2nd Annual VIMS Datathon Competition 2021

Ferdowsi University of Mashhad (FUM)
Mashhad, Iran

September 2021

Objectives

- Train Machine Learning Algorithm

- Automatic Detection

Approach

1. Getting to know data
2. Visualize Data
3. Model Selection
4. Manual Labeling
5. Training Model
6. Model Performance
7. Proposed Ideas

Get to Know Data

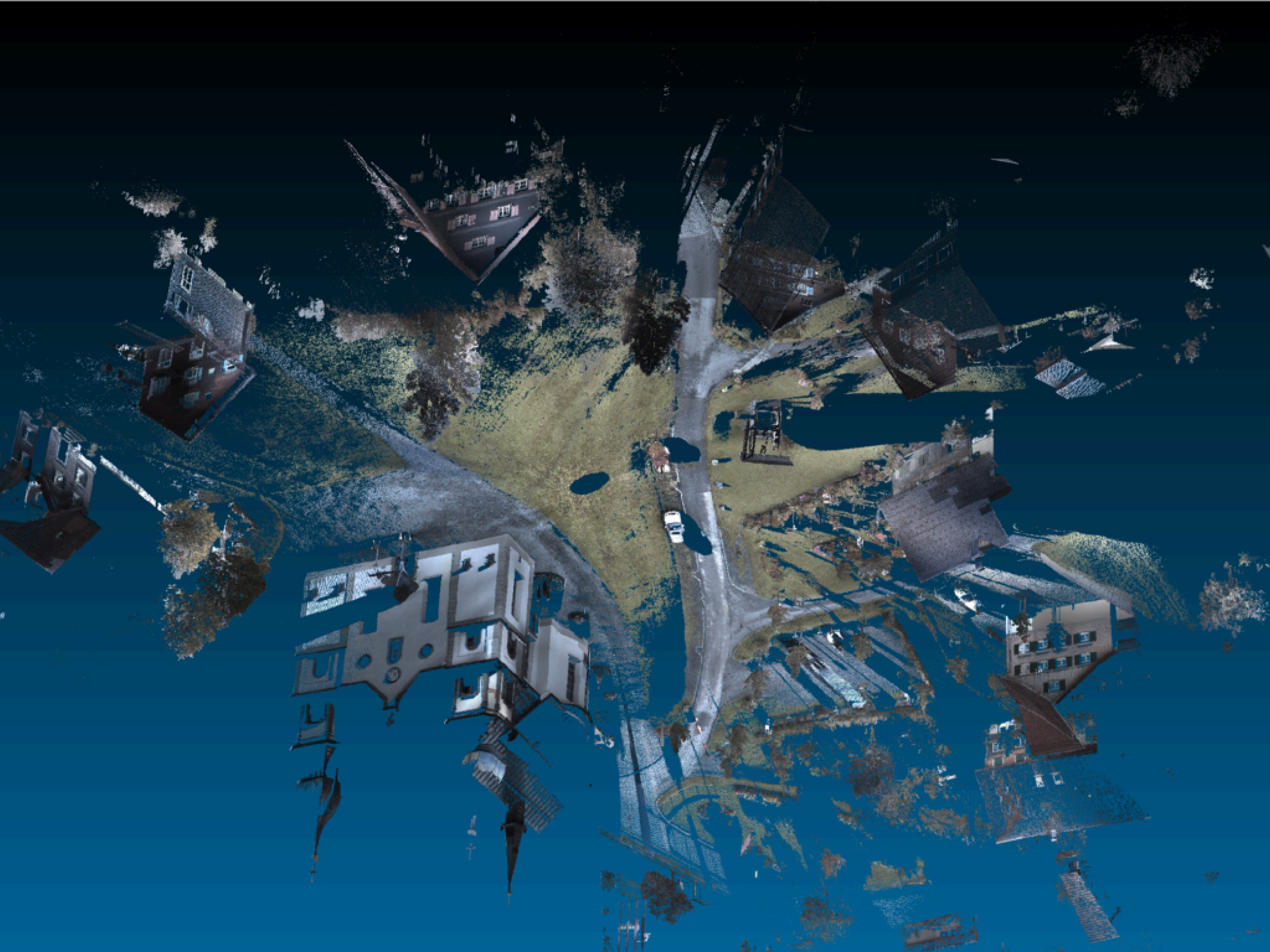
- Pointcloud data

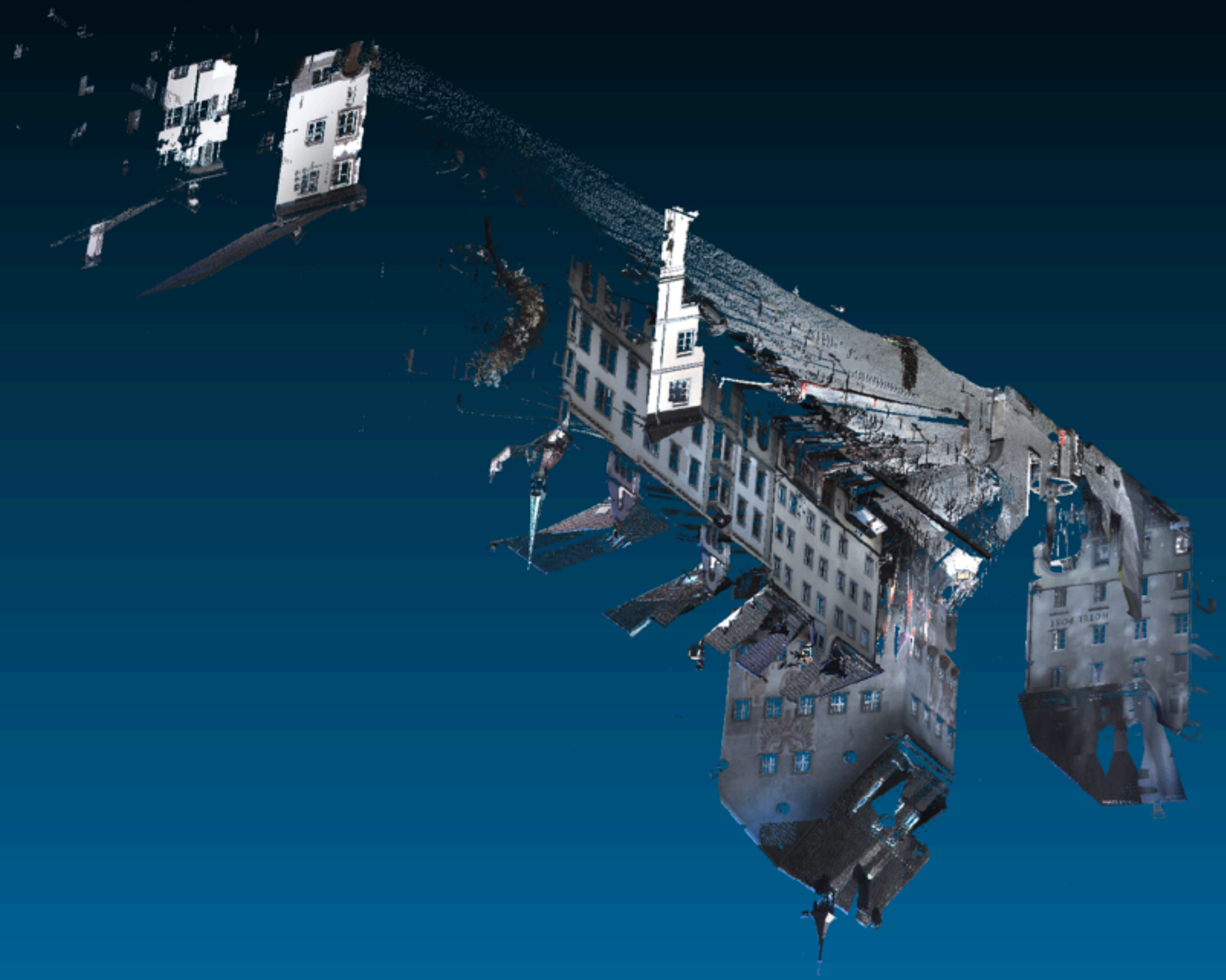
X	Y	Z	i	R	G	B

- Each record has 7 numerical attributes:
  - 3 float numerical format for XYZ coordinates
  - 1 integer for intensity
  - 3 integers for “Red”, “Green”, “Blue” (range 0-255)
  - No Nan-field detected.

