MPT-Net: Mask Point Transformer Network for Large Scale Point Cloud Semantic Segmentation

Zhe Jun Tang[1] and Tat-Jen Cham[2]

***Abstract*— Point cloud semantic segmentation is important for road scene perception, a task for driverless vehicles to achieve full fledged autonomy. In this work, we introduce Mask Point Transformer Network (MPT-Net), a novel architecture for point cloud segmentation that is simple to implement. MPT-Net consists of a local and global feature encoder and a transformer based decoder; a 3D Point-Voxel Convolution encoder backbone with voxel self attention to encode features and a Mask Point Transformer module to decode point features and segment the point cloud. Firstly, we introduce the novel MPT designed to specifically handle point cloud segmentation. MPT offers two benefits. It attends to every point in the point cloud using mask tokens to extract class specific features globally with cross attention, and provide inter-class feature information exchange using self attention on the learned mask tokens. Secondly, we design a backbone to use sparse point voxel convolutional blocks and a self attention block using transformers to learn local and global contextual features. We evaluate MPT-Net on large scale outdoor driving scene point cloud datasets, SemanticKITTI and nuScenes. Our experiments show that by replacing the standard segmentation head with MPT, MPT-Net achieves a state-of-the-art performance over our baseline approach by 3.8% in SemanticKITTI and is highly effective in detecting 'stuffs' in point cloud.**

I. Introduction

Road scene understanding is a critical task for driverless vehicles to achieve full autonomy that is paramount to passengers' safety. In recent years, many autonomous vehicles are equipped with laser range sensors called 'LiDAR'. These sensors capture a 3D point cloud of the surrounding environment with details exceeding conventional images based systems in range and precision. To achieve better road perception, research in point cloud semantic segmentation has gained interest. The task of point cloud semantic segmentation is to map individual points to their predefined classes e.g. roads, pedestrians, and trees etc.

Advances in deep learning for point cloud has resulted in several new architectures that pushed the state-of-the-art results in semantic segmentation. Notably, many architectures are reliant on converting points to voxels for processing via a 3D Convolutional Neural Network (CNN). While converting points in voxels is an effective way to process point cloud, a key parameter which influences segmentation results is voxel resolution. Increasing voxel resolution improves segmentation performance. However the downside is the cost of more computation. Addressing this large computation cost,

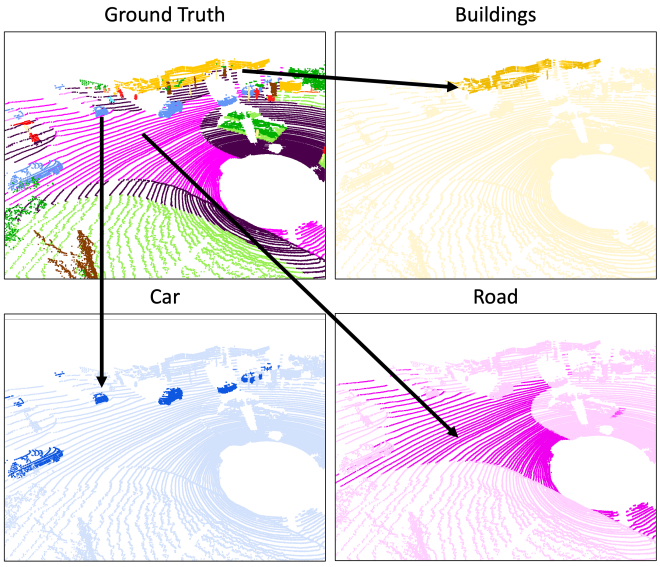

Fig. 1. We design a segmentation head called Mask Point Transformer which leverages on cross attention and self attention mechanisms. By introducing a set of mask tokens that corresponds to the number of classes in the point cloud, each mask token attend to every single point in the point cloud to learn class specific features. In the figure above, we show attention maps of what the 'car', 'buildings', and 'road' mask tokens focuses on in the entire point cloud scene.

networks like [1] utilize a sparse voxel convolution which skips non-activate regions to reduce memory consumption. Other architectures like SPVCNN [2] include a Sparse Point-Voxel Convolution (SPVC) branch to retain fine grained point details by mapping voxel features to points and process these individual points using Multi Layer Perceptron (MLP).

Leveraging on the use of sparse point voxel convolution technique, we utilize SPVCNN's SPVC module to learn voxel and point level features which we adopt as a baseline. We dispose the transposed convolution blocks and only extract features at the last downsampled layer. In doing so, we observe little discrepancies in the segmentation results. As the SPVC blocks seek to learn local features based on the local convolutional kernels, we introduce a Voxