RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds

Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, Andrew Markham

CVPR 2020 Oral.

2020-05-15

SKKU Yang Yong Jun

Figure

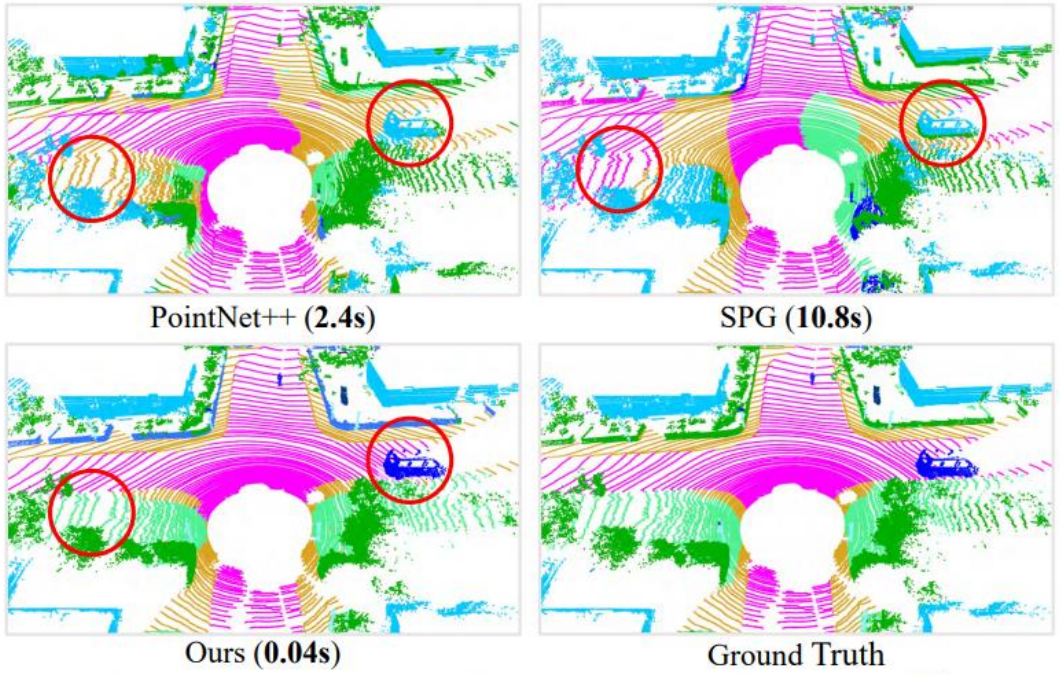
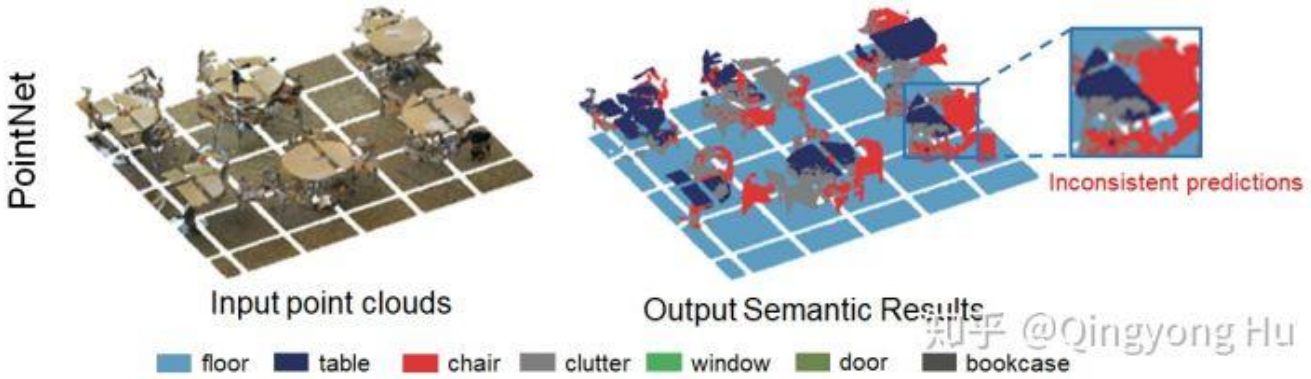


Figure 1. Semantic segmentation results of PointNet++ [44], SPG [26] and our approach on SemanticKITTI [3]. Our RandLA-Net takes only 0.04s to directly process a large point cloud with 10^5 points over $150\times130\times10$ meters in 3D space, which is up to $200\times$ faster than SPG. Red circles highlight the superior segmentation accuracy of our approach.



PointNet approach

- You need to cut into small 1m × 1m point cloud blocks, then sample each point cloud block to get a 2048 point input network.
- When cutting into zones, the overall shape of the object is lost
- Cannot learn shape information of objects larger than the size of the zone

Abstract

- Most existing approaches are relying on expensive sampling techniques or computationally heavy pre/post-processing steps
- we introduce RandLA-Net, an efficient and lightweight neural architecture to directly infer per-point semantics for large-scale point clouds.
- The key to our approach is to use random point sampling instead of more complex point selection approaches.
- we introduce a novel local feature aggregation module to progressively increase the receptive field for each 3D point, thereby effectively preserving geometric details.

Introduction

- The existing point cloud segmentation approach cannot capture a wide range of context information
- Cannot scale from a small point cloud to a wide poitn cloud

Reason

1) Point cloud sampling is computationally expensive or memory inefficient

2) Most local feature learners rely on expensive computational methods

3) In the case of a large-scale point cloud with many objects, existing local learners cannot learn complex structures or have an acceptable size.

Solved

1) Design a memory and computationally efficient deep learning network that can process large-scale poitn clouds without preprocessing of point clouds

2) By gradually increasing the receiving field size of the local layer, you can effectively learn the complex local structure.

Approach & Contribution

Approach

1) For each 3D point, a local spatial structure (LocSE) unit is first introduced to explicitly preserve the local geometry.

2) Utilizes meticulous pooling to automatically maintain useful local functions.

3) Multiple LocSE devices and meticulous pooling are stacked into an extended residual block to significantly increase the effective acceptance field of each point. All of these neural components are implemented as a shared MLP, making them highly memory and computationally efficient.

Contribution

1) We analyze and compare existing sampling approaches, identifying random sampling as the most suitable component for efficient learning on large-scale point clouds.

2) We propose an effective local feature aggregation module to preserve complex local structures by progressively increasing the receptive field for each point.

3) We demonstrate significant memory and computational gains over baselines, and surpass the state-of-the-art semantic segmentation methods on multiple large-scale benchmarks.

RandLA-Net: Sampling

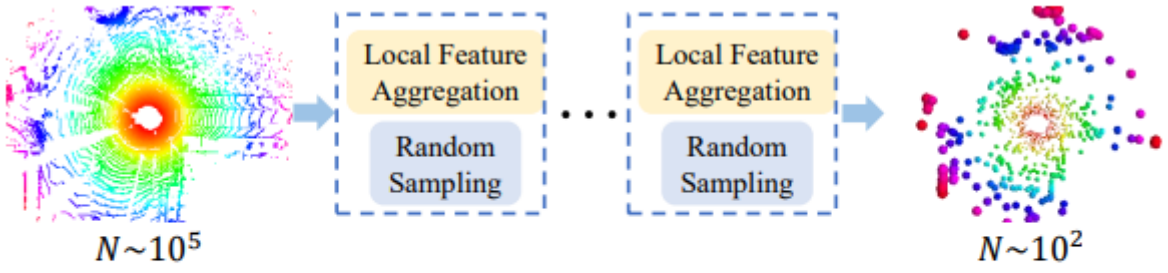


Figure 2. In each layer of RandLA-Net, the large-scale point cloud is significantly downsampled, yet is capable of retaining features necessary for accurate segmentation.

- Summary
- Many sampling algorithms are computationally expensive or require a lot of memory
- Many sampling algorithms are computationally expensive or require a lot of memory
- Random sampling is computationally efficient and is independent of the number of input points.
- Memory efficiency.
- However, random sampling loses a lot of useful information.

- When processing hundreds of meters of point cloud, you should downsample while maintaining as much geometric information as possible.
- -Heuristic sampling
- 1. Farthest Point Sampling (FPS)
- Select the farthest point from the sample
- The computational complexity is O(N ^ 2), which makes it unsuitable for large point clouds

- 2. IDIS (Inverse Density Importance Sampling)
- Find the density of the point cloud to make the density of each point uniform
- Influenced by outliers

- 3. Random Sampling
- Select K points from the entire point cloud
- Select random from uniform distribution

