MeshNet-SP: A Semantic Urban 3D Mesh Segmentation Network with Sparse Prior

Guangyun Zhang and Rongting Zhang *

School of Geomatics Science and Technology, Nanjing Tech University, Nanjing 211800, China; zgy@njtech.edu.cn
* Correspondence: zrt@njtech.edu.cn

Abstract: A textured urban 3D mesh is an important part of 3D real scene technology. Semantically segmenting an urban 3D mesh is a key task in the photogrammetry and remote sensing field. However, due to the irregular structure of a 3D mesh and redundant texture information, it is a challenging issue to obtain high and robust semantic segmentation results for an urban 3D mesh. To address this issue, we propose a semantic urban 3D mesh segmentation network (MeshNet) with sparse prior (SP), named MeshNet-SP. MeshNet-SP consists of a differentiable sparse coding (DSC) subnetwork and a semantic feature extraction (SFE) subnetwork. The DSC subnetwork learns low-intrinsic-dimensional features from raw texture information, which increases the effectiveness and robustness of semantic urban 3D mesh segmentation. The SFE subnetwork produces high-level semantic features from the combination of features containing the geometric features of a mesh and the low-intrinsic-dimensional features of texture information. The proposed method is evaluated on the SUM dataset. The results of ablation experiments demonstrate that the low-intrinsic-dimensional feature is the key to achieving high and robust semantic segmentation results. The comparison results show that the proposed method can achieve competitive accuracies, and the maximum increase can reach 34.5%, 35.4%, and 31.8% in mR, mF1, and mIoU, respectively.

Keywords: 3D real scene; urban 3D mesh; semantic segmentation; sparse prior; low intrinsic dimension; convolutional neural network

1. Introduction

A textured urban 3D mesh, created mostly by a dense image match of oblique aerial images, is one of the final user products in the photogrammetry and remote sensing (PRS) community, and has been widely applied in city management [1], urban and rural planning [2], heritage protection [3], building damage assessment [4], estimation of the potential achievable solar energy of buildings [5], and so forth. Although 3D meshes have advantages in visualization over other 3D data (such as point clouds and voxels) [6], it is hard to use 3D meshes to conduct complex spatial analysis as they lack semantic information [7].

The semantic segmentation of 3D data (such as 3D point clouds and 3D meshes) is a central task in PRS. With the introduction of PointNet [8], a large number of deep learning methods for directly consuming unordered point clouds have emerged [9,10]. For example, in order to make a network provide high representativeness and remarkable robustness, Ma et al. [11] proposed an end-to-end feature extraction framework for 3D point-cloud segmentation by using dynamic point-wise convolutional operations in multiple scales. Lai et al. [12] presented a stratified transformer for 3D point-cloud segmentation. The stratified transformer addressed the issue that current methods failed to directly model long-range dependencies. Chibane et al. [13] provided a weakly supervised 3D semantic instance segmentation method (named Box2Mask) using bounding boxes. The core of the Box2Mask involves a deep model that directly votes for bounding box parameters, and a clustering method specifically tailored to bounding box votes. However, there is...
comparably limited research on the semantic segmentation of urban 3D meshes in the PRS field. The complexity and irregular geometric structure in 3D meshes make it challenging to perform convolution operations directly on them [6]. In the early days, the Markov random field-based random forest was used to perform semantic segmentation of 3D meshes, which regarded handcrafted features as inputs [14]. The handcrafted features included geometric features (elevation, planarity, and verticality) and photometric features (average color, standard deviation, and color distribution in the HSV color space). To the best of our knowledge, [14] was the first study that combined geometric and photometric features for the semantic segmentation of 3D meshes in the PRS community. With the development of deep learning technology, deep-learning-based methods for the semantic segmentation of 3D meshes were proposed. According to the type of input data, these methods can be grouped into three classes: center of gravity point (CoGP)-based, voxel-based, mesh-based, and view-based methods. CoGP-based methods [6,15–19] use the center of gravity (CoG) per facet to denote the facet and generate CoG point clouds. CoG usually contains the geometric and photometric features of the corresponding facet. CoG point clouds are then used as input of 1D CNN or the state-of-the-art point-cloud semantic segmentation network, such as PointNet++ [20], KPConv [21], etc. Voxel-based methods [22–26] convert the irregular 3D meshes into regular 3D grids, i.e., 3D voxels, and then 3D CNNs are applied to these 3D voxels. Different from CoGP-based and voxel-based methods, mesh-based methods [27–35] directly perform convolution on 3D meshes, using the topological information of vertices/edges/facets within 3D meshes. In view-based methods [36–38], different virtual views of the 3D mesh were used to render multiple 2D channels for training an effective 2D semantic segmentation model. The per-view predictions generated features that were fused on 3D mesh vertices to predict mesh semantic segmentation labels.

Among most of the methods mentioned above, texture information is one of the most important inputs, which has a significant effect on improving the accuracy of semantic 3D mesh segmentation [7,14,16,17,39]. Although the above methods emphasize the importance of texture information, they do not take into account the sparse characteristics of texture information. Researchers have demonstrated that nature image data show a low-dimensional structure despite the high dimensionality of traditional pixel representations, and the discriminative information of image data has sparse characteristics [40,41]. Moreover, it has been proven that deep networks can learn more easily from low-intrinsic-dimensional datasets, and the learned models generalize better from training to test data [40]. However, the presence of noise in image data can lead to an increase in its intrinsic dimensionality. This can pose challenges for training deep networks and result in decreased performance. The added noise introduces additional variability and complexity, making it harder for the network to extract meaningful features and patterns, thereby hindering the learning process and negatively impacting the network’s performance.

