Appendices

A. Details for the Evaluation of Sampling.

We provide the implementation details of different sampling approaches evaluated in Section 4.1. To sample K points (point features) from a large-scale point cloud $\boldsymbol{P}$ with N points (point features):

1. *Farthest Point Sampling (FPS):* We follow the implementation[2] provided by PointNet++ [44], which is also widely used in [33, 60, 36, 6, 70]. In particular, FPS is implemented as an operator running on GPU.

2. *Inverse Density Importance Sampling (IDIS):* Given a point p_i, its density ρ is approximated by calculating the summation of the distances between p_i and its nearest t points [15]. Formally:

$$\rho(p_i) = \sum_{j=1}^{t} \left|\left| p_i - p_i^j \right|\right|, p_i^j \in \mathcal{N}(p_i) \tag{4}$$

   where p_i^j represents the coordinates (i.e. x-y-z) of the j^{th} point of the neighbour points set $\mathcal{N}(p_i)$, t is set to 16. All the points are ranked according to the inverse density $\frac{1}{\rho}$ of points. Finally, the top K points are selected.

3. *Random Sampling (RS):* We implement random sampling with the python numpy package. Specifically, we first use the numpy function *numpy.random.choice()* to generate K indices. We then gather the corresponding spatial coordinates and per-point features from point clouds by using these indices.

4. *Generator-based Sampling (GS):* The implementation follows the code[3] provided by [12]. We first train a ProgressiveNet [12] to transform the raw point clouds into ordered point sets according to their relevance to the task. After that, the first K points are kept, while the rest is discarded.

5. *Continuous Relaxation based Sampling (CRS): CRS* is implemented with the self-attended gumbel-softmax sampling [1][66]. Given a point feature set $\boldsymbol{P} \in \mathbb{R}^{N\times(d+3)}$ with 3D coordinates and per point features, we firstly estimate a probability score vector $\mathbf{s} \in \mathbb{R}^N$ through a score function parameterized by a MLP layer, i.e., $\mathbf{s} = softmax(MLP(\boldsymbol{P}))$, which learns a categorical distribution. Then, with the Gumbel noise $\mathbf{g} \in \mathbb{R}^N$ drawn from the distribution $Gumbel(0, 1)$. Each sampled point feature vector $\mathbf{y} \in \mathbb{R}^{d+3}$ is calculated as follows:

$$\mathbf{y} = \sum_{i=1}^{N} \frac{\exp\left((log(\mathbf{s}^{(i)}) + \mathbf{g}^{(i)})/\tau\right)\boldsymbol{P}^{(i)}}{\sum_{j=1}^{N}\exp\left((log(\mathbf{s}^{(j)}) + \mathbf{g}^{(j)})/\tau\right)}, \tag{5}$$

   where $\mathbf{s}^{(i)}$ and $\mathbf{g}^{(i)}$ indicate the i^{th} element in the vector $\mathbf{s}$ and $\mathbf{g}$ respectively, $\boldsymbol{P}^{(i)}$ represents the i^{th} row vector in the input matrix $\boldsymbol{P}$. $\tau > 0$ is the annealing temperature. When $\tau \to 0$, Equation 5 approaches the discrete distribution and samples each row vector in $\boldsymbol{P}$ with the probability $p(\mathbf{y} = \boldsymbol{P}^{(i)}) = \mathbf{s}^{(i)}$.

6. *Policy Gradients based Sampling (PGS):* Given a point feature set $\boldsymbol{P} \in \mathbb{R}^{N\times(d+3)}$ with 3D coordinates and per point features, we first predict a score $\mathbf{s}$ for each point, which is learnt by an MLP function, i.e., $\mathbf{s} = softmax(MLP(\boldsymbol{P})) + \epsilon$, where $\epsilon \in \mathbb{R}^N$ is a zero-mean Gaussian noise with the variance $\boldsymbol{\Sigma}$ for random exploration. After that, we sample K vectors in $\boldsymbol{P}$ with the top K scores. Sampling each point/vector can be regarded as an independent action and a sequence of them form a sequential Markov Decision Process (MDP) with the following policy function π:

$$a_i \sim \pi(a|\boldsymbol{P}^{(i)}; \theta, \mathbf{s}) \tag{6}$$

   where a_i is the binary decision of whether to sample the i^{th} vector in $\boldsymbol{P}$ and θ is the network parameter of the MLP. Hence to properly improve the sampling policy with an underivable sampling process, we apply REINFORCE algorithm [50] as the gradient estimator. The segmentation accuracy R is applied as the reward value for the entire sampling process as $\mathcal{J} = R$. It is optimized with the following estimated gradients:

$$\frac{\partial \mathcal{J}}{\partial \theta} \approx \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{N}\frac{\partial}{\partial\theta}\log\pi(a_i|\boldsymbol{P}^{(i)};\theta,\boldsymbol{\Sigma})\times (R - b^c - b(\boldsymbol{P}^{(i)})), \tag{7}$$

   where M is the batch size, b^c and $b(\boldsymbol{P}^{(i)})$ are two control variates [41] for alleviating the high variance problem of policy gradients.

[2] https://github.com/charlesq34/pointnet2
[3] https://github.com/orendv/learning_to_sample

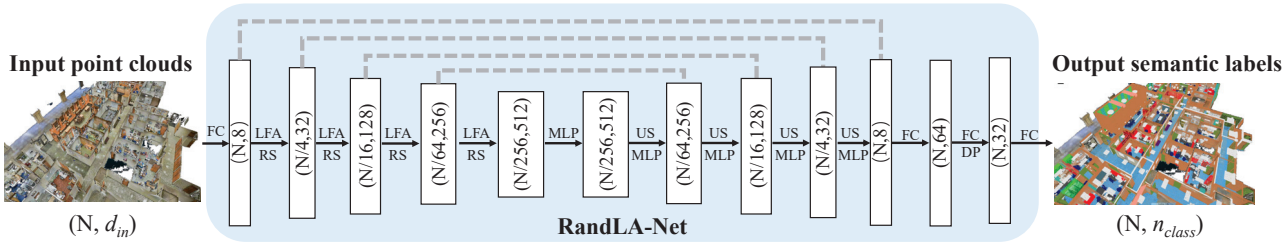

Figure 7. The detailed architecture of our RandLA-Net. (N, D) represents the number of points and feature dimension respectively. FC: Fully Connected layer, LFA: Local Feature Aggregation, RS: Random Sampling, MLP: shared Multi-Layer Perceptron, US: Up-sampling, DP: Dropout.

B. Details of the Network Architecture

Figure 7 shows the detailed architecture of RandLA-Net. The network follows the widely-used encoder-decoder architecture with skip connections. The input point cloud is first fed to a shared MLP layer to extract per-point features. Four encoding and decoding layers are then used to learn features for each point. At last, three fully-connected layers and a dropout layer are used to predict the semantic label of each point. The details of each part are as follows:

Network Input: The input is a large-scale point cloud with a size of $N \times d_{in}$ (the batch dimension is dropped for simplicity), where N is the number of points, d_{in} is the feature dimension of each input point. For both S3DIS [2] and Semantic3D [17] datasets, each point is represented by its 3D coordinates and color information (i.e., x-y-z-R-G-B), while each point of the SemanticKITTI [3] dataset is only represented by 3D coordinates.

Encoding Layers: Four encoding layers are used in our network to progressively reduce the size of the point clouds and increase the per-point feature dimensions. Each encoding layer consists of a local feature aggregation module (Section 3.3) and a random sampling operation (Section 3.2). The point cloud is downsampled with a four-fold decimation ratio. In particular, only 25% of the point features are retained after each layer, i.e., $(N \rightarrow \frac{N}{4} \rightarrow \frac{N}{16} \rightarrow \frac{N}{64} \rightarrow \frac{N}{256})$. Meanwhile, the per-point feature dimension is gradually increased each layer to preserve more information, i.e., $(8 \rightarrow 32 \rightarrow 128 \rightarrow 256 \rightarrow 512)$.

Decoding Layers: Four decoding layers are used after the above encoding layers. For each layer in the decoder, we first use the KNN algorithm to find one nearest neighboring point for each query point, the point feature set is then upsampled through a nearest-neighbor interpolation. Next, the upsampled feature maps are concatenated with the intermediate feature maps produced by encoding layers through skip connections, after which a shared MLP is applied to the concatenated feature maps.

Final Semantic Prediction: The final semantic label of each point is obtained through three shared fully-connected layers $(N,\ 64) \rightarrow (N,\ 32) \rightarrow (N,\ n_{class})$ and a dropout layer. The dropout ratio is 0.5.

Network Output: The output of RandLA-Net is the predicted semantics of all points, with a size of $N \times n_{class}$, where n_{class} is the number of classes.

C. Additional Ablation Studies on LocSE

In Section 3.3, we encode the relative point position based on the following equation:

$$\mathbf{r}_i^k = MLP\Big(p_i \oplus p_i^k \oplus (p_i - p_i^k) \oplus ||p_i - p_i^k||\Big) \quad (8)$$

We further investigate the effects of different spatial information in our framework. Particularly, we conduct the following more ablative experiments for LocSE:

- 1) Encoding the coordinates of the point p_i only.
- 2) Encoding the coordinates of neighboring points p_i^k only.
- 3) Encoding the coordinates of the point p_i and its neighboring points p_i^k.
- 4) Encoding the coordinates of the point p_i, the neighboring points p_i^k, and Euclidean distance $||p_i - p_i^k||$.
- 5) Encoding the coordinates of the point p_i, the neighboring points p_i^k, and the relative position $p_i - p_i^k$.

Table 6 compares the mIoU scores of all ablated networks on the SemanticKITTI [3] dataset. We can see that: 1) Explicitly encoding all spatial information leads to the best mIoU performance. 2) The relative position $p_i - p_i^k$ plays an important role in this component, primarily because the relative point position enables the network to be aware of the local geometric patterns. 3) Only encoding

LocSE	mIoU(%)
(1) (p_i)	48.9
(2) (p_i^k)	50.7
(3) (p_i, p_i^k)	52.5
(4) $(p_i, p_i^k, \|\|p_i - p_i^k\|\|)$	53.7
(5) $(p_i, p_i^k, p_i - p_i^k)$	56.8
(6) $(p_i, p_i^k, p_i - p_i^k, \|\|p_i - p_i^k\|\|)$ (**The Full Unit**)	57.1

Table 6. The mIoU result of RandLA-Net by encoding different kinds of spatial information.

the point position p_i or p_i^k is unlikely to improve the performance, because the relative local geometric patterns are not explicitly encoded.

D. Additional Ablation Studies on Dilated Residual Block

In our RandLA-Net, we stack two LocSE and Attentive Pooling units as the standard dilated residual block to gradually increase the receptive field. To further evaluate how the number of aggregation units in the dilated residual block impact the entire network, we conduct the following two more groups of experiments.

- 1) We simplify the dilated residual block by using only one LocSE unit and attentive pooling.
- 2) We add one more LocSE unit and attentive pooling, i.e., there are three aggregation units chained together.

Dilated residual block	mIoU(%)
(1) one aggregation unit	52.9
(2) three aggregation units	54.2
(3) two aggregation units (**The Standard Block**)	57.1

Table 7. The mIoU scores of RandLA-Net regarding different number of aggregation units in a residual block.

Table 7 shows the mIoU scores of different ablated networks on the validation split of the SemanticKITTI [3] dataset. It can be seen that: 1) Only one aggregation unit in the dilated residual block leads to a significant drop in segmentation performance, due to the limited receptive field. 2) Three aggregation units in each block do not improve the accuracy as expected. This is because the significantly increased receptive fields and the large number of trainable parameters tend to be overfitted.

