ASAP-Net: Attention and Structure Aware Point Cloud Sequence Segmentation

Hanwen Cao *[1]
mbd_chw@sjtu.edu.cn
Yongyi Lu †[2]
yylu1989@gmail.com
Cewu Lu[1]
lucewu@sjtu.edu.cn
Bo Pang[1]
pangbo@sjtu.edu.cn
Gongshen Liu[1]
lgshen@sjtu.edu.cn
Alan Yuille[2]
alan.l.yuille@gmail.com

[1] Shanghai Jiaotong University
Shanghai, China

[2] Johns Hopkins University
Baltimore, MD, USA

* Work done at Johns Hopkins Univ.
†corresponding author

Abstract

Recent works of point clouds show that mulit-frame spatio-temporal modeling outperforms single-frame versions by utilizing cross-frame information. In this paper, we further improve spatio-temporal point cloud feature learning with a flexible module called ASAP considering both attention and structure information across frames, which we find as two important factors for successful segmentation in dynamic point clouds. Firstly, our ASAP module contains a novel attentive temporal embedding layer to fuse the relatively informative local features across frames in a recurrent fashion. Secondly, an efficient spatio-temporal correlation method is proposed to exploit more local structure for embedding, meanwhile enforcing temporal consistency and reducing computation complexity. Finally, we show the generalization ability of the proposed ASAP module with different backbone networks for point cloud sequence segmentation. Our ASAP-Net (backbone plus ASAP module) outperforms baselines and previous methods on both Synthia and SemanticKITTI datasets (+**3.4** to +**15.2** mIoU points with different backbones). Code is availabe at https://github.com/intrepidChw/ASAP-Net

1 Introduction

Dynamic point cloud sequences are readily-available input sources for many vision tasks. The ability to segment dynamic point clouds is a fundamental part of the perception system and will have a significant impact on applications such as autonomous driving [1], robot navigation [28] and augmented reality [16].

While great success has been achieved in static point clouds [8, 23, 24, 31, 33], the literature on modeling point cloud sequence has not been fully-explored. For static point

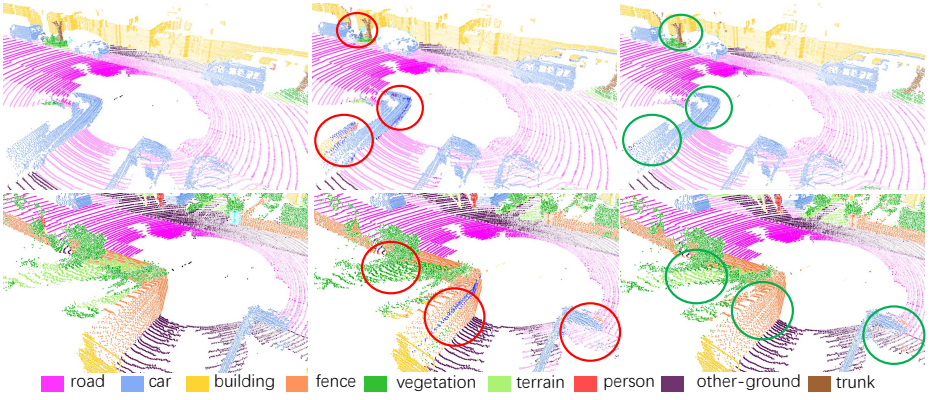

Figure 1: Segmentation results on SemanticKITTI [3]. From left to right: groundtruth, SqueezeSegV2 [38] before and after incorporating our ASAP module. RED circles highlight the failure cases by SqueezeSegV2 and GREEN circles highlight the successful cases. Backbone's performance gets improved after incorporating our ASAP module.

clouds, the pioneer work PointNet [23] has shown its capability of directly learning point-wise features from the raw point clouds. However, a single point does not have semantic meaning until it is considered with its neighbouring points. Subsequent works on static point clouds [8, 24, 31] encode local structures from set of neighboring points and capture interactions among local structures. For modeling dynamic point cloud sequence, an intuitive and straightforward way is to encode the local structures both spatially and temporally, *i.e.*, within a spatio-temporal neighborhood. Latest works [5, 19] follow this direction. 4D MinkNet [5] first transforms the point cloud into regular voxels and applies a generalized sparse convolution along spatial and temporal dimensions. MeteorNet [19] directly stacks multi-frame point clouds and computes local features by grouping spatio-temporal neighboring points. Although both methods are capable of capturing spatio-temporal features, it is hard for them to decouple spatial and temporal information in sequences [21, 22]. There is evidence that in real world, the human visual system largely relies on temporal information [20] and treats it as a primary source of information. Without decoupling spatial and temporal structures, it is not well understood to which degree temporal information can contribute to dynamic scene understanding [7]. Therefore, it will be beneficial to related research if we can decouple the two aspects of information. Also, decoupling spatial and temporal structures can make the model more extensible and flexible, *i.e.*, we can easily apply the static point cloud methods as backbone networks, which can be seen in Section 3.1 and Section 4.2.