Sparse coding is one of the methods to acquire low-intrinsic-dimensional data from raw data [42,43]. The idea behind sparse coding is to find a concise and compact representation (i.e., sparse representation) of the raw data by using a small number of basis functions or features. The goal is to capture the essential information and discard the redundant or irrelevant components, thereby reducing the dimensionality of the data. Thus, sparse representation is a form of data that is typically low dimensional, capable of effectively representing raw images regardless of the presence of noise in the raw images. Classical sparse coding has been widely used in signal and image restoration/denoising tasks because of the ability to learn interpretable low-intrinsic-dimensional representations and the strong theoretical support [44]. Although deep-learning-based image restoration/denoising methods have surpassed the performance of classical sparse coding methods in modern image datasets (such as ImageNet and CIFAR), deep learning networks are still “black boxes” that are not clearly understood. Thus, the research on integrated sparse coding and deep learning networks has attracted significant attention [45–47] due to their complementary advantages, which is one of the prominent research directions for constructing end-to-end deep network integrating image denoising and high-level tasks in CV field. However, to
the best of our knowledge, the research on end-to-end deep learning networks integrating sparse prior for semantic 3D mesh segmentation is blank in the PRS field.

In this paper, a deep learning architecture is proposed for the semantic segmentation of urban 3D meshes. The contributions of our work can be summarized as follows.

1. Considering the importance of texture images for the semantic segmentation of urban 3D meshes, we propose a differentiable sparse coding (DSC) subnetwork to obtain low-intrinsic-dimensional features from texture images by using the unrolled optimization algorithm. Moreover, we propose a semantic feature extraction (SFE) subnetwork to extract high-level features that are used to predict a label for each facet.

2. We propose an end-to-end deep learning architecture (named MeshNet-SP) that integrates the DSC subnetwork and the SFE subnetwork to perform semantic segmentation of urban 3D meshes. And an end-to-end training strategy is also proposed.

3. Comprehensive experiments are conducted to demonstrate that the proposed end-to-end deep learning architecture can achieve competitive results in semantic segmentation of urban 3D meshes despite the presence of noise in the texture images.

2. Method

In this section, we describe the proposed end-to-end architecture for semantic urban 3D mesh segmentation. We evaluate the proposed architecture in Section 3, as well as ablated models where the differentiable sparse coding module is introduced or not. We assess the performance of the proposed model on SUM (a benchmark dataset of semantic urban meshes) [48], showing the competitive results of the proposed method.

The architecture of the proposed MeshNet-SP is shown in Figure 1, which combines jointly differentiable sparse coding (DSC) modules and semantic feature extraction (SFE) modules. The proposed MeshNet-SP takes both geometrical information (such as coordinates of CoGs and normals of facets) and texture information of urban 3D mesh as input and outputs a label for each facet within an urban 3D mesh. The DSC module exploits sparse prior and a differential structure using MLP. This allows our model to extract low-intrinsic-dimensional features from raw texture images for improving semantic segmentation accuracy. The SFE module extracts high-level semantic features of urban 3D mesh by constructing graph structure from urban 3D mesh and performing convolution operations on the constructed graph.

![Figure 1. The architecture of the proposed MeshNet-SP.](image)

We base the DSC module on an optimization algorithm $\Lambda$ that solves the problem of getting sparse representations from raw texture images. Usually, the obtained sparse repre-
sentations are low-intrinsic-dimensional data [42,43]. We express the joint representation and perception problem as a bi-level optimization problem

$$\min_{D, \omega} \mathcal{L}(\langle \Lambda(y, D), g \rangle, c, \omega) \quad s.t. \quad \Lambda(y, D) = \arg \min_z \mathcal{G}(z, y, D)$$

(1)

where, here, $\Lambda$ minimizes a sparse representation problem $\mathcal{G}$. The outputs of this DSC module are sparse codes $\Lambda(y, D)$ of raw texture information, which are concatenated with geometrical information $g$ of the urban 3D mesh and input into later SFE modules, and associated semantic segmentation loss $\mathcal{L}$, which is used to calculate the loss between predict labels and ground-truth labels $c$. Here, the model parameters $\omega$ of semantic urban 3D mesh segmentation are absorbed in $\mathcal{L}$ as an argument.

For the nested objective $\mathcal{G}$, we follow the sparse coding model as the architecture backbone. The sparse coding model aims to discover a latent sparse representation $z \in \mathcal{R}^d$ that can be utilized to reconstruct an input $y \in \mathcal{R}^d$ by using a learned decoder $D$. We address the sparse representation problem by using an unrolled iterative optimization algorithm. To achieve this, we parameterize the encoder and decoder with unknown, learned parameters, and truncate the iterations to yield the operator $\Lambda$.

Any differentiable semantic 3D mesh segmentation method can be utilized in the proposed stack. In this paper, we develop a semantic 3D mesh segmentation network by stacking several semantic feature extraction modules. The semantic segmentation loss is a standard cross-entropy loss.

2.1. Differentiable Sparse Coding Module

2.1.1. Sparse Coding System with Variance Regularization

Usually, when both the sparse code $z$ and the dictionary $D$ are unknown, $z$ and $D$ can be obtained simultaneously by alternately performing the sparse coding algorithm and the dictionary learning algorithm.