Visualize Data

- CloudCompare V. 2.11.1 Application used for visualization





Model Selection

- Assessed Models:

1. PointNet

2. PointRCNN

3. RandLA-Net

Model Selection

	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

- Better Accuracy rate
- Better mIOU

Manual Labeling

- Tool Selection

Tools	Website	Explanation
CloudCompare	**http://www.cloudcompare.org/**	**Free with a great UI**
Amazon SageMaker	https://aws.amazon.com/sagemaker	Not accessible in Iran
Supervisely	https://supervise.ly/lidar-3d-cloud/	Payment
cloudfactory	https://info.cloudfactory.com/thankyou?submissionGuid=f167efb0-6dc3-4c12-9499-51103619dae3	Payment
MeshLab	https://www.meshlab.net/	Free but very slow
LATTE	https://github.com/bernwang/latte	Generating binary point cloud instead of 3D
labelCloud	https://github.com/ch-sa/labelCloud	Suitable for lightweight 3D point cloud
LIDAR PCD	https://github.com/DEEPI-LAB/LiDAR-Point-Cloud-Preprocessing-matlab	Very slow
ByteBridge	https://docsv2.bytebridge.io/#payment-refund	Payment

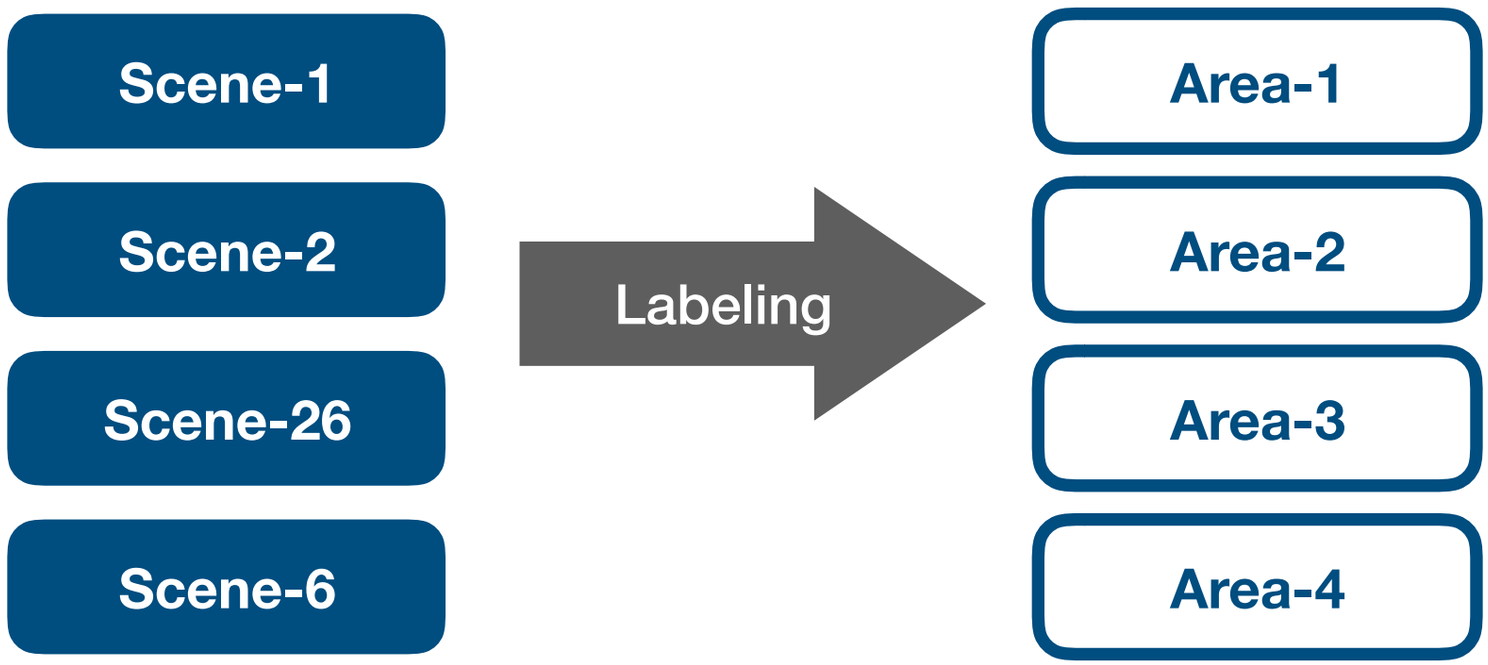
Manual Labeling

- Output: TXT file format

label1.txt

```
13.31000042 31.99399948 16.99200058 21 21 23 -1350.000000
13.27499962 31.99500084 17.00200081 24 24 26 -1349.000000
15.19799995 31.95499992 15.99899960 19 19 19 -1452.000000
15.16600037 31.93199921 15.99400043 19 19 19 -1381.000000
13.02600002 31.94700050 10.04100037 94 90 91 -1347.000000