To effectively encode temporal structure, we need to tackle the following challenges:

(a) **Feature fusion with different frames**: The features from different frames may contribute differently to the results and they are all likely to contain undesired noise or mistakes. Ideally, the network should automatically identify the importance or confidence degree of different frames to achieve better fusing results.

(b) **Point correlation across frames**: To fuse features from different frames, we need to correlate points across frames. However, the distribution of dynamic points varies from time to time and they are unordered, making it challenging to correlate.

In our ASAP module, we propose a novel attentive temporal embedding layer and spatio-temporal correlation strategy to tackle the above two challenges respectively. We also conduct thorough experiments to show our ASAP module's advantage over state-of-the-art methods [5, 19] and its generalization ability of improving the performance of different backbones. Figure 1 provides qualitative results of our approach, showing the effectiveness of our ASAP module. Our key contributions are:

- We run a pilot study towards segmentation in point cloud sequences, by proposing a flexible architecture called ASAP module which can be easily plugged into previous static point cloud pipeline and achieve improvement by a large margin.
- We introduce a novel attentive temporal embedding layer to fuse the spatial local features across frames effectively and robustly by automatically calculating attentions.
- We present a spatio-temporal correlation strategy to exploit structural information, enforce temporal consistency and reduce computation.

2 Related Work

2.1 Semantic Segmentation in Static Point Clouds

Semantic segmentation in point clouds is a long-standing topic. Deep neural networks designed for the task can be divided into four categories: a) **Voxelization based methods**. These methods transform the point cloud into regular voxel grids and apply 3D convolution network to do semantic segmentation [12]. However, the sparsity of point clouds can cause exploding memory consumption problem. More advanced methods [9, 25, 31, 36] are proposed to tackle the problem. b) **Range image based methods**. These methods [4, 37, 38] first project the point cloud onto sphere according to the azimuth and zenith angles and then apply traditional convolutional networks like SqueezeNet [13] and U-Net [26]. c) **Direct point cloud methods.** These methods directly take point clouds as input. [23, 24] extract point-wise features using multi-layer perceptron and symmetric function (max pooling) to aggregate features. [14] proposes an orientation-encoding unit to improve PointNet-based architectures' representation ability. [8] proposes new grouping techniques to better define neighborhoods for PointNet-based feature network. [10, 11, 30] propose new convolution operators to directly deal with point clouds. d) **Other representation based methods**. [15, 39] use kd-trees to represent point clouds. [17, 32, 35] are graph-based. [29] interpolates input features onto a permutohedral lattice then does convolution.

2.2 Temporal Information in Point Cloud Sequences

To deal with 3D-videos (a continuous sequence of depth images, or LiDAR scans), [5] adopts sparse tensors and proposes the generalized sparse convolution. [19] is based on PointNet++ [24]. It first stacks different frames of point clouds and get neighbor points in different frames by either searching with a time-dependent radius or tracking the points.

Other methods include scene flow and motion estimation. [6] formulates the problem as an energy minimization problem of a factor graph, with hand-crafted SHOT [33] descriptors for correspondence search. [34] uses EM algorithm to estimate a locally rigid and non-deforming flow. [18] is based on PointNet [23] and PointNet++ [24] and proposes a new flow embedding layer and set upconv layer to estimate scene flow in point clouds in an end-to-end way. [2] estimates scene flow as rigid motions of individual objects or background with

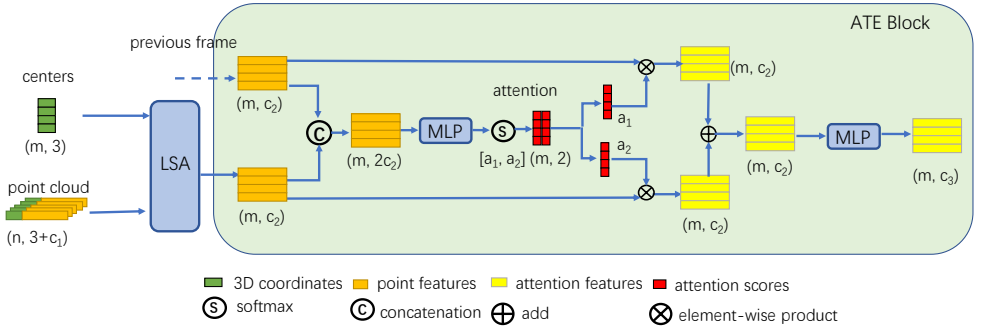

Figure 2: The proposed ASAP module. Given the current point cloud and corresponding center points, we first use the LSA block to compute the center features. We then fuse the center features from consecutive frames with the ATE block. The correspondence between the two sets of center features are guaranteed by our spatio-temporal correlation strategy.

network that jointly learns to regress ego-motion and detect 3D objects. All these methods focus on different tasks and can not be directly applied to the task of semantic segmentaion.