Traditionally, sparse coding algorithms utilize an $l_1$ sparsity penalty and a linear decoder $D \supset \mathcal{R}^d \times 1$ to conduct inference and identify a latent sparse representation $z \supset \mathcal{R}^d$ of a given texture information $y \supset \mathcal{R}^d$. This representation is founded by minimizing the energy function given below:

$$\mathcal{G}(z, y, D) = f(z) + g(z) = \frac{1}{2} \|y - zD\|_2^2 + \lambda \|z\|_1.$$  

(2)

In (2), the term $f(z) = \frac{1}{2} \|y - zD\|_2^2$ shows the reconstruction error for the raw texture information by utilizing the sparse codes $z$ and dictionary $D$. The term $g(z) = \lambda \|z\|_1$ is a regularization term. It penalizes the sparse code $z$ using the $l_1$ norm. $\lambda$ controls the sparsity level of sparse code $z$. The larger the $\lambda$, the more sparse the sparse code becomes. In essence, the columns of the dictionary $D$ are selectively linear combined by each sparse code $z$. Finding an optimal sparse code $z^*$ is to discover a solution of optimization problem of (2), i.e.,

$$z^* = \arg \min_z \mathcal{G}(z, y, D) = \arg \min_z f(z) + g(z)$$

(3)

The dictionary learning algorithm uses gradient-based optimization method to update the elements of dictionary $D$ by minimizing the MSE between the reconstructed texture information $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_N\} \supset \mathcal{R}^d$ and the training raw texture information $Y = \{y_1, y_2, \cdots, y_N\} \supset \mathcal{R}^d$ from the sparse codes $Z = \{z_1, z_2, \cdots, z_N\} \supset \mathcal{R}^d$ obtained from the sparse coding algorithm, i.e., (4),

$$\mathcal{L}_D(D, Z^*, Y) = \arg \min_D \frac{1}{N} \sum_{i=1}^{N} \|y_i - Dz_i^*\|_2^2$$

(4)

Typically, in order to prevent a collapse in the $l_1$ norm of the codes and successfully train a sparse coding system, it is necessary to regularize the dictionary $D$ by bounding the Euclidean norms of the dictionary’s elements. However, it is hard to conduct normalization.
procedures for sparse coding systems in which the decoder is a non-linear multi-layer neural network. Similar to [49], we apply variance regularization to each latent code component to prevent a collapse in the $l_1$ norm of the codes. To this end, a regularization term $\gamma$ is added to the energy function in (2). The regularization term $\gamma$ ensures that the variance of all latent components across a mini-batch of codes is greater than a pre-set threshold $\sqrt{T}$. Thus, the (2) is rewritten as

$$\tilde{g}(z, y, D) = \hat{f}(z) + \bar{g}(z) = \sum_{i=1}^{n} \frac{1}{2} ||y_i - Dz_i||^2_2 + \sum_{j=1}^{l} \beta \left[ (T - \sqrt{\text{var}(z_j)})_+ \right]^2 + \sum_{i=1}^{n} \lambda ||z_i||_1$$

(5)

where

$$\hat{f}(z) = \sum_{i=1}^{n} \frac{1}{2} ||y_i - Dz_i||^2_2 + \sum_{j=1}^{l} \beta \left[ (T - \sqrt{\text{var}(z_j)})_+ \right]^2,$$

$$\bar{g}(z) = \sum_{i=1}^{n} \lambda ||z_i||_1.$$

The first term of $\hat{f}(z)$ is the sum of reconstruction error for data samples $Y = \{y_1, \ldots, y_n\}$ based on each code $z_i \in \mathbb{R}^l$ within a mini-batch. The second term of $\hat{f}(z)$ is the added regularization term $\gamma$ involving the variance of latent component $z_j \in \mathbb{R}^n$ across the mini-batch. In variance $\text{var}(z_i) = \frac{1}{T} \sum_{i=1}^{n} (z_i - u_i)^2$, $u_i$ is the mean across the $j$-th component.

In this paper, a fast iterative shrinkage threshold algorithm (FISTA) is applied to solve the optimization problem of (6). The details of FISTA can be found in [50].

$$\arg \min \tilde{g}(z, y, D) = \arg \min \hat{f}(z) + \bar{g}(z)$$

(6)

2.1.2. Architecture of Differentiable Sparse Coding Module

The architecture of the DSC module is shown in Figure 2. Given texture information and a fixed decoder $D$, the FISTA algorithm is applied to obtain a sparse code $z^*$ which can best reconstruct the texture information using $D$'s elements. The encoder $\epsilon$ is trained to predict the sparse code $z^*$, which is the output of FISTA. On the other hand, the decoder is trained by minimizing the mean square error (MSE) between the reconstructed texture information and the raw texture information by using the sparse code $z^*$. The details are described as follows.

![Figure 2: The architecture of the DSC module.](image)

Inspired by [51], an architecture of the encoder $\epsilon$ is designed based on the unrolled FISTA shown in Algorithm 1. As shown in Figure 2, the encoder $\epsilon$ has two multi-layer perceptron (MLP) layers $U \in \mathbb{R}^{d \times l}$ and $S \in \mathbb{R}^{l \times l}$, a bias term $b \in \mathbb{R}^l$, and non-linear functions ReLU. The encoder $\epsilon$ is similar to a recurrent neural network, which can be trained by using mini-batch gradient descent to minimize the MSE between FISTA’s output $z^* \in \mathbb{R}^l \times n$ and the encoder’s output $z_\epsilon$ from $Y = \{y_1, y_2, \cdots, y_N\} \subseteq \mathbb{R}^d$, i.e.,

$$\mathcal{L}_\epsilon(\epsilon, Z^*, Y) = \arg \min_\epsilon \frac{1}{N} \sum_{i=1}^{N} ||y_i - z_\epsilon^*||^2_2$$

(7)

In this paper, a non-linear decoder $D$, consisting of two MLP layers, a bias term following the first MLP layer, and a non-linear activation function ReLU (see Figure 2),
is regarded as the dictionary in the sparse coding system. The first MLP layer maps the outputs of FISTA to the hidden representations. And the second MLP layer maps the hidden representations to the reconstructed texture information. The non-linear decoder $D$ is trained using gradient descent to minimize the MSE between raw texture information and the reconstructed texture information (see (4)).

Algorithm 1 Unrolled fast iterative shrinkage threshold algorithm.

