Point 4D Transformer Networks for Spatio-Temporal Modeling in Point Cloud Videos

SUPPLEMENTARY MATERIAL

1. Implementation Details

1.1. 3D Action Recognition

For 3D action recognition, videos usually contains many frames, *e.g.*, 16, 20 and 24 frames. Following previous works, we sample 2,048 points for each frame. To reduce the number of frames and points to be processed by the subsequent transformer, we set temporal stride s_t to 2 and spatial subsampling rate s_s to 32. In this way, for a video with 24 frames, the transformer processes $\frac{2048}{32} \times \frac{24}{2} = 768$ points. The temporal radius r_t is set to 1 to capture temporal local structure. The spatial radius r_s is set to 0.5 for MSR-Action3D and 0.1 for NTU RGB+D 60/120. The transformer contains 2 self-attention ($m = 5$) blocks, with 8 heads ($h = 8$) per block. We train our models for 50 epochs with the SGD optimizer. Batch size is set to 16. Learning rate is set to 0.01, and decays with a rate of 0.1 at the 20th and and 30th epochs, respectively.

1.2. 4D Semantic Segmentation

For 4D Semantic Segmentation, each video clip only contains 3 frames while each frame contains 16,384 points. Due to the high resolution, we stack four 4D point convolution layers, with spatial subsampling rate (s_s) of 4, 4, 4 and 2, to considerably merge points. The spatial radius r_s progressively increases as 0.9, 1.8, 2.7 and 3.6, respectively. However, given the short video clip length, we do not subsample frames and thus set temporal stride s_t to 1. The temporal radius r_t is set to 1 for the 3rd 4D point convolution layer and 0 for other convolution layers. In this way, the transformer takes $\frac{16384}{4\times4\times4\times2} \times 3 = 384$ points as input. The transformer contains 2 self-attention ($m = 2$) blocks, with 4 heads ($h = 4$) per block. We train our models for 75 epochs with the SGD optimizer. Batch size is set to 8. Learning rate is set to 0.01, and decays with a rate of 0.1 at the 30th, 40th and 50th epochs, respectively.

2. Impact of Temporal Modeling on 4D Semantic Segmentation

In our P4Transformer architecture for 4D semantic segmentation, we set the temporal radius r_t of the 3rd point 4D convolution layer to 1 to capture the temporal dependency. Then, the transformer is employed to capture the spatio-temporal structure. In this way, our P4Transformer captures the temporal correlation.

Table 1. Influence of temporal modeling on 4D semantic segmentation.

Temporal radius r_t	Transformer	Accuracy (mIoU%)
0	✗	81.87
1	✗	82.26
0	✓	82.75
1	✓	83.16

We investigate the influence of these two temporal modeling techniques on 4D semantic segmentation. As shown in Table 1, temporal modeling effectively improves the accuracy of 4D semantic segmentation.

3. 4D Semantic Segmentation Visualization

We visualize a few 4D semantic segmentation examples in Fig. 1. We also compare our method with the second best method, *i.e.*, MeteorNet, by visualization in Fig. 2. In most instances, both MeteorNet and our P4Transformer achieve satisfied results. However, probably, due to illumination, MeteorNet makes some incorrect predictions in the examples, while our method avoids these errors by better exploiting spatio-temporal structure.

