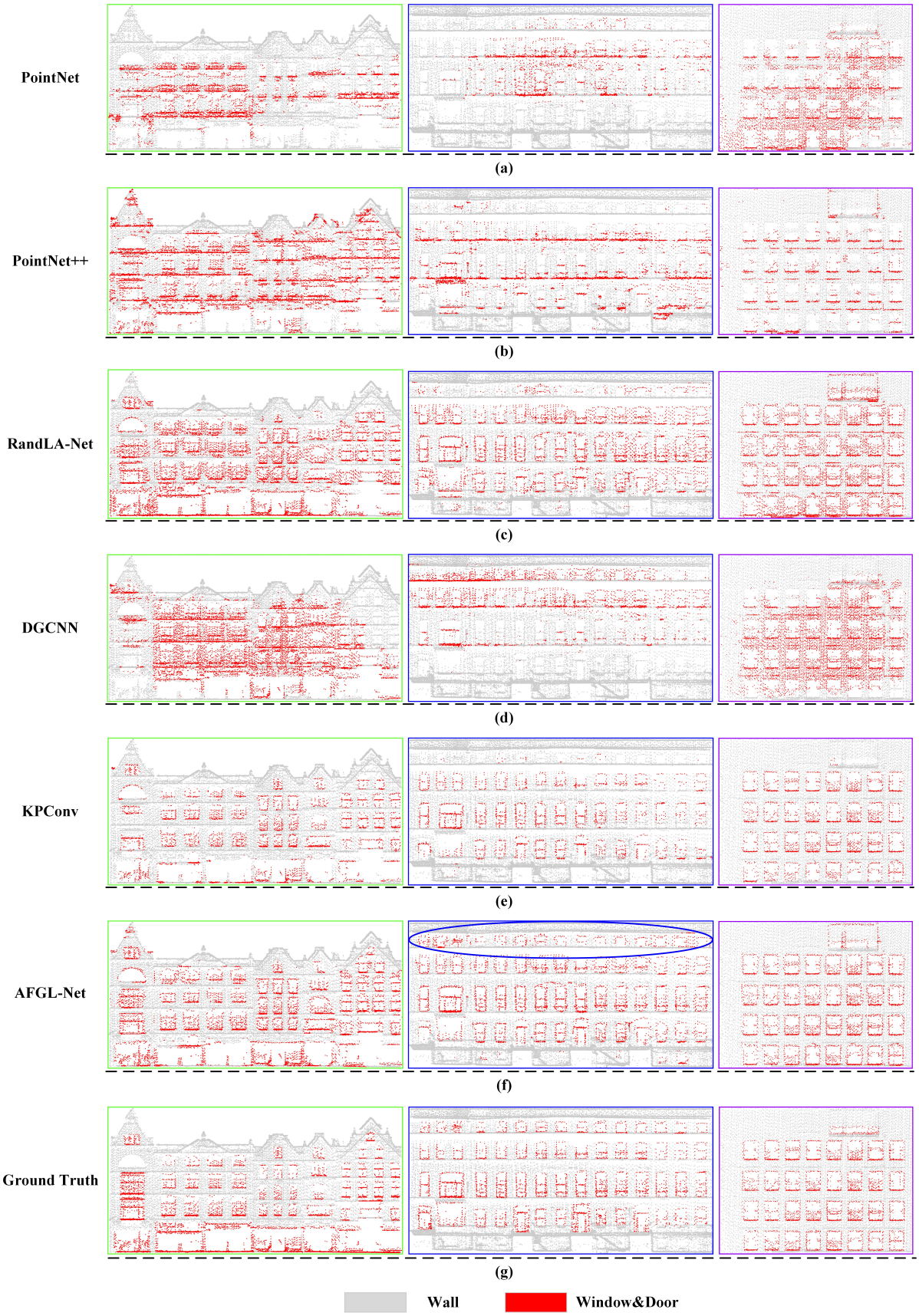

Figure 13. The comparisons with the state-of-the-art methods based on highlighted regions in three rectangles in Figure 12. Subfigures from (**a**–**f**) represent the zoomed results of the corresponding algorithms. Subfigure (**g**) is the corresponding ground truth.