3 ASAP-Net

In this section, we will present our ASAP module and the overall architecture.

3.1 ASAP module

The architecture of our ASAP module is illustrated in Figure 2. To illustrate our module, we denote a point cloud sequence of length T as $\{S_1, S_2, ..., S_T\}$. Each frame of point cloud is represented as $S_t = \{p_i^{(t)}\}_{i=1}^{n}$, where n is the number of points and each point $p_i^{(t)}$ consists of its Euclidean coordinate $x_i^{(t)} \in \mathbb{R}^3$ and feature $f_i^{(t)} \in \mathbb{R}^c$. We also denote a set of center points in S_t as $\{c_j^{(t)}\}_{j=1}^{m}$, where m is the number of center points and $m < n$.

3.1.1 Local Structure Aggregation (LSA)

Given a single frame of point cloud represented as $\{p_i^{(t)}\}_{i=1}^{n}$ and corresponding center points $\{c_j^{(t)}\}_{j=1}^{m}$, we follow PointNet++ [24] to extract local features. For each center point, we gather its neighbor points within a radius, denoted as $\mathcal{N}(c_j^{(t)})$, and compute its feature with the following symmetric function:

$$f_j^{(t)} = \underset{p_i^{(t)} \in \mathcal{N}(c_j^{(t)})}{MAX} \{\eta(f_i^{(t)}, x_i^{(t)} - x_j^{(t)})\}. \quad (1)$$

where η is an MLP (multi-layer perceptron) with concatenated feature vector $f_i^{(t)}$ and difference of spatial positions $x_i^{(t)} - x_j^{(t)}$ as inputs. MAX is element-wise max pooling.

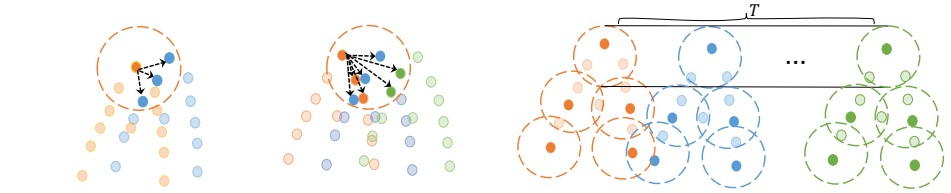

Figure 3: Illustration of the proposed spatio-temporal correlation. Different from (a) two-frame direct grouping [18] and (b) multi-frame direct grouping [19], each center (the solid points inside the cycles in (c)) in our spatio-temporal correlation are correlated across T frames either by (i) per-frame nearest center search or (ii) inter-frame constant center iterate, followed by the temporal embedding layer. Compared to (a) and (b), our tubular-alike correlation is able to decouple spatial and temporal structures.

3.1.2 Spatio-temporal Correlation (STC)

Recall that we need to tackle two main challenges in dynamic point clouds: (a) Feature fusion with different frames and (b) point correlation across frames. Here we first introduce (b) how to correlate points between frames and leave (a) in the next section. Specifically, given two consecutive point clouds, we need to obtain pairwise correspondence between them to leverage temporal information. Latest methods like [19] use either direct grouping of neighboring points or flow-based grouping. The former is incapable of knowing exactly the correspondences across frames, while the latter are computationally expensive. In the followings we present two simple yet efficient correlation approaches:

(i) **Per-frame Nearest Center Search**. Use farthest point sampling to generate center points each frame and compute local features separately. For each center in the current frame, we correlate it with the closest center in the previous frame based on Euclidean distance.

(ii) **Inter-frame Constant Center Iteration**. Only generate the center coordinates in the first frame using farthest point sampling and use the same center coordinates in subsequent frames so the center features are naturally correlated.

Our insight to design (ii) is that, to capture temporal information effectively, the network should have stable focus regions to achieve temporal consistency. It may seem uncommon to use the same center coordinates since they don't exist in frames except the first one. However, their neighbor regions do exist. Although the points are dynamic, their boundary is actually stable (defined by the intrinsic parameters of LiDAR scanner or depth camera). Therefore, there's actually no worry that some center points get too far away and no longer have neighbor points due to the dynamics the point cloud. Each center point forms a spatio-temporal tube in time (see Figure 3) and by computing the center features in a recurrent way, we can capture temporal information and adapt to the neighborhood changes.