E. Visualization of Attention Scores

To better understand the attentive pooling, it is desirable to visualize the learned attention scores. However, since the attentive pooling operates on a relatively small local point set (i.e., K=16), it is hardly able to recognize meaningful shapes from such small local regions. Alternatively, we visualize the learned attention weight matrix W defined in Equation 2 in each layer. As shown in Figure 8, the attention weights have large values in the first encoding layers, then gradually become smooth and stable in subsequent layers. This shows that the attentive pooling tends to choose prominent or key point features at the beginning. After the point cloud being significantly downsampled, the attentive pooling layer tends to retain the majority of those point features.

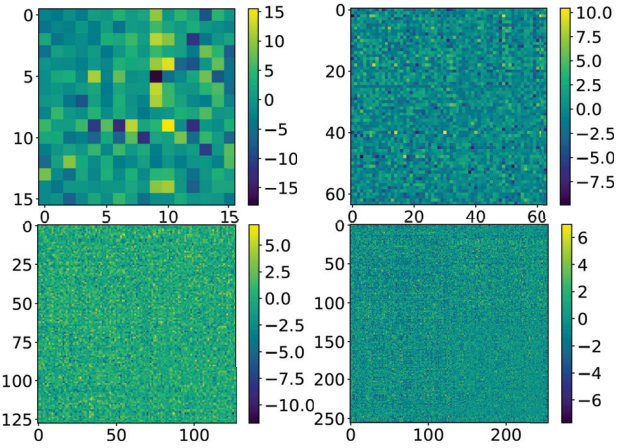

Figure 8. Visualization of the learned attention matrix in different layers. From top left to bottom right: 16×16 attention matrix, 64×64 attention matrix, 128×128 attention matrix, 256×256 attention matrix. The yellow color represents higher attention scores.

F. Additional Results on Semantic3D

More qualitative results of RandLA-Net on the Semantic3D [17] dataset (*reduced-8*) are shown in Figure 9.

G. Additional Results on SemanticKITTI

Figure 10 shows more qualitative results of our RandLA-Net on the validation set of SemanticKITTI [3]. The red boxes showcase the failure cases. It can be seen that, the points belonging to *other-vehicle* are likely to be misclassified as *car*, mainly because the partial point clouds without colors are extremely difficult to be distinguished between the two similar classes. In addition, our approach tends to fail in several minority classes such as *bicycle*, *motorcycle*, *bicyclist* and *motorcyclist*, due to the extremely imbalanced point distribution in the dataset. For example, the number of points for *vegetation* is 7000 times more than that of *motorcyclist*.