Based on the quantitative results in Table 6 and the qualitative results in Figures 12 and 13, the comparison results for building façade parsing of Dublin dataset are as follows:

Both PointNet and DGCNN tend to incorrectly predict large-sized local areas such as doors, windows, and/or building façades (see Figure 13a–d, which leads to severe commission and omission semantic segmentation error. This is probably because neither of them adopts an autoencoder which can effectively enhance the receptive field of individual façade point, thereby weakening the ability of feature encoding. As the PointNet++ has the ability to capture local context by introducing a hierarchical feature leaning autoencoder, it can capture the rough positions of the most façade components such as windows and doors. However, it is unable to precisely detect façade component positions as evident in Figure 13b.

RandLA-Net, KPConv, and AFGL-Net have an obvious advantage, having mIoU from 63.98% to 67.02% and OA from 86.02% to 89.72% than other methods, which substantiate the ability of local feature encoding by these three methods. More specifically, AFGL-Net's IoU for "Window & Door" is improved by 3.42% and 4.27% than that of RandLA-Net and KPConv (see Table 6) demonstrating the superiority of the proposed AFGL-Net. RandLA-Net tends to mistakenly predict wall points as the window/door points, thereby leading to large commission semantic errors, as demonstrated in Figure 13b. In contrast, KPConv tends to predict the window and door points as a wall class, thereby resulting in large omission semantic errors, as evident in Figure 13e. AFGL-Net can make a good tradeoff between commission and omission semantic errors through introducing powerful LSE encoding and GTA feature representations at the global scale. Because of this, AFGL-Net can better perceive the geometric shapes of façades and the obtained semantic parsing results are approaching the ground truth as close as possible.

It is worth noting that our AFGL-Net can effectively recognize a series of small unnoticeable windows with varying density of point clouds (see regions denoted by the blue ovals in Figure 13f). This shows that the global Transformer awareness module can perceive the geometric structures of façades and capture the global contextual features between windows. In contrast, KPConv and RandLA-Net lack the ability to correctly detect these small windows due to a lack of global feature perceptron. Although our result is competing, it extremely requires 253.95 mins to train the model using training point clouds from regions B_4 to B_7. The high computational complexity occurs because of posing sophisticated LSE module to depict complex building façades and introducing the Transformer module to describe the global features of building façades.

3.7. Robustness

To evaluate the robustness and generalization of our AFGL-Net, we conduct another comparison experiment using the RueMonge2014 dataset. The used training and testing sets are shown in Figure 7. The optimal hyperparameters are obtained through the strategy in Section 3.3. Table 7 shows the quantitative comparison between AFGL-Net and other state-of-the-art methods including pointwise MLP methods, such as PointNet [15], PointNet++ [16] and RandLA-Net [23], graph convolution method, such as DGCNN [30] and point convolution method such as KPConv [28]. Figure 14 shows the qualitative results of AFGL-Net, and Figure 15 shows the enlarged details of each method.

Table 7. The comparison between AFGL-Net and other state-of-the-art methods.

Methods	Wall IoU (%)	Sky IoU (%)	Balcony IoU (%)	Window IoU (%)	Door IoU (%)	Shop IoU (%)	Roof IoU (%)	mIoU (%)	OA (%)	Training Time (min)	Testing Time (min)
PointNet	48.09	45.39	24.41	24.12	12.63	47.52	33.62	33.68	55.33	49.90	0.33
PointNet++	73.47	63.42	51.44	45.49	16.19	57.94	54.48	51.78	76.81	153.02	0.88
RandLA-Net	74.81	**70.67**	**54.15**	48.64	21.57	59.29	60.89	55.72	78.68	69.05	1.31
DGCNN	67.45	64.71	38.89	38.32	18.53	57.60	58.81	49.19	72.19	170.20	0.40
KPConv	74.41	62.74	39.26	40.44	6.34	43.95	58.96	46.59	75.69	225.45	0.97
Ours	**77.28**	70.07	54.03	**53.31**	**37.01**	**64.08**	**62.80**	**59.80**	**80.71**	77.12	1.50