We adopt (ii) as our final strategy. It can be seen that (i) needs to do sampling per frame and compute the distances while (ii) is more computationally efficient. We will show the advantages of our spatio-temporal correlation strategy with experiments in Section 4.5.1.

3.1.3 Temporal Embedding (TE)

The temporal embedding layer is used to fuse local structure features across frames. The layer will take in the current center points $\{c_j^{(t)}\}_{j=1}^m$ and the matched previous ones $\{c_j^{(t-1)}\}_{j=1}^m$

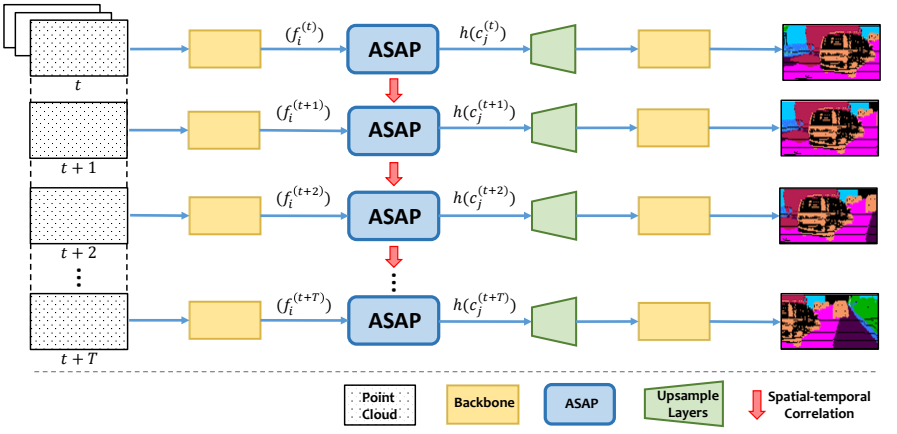

Figure 4: Overall network structure. Our framework consists of the backbone network and the proposed ASAP module and upsampling layers. We also use our spatio-temporal correlation strategy to guarantee the correlation between consecutive frames.

after spatio-temporal correlation and update the feature of each current center point which we denote as $h(c_j^{(t)})$. We also propose two temporal embedding methods:

(i) **Direct Temporal Embedding (DTE)**: We directly concatenate the features of the matched center points and use a shared MLP ζ to update the feature of current center point: $h(c_j^{(t)}) = \zeta(f_j^{(t-1)}, f_j^{(t)})$.

(ii) **Attentive Temporal Embedding (ATE)**: We compute two scalar attentions by feeding the concatenation of two correlated features into a shared MLP γ followed by a $Softmax$ function: $[a_1, a_2] = Softmax(\gamma(f_j^{(t-1)}, f_j^{(t)}))$. Then we calculate an weighted-sum feature by multiply the two attentions with corresponding features: $f_j^{'(t)} = a_1 f_j^{(t-1)} + a_2 f_j^{(t)}$. Finally we get the updated feature of current center point with another MLP ζ: $h(c_j^{(t)}) = \zeta(f_j^{'(t)})$.

Our insight to design the ATE block is that different frames may contribute differently to the results. For slowly-moving objects, features from previous frame can be more reliable since they contain multi-frame information due to the recurrent computation while for fast-moving objects, it's better to rely more on current frame. Also, there can be undesired mistakes and noise in each frame. Therefore, the network can make better and more robust predictions by estimating the importance and reliability of features from different frames. ATE can achieve better results with even less parameters, which will be shown in Section 4.

The LSA and TE can be stacked hierarchically. Specifically, we call the module with DTE as SAP-x (no attention) and the module with ATE as ASAP-x (with attention), where x is the number of LSA and TE.

3.2 Overall Structure

The overall structure is shown in Figure 4. We first use a backbone network like [24, 38] to extract point features in each frame. Then our ASAP module will compute the center features and fuse across frames in a recurrent way. Next, we use the *feature propagation* layer proposed by [24] to upsample the point cloud to original size and return it to the backbone network. Finally, the backbone network will continue to run and return the predictions.

According to our spatio-temporal correlation strategy, for each sequence, we sample center points with farthest point sampling in the first frame and use the same coordinates in subsequent frames to achieve temporal consistency and reduce computation.