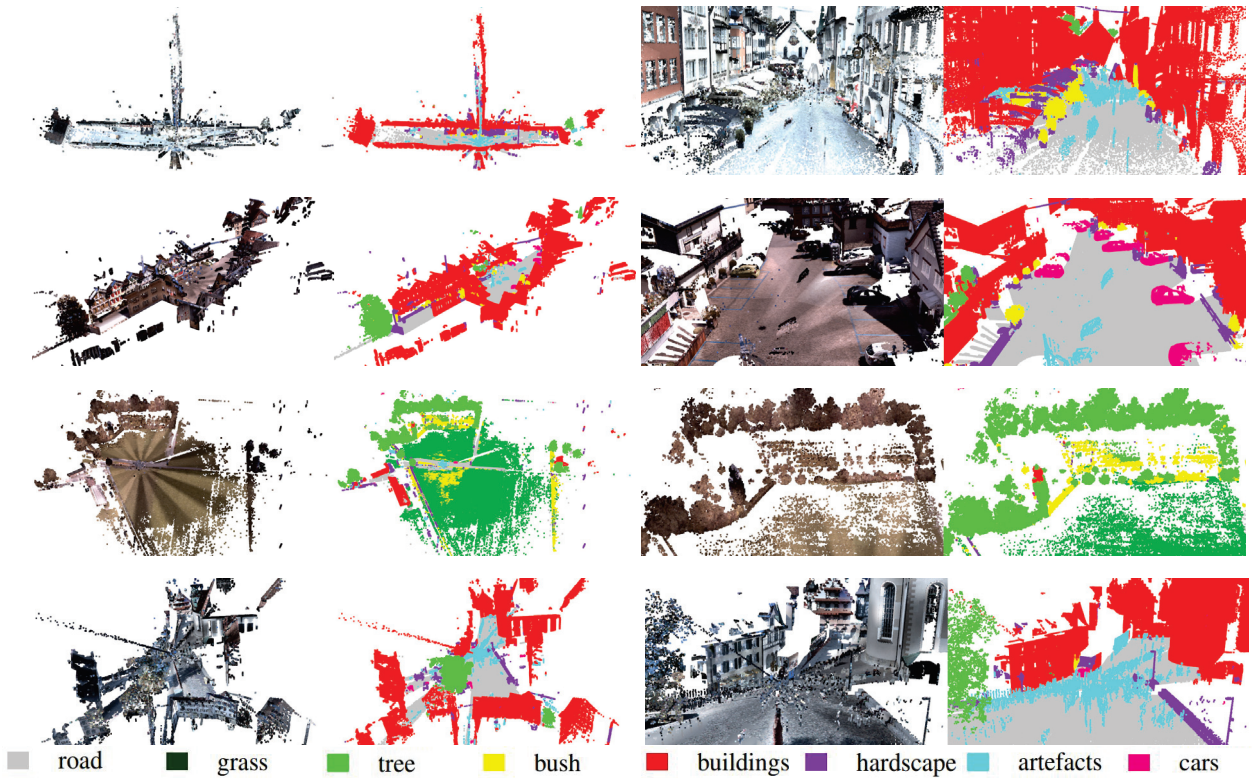

Figure 9. Qualitative results of RandLA-Net on the *reduced-8* split of Semantic3D. From left to right: full RGB colored point clouds, predicted semantic labels of full point clouds, detailed view of colored point clouds, detailed view of predicted semantic labels. Note that the ground truth of the test set is not publicly available.

H. Additional Results on S3DIS

We report the detailed 6-fold cross validation results of our RandLA-Net on S3DIS [2] in Table 8. Figure 11 shows more qualitative results of our approach.

I. Video Illustration

We provide a video to show qualitative results of our RandLA-Net on both indoor and outdoor datasets, which can be viewed at https://www.youtube.com/watch?v=Ar3eY_lwzMk&t=9s.