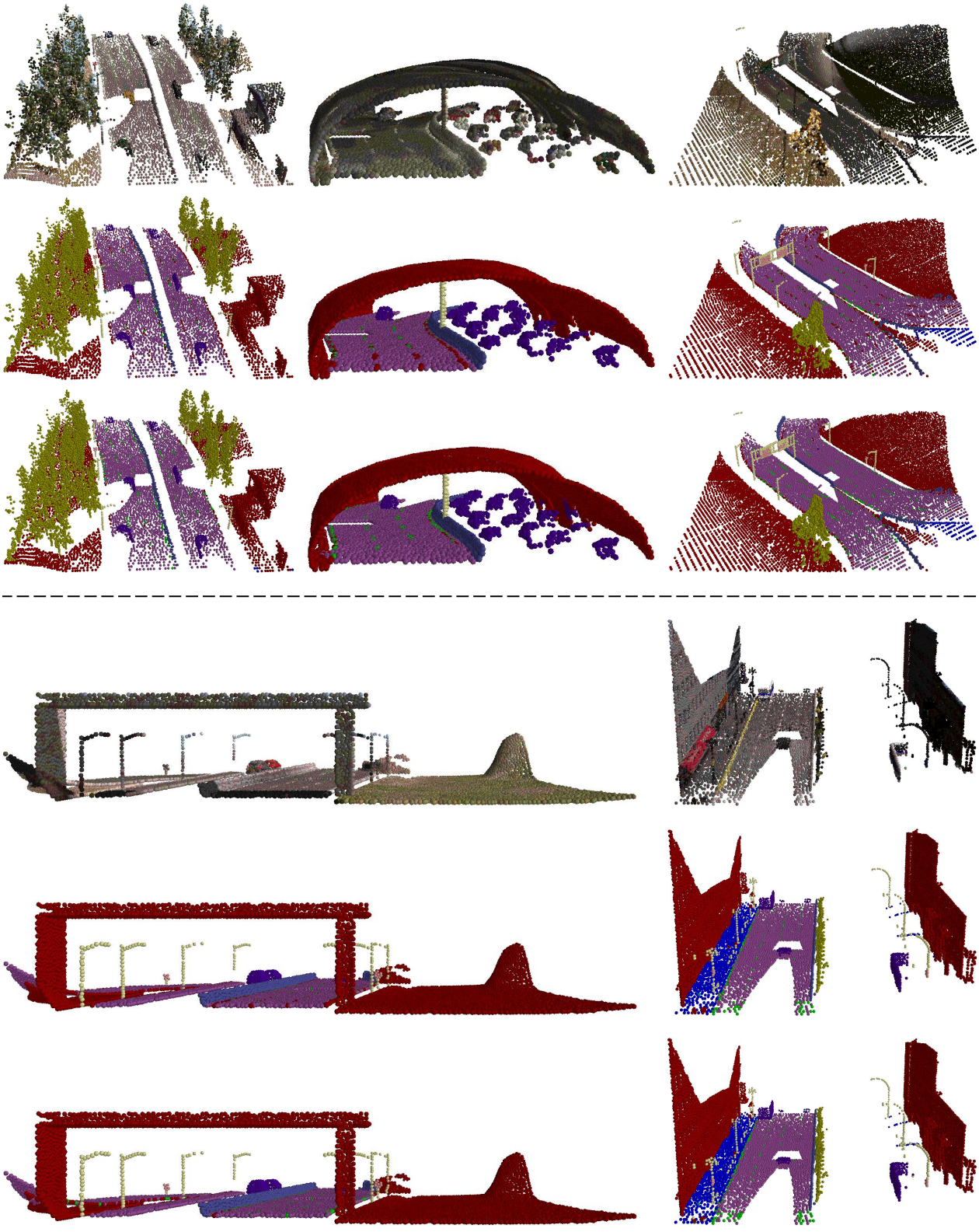

Figure 1. Visualization of 4D semantic segmentation. In each part, top: inputs; middle: ground truth; bottom: P4Transformer predictions.

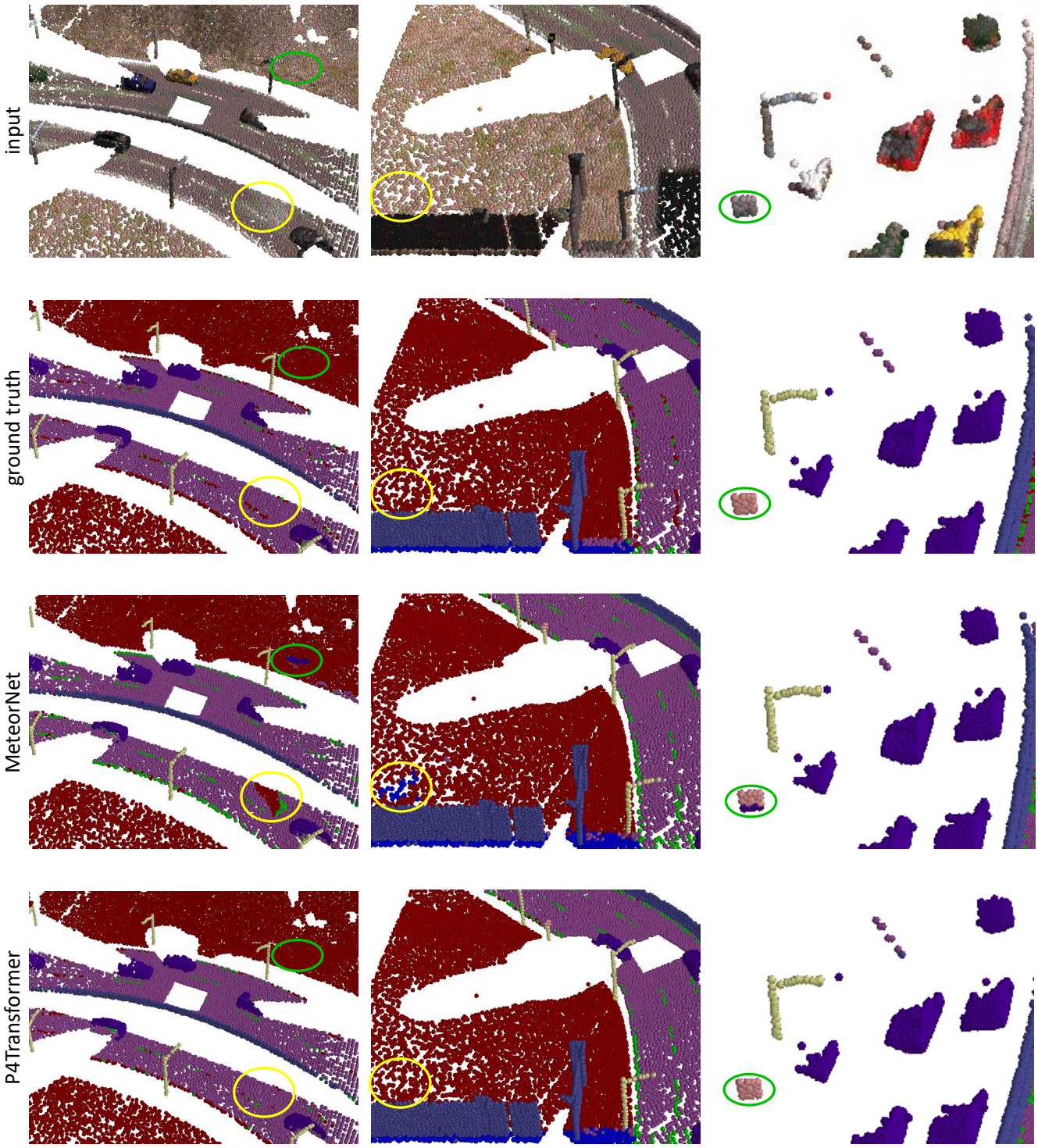

Figure 2. Qualitative comparison of 4D semantic segmentation between MeteorNet and our P4Transformer.