4 Experiments

4.1 Datasets and Settings

We conduct semantic segmentation experiments on two datasets. We first test our method on the large-scale synthetic dataset Synthia [27], compare with meteornet [19] and a sparse 4D CNN [5] baselines and do ablation study. We then conduct experiments on the largest publicly available real LiDAR dataset semanticKITTI [3] to show our module's effectiveness and generalization ability.

Synthia It consists of six sequences of driving scenarios in nine different weather conditions. Each sequence consists of four stereo RGBD images from four viewpoints captured from the top of a moving car. We reconstruct 3D point clouds from RGBD images and create point cloud sequences the same way with [19]. We used the same train/validation/test split as [5, 19]. The train, validation and test set contain 19,888, 815 and 1,886 frames respectively.

SemanticKITTI The dataset provides 23,201 full 3D scans for training and 20,351 for testing, which makes it by a wide margin the largest dataset publicly available. We follow the split in [3] for training, validation and testing. To compare with the backbone network, we follow the single scan experiments proposed in [3] and make predictions for the 19 classes. Since our model take a sequence as input, we split the testing set into sequences with a fixed length and save the result of each scan.

4.2 Choice of Backbone Networks

Our proposed ASAP module is a flexible architecture and can be easily incorporated into different backbones. We call the overall network ASAP-Net (Backbone + ASAP module).

To show the flexibility of our ASAP module, the backbones should be different in nature. Therefore, PointNet++[24] and SqueezeSegV2 [38] are chosen for the simplicity of architecture and representativeness in two different categories: 1) directly processing and 2) range image based, as per-frame point cloud methods mainly fall into these two groups. For PointNet++ [24], our ASAP module processes the point cloud in the same way so it can be incorporated directly; For SqueezeSegV2 [38], the network first projects the LiDAR point cloud onto a sphere for a dense, grid-based representation as $\theta = \arcsin \frac{z}{\sqrt{x^2+y^2+z^2}}, \tilde{\theta} = \lfloor \theta / \Delta\theta \rfloor$ and $\phi = \arcsin \frac{y}{\sqrt{x^2+y^2}}, \tilde{\phi} = \lfloor \phi / \Delta\theta \rfloor$. Here $\Delta\theta$ and $\Delta\phi$ are resolutions for discretization and $(\tilde{\theta}, \tilde{\phi})$ denotes the position of a point on a 2D spherical grid. After applying these equations on each point cloud, we can obtain a 3D tensor of size $H \times W \times C$. Then they adopt an encoder-decoder structure using CNN module called *fireModule* and *fireDeconvs*, which are introduced in SqueezeNet [13].

To reconstruct the point cloud from 2D spherical grid, we simply reshape the feature map of the middle layer from $H \times W \times C$ to $N \times C$, where $N = H \times W$ is the number of points and C is the feature length. We use max pooling layer on the original projected spherical grid (the channel contains coordinates) to get the corresponding 3D coordinates approximately.

Approach	param (M)	frame num	mIoU	mAcc	IoU Bldg	Road	Sdwlk	Fence	Vegitn	Pole	Car	T.Sign	Pdstr	Bicyc	Lane	T.light
3D MinkNet [6]	19.31	1	76.24	**89.31**	89.39	97.68	69.43	86.52	98.11	97.26	93.50	79.45	92.27	0.00	44.61	66.69
4D MinkNet [6]	23.72	3	77.46	88.01	90.13	**98.26**	73.47	87.19	**99.10**	97.50	94.01	79.04	**92.62**	0.00	50.01	68.14
PointNet++ [24]	0.88	1	79.35	85.43	96.88	97.72	86.20	92.75	97.12	97.09	90.85	66.87	78.64	0.00	72.93	75.17
MeteorNet [19]	1.78	3	81.80	86.78	**98.10**	97.72	88.65	94.00	97.98	**97.65**	93.83	**84.07**	80.90	0.00	71.14	77.60
PNv2 with SAP-1	1.97	3	82.31	86.72	97.68	97.99	**90.16**	94.84	97.25	97.34	94.77	80.35	83.54	0.00	**75.13**	**78.62**
PNv2 with ASAP-1	1.84	3	**82.73**	87.02	97.67	98.15	89.85	**95.50**	97.12	97.59	**94.90**	80.97	86.08	0.00	74.66	77.51

Table 1: Segmentation results on the Synthia dataset [27]. The evaluation metrics are mean IoU and mean accuracy (%). PNv2 represents PointNet++ [24]. SAP-x and ASAP-x are defined in Section 3.1.3.