**Input:** Texture information $y \in \mathbb{R}^d$; number of iterations $L$; **Parameters:** $U \in \mathbb{R}^{d \times l}$; $S \in \mathbb{R}^{l \times l}$; $b \in \mathbb{R}^l$

**Output:** Sparse code $z_e \in \mathbb{R}^l$

1: $u = Uy + b$
2: $z_0 = ReLU(u)$
3: for each $i \in [1, L]$ do
4: $z_i = ReLU(u + Sz_{i-1})$
5: end for
6: return $z_e = z_L$

2.2. Semantic Feature Extraction Module

The purpose of the cascade of the prior DSC module in our architecture is to obtain low-intrinsic-dimensional data, which are good identifiable features for semantic urban 3D mesh segmentation, from raw texture images. The prior steps from Algorithm 1 must be flexible enough to learn the low-intrinsic-dimensional data from raw texture images but also project on a subset according to the semantic urban 3D mesh segmentation loss $L_s$. The obtained low-intrinsic-dimensional data (i.e., sparse codes of raw texture images) are concatenated with geometrical information $g$ of the urban 3D mesh and input into the later SFE to get high-level semantic features. In this paper, we construct the SFE module using the CoGP-based method. The architecture of the SFE module is shown in Figure 3. Every facet within an urban 3D mesh is represented by a CoGP with features consisting of the corresponding sparse codes of raw texture images, the coordinates of the CoGP, and the normal vector of the facet. In order to learn the local features of the urban 3D mesh, we apply the knn method to build a directed graph per CoGP, and propose an edge convolution method performing on the direction graphs. At last, the outputs of edge convolution are aggregated by using a symmetry function. The details are described as follows.

![Figure 3. The architecture of the SFE module.](image)

Edge Convolution

Given $G = (v, e)$ is a directed graph constructed by the $i$-th CoGP $p_i$ and its $k$ neighbor CoGPs in a min-batch including $n$ CoGPs. $v$ and $e$ are the set of vertexes and edges of the directed graph $G$, respectively. The edge features between the $i$-th CoGP $p_i$ and its neighbors can be learned by a shared MLP. The edge convolution operation can be described mathematically as

$$
\epsilon_{ij} = ReLU(\phi(f_j - f_i) + \psi f_i), \quad j = 1, \cdots, k
$$

(8)
where $\epsilon_{ij}$ is the learned edge feature between the $i$-th CoGP $p_i$ and the $j$-th neighbor; $f_i$ and $f_j$ are the feature vectors of the $i$-th CoGP $p_i$ and the $j$-th neighbor, respectively; $\phi_1, \phi_2, \ldots, \phi_{m_o}$ and $\psi_1, \psi_2, \ldots, \psi_{m_o}$ are learnable parameters; $m_o$ is the output dimension of the shared MLP. In order to aggregate the $k$ edge features between the $i$-th CoGP $p_i$ and its $k$ neighbor CoGPs, we use a symmetric function, i.e., maximum function, to perform the pooling operation. The pooling operation can be described mathematically as

$$f'_i = \max(\epsilon_{ij}), \quad j = 1, \ldots, k$$

where $f'_i$ is the higher-level feature for the $i$-th CoGP.

As shown in Figure 1, the proposed architecture contains three SFE modules. Their outputs are concatenated by a skip connection way, which is useful to solve the vanishing gradient problem. After obtaining the outputs of the last SFE module, the outputs of these three SFE modules are concatenated and then put into a SoftMax layer to predict the label per facet. The cross-entropy loss is regarded as the prediction loss. The object of prediction is to minimize the cross-entropy loss, i.e.,

$$L_C = \arg\min \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{M} s_{ic} \log(p_{ic})$$

where $N$ and $M$ are the number of samples in a mini-batch and classes, respectively; $s_{ic}$ is a sign function, which is 1 when the true label of the $i$-th samples is the same as label $c$, otherwise 0; $p_{ic}$ is the prediction probability of the $i$-th sample belonging to class $c$.

The main term of the MeshNet-SP loss function is the $L_C$ defined in (10), which is complemented by the dictionary learning term $L_D$ (4) and the sparse coding term $L_\epsilon$ (7). The complete loss function can be expressed as

$$L = L_C + L_D + L_\epsilon$$

### 2.3. End-to-End Training Strategy

We propose a training strategy to speed up the training of MeshNet-SP. This strategy involves training the DSC subnetwork and the SFE subnetwork sequentially, followed by end-to-end fine-tuning. After the sequential training, a certain number of epochs of the training of (1) is conducted with initial values set as $\theta_1$ and $\omega_1$. The gradient backpropagation from the SFE subnetwork to the DSC subnetwork is derived by the chain rule of derivatives, which allows the DSC subnetwork to output useful sparse codes for the semantic urban 3D mesh segmentation task. The proposed training strategy is more efficient than directly solving (1) due to the heavy computational loads of the DSC subnetwork and the relatively slow convergence of the SFE subnetwork.

### 3. Experiments

#### 3.1. Dataset

The proposed MeshNet-SP is evaluated on a semantic urban meshes (SUM) benchmark dataset[48]. The SUM dataset encompasses an area of approximately 4 square kilometers in Helsinki, Finland, and consists of six classes: ground, vegetation, building, water, car, and boat. The textured mesh data were derived from oblique aerial images with a ground sample distance of 7.5 cm using ContextCapture software. The SUM dataset comprises 64 tiles. We randomly selected 12 tiles for training and another 12 tiles for testing from the SUM dataset provided by [48], taking into account the computer’s processing capacity. Figure 4 shows the spatial distribution of the selected training dataset (yellow areas) and testing dataset (blue areas). The distribution of each class in the training and validation datasets is illustrated in Figure 5, with a total of 7,479,164 faces (3,787,315 for training and 3,691,849 for testing). Figure 5 presents detailed statistics on class frequencies in both datasets: ground (17.17%, 17.02%), vegetation (22.19%, 23.57%), building (54.21%, 56.63%), water (1.17%, 0.31%), car (2.95%, 2.22%), and boat (2.32%, 0.26%). Figure 5 demonstrates
that there exists a significant imbalance among different classes regarding their relative number of facets; specifically, the proportion of water-related facets only accounts for merely around one percent or even less than that across both datasets while buildings occupy over half of all facets therein, which poses a great challenge to semantic segmentation tasks involving textured meshes with unbalanced classes as such.