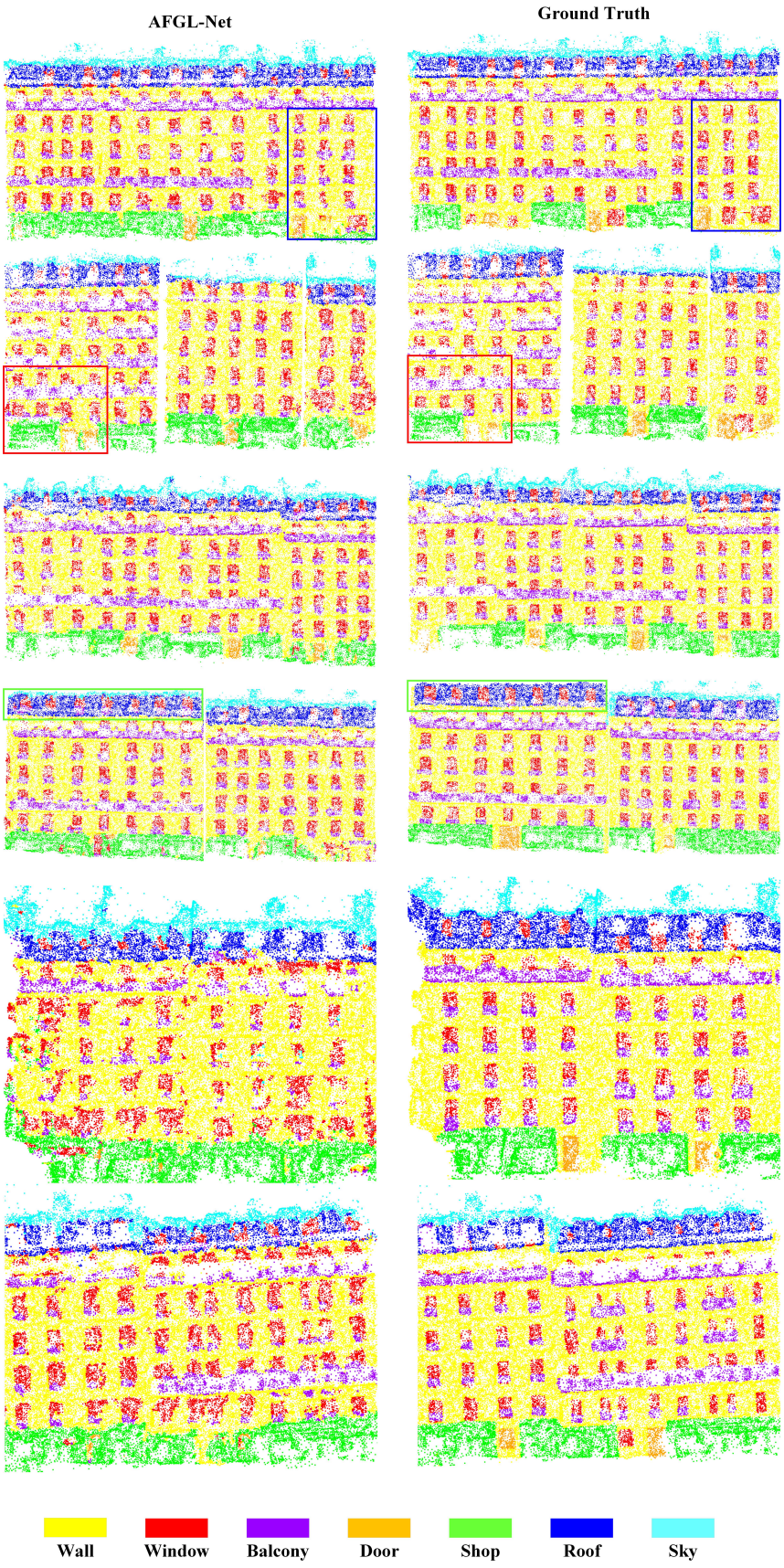

Figure 14. Part of semantic results of RueMonge2014 using the proposed AFGL-Net. The enlarged views of the rectangles are given in Figure 15.