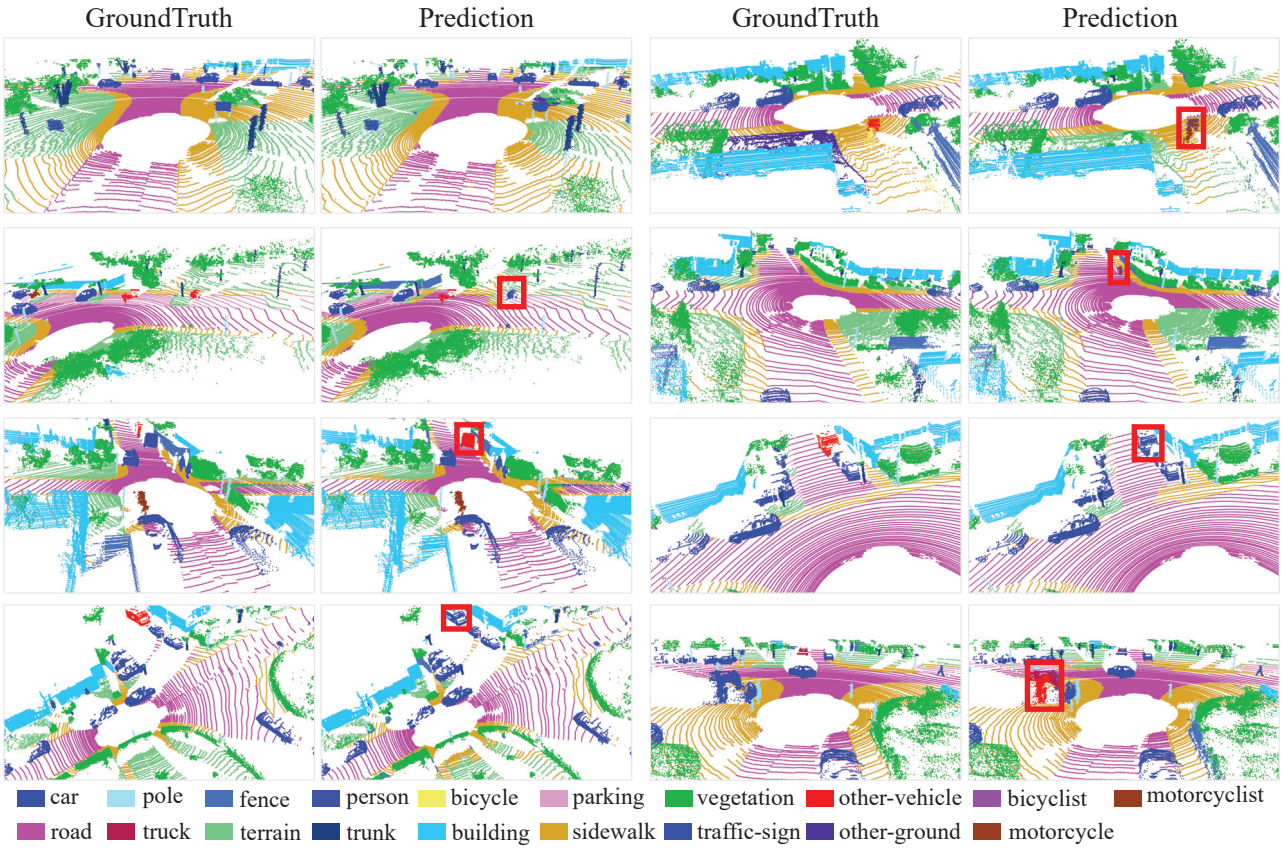

Figure 10. Qualitative results of RandLA-Net on the validation split of SemanticKITTI [3]. Red boxes show the failure cases.

	OA(%)	mAcc(%)	mIoU(%)	ceil.	floor	wall	beam	col.	wind.	door	table	chair	sofa	book.	board	clut.
PointNet [43]	78.6	66.2	47.6	88.0	88.7	69.3	42.4	23.1	47.5	51.6	54.1	42.0	9.6	38.2	29.4	35.2
RSNet [21]	-	66.5	56.5	92.5	92.8	78.6	32.8	34.4	51.6	68.1	59.7	60.1	16.4	50.2	44.9	52.0
3P-RNN [67]	86.9	-	56.3	92.9	93.8	73.1	42.5	25.9	47.6	59.2	60.4	66.7	24.8	57.0	36.7	51.6
SPG [26]	86.4	73.0	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	**73.5**	69.2	63.2	45.9	8.7	52.9
PointCNN [33]	**88.1**	75.6	65.4	**94.8**	**97.3**	75.8	63.3	51.7	58.4	57.2	71.6	69.1	39.1	61.2	52.2	58.6
PointWeb [70]	87.3	76.2	66.7	93.5	94.2	80.8	52.4	41.3	64.9	68.1	71.4	67.1	50.3	62.7	62.2	58.5
ShellNet [69]	87.1	-	66.8	90.2	93.6	79.9	60.4	44.1	64.9	52.9	71.6	**84.7**	53.8	64.6	48.6	59.4
KPConv [54]	-	79.1	**70.6**	93.6	92.4	**83.1**	**63.9**	**54.3**	**66.1**	**76.6**	57.8	64.0	**69.3**	**74.9**	61.3	**60.3**
RandLA-Net (Ours)	88.0	**82.0**	70.0	93.1	96.1	80.6	62.4	48.0	64.4	69.4	69.4	76.4	60.0	64.2	**65.9**	60.1

Table 8. Quantitative results of different approaches on S3DIS [2] (6-fold cross-validation). Overall Accuracy (OA, %), mean class Accuracy (mAcc, %), mean IoU (mIoU, %), and per-class IoU (%) are reported.