![Image of spatial distribution](image_url)

**Figure 4.** The spatial distribution of the selected training dataset (yellow area) and testing dataset (blue area). The training dataset and testing dataset contain 12 tiles, respectively.

![Proportion of each class](image_url)

(a) Proportion of each class in the training datasets

(b) Proportion of each class in the testing datasets

**Figure 5.** Proportion of each class in the training and testing datasets: ground (17.17%, 17.02%), vegetation (22.19%, 23.57%), buildings (54.21%, 56.63%), water (1.17%, 0.31%), cars (2.95%, 2.22%), and boats (2.32%, 0.26%). Unlabeled faces are not considered.
3.2. Experimental Design

We evaluate the proposed MeshNet-SP by setting three category configurations. Table 1 shows the details of different experiments. First, we evaluate our joint DSC subnetwork and SFE subnetwork on the semantic segmentation of urban 3D meshes. We include ablated studies that show the importance of low-intrinsic-dimensional data obtained by the proposed DSC subnetwork to improve the accuracy of semantic urban 3D mesh segmentation. Second, we assess the robustness of the proposed MeshNet-SP for semantic urban 3D mesh segmentation under varying levels of textured image noise. Third, we analyze the influence of sparse code’s dimension on the accuracy of semantic urban 3D mesh segmentation.

<table>
<thead>
<tr>
<th>#</th>
<th>DSC Subnetwork</th>
<th>Textured Image Noise Level</th>
<th>Sparse Code Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>with</td>
<td>without</td>
<td>0.0 $\sigma$</td>
</tr>
<tr>
<td>Category 1</td>
<td>Exps.1</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>Exps.2</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Category 2</td>
<td>Exps.3</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>Exps.4</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>Exps.5</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Exps.6</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Category 3</td>
<td>Exps.7</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>Exps.8</td>
<td>✓</td>
<td>×</td>
</tr>
</tbody>
</table>

* These experiments can be grouped into three categories. Category 1 evaluates the importance of the DSC subnetwork. Category 2 assesses the robustness of MeshNet-SP under varying textured image noise. Category 3 evaluates the influence of sparse code’s dimension on the accuracy of semantic urban 3D mesh segmentation.

3.3. Evaluation Metrics

In this paper, similar to [19,48], over accuracy (OA), Kappa coefficient (Kappa), mean precision (mP), mean recall (mR), mean F1 score (mF1), and mean intersection over union (mIoU) are used to quantitatively evaluate the proposed method. Considering the triangle facets within a mesh have different sizes, we calculate these evaluation indices by the area of triangle facets instead of the number of triangle facets. Mathematically, these evaluation indices can be expressed as

$$OA = \frac{\sum_{i=1}^{c} TP_i}{S}$$

$$Kappa = \frac{OA - pe}{1 - pe}, \quad pe = \frac{\sum_{i=1}^{c} (TP_i + FN_i) \times (TP_i + FP_i)}{S \times S}$$

$$mP = \frac{1}{c} \sum_{i=1}^{c} \frac{TP_i}{TP_i + FP_i}$$

$$mR = \frac{1}{c} \sum_{i=1}^{c} \frac{TP_i}{TP_i + FN_i}$$

$$mF1 = \frac{1}{c} \sum_{i=1}^{c} \frac{2TP_i}{TP_i + FP_i + FN_i}$$

$$mIoU = \frac{1}{c} \sum_{i=1}^{c} \frac{TP_i}{TP_i + FP_i + FN_i}$$
where \( c \) is the number of class; \( S \) is the total area of facets within the mesh; \( TP_i, FP_i, \) and \( FN_i \) are the areas of true positive, false positive, and false negative for the \( i \)-th class, respectively; \( P_i \) and \( R_i \) are the precision and recall for the \( i \)-th class, respectively.

### 3.4. Implementation Details

The proposed MeshNet-SP is implemented by Pytorch [52] on a 64-bit Windows 10 operating system. The MeshNet-SP is trained and tested on a machine equipped with an NVIDIA Quadro RTX 4000 GPU with 8 GB memory, and two 16-core Intel(R) Xeon(R) Gold 5218 CPUs with a 2.3 GHz main frequency and 128 GB RAM.

During the training and testing stages, dropout and learning ratios are empirically set to be 0.6 and 0.001, respectively. Considering the hardware processing capability, the training and testing dataset are split as small patches, and the batch size is set to be six. In the training phase, the first 50 epochs are used for training the DSC subnetwork, followed by 500 epochs for training the SFE subnetwork, and the last 50 epochs are used for end-to-end fine-tuning. The dimension of sparse code is set to be 128. During training, the mean utilization and memory usage of CPU and GPU are (9.0\%, 47.4\%) and (87.9\%, 61.1\%), respectively; and the maximum utilization and memory usage of CPU and GPU are (100.0\%, 78.4\%) and (94.0\%, 63.3\%), respectively. It should be noted that the results of each experimental configuration are based on a single experiment.