Local Feature Aggregation module

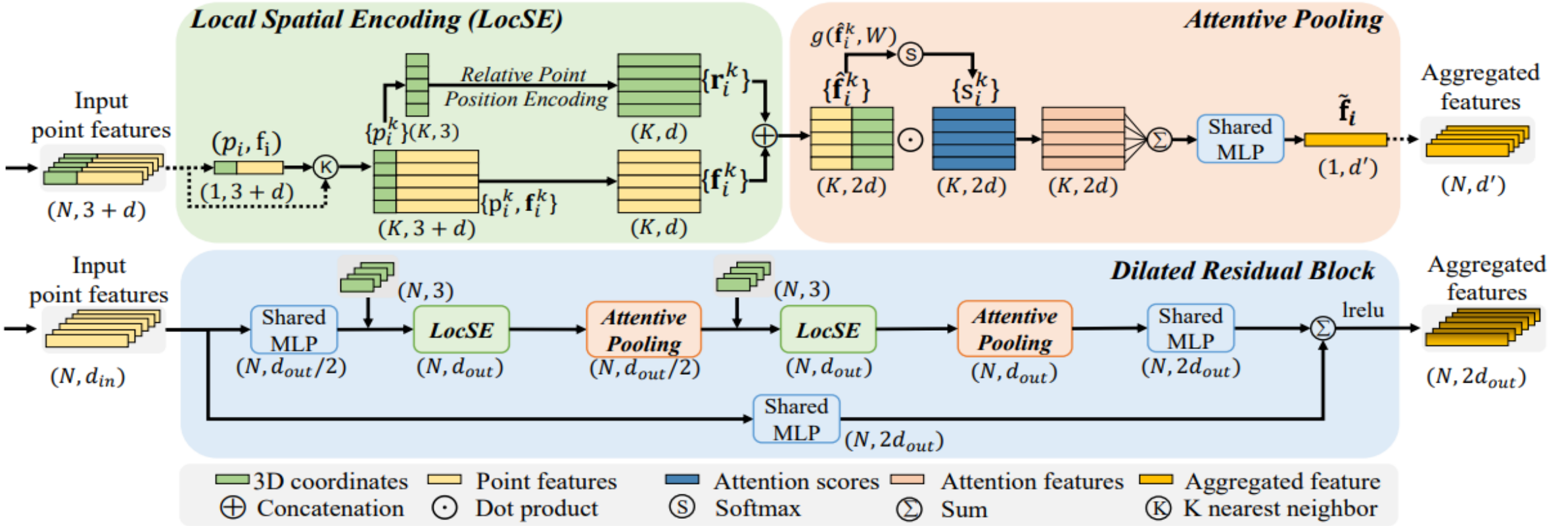
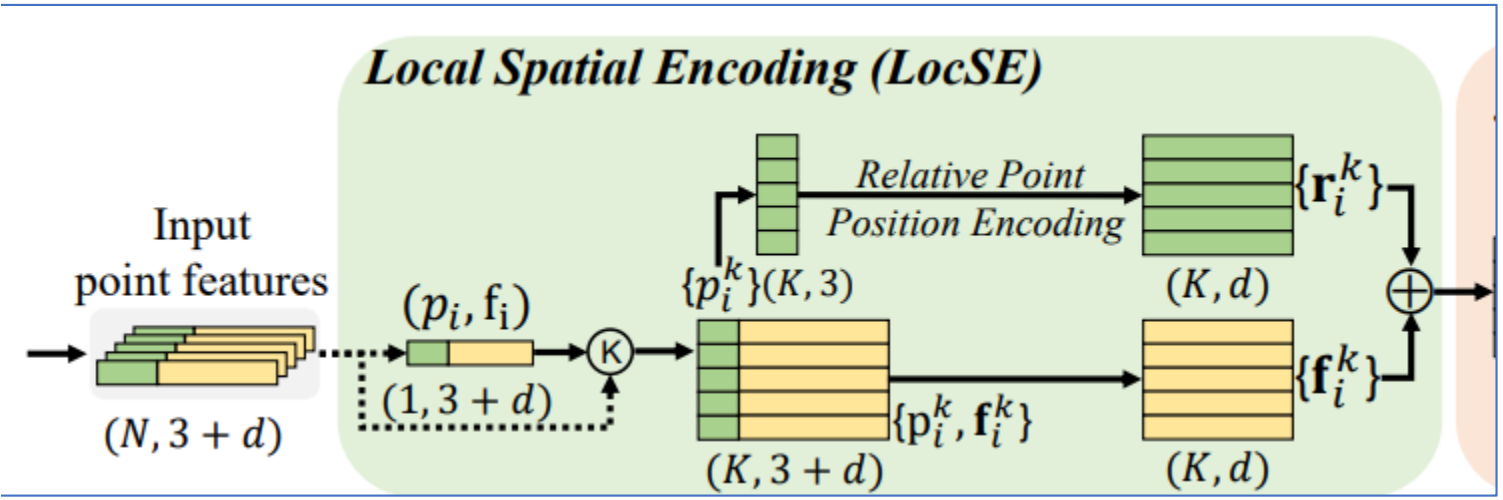


Figure 3. The proposed local feature aggregation module. The top panel shows the location spatial encoding block that extracts features, and the attentive pooling mechanism that weights the most important neighbouring features, based on the local context and geometry. The bottom panel shows how two of these components are chained together, to increase the receptive field size, within a residual block.

Local Spatial Encoding



Finding Neighbouring Points – KNN algorithm
Find p_i 's nearst k point, K = {p_i^1, p_i^2, … p_i^k}

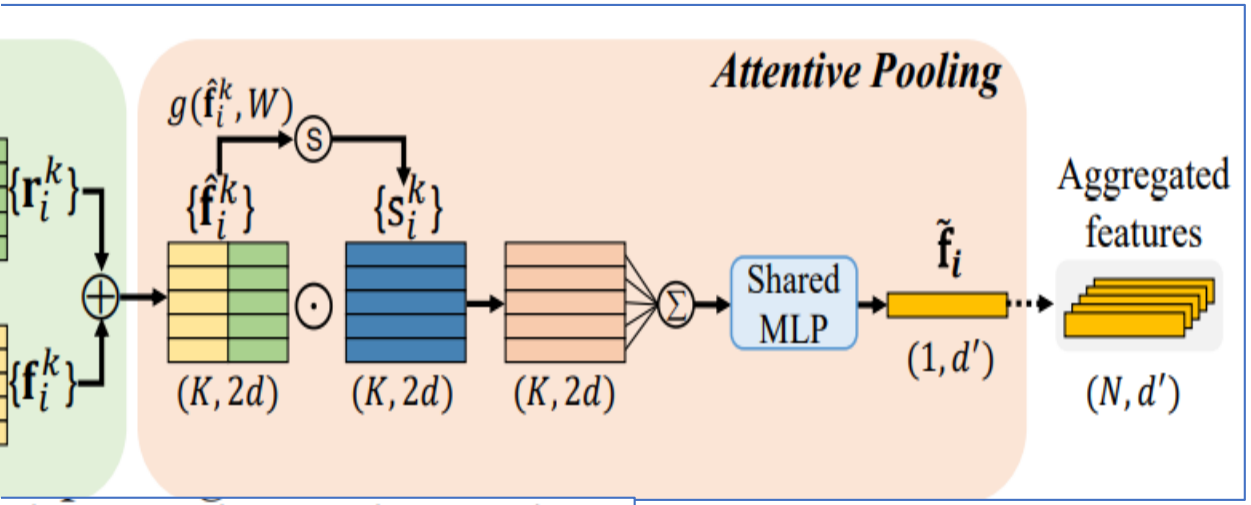
Relative Point Position Encoding

$$\mathbf{r}_i^k = MLP\left(p_i \oplus p_i^k \oplus (p_i - p_i^k) \oplus ||p_i - p_i^k||\right)$$