Figure 15. The comparisons with the state-of-the-art methods based on three highlighted regions in rectangles in Figure 14.

In Table 7, it can be obviously found that the semantic parsing accuracy for PointNet++, RandLA-Net, and AFGL-Net is superior to others. The good performance of PointNet++ is due to the dense and homogeneous point clouds of RueMonge2014. Based on these photogrammetric point clouds, PointNet++ can capture geometric details of building façades along Rue Monge street. However, it should be noted that PointNet++ only achieves 51.78% mIoU and 76.81% OA in the RueMonge2014 dataset, which are lower than 56.81% mIoU and 81.66% OA in Dublin dataset. This is possible because of an insufficient training dataset, as demonstrated in Figure 10. The competing performance regarding RandLA-Net and AFGL-Net is probably because they not only have strong local spatial encoding but adopt an attentive pooling strategy to aggregate features of nearest neighbor points with different attention weights, thereby making the local features more distinguishable. It is worth noting that our AFGL-Net can sufficiently learn topologies between façade components and infer the contextual semantic, which is beneficial to counter the potentially detrimental impact of class imbalance (see the class door in Figure 7). Fortunately, our method can cope with this situation. As can be seen from Table 7, the IoU of the class door for AFGL-Net is the maximum of 37.01%, which is much higher than the second largest value of 21.75% IoU for RandLA-Net.

For PointNet, DGCNN, and KPConv, the semantic accuracy regarding classes such as windows, doors, and balconies is inferior to other counterparts. The poor performance of PointNet and DGCNN can be attributed to the insufficient consideration of relevance between façade components, thereby weakening contextual feature representation. KPConv's feature encoding and representation capabilities are significantly subject to insufficient training data in RueMonge2014 and similarity of the geometric shapes of building façades.

4. Conclusions and Suggestions for Future Research

We present a deep learning framework, namely AFGL-Net for small-sized façade component parsing. Unlike other mainstream deep learning frameworks that learn semantic information from the relatively balanced classes, the AFGL-Net can effectively recognize the imbalanced classes such as windows and doors from the large-scale Dublin benchmark and photogrammetric point clouds, e.g., RueMonge2014. More precisely, in the Dublin dataset, the proposed AFGL-Net achieves 47.06% IoU for window/door class, which improves by 3.42% and 4.27% window/door IoU compared with RandLA-Net and KPConv frameworks. For predicted window/door in RueMonge2014 dataset, we achieve 55.31% and 37.01% IoU for windows and doors, increasing by 4.67% and 15.44% compared with RandLA-Net. The superiority of the proposed AFGL-Net is probably because of adopting local geometric features and global contextual features. These two kinds of features are provided as input into the attention-based feature fusion mechanism to obtain more informative features for façade parsing from contaminated point clouds with varying point density and a certain degree of missing data.

Although we obtain a high parsing accuracy for building façade, it should be noted that the proposed framework tends to be encumbered as it is loosely combined by KNN searching, autoencoder based on a U-Net backbone with 5-logical layers, global Transformer, and feature fusion based on an attention mechanism. The complexity of the composition makes the framework inflexible and computationally inefficient. To further refine this paper, in future work, we plan to explore light-weighted autoencoding and feature aggregation methods to make a tradeoff between façade parsing accuracy and efficiency of large-scale point cloud processing. In addition, we notice that there is a lack of public annotated datasets for small façade component parsing. Although some parts of the Dublin dataset have been annotated, it has some flaws, such as homogenous geometry shapes, low level of labeling granularity, and incorrect labeling. RueMonge2014 [37], as an existing photogrammetric annotated façade point set, has a relatively small size, thereby causing insufficient learning. In future work, we plan to expand the existing 0.75 km^2 annotated Dublin points to the whole dataset covering 5.6 km^2. Meanwhile, we will implement fine-grained labelings, such as labeling of different façade planes, recognition of other possible

façade components (balcony, air conditioning host), the distinction of rooftop and façade windows, and maintenance of topology relations between components and their attached façade/rooftop planes. To alleviate the training using insufficient photogrammetric points, we plan to acquire and label large-scale urban building façades with varying shapes using the DJI (https://www.dji.com/cn, accessed on 7 December 2021) Matrice 300 RTK and DJI Zenmuse P1 photogrammetry camera.