### 3.5. Results and Analysis

Table 2 summarizes the results of the ablation experiments with various configurations. From Table 2, we can see that, in each category experiment, the proposed MeshNet-SP can obtain the highest accuracy in all metrics, such as IoU per class, OA, Kappa, mP, mR, mF1, and mIoU. These results demonstrate the effectiveness and robustness of the proposed MeshNet-SP. In particular, the proposed MeshNet-SP can maintain relatively high accuracy in IoU for the difficulty classes (car and boat) by extracting low-intrinsic-dimensional features. Specifically, we describe our findings from these ablation experiments below.

Table 2. Accuracies (%) based on the area of triangle facets. The results reported in this table are IoU of per class, overall accuracy (OA), Kappa coefficient (Kappa), mean precision (mP), mean recall (mR), mean IoU (mIoU), and mean F1 score (mF1). Exps.: experiment, \( i = 1, 2, \ldots, 8 \).

<table>
<thead>
<tr>
<th>Category</th>
<th>#</th>
<th>Ground</th>
<th>Vegetation</th>
<th>Building</th>
<th>Water</th>
<th>Car</th>
<th>Boat</th>
<th>OA</th>
<th>Kappa</th>
<th>mP</th>
<th>mR</th>
<th>mF1</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exps.1</td>
<td>81.04</td>
<td>84.93</td>
<td>91.75</td>
<td>57.84</td>
<td>57.45</td>
<td>39.38</td>
<td>93.17</td>
<td>88.42</td>
<td>79.90</td>
<td>80.58</td>
<td>79.97</td>
<td>68.73</td>
</tr>
<tr>
<td></td>
<td>Exps.2</td>
<td>79.91</td>
<td>80.95</td>
<td>89.56</td>
<td>53.18</td>
<td>53.19</td>
<td>35.43</td>
<td>91.72</td>
<td>85.96</td>
<td>77.73</td>
<td>77.83</td>
<td>77.33</td>
<td>65.37</td>
</tr>
<tr>
<td>Category 2</td>
<td>Exps.3</td>
<td>80.93</td>
<td>84.95</td>
<td>91.71</td>
<td>59.56</td>
<td>58.23</td>
<td>32.88</td>
<td>90.39</td>
<td>83.87</td>
<td>77.97</td>
<td>81.63</td>
<td>79.13</td>
<td>68.04</td>
</tr>
<tr>
<td></td>
<td>Exps.4</td>
<td>79.66</td>
<td>82.29</td>
<td>88.15</td>
<td>59.38</td>
<td>54.52</td>
<td>10.44</td>
<td>90.74</td>
<td>84.79</td>
<td>73.32</td>
<td>81.46</td>
<td>72.78</td>
<td>62.41</td>
</tr>
<tr>
<td></td>
<td>Exps.5</td>
<td>73.67</td>
<td>48.37</td>
<td>74.55</td>
<td>36.27</td>
<td>27.37</td>
<td>10.96</td>
<td>79.85</td>
<td>64.63</td>
<td>64.06</td>
<td>56.28</td>
<td>58.57</td>
<td>45.20</td>
</tr>
<tr>
<td></td>
<td>Exps.6</td>
<td>64.59</td>
<td>36.37</td>
<td>54.73</td>
<td>21.63</td>
<td>7.83</td>
<td>2.64</td>
<td>65.32</td>
<td>45.20</td>
<td>48.87</td>
<td>43.38</td>
<td>42.97</td>
<td>31.30</td>
</tr>
<tr>
<td>Category 3</td>
<td>Exps.7</td>
<td>79.23</td>
<td>81.51</td>
<td>90.00</td>
<td>46.75</td>
<td>50.72</td>
<td>34.21</td>
<td>91.77</td>
<td>86.02</td>
<td>74.82</td>
<td>79.30</td>
<td>75.83</td>
<td>63.74</td>
</tr>
<tr>
<td></td>
<td>Exps.8</td>
<td>78.44</td>
<td>79.13</td>
<td>88.49</td>
<td>46.00</td>
<td>48.84</td>
<td>33.95</td>
<td>90.75</td>
<td>84.55</td>
<td>73.55</td>
<td>77.63</td>
<td>74.92</td>
<td>62.48</td>
</tr>
</tbody>
</table>

Compared to raw image data, the low-intrinsic-dimensional data are more useful to improve the precision of semantic 3D mesh segmentation. The proposed method, which incorporates the DSC subnetwork, is compared to the method without such a subnetwork; see the first and second rows of Table 1. Exps.1 and Exps.2 take both geometric information (coordinates of CoGs and normals of facets) and texture information of the urban 3D mesh as input. In Exps.1, texture information is processed by the DSC subnetwork to produce low-intrinsic-dimensional data, which are then concatenated with geometric information and used to train the SFE subnetwork. Meanwhile, in Exps.2, the SFE subnetwork is trained directly on raw texture information along with geometric information. We observe that
Exps.1 converges faster than Exps.2, which is consistent with the conclusion that deep networks can learn more easily from low-intrinsic-dimensional datasets [40]. We note that our joint network (i.e., Exps.1) obtains higher OA, Kappa, mP, mR, mF1, and mIoU compared to Exps.2, which increase by 1.45%–3.36%. Figures 6 and 7 show the partial visualization results and normalized confusion matrix of Exps.1 and Exps.2, respectively. From Figure 6 and 7, it can be found that Exps.2 produces more prediction errors than Exps.1 does, especially in water class and car class, of which the recall index decreases 11% and 5% compared to Exps.1, respectively. Moreover, according to the normalized confusion matrices shown in Figure 7, it can be found that the water and car classes are mainly misclassified as the ground class, while the boat class is mainly misclassified as the building class. The reason may be that the geometric features of water are similar to the ground’s, i.e., they are flat in the local area. Another main reason is that the number of samples of the water, car, and boat classes is small, their discriminant features are effectively learned hard by the deep-learning-based method. To some extent, the proposed MeshNet-SP decreases the misclassification of these classes. These results validate that the DSC subnetwork producing low-intrinsic-dimensional data can efficiently improve the accuracy of the semantic 3D mesh urban segmentation.