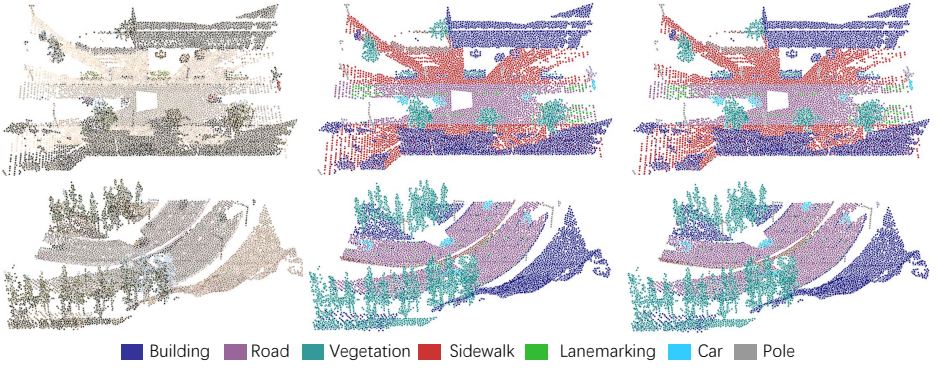

Figure 5: Visualization of two example results from the Synthia dataset. From left to right: RGB input, groundtruth, predictions.

Note that, Although as claimed in [3], *DarkNet53Seg* achieves the best mIoU result on SemanticKITTI [3], it's not a published work. According to the author, it's the same type with SqueezeSegV2 [33] and it performs better just because it has more layers. Therefore, improvement on SqueezeSegV2 [33] should guarantee the improvement on DarkNet53Seg.

4.3 Evaluation on Synthia

The results are listed in Table 1, our ASAP-Net consistently surpasses both the sparse 4D CNN [6] and MeteorNet [19] and establishes a new state-of-art result in term of mIoU. Figure 5 visualizes two samples. Our model can accurately segment most objects.

We can see that our ASAP module can achieve better results with even less parameters than SAP module. The reduction of parameters has two aspects of reasons. Firstly, the extra number of parameters γ to compute attention is really small since the output attention length is just two. Secondly, the features from two consecutive frames are fused by addition instead of concatenation so the input feature length to ζ is half of that in SAP.

4.4 Evaluation on SemanticKITTI

The IoU results of the backbone networks as well as the results with our ASAP-Net are shown in Table 2. The performances of both backbone networks get improved when combined with our ASAP-module. Some quantitative visualizations can be found in Figure 1.

Approach	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
PointNet [24]	14.6	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7
PointNet++ [24]	20.1	53.7	1.9	0.2	0.9	0.2	0.9	1.0	0.0	72.0	18.7	41.8	5.6	62.3	16.9	46.5	13.8	20.0	6.0	8.9
PNv2 with SAP-1	31.4	79.7	9.5	5.5	0.1	8.1	7.9	22.3	1.1	81.2	**34.2**	56.3	7.9	75.7	38.5	58.8	26.8	50.1	16.1	16.9
PNv2 with ASAP-1	33.3	84.1	**11.6**	**7.5**	3.2	11.4	7.8	18.5	3.0	**81.8**	28.1	53.1	7.8	74.9	37.6	64.4	27.2	51.7	22.8	**30.8**
PNv2 with ASAP-2	**35.3**	**86.4**	9.3	6.4	**8.1**	**13.0**	**12.8**	**25.2**	**3.8**	80.3	29.7	**57.6**	**13.2**	**77.7**	**40.1**	**66.7**	**31.9**	**52.9**	**26.7**	28.5
SqueezeSeg [34]	29.5	68.8	16.0	4.1	3.3	3.6	12.9	13.1	0.9	85.4	26.9	54.3	4.5	57.4	29.0	60.0	24.3	53.7	17.5	24.5
SqueezeSegV2 [33]	39.7	81.8	18.5	17.9	13.4	14.0	20.1	25.1	3.9	88.6	45.8	67.6	17.7	73.7	41.1	71.8	35.8	60.2	20.2	36.3
SSv2 with SAP-1	40.7	83.0	21.1	**22.4**	4.1	15.5	25.7	**33.9**	1.3	89.5	52.9	68.9	19.2	73.7	40.1	69.2	36.1	61.4	14.0	**40.9**
SSv2 with ASAP-1	41.3	83.3	19.8	21.1	9.2	15.5	24.4	31.3	2.5	89.7	53.3	69.0	18.1	79.3	45.0	71.7	**38.3**	61.6	14.0	37.1
SSv2 ASAP-1-msg	42.5	**84.0**	19.8	22.3	13.1	17.2	24.0	31.9	**5.5**	90.4	**55.5**	**70.9**	**22.2**	80.5	46.2	71.5	37.0	63.1	15.3	36.5
SSv2 with ASAP-2	**43.1**	83.1	**22.5**	20.8	**16.0**	**18.1**	**26.6**	32.8	4.6	**90.5**	55.2	69.8	20.4	**81.7**	**49.0**	**72.0**	**38.3**	**63.2**	**16.4**	38.4
SPGraph [15]	17.4	49.3	0.2	0.2	0.1	0.8	0.3	2.7	0.1	45.0	0.6	39.1	0.6	64.3	20.8	48.9	27.2	24.6	15.9	0.8
SPLATNet [25]	18.4	58.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	64.6	0.4	39.1	0.0	58.3	23.1	71.1	9.9	19.3	5.6	0.0
TangentConv [30]	40.9	**90.8**	2.7	16.5	15.2	12.1	23.0	28.4	**8.1**	83.9	33.4	63.9	15.4	83.4	49.0	**79.5**	49.3	58.1	35.8	28.5
DarkNet21Seg [3]	47.4	85.4	**26.2**	26.5	18.6	15.6	31.8	**33.6**	4.0	91.4	57.0	74.0	26.4	81.9	52.3	77.6	48.4	63.6	24.6	50.0
DarkNet53Seg [3]	**49.9**	86.4	24.5	**32.7**	**25.5**	**22.6**	**36.2**	**33.6**	4.7	**91.8**	**64.8**	**74.6**	**27.9**	**84.1**	**55.0**	78.3	**50.1**	**64.0**	**38.9**	**52.2**