13.10799980 31.87400055 9.79899979 106 105 103 -1268.000000
12.99100018 31.95299911 9.79399967 74 79 85 -1253.000000
12.99300003 31.91300011 9.79399967 75 82 90 -1368.000000
13.00500011 31.94300079 9.78400040 70 83 91 -1306.000000
13.00500011 31.98800087 9.78499985 100 101 106 -1290.000000
13.00899982 31.99799919 9.73299980 64 71 79 -1394.000000
13.00500011 31.98800087 9.69200039 69 74 80 -1376.000000
13.01000023 32.00000000 9.71399975 72 72 84 -1427.000000
13.11400032 31.97999954 9.74899960 108 104 105 -1468.000000
13.14299965 31.95899963 9.73200035 123 123 121 -1314.000000
13.18799973 31.93199921 9.71300030 54 61 71 -1332.000000
13.13799953 31.94899940 9.74699974 100 111 107 -1275.000000
13.16199970 31.96100044 9.72000027 112 118 114 -1288.000000
13.15600014 31.94700050 9.73400021 111 122 118 -1329.000000
13.17399979 31.94499969 9.72000027 86 97 93 -1391.000000
13.17199993 31.94000053 9.73700047 66 73 79 -1396.000000
13.12800026 31.96899986 9.73799992 121 119 122 -1316.000000
13.23700047 31.94599915 9.72299957 56 60 71 -1291.000000
13.23799992 31.95000076 9.70600033 56 60 71 -1403.000000
13.24100018 31.95499992 9.74499989 56 60 71 -1345.000000
13.04899979 31.95700073 9.76700020 95 93 96 -1376.000000
13.02099991 31.98200035 9.77799988 110 112 109 -1299.000000
```

Manual Labeling



Training Model

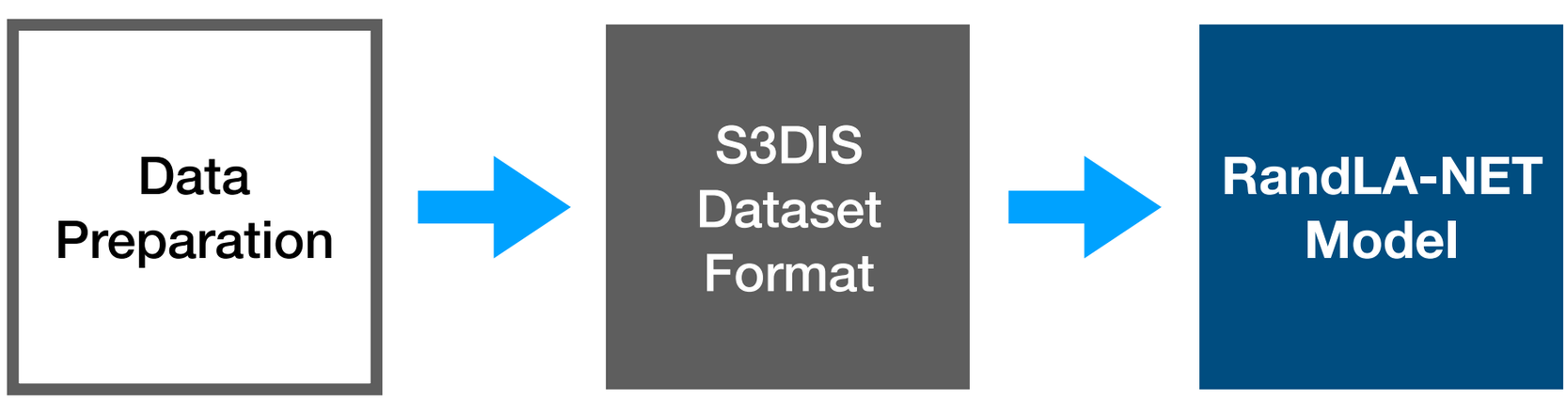
S3DIS Dataset format comprises
of 6-column point cloud
consisted of XYZ and RGB