The proposed MeshNet-SP has higher robustness for semantic urban 3D mesh segmentation under varying levels of texture image noise. In order to evaluate the robustness of the proposed MeshNet-SP, four experiments (i.e., Exps.3~Exps.6) are set. In Exps.3 and Exps.4, the DSC subnetwork is applied to get low-intrinsic-dimensional data, while Exps.5 and Exps.6 are without such a subnetwork. The experiments involve the addition of varying levels of noise, such as 1× standard deviation (1σ) and 2× standard deviation (2σ), which follow a Gaussian distribution, to the raw texture images. The results in Table 2 and Figures 8 and 9 validate the effectiveness of our proposed MeshNet-SP to obtain low-intrinsic-dimensional data from raw/noised texture images to improve the accuracy of the semantic 3D mesh segmentation. We can see in Table 2 and Figure 8 that while the accuracy of the models without the DSC subnetwork drastically decreases over the 0σ~2σ noise levels, the accuracy
of our proposed MeshNet-SP remains stable. Specifically, from $0\sigma$ to $2\sigma$, the accuracy of the models without the DSC subnetwork decreases by $\sim30\%$, where the Kappa coefficient decreases by $\sim41\%$. The primary factor contributing to the decline in the accuracy of the model without the DSC subnetwork is the presence of noise in texture image data, which can result in an increase in its intrinsic dimensionality. However, the proposed MeshNet-SP can still effectively obtain low-intrinsic-dimensional data through the utilization of the DSC subnetwork. From the partial visualization results (see Figure 10) and the normalized confusion matrices (see Figure 9), we can see that as noise levels increase, the number of misclassification in the water, car, and boat classes increases dramatically in Exps.5 and Exps.6 due to the absence of the DSC subnetwork. Meanwhile in Exps.3 and Exps.4 with the DSC subnetwork, as noise levels increase, the increase of misclassification of water, car, and boat classes is relatively small. These results validate that the DSC subnetwork producing low-intrinsic-dimensional data obtained can make MeshNet-SP more robust.

![Normalized confusion matrix of Exps.1](a) ![Normalized confusion matrix of Exps.2](b)

**Figure 7.** Normalized confusion matrix of Exps.1 and Exps.2.

![Variation trend of semantic segmentation accuracy along with varying noise levels of the texture images](a) ![Variation trend of semantic segmentation accuracy along with varying noise levels of the texture images](b)

**Figure 8.** Variation trend of semantic segmentation accuracy along with varying noise levels of the texture images.
The increase of the sparse code’s dimension would not always improve the performance of the proposed MeshNet-SP for semantic 3D mesh segmentation. In order to analyze the influence of sparse code’s dimension on the accuracy of semantic urban 3D mesh segmentation, Exps.3, Exps.7, and Exps.8 are conducted, where the dimension of sparse code is 128, 256, and 512, respectively. From Table 2, we observe that the performance of the proposed MeshNet-SP decreases along with the increase in the dimension of the sparse code. From the partial visualization results (see Figure 11) and the normalized confusion matrices (see Figure 12), we observe that the number of difficulty classes (water, car, and boat classes) misclassified as ground are also increased with the increase in the dimension of the sparse code. The main reason for this phenomenon is that the intrinsic dimensionality of the higher-dimensional sparse code may be greater than that of the lower-dimensional sparse code. These results validate again that low-intrinsic-dimensional data can improve the performance of deep learning networks.
Figure 10. Results of the semantic 3D mesh segmentation on the validation dataset with Exps.3 ~ Exps.6.
3.6. Comparison to other Competition Methods

To evaluate the performance of the proposed MeshNet-SP, we compare the proposed MeshNet-SP with seven current state-of-the-art 3D semantic segmentation methods that can process large-scale urban datasets, i.e., Wilk et al. [19], Gao et al. [48], Hu et al. [53], Thomas et al. [21], Landrieu et al. [54], Qi et al. 2017b [20], and Qi et al. 2017a [8]. Specifically, Wilk et al. [19] proposed a hybrid method for semantic urban mesh segmentation. The main idea of the hybrid method was to semantically segment the point clouds sampled from urban mesh and the oblique images, using a fully convolutional neural network [55].
and a pyramid scene parsing network (PSP-Net) [56], respectively; and then map the acquired labels of point clouds and oblique images to mesh. Gao et al. [48] released the urban 3D mesh data used in this paper, and proposed a pipeline to perform semantic 3D mesh segmentation. In the pipeline, they first used region growing to group triangle facets into homogeneous regions, and then extracted 11 types of geometric and radiometric features from those mesh segments. Finally, these geometric and radiometric features were concatenated into a 44-dimensional feature vector, which is used by a random forest (RF) classifier. The rest compared methods (RandLA-Net [53], KPConv [21], SPG [54], PointNet++ [20], and PointNet [8]) belonged to point-based approaches, where points were sampled from the facets and used to generate point clouds.

The results of the comparison with other competition methods are presented in Table 3, which clearly demonstrate the superior performance of the proposed method in this article over all other competing methods. Specifically, the MeshNet-SP method surpasses Gao et al.’s [48] proposed method by a significant margin of 10.0% mR, 6.2% mF1, and 2.5% mIoU. Furthermore, it outperforms the remaining six methods with improvements ranging from 5.5% to 34.5% in terms of mR, from 2.3% to 35.4% in terms of mF1, and from −0.3% to 31.8% in terms of mIoU. Moreover, from Table 3, we can see that boat and car are difficult classes for all kinds of methods because of the small sample size. In particular, the methods of Qi et al. (2017a) [8] and Qi et al. (2017b) [20] almost cannot acquire these two classes. Meanwhile, the proposed MeshNet-SP can acquire the highest IoU for boat and car classes.