Table 2: Segmentation results of backbone networks and backbones combined with ASAP module on the Semantic KITTI dataset [3]. Metric is mean IoU. PNv2 represents PointNet++ [24] and SSv2 represents SqueezeSegV2 [33]. SAP-x and ASAP-x are defined in Section 3.1.3. All methods were trained on sequences 00 to 10, except for sequence 08 which is used as validation set. We also list the results of other published works to give the readers an overview of the performances on the dataset.

4.5 Model Analysis

4.5.1 Ablation on Spatio-temporal Correlation Strategy

In this section, we do ablation analysis on the two spatio-temporal correlation strategies and input sequence length. Without loss of generality, we use PointNet++ [24] with ASAP-1.

The results are shown in Table 3. As we can see, STC (i) is worse. We suppose it's because the focal regions (regions within a radius of center points) keep changing in different frames, making it hard to achieve temporal consistency. We can also see that the results get improved as the sequence length increases, which shows the effectiveness of our ASAP module to utilize temporal information.

Approach	param (M)	frame num	mIoU	mAcc	IoU Bldg	Road	Sdwlk	Fence	Vegitn	Pole	Car	T.Sign	Pdstr	Bicyc	Lane	T.light
PointNet++ [24]	0.88	1	79.35	85.43	96.88	97.72	86.20	92.75	97.12	97.09	90.85	66.87	78.64	0.00	72.93	75.17
PNv2 with STC (i)	1.84	3	81.78	86.47	97.18	97.73	89.91	94.46	96.61	97.01	94.25	78.28	83.74	0.00	74.69	**77.69**
PNv2 with STC (ii)	1.84	3	82.73	87.02	97.67	98.15	89.85	95.50	97.12	97.59	94.90	80.97	86.08	0.00	74.66	77.51
PNv2 with STC (ii)	1.84	4	82.82	86.71	**98.01**	98.33	92.14	**95.54**	99.12	**97.69**	95.65	81.62	84.84	0.00	74.91	76.04
PNv2 with STC (ii)	1.84	5	82.90	**87.14**	97.90	**98.36**	92.05	95.43	**99.16**	97.51	95.21	**82.27**	84.03	0.00	75.69	77.15
PNv2 with STC (ii)	1.84	6	**83.00**	87.00	97.63	98.34	**92.49**	94.85	97.38	97.53	**95.76**	80.65	**87.48**	0.00	**77.07**	76.81

Table 3: Segmentation results on the Synthia dataset [27]. The evaluation metrics are mean IoU and mean accuracy (%). PNv2 represents PointNet++ [24].

4.5.2 Improvements on Different Backbones

As we can see in Table 2, the improvement of PointNet++ [24] is much larger than that of SqueezeSegV2 [33]. We assume that's because our ASAP module architecture is similar

with PointNet++ [24] but totally different from SqueezeSegV2 [38]. SqueezeSegV2 [38] converts the point cloud to a 2D image. To combine it with our ASAP module, we have to first reconstruct point cloud from the image and then project the point cloud back to 2D. The process can cause information loss. For example, according to Section 4.2, we use a max pooling layer to reconstruct the point cloud so the coordinates of output points can come from different points. In other words, those output points may not exist in the original point cloud at all, thus reducing the performance to some extent.