$\oplus$ is concatenation operation, || x || is Euclidean Distance

- Given Point cloud P (x-y-z + R,G,B or learnt feature)
- LocSE can observe the local geometric patterns, thus eventually benefiting the entire network to effectively learn complex local structures.
- LocSE tends to aid the network to learn local features and obtains good performance in practice

Attentive Pooling



$$\hat{\mathbf{F}}_i = \{\hat{\mathbf{f}}_i^1 \cdots \hat{\mathbf{f}}_i^k \cdots \hat{\mathbf{f}}_i^K\}$$

$$\mathbf{s}_i^k = g(\hat{\mathbf{f}}_i^k, \boldsymbol{W})$$

$$\tilde{\mathbf{f}}_i = \sum_{k=1}^{K} (\hat{\mathbf{f}}_i^k \cdot \mathbf{s}_i^k)$$

- Pointnet typically use max pooling to integrate the **local features,** information being lost.
- Computing attention scores
  - Function ***g*** learns a unique attention score for each feature (score: 0 ~ 1)
- Weighted Summation
  - Network learn selection of important feature automatically

- LocSE & Attentive Pooling learn to aggregate the geometric patterns and features of its K nearest points, and finally generate an informative feature

Dilated Residual Block

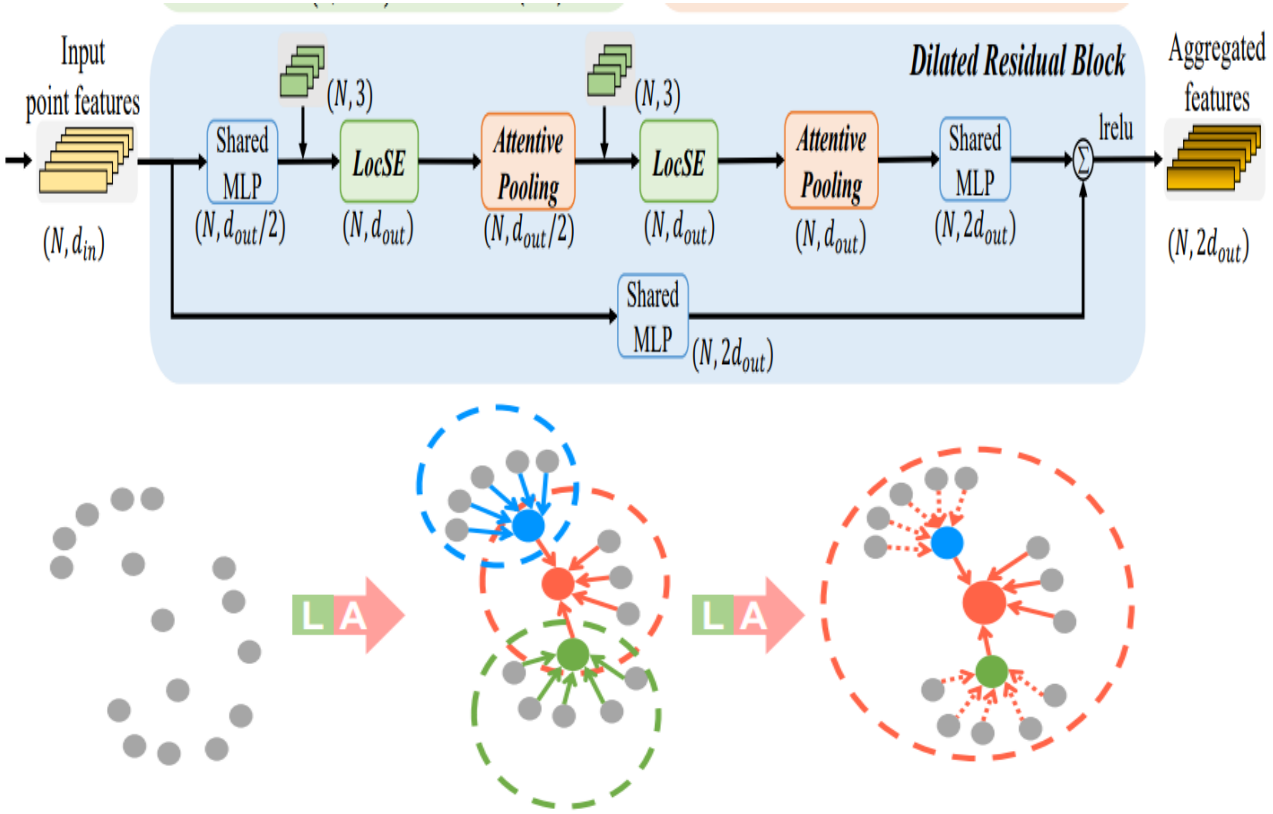


Figure 4. Illustration of the dilated residual block which significantly increases the receptive field (dotted circle) of each point, colored points represent the aggregated features. L: Local spatial encoding, A: Attentive pooling.