Table 3. Accuracy comparisons among different methods. The results reported in this table are IoU of per class, overall accuracy (OA), mean recall (mR), mean F1 score (mF1), and mean IoU (mIoU).

<table>
<thead>
<tr>
<th>#</th>
<th>Ground</th>
<th>Veg.</th>
<th>Building</th>
<th>Water</th>
<th>Car</th>
<th>Boat</th>
<th>OA</th>
<th>mR</th>
<th>mF1</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gao et al. [48]</td>
<td>83.3</td>
<td>90.5</td>
<td>92.5</td>
<td>86.0</td>
<td>37.3</td>
<td>7.4</td>
<td>93.0</td>
<td>70.6</td>
<td>73.8</td>
<td>66.2</td>
</tr>
<tr>
<td>Wilk et al. [19]</td>
<td>83.8</td>
<td>88.1</td>
<td>91.5</td>
<td>85.0</td>
<td>49.9</td>
<td>15.7</td>
<td>91.4</td>
<td>75.1</td>
<td>77.7</td>
<td>69.0</td>
</tr>
<tr>
<td>Qi et al. (2017a) [8]</td>
<td>56.3</td>
<td>14.9</td>
<td>66.7</td>
<td>83.8</td>
<td>0.0</td>
<td>0.0</td>
<td>71.4</td>
<td>46.1</td>
<td>44.6</td>
<td>36.9</td>
</tr>
<tr>
<td>Qi et al. (2017b) [20]</td>
<td>68.0</td>
<td>73.1</td>
<td>84.2</td>
<td>69.9</td>
<td>0.5</td>
<td>1.6</td>
<td>85.5</td>
<td>57.8</td>
<td>57.1</td>
<td>49.5</td>
</tr>
<tr>
<td>Hu et al. [53]</td>
<td>38.9</td>
<td>59.6</td>
<td>81.5</td>
<td>27.7</td>
<td>22.0</td>
<td>2.1</td>
<td>74.9</td>
<td>53.3</td>
<td>49.9</td>
<td>38.6</td>
</tr>
<tr>
<td>Landrieu et al. [54]</td>
<td>56.4</td>
<td>61.8</td>
<td>87.4</td>
<td>36.5</td>
<td>34.4</td>
<td>6.2</td>
<td>79.0</td>
<td>64.8</td>
<td>59.6</td>
<td>47.1</td>
</tr>
<tr>
<td>Thomas et al. [21]</td>
<td>86.5</td>
<td>88.4</td>
<td>92.7</td>
<td>77.2</td>
<td>54.3</td>
<td>13.3</td>
<td>93.3</td>
<td>73.7</td>
<td>76.7</td>
<td>68.8</td>
</tr>
<tr>
<td>Ours</td>
<td>81.04</td>
<td>84.9</td>
<td>91.8</td>
<td>57.8</td>
<td>57.5</td>
<td>39.4</td>
<td>93.2</td>
<td>80.6</td>
<td>80.0</td>
<td>68.7</td>
</tr>
</tbody>
</table>

4. Conclusions

The semantic segmentation of urban 3D meshes is a key task in the photogrammetry and remote sensing field. Urban 3D meshes are usually irregular, which makes it hard to use traditional convolution networks to perform semantic segmentation, and have abundant texture information, which is useful for improving the accuracy of semantic segmentation. How to robustly obtain high-accuracy semantic segmentation results for urban 3D meshes has become a challenging issue. In this paper, we propose a MeshNet-SP consisting of the differentiable sparse coding (DSC) subnetwork and the semantic feature extraction (SFE) subnetwork. The DSC subnetwork is used to produce low-intrinsic-dimensional data from raw texture images. Pioneer researchers have evaluated that low-intrinsic-dimensional data are helpful in making deep learning networks train easier and achieve higher accuracy. The SFE subnetwork constructs a graph based on the center of gravity points of facet by the k nearest neighbors (knn) method, and performs edge convolution on that graph to obtain high-level features used to predict a label per facet. The effectiveness and robustness of the proposed MeshNet-SP have been evaluated through ablation experiments. From the results of ablation experiments, we observe that the low-intrinsic-dimensional data (sparse codes) produced by the DSC subnetwork are key for obtaining high and robust semantic segmentation results, but the accuracy is not proportional to the dimension of sparse codes. Moreover, compared to other competition methods, the proposed MeshNet-SP can achieve...
higher accuracy for semantic urban 3D mesh segmentation. Specifically, the maximum increase reaches 34.5%, 35.4%, and 31.8% in mR, mF1, and mIoU, respectively.

**Author Contributions:** Conceptualization, G.Z. and R.Z.; methodology, G.Z. and R.Z.; validation, R.Z.; writing—original draft preparation, R.Z.; writing—review and editing, G.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Natural Science Foundation of Jiangsu Province (Grant Number BK20230338); National Natural Science Foundation of China (Grant Number 41601365) and the Key Laboratory Independent Research Foundation (Grant Number 2022-ZZKY-JJ-05-02).

**Data Availability Statement:** The public urban 3D mesh dataset released by Gao et al. [48] can be downloaded at https://3d.bk.tudelft.nl/projects/meshannotation/, (accessed on 5 October 2021).

**Acknowledgments:** We would like to thank the National Natural Science Foundation of China for providing the funding for this project. The authors appreciate Gao et al. [48] for releasing the public urban 3D mesh dataset.

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

**References**


55. Dominik, W.; Bozyczko, M.; Tulacz-Maziarz, K. Deep learning for automatic lidar point cloud processing 2021. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.