4.5.3 Failed Cases

As we can see in Table 2, the IoU results of class truck and pole with backbone SqueezeSegV2 [38] drop after incorporating ASAP-1. We hypothesize it's because there's not enough fine-scale information since the query radius of ASAP-1 is relatively large (1 meter). To verify that, we use multi-scale grouping proposed by [24] in the LSA block to extract center point features. As shown in Table 2, results of both classes get improved.

5 Conclusions

We run a pilot study towards segmenting dynamic point cloud sequences by proposing a flexible module called ASAP. Based on properties of point cloud sequences, we introduce a novel attentive temporal embedding layer to fuse the relatively informative local features across frames. We further present a spatio-temporal correlation to efficiently exploit more structure information for embedding. We show how we can easily insert the proposed ASAP module into former static point cloud pipelines and achieve great improvements on large-scale datasets. We expect our ASAP module can benefit research on point cloud sequence segmentation and related studies.

References

[1] Shoaib Azam, Farzeen Munir, Aasim Rafique, YeongMin Ko, Ahmad Muqeem Sheri, and Moongu Jeon. Object modeling from 3d point cloud data for self-driving vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 409–414. IEEE, 2018.

[2] Aseem Behl, Despoina Paschalidou, Simon Donne, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[4] Pierre Biasutti, Aurélie Bugeau, Jean-François Aujol, and Mathieu Brédif. Riu-net: Embarrassingly simple semantic segmentation of 3d lidar point cloud. *arXiv:1905.08748*, 2019.

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[6] Ayush Dewan, Tim Caselitz, Gian Diego Tipaldi, and Wolfram Burgard. Rigid scene flow for 3d lidar scans. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1765–1770. IEEE, 2016.

[7] Li Ding, Jack Terwilliger, Rini Sherony, Bryan Reimer, and Lex Fridman. Value of temporal dynamics information in driving scene segmentation. *arXiv:1904.00758*, 2019.

[8] Francis Engelmann, Theodora Kontogianni, Jonas Schult, and Bastian Leibe. Know what your neighbors do: 3d semantic segmentation of point clouds. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[9] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[10] Fabian Groh, Patrick Wieschollek, and Hendrik Lensch. Flex-convolution (million-scale point-cloud learning beyond grid-worlds). *arXiv:1803.07289*, 2018.

[11] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] Jing Huang and Suya You. Point cloud labeling using 3d convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2670–2675. IEEE, 2016.

[13] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv:1602.07360*, 2016.

[14] Mingyang Jiang, Yiran Wu, Tianqi Zhao, Zelin Zhao, and Cewu Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv:1807.00652*, 2018.

[15] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[16] Yen-Cheng Kung, Yung-Lin Huang, and Shao-Yi Chien. Efficient surface detection for augmented reality on 3d point clouds. In *Proceedings of the 33rd Computer Graphics International*, pages 89–92. 2016.

[17] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[18] Xingyu Liu, Charles R. Qi, and Leonidas J. Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[19] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9246–9255, 2019.

[20] Jing Samantha Pan and Geoffrey P Bingham. With an eye to low vision: Optic flow enables perception despite image blur. *Optometry and Vision Science*, 90(10):1119–1127, 2013.

[21] Bo Pang, Kaiwen Zha, Hanwen Cao, Chen Shi, and Cewu Lu. Deep rnn framework for visual sequential applications. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 423–432, 2019.

[22] Bo Pang, Kaiwen Zha, Hanwen Cao, Jiajun Tang, Minghui Yu, and Cewu Lu. Complex sequential understanding through the awareness of spatial and temporal concepts. *Nature Machine Intelligence*, 2(5):245–253, 2020.

[23] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[24] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[25] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[27] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[28] Imeen Ben Salah, Sebastien Kramm, Cédric Demonceaux, and Pascal Vasseur. Summarizing large scale 3d point cloud for navigation tasks. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2017.

[29] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[30] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[31] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *2017 international conference on 3D vision (3DV)*, pages 537–547. IEEE, 2017.

[32] Gusi Te, Wei Hu, Amin Zheng, and Zongming Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 746–754, 2018.

[33] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010.

[34] Arash K Ushani, Ryan W Wolcott, Jeffrey M Walls, and Ryan M Eustice. A learning approach for real-time temporal scene flow estimation from lidar data. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5666–5673. IEEE, 2017.

[35] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):1–12, 2019.

[36] Zongji Wang and Feng Lu. Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes. *IEEE transactions on visualization and computer graphics*, 2019.

[37] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.

[38] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE, 2019.

[39] Wei Zeng and Theo Gevers. 3dcontextnet: K-d tree guided hierarchical learning of point clouds using local and global contextual cues. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.