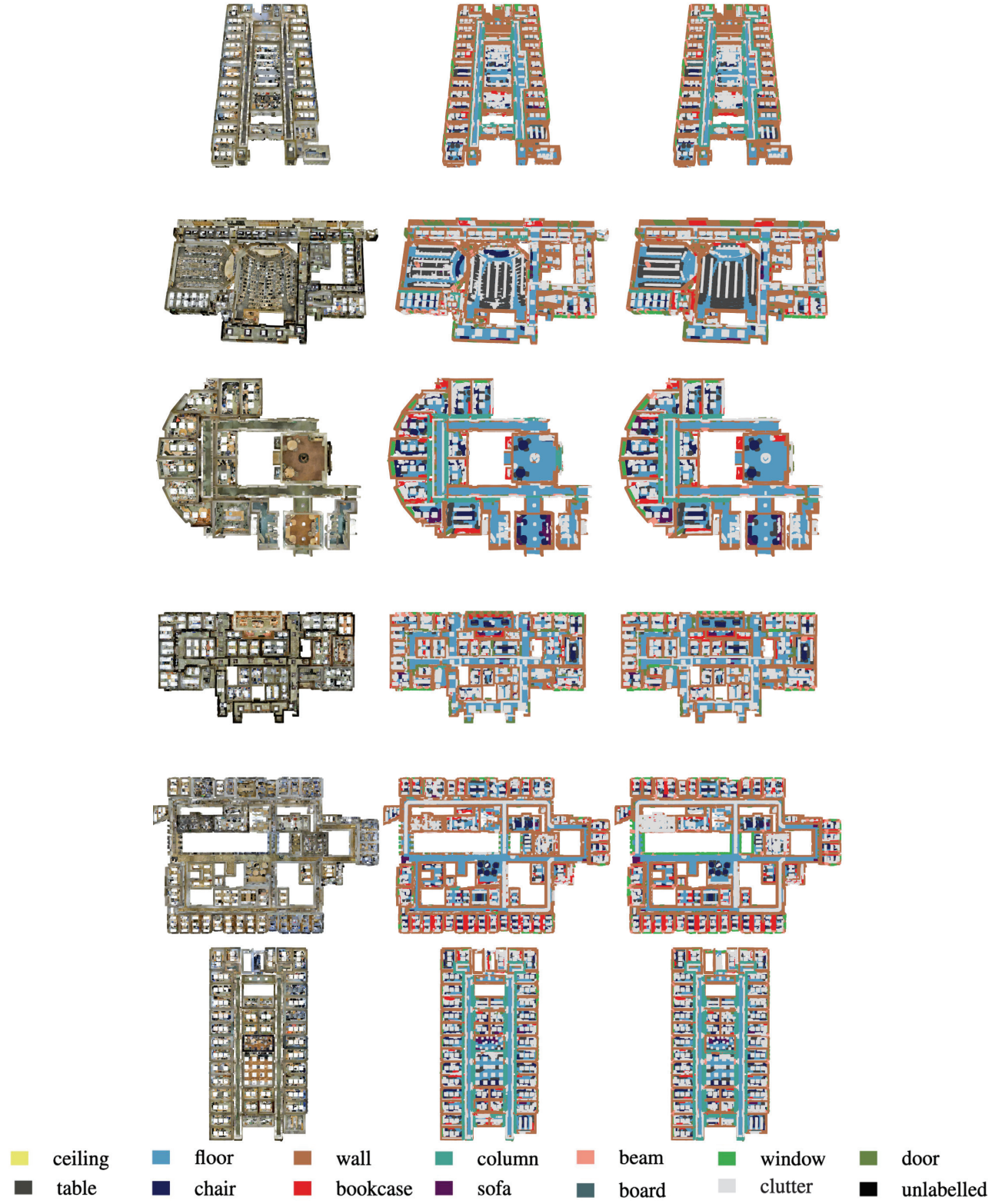

Figure 11. Semantic segmentation results of our RandLA-Net on the complete point clouds of Areas 1-6 in S3DIS. Left: full RGB input cloud; middle: predicted labels; right: ground truth.