Author Contributions: D.C. and G.X. analyzed the data and wrote Python source code. L.Z. and D.C. helped in project and study design, paper writing, and data analysis. J.P., F.H. and J.L. helped in proofreading, data acquisition, and experiment comparison. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Science Fund for Distinguished Young Scholars under Grant 41925006, in part by the National Natural Science Foundation of China under Grant 41971415, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20201387, and in part by the Open Fund of State Key Laboratory of Remote Sensing Science under Grant OFSLRSS202010. This work was performed while the first author Dr. Dong Chen acted as an awardee of the 2021 Qinglan Project, sponsored by Jiangsu Province, China.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The annotated Dublin benchmark is released on the University of Dublin website (https://v-sense.scss.tcd.ie/DublinCity/, accessed on 7 December 2021). RueMonge2014 annotated point dataset is publicly available from the website (https://people.ee.ethz.ch/~rhayko/paper/eccv2014_riemenschneider_multiviewsemseg/, accessed on 7 December 2021) of the Computer Vision Laboratory of ETH Zürich.

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

References

1. Abou Diakité, A.; Zlatanova, S. Valid space description in BIM for 3D indoor navigation. *Int. J. 3-D Inf. Model. (IJ3DIM)* **2016**, *5*, 1–17. [CrossRef]
2. Chen, M.; Lin, H.; Liu, D.; Zhang, H.; Yue, S. An object-oriented data model built for blind navigation in outdoor space. *Appl. Geogr.* **2015**, *60*, 84–94. [CrossRef]
3. Reinhart, C.F.; Davila, C.C. Urban building energy modeling–A review of a nascent field. *Build. Environ.* **2016**, *97*, 196–202. [CrossRef]
4. Mao, B.; Ban, Y.; Harrie, L. A multiple representation data structure for dynamic visualisation of generalised 3D city models. *ISPRS J. Photogramm. Remote Sens.* **2011**, *66*, 198–208. [CrossRef]
5. Verdie, Y.; Lafarge, F.; Alliez, P. LOD generation for urban scenes. *ACM Trans. Graph.* **2015**, *34*, 30. [CrossRef]
6. Wang, R.; Peethambaran, J.; Chen, D. Lidar point clouds to 3-D urban models: A review. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *11*, 606–627. [CrossRef]
7. Du, J.; Chen, D.; Wang, R.; Peethambaran, J.; Mathiopoulos, P.T.; Xie, L.; Yun, T. A novel framework for 2.5-D building contouring from large-scale residential scenes. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 4121–4145. [CrossRef]
8. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 945–953.
9. Lawin, F.J.; Danelljan, M.; Tosteberg, P.; Bhat, G.; Khan, F.S.; Felsberg, M. Deep projective 3D semantic segmentation. In Proceedings of the International Conference on Computer Analysis of Images and Patterns, Ystad, Sweden, 22–24 August 2017; pp. 95–107.
10. Boulch, A.; Le Saux, B.; Audebert, N. Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks. In Proceedings of the Workshop on 3D Object Retrieval, Lyon, France, 23–24 April 2017; Volume 3.
11. Maturana, D.; Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 922–928.
12. Graham, B.; Engelcke, M.; Van Der Maaten, L. 3d semantic segmentation with submanifold sparse convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9224–9232.
13. Riegler, G.; Osman Ulusoy, A.; Geiger, A. Octnet: Learning deep 3d representations at high resolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3577–3586.