- Since the large point clouds are going to be substantially downsampled, receptive field for each point are increased, geometric details of input point clouds are more likely to be reserved
- EX) red point is able to receive information from up to K2 neighbouring points

Local Feature Aggregation module

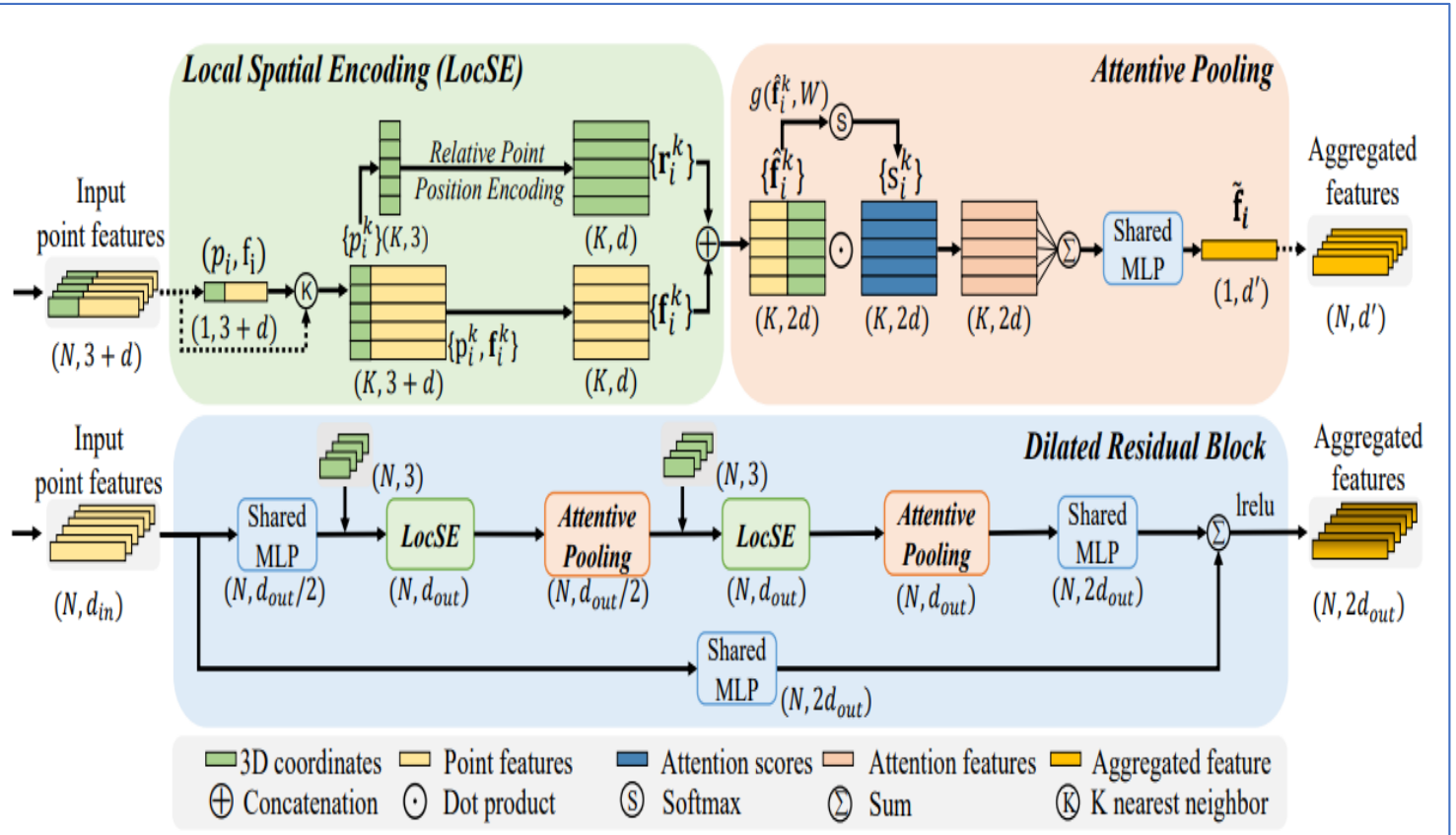
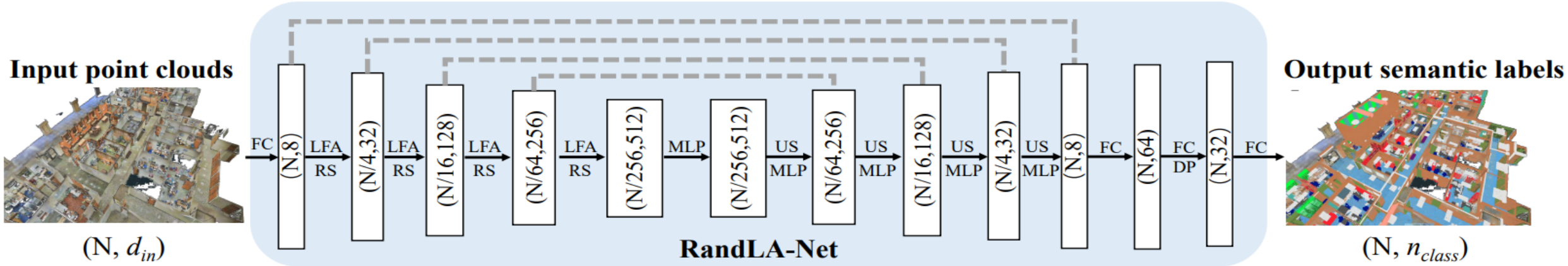


Figure 3. The proposed local feature aggregation module. The top panel shows the location spatial encoding block that extracts features, and the attentive pooling mechanism that weights the most important neighbouring features, based on the local context and geometry. The bottom panel shows how two of these components are chained together, to increase the receptive field size, within a residual block.

- Local Feature Aggregation module is designed to effectively preserve complex local structures considering neighbouring geometries and significantly increasing receptive fields.
- Moreover, this module only consists of feed-forward MLPs, thus being computationally efficient.

Segmentation Network



- MLP: Shared Multi Layer Perceptron FC: Fully Connected Layer, LFA: Local Feature Aggrestion, RS: Random Sampling, US: Upsampling, DP: Dropout
- The network follows the widely-used encoder-decoder architecture with skip connections. Shared MLP per points. Predict point label
- Input: N x d_{in} (x-y-z, r-g-b)
- Encoding Layer: LFA, RS
- Decoding Layer: US: Find KNN point & interpolation, Skip Connection
- Segmentation Prediction: shared fully-connected layers (N, n_{class})

Experiments - Sampling

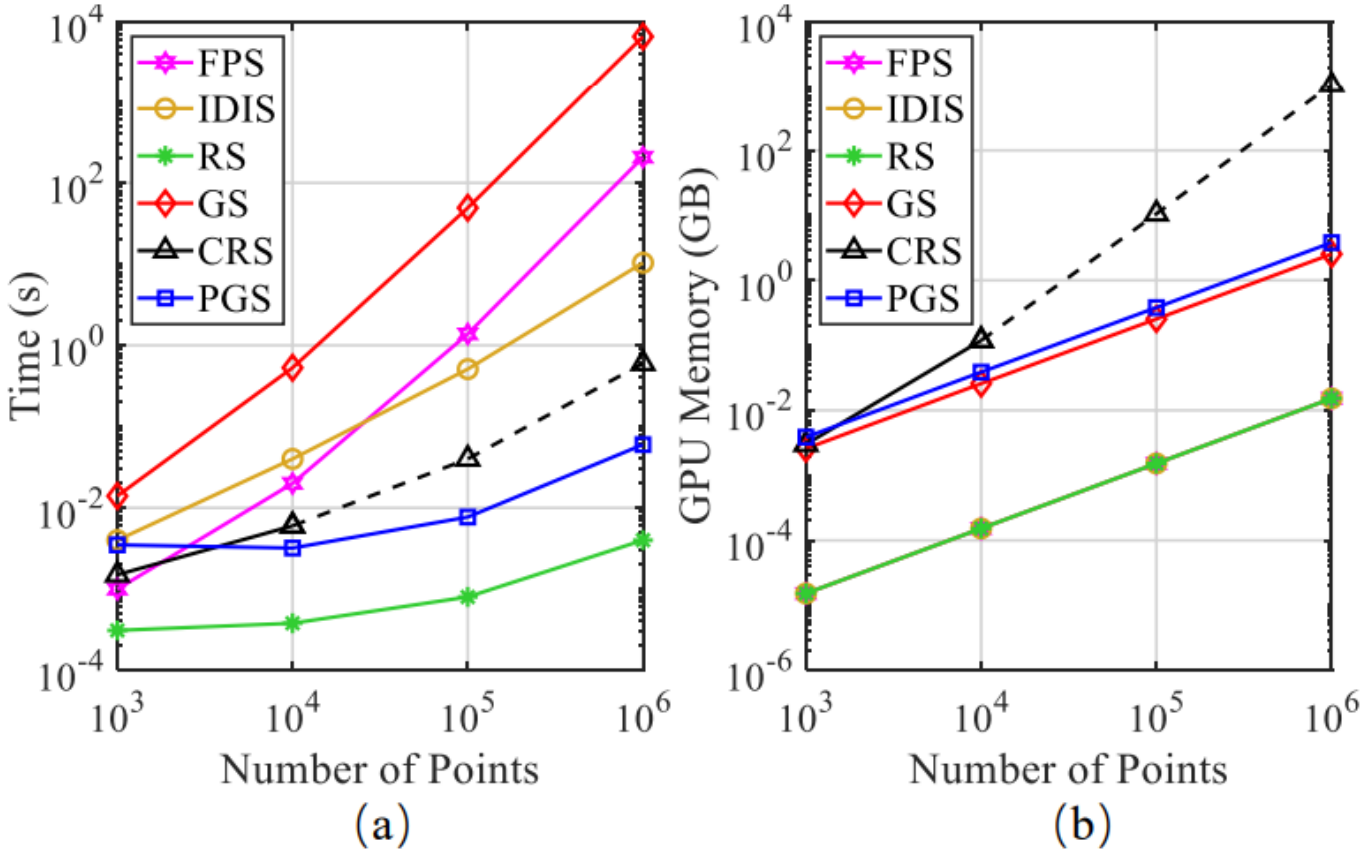


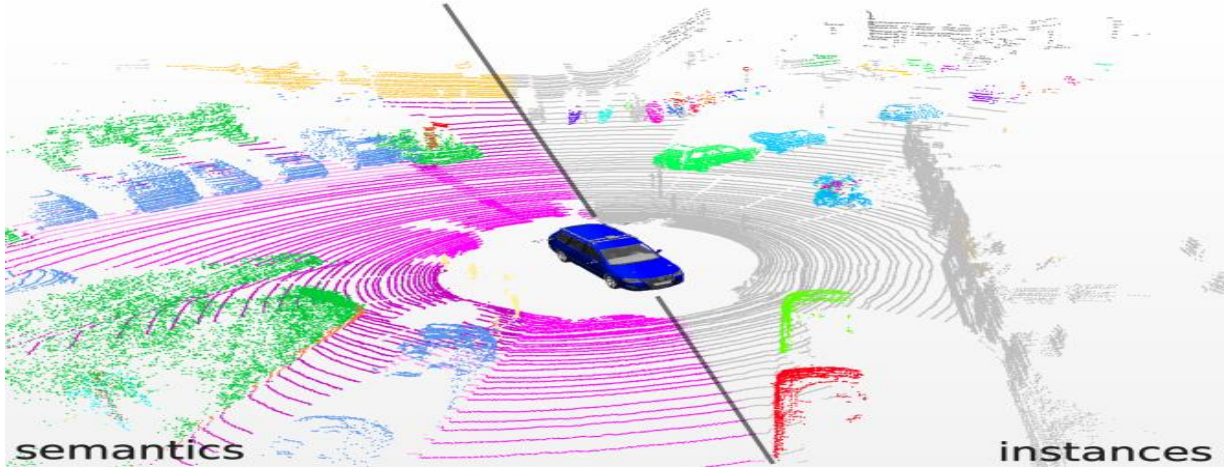
Figure 5. Time and memory consumption of different sampling approaches. The dashed lines represent estimated values due to the limited GPU memory.

- For largescale point clouds, another method use extremely time-consuming or memory-costly
- rely on the expensive sampling approaches

Experiments - Efficiency

	Total time (seconds)	Parameters (millions)	Maximum inference points (millions)
PointNet (Vanilla) [43]	192	0.8	0.49
PointNet++ (SSG) [44]	9831	0.97	0.98
PointCNN [33]	8142	11	0.05
SPG [26]	43584	**0.25**	-
KPConv [54]	717	14.9	0.54
RandLA-Net (Ours)	**185**	1.24	**1.03**

Table 1. The computation time, network parameters and maximum number of input points of different approaches for semantic segmentation on Sequence 08 of the SemanticKITTI [3] dataset.



- Data: 4071 LIDAR scan
- Eval time consumption & Network size
- Feed Point cloud (size: 81920)

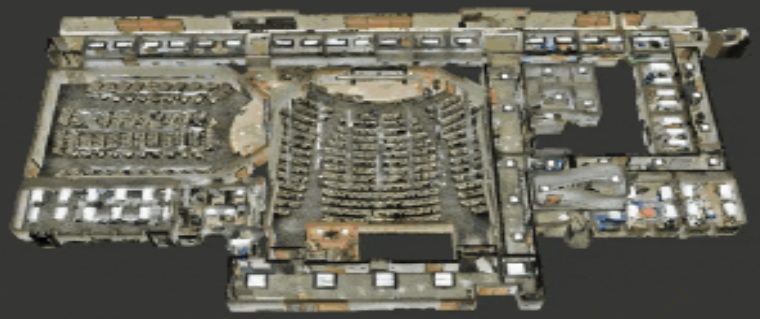
- RandLA-Net: 22FPS

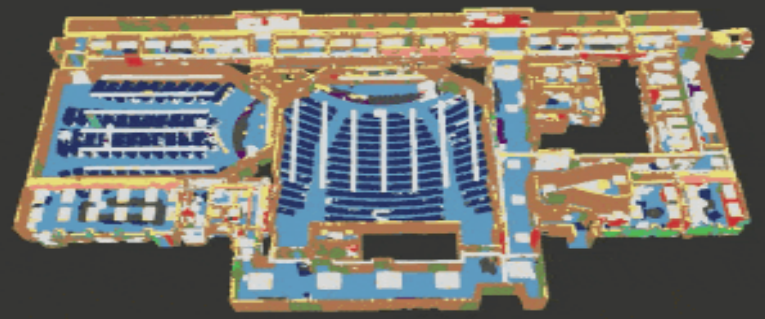
Experiments - Sementation

	mIoU (%)	OA (%)	man-made.	natural.	high veg.	low veg.	buildings	hard scape	scanning art.	cars
SnapNet_ [4]	59.1	88.6	82.0	77.3	79.7	22.9	91.1	18.4	37.3	64.4
SEGCloud [52]	61.3	88.1	83.9	66.0	86.0	40.5	91.1	30.9	27.5	64.3
RF_MSSF [53]	62.7	90.3	87.6	80.3	81.8	36.4	92.2	24.1	42.6	56.6
MSDeepVoxNet [46]	65.3	88.4	83.0	67.2	83.8	36.7	92.4	31.3	50.0	78.2
ShellNet [69]	69.3	93.2	96.3	90.4	83.9	41.0	94.2	34.7	43.9	70.2
GACNet [56]	70.8	91.9	86.4	77.7	**88.5**	**60.6**	94.2	37.3	43.5	77.8
SPG [26]	73.2	94.0	**97.4**	**92.6**	87.9	44.0	83.2	31.0	63.5	76.2
KPConv [54]	74.6	92.9	90.9	82.2	84.2	47.9	94.9	40.0	**77.3**	**79.7**
RandLA-Net (Ours)	**77.4**	**94.8**	95.6	91.4	86.6	51.5	**95.7**	**51.5**	69.8	76.8

Methods	Size	**mIoU(%)**	Params(M)	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic-sign
PointNet [43]	50K pts	14.6	3	61.6	35.7	15.8	1.4	41.4	46.3	0.1	1.3	0.3	0.8	31.0	4.6	17.6	0.2	0.2	0.0	12.9	2.4	3.7
SPG [26]		17.4	**0.25**	45.0	28.5	0.6	0.6	64.3	49.3	0.1	0.2	0.2	0.8	48.9	27.2	24.6	0.3	2.7	0.1	20.8	15.9	0.8
SPLATNet [49]		18.4	0.8	64.6	39.1	0.4	0.0	58.3	58.2	0.0	0.0	0.0	0.0	71.1	9.9	19.3	0.0	0.0	0.0	23.1	5.6	0.0
PointNet++ [44]		20.1	6	72.0	41.8	18.7	5.6	62.3	53.7	0.9	1.9	0.2	0.2	46.5	13.8	30.0	0.9	1.0	0.0	16.9	6.0	8.9
TangentConv [51]		40.9	0.4	83.9	63.9	33.4	15.4	83.4	90.8	15.2	2.7	16.5	12.1	79.5	49.3	58.1	23.0	28.4	**8.1**	49.0	35.8	28.5
SqueezeSeg [58]	64*2048 pixels	29.5	1	85.4	54.3	26.9	4.5	57.4	68.8	3.3	16.0	4.1	3.6	60.0	24.3	53.7	12.9	13.1	0.9	29.0	17.5	24.5
SqueezeSegV2 [59]		39.7	1	88.6	67.6	45.8	17.7	73.7	81.8	13.4	18.5	17.9	14.0	71.8	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.3
DarkNet21Seg [3]		47.4	25	91.4	74.0	57.0	26.4	81.9	85.4	18.6	**26.2**	26.5	15.6	77.6	48.4	63.6	31.8	33.6	4.0	52.3	36.0	50.0
DarkNet53Seg [3]		49.9	50	**91.8**	74.6	64.8	**27.9**	84.1	86.4	25.5	24.5	32.7	22.6	78.3	50.1	64.0	36.2	33.6	4.7	55.0	38.9	52.2
RangeNet53++ [40]		52.2	50	**91.8**	**75.2**	**65.0**	27.8	**87.4**	91.4	25.7	25.7	**34.4**	23.0	80.5	55.1	64.6	38.3	38.8	4.8	**58.6**	47.9	**55.9**
RandLA-Net (Ours)	50K pts	**53.9**	1.24	90.7	73.7	60.3	20.4	86.9	**94.2**	**40.1**	26.0	25.8	**38.9**	**81.4**	**61.3**	**66.8**	**49.2**	**48.2**	7.2	56.3	**49.2**	47.7

S3DIS: Area2



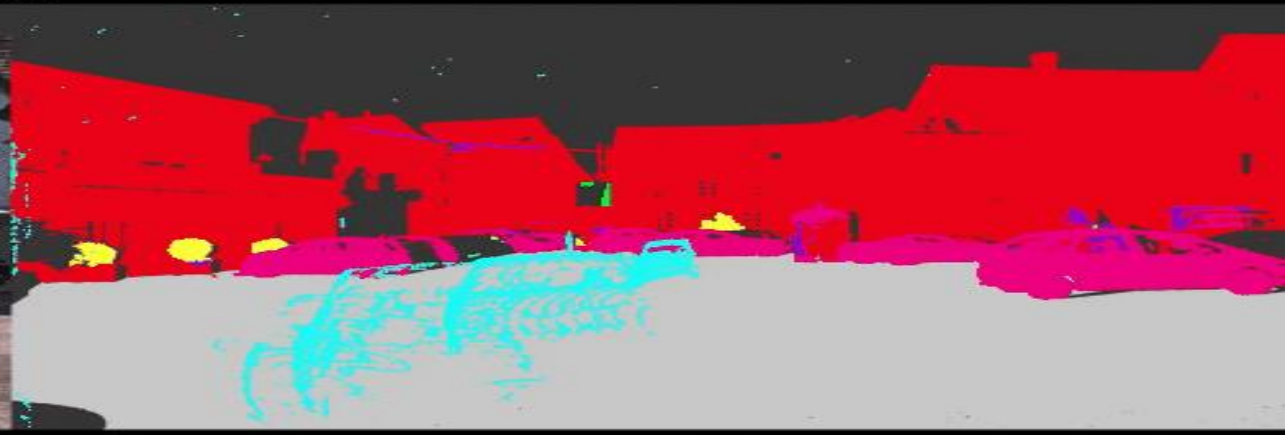


Input Point Cloud

Semantic Segmentation Result

floor beam window table sofa board
wall column door chair bookcase clutter

Semantic3D: sg27-10-reduced



Input Point Clouds

Semantic Segmentation Results

road grass tree bush buildings hardscape artefacts cars

Conclusion

- paper demonstrated that it is possible to efficiently and effectively segment large-scale point clouds by using a lightweight network architecture.

- Using Random Sampling
- local feature aggregation module is also introduced to effectively preserve useful features from a wide neighbourhood.