Input Data Format

Training Model

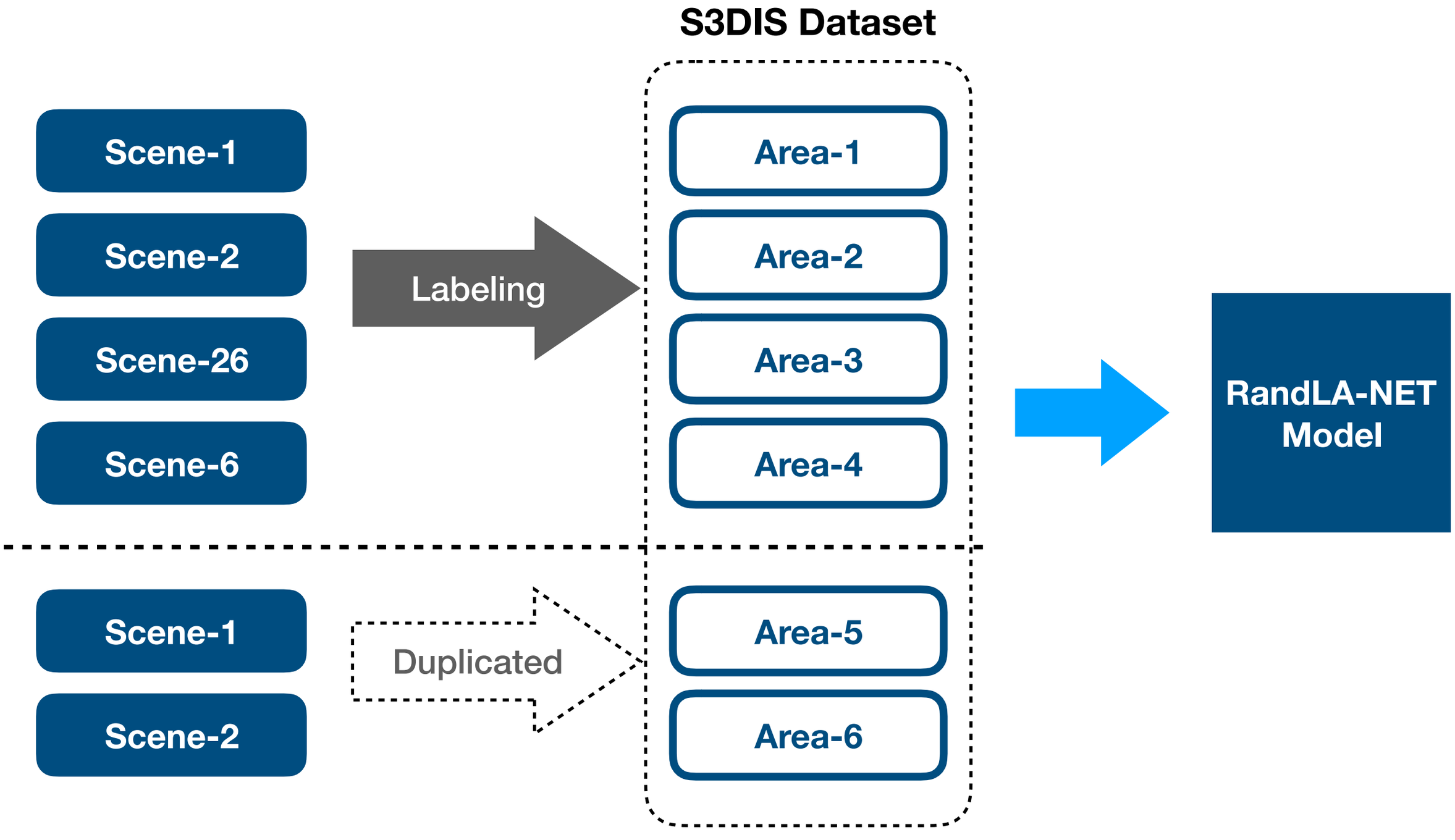
“intensity” feature
removed from dataset

S3DIS Dataset format comprises
of 6-column point cloud
consisted of XYZ and RGB



Input Data Format

Training Model



Model Performance

- IoU_{Class1}: 16.23
- IoU_{Class2}: 36.18
- IoU_{Class3}: 2.67
- IoU_{Class4}: 30.65
- **Mean IoU: 21.43**

Proposed Ideas for Future Research

Proposed Ideas (1)

1. CIoU vs. IoU

CIoU has better convergence rate in comparison to IoU

Generally, the IoU-based loss is defined as (Zheng et al., 2020):

$$\mathcal{L} = 1 - IoU + \mathcal{R}(B, B^{gt})$$

$$IoU = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|}$$

Proposed Ideas (1)

1. CIoU vs. IoU

The proposed approach applied an improved on CIoU based on Pseuso-Huber loss (Barron et al.,2019). Pseuso-Huber loss used in robust regression. It can be defined:

$$L_\delta(y) = \delta^2\left(\sqrt{1+\left(\frac{y}{\delta}\right)^2}-1\right)$$