14. Klokov, R.; Lempitsky, V. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In Proceedings of the IEEE International Conference on Computer Vision, Honolulu, HI, USA, 21–26 July 2017; pp. 863–872.
15. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
16. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv* **2017**, arXiv:1706.02413.
17. Jiang, M.; Wu, Y.; Zhao, T.; Zhao, Z.; Lu, C. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv* **2018**, arXiv:1807.00652.
18. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
19. Zhao, H.; Jiang, L.; Fu, C.W.; Jia, J. Pointweb: Enhancing local neighborhood features for point cloud processing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5565–5573.
20. Li, J.; Chen, B.M.; Lee, G.H. So-net: Self-organizing network for point cloud analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9397–9406.
21. Chiang, H.Y.; Lin, Y.L.; Liu, Y.C.; Hsu, W.H. A unified point-based framework for 3d segmentation. In Proceedings of the 2019 International Conference on 3D Vision (3DV), Quebec City, QC, Canada, 16–19 September 2019; pp. 155–163.
22. Geng, X.; Ji, S.; Lu, M.; Zhao, L. Multi-Scale Attentive Aggregation for LiDAR Point Cloud Segmentation. *Remote Sens.* **2021**, *13*, 691. [CrossRef]
23. Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, N.; Markham, A. Randla-net: Efficient semantic segmentation of large-scale point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11108–11117.
24. Hua, B.S.; Tran, M.K.; Yeung, S.K. Pointwise convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 984–993.
25. Tatarchenko, M.; Park, J.; Koltun, V.; Zhou, Q.Y. Tangent convolutions for dense prediction in 3d. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3887–3896.
26. Zhang, Z.; Hua, B.S.; Yeung, S.K. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 1607–1616.
27. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. Pointcnn: Convolution on x-transformed points. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 820–830.
28. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. Kpconv: Flexible and deformable convolution for point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 6411–6420.
29. Komarichev, A.; Zhong, Z.; Hua, J. A-cnn: Annularly convolutional neural networks on point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7421–7430.
30. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. *Dynamic Graph Cnn for Learning on Point Clouds*; ACM: New York, NY, USA, 2019; Volume 38, pp. 1–12.
31. Te, G.; Hu, W.; Zheng, A.; Guo, Z. Rgcnn: Regularized graph cnn for point cloud segmentation. In Proceedings of the 26th ACM International Conference on Multimedia, Seoul, Korea, 22–26 October 2018; pp. 746–754.
32. Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; Shan, J. Graph attention convolution for point cloud semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 10296–10305.
33. Landrieu, L.; Simonovsky, M. Large-scale point cloud semantic segmentation with superpoint graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4558–4567.
34. Zhang, S.; Tong, H.; Xu, J.; Maciejewski, R. Graph convolutional networks: A comprehensive review. *Comput. Soc. Netw.* **2019**, *6*, 1–23. [CrossRef]
35. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
36. Zolanvari, S.; Ruano, S.; Rana, A.; Cummins, A.; da Silva, R.E.; Rahbar, M.; Smolic, A. DublinCity: Annotated LiDAR point cloud and its applications. *arXiv* **2019**, arXiv:1909.03613.
37. Riemenschneider, H.; Bódis-Szomorú, A.; Weissenberg, J.; Van Gool, L. Learning where to classify in multi-view semantic segmentation. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 516–532.
38. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241.
39. Fix, E.; Hodges, J.L. Discriminatory analysis. Nonparametric discrimination: Consistency properties. *Int. Stat. Rev. Int. De Stat.* **1989**, *57*, 238–247. [CrossRef]

40. Rusu, R.B.; Blodow, N.; Marton, Z.C.; Beetz, M. Aligning point cloud views using persistent feature histograms. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 3384–3391.
41. Rusu, R.B.; Blodow, N.; Beetz, M. Fast point feature histograms (FPFH) for 3D registration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 3212–3217.
42. Laefer, D.F.; Abuwarda, S.; Vo, A.V.; Truong-Hong, L.; Gharibi, H. 2*015 Aerial Laser and Photogrammetry Survey of Dublin City Collection Record*; Technical Report; New York University: New York, NY, USA, 2017.