Where y is difference between the target and predicted values and δ is an adjustable parameter. In this way, the penalty operates like squared error loss when the error is within [-δ,δ] but becomes linear outside this range (Barron et al., 2019). Therefore, we improved CIoU loss by replacing Pseudo-Huber formula instead of Euclidean distance:

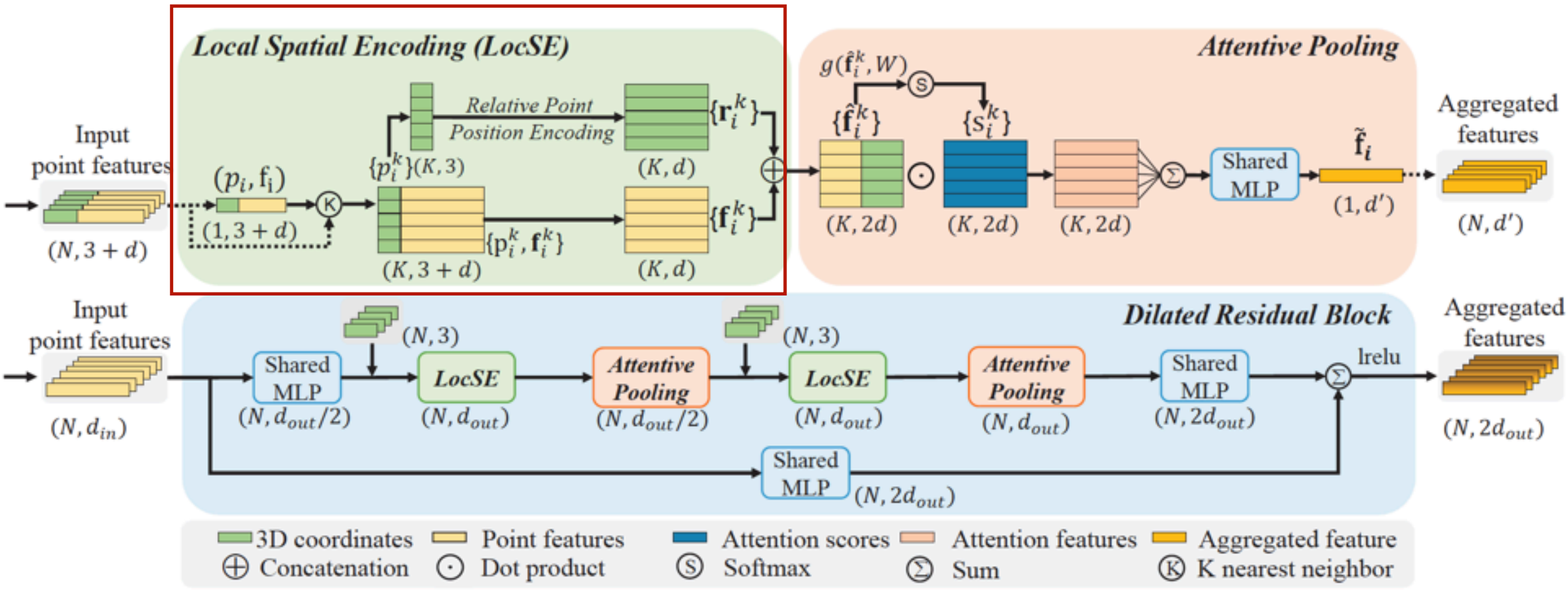
$$\mathcal{R}_{improved_{CIoU}} = \left(\frac{\delta^2}{c^2}\right)\left(\sqrt{1+\frac{\rho^2(b,b^{gt})}{\delta^2}}-1\right)+\alpha v$$

Finally, the proposed loss function can be written as:

$$L_{improved\_CIoU} = 1 - \mathrm{IoU} + \left(\frac{\delta^2}{c^2}\right)\left(\sqrt{1+\frac{\rho^2(b,b^{gt})}{\delta^2}}-1\right)+\alpha v$$

Proposed Ideas (2)

2. Optimize Model (Heuristics)



Proposed Ideas (2)

2. Optimize Model (Heuristics)

For Local Spatial Encoding (LocSE) Unit:

1. Define L-levels of neighbourhood
2. Select K/L neighbours form each level
3. Each level is K^L of nearest neighbour

It is expected that this heuristic method will leader to better segmentation performance of the mode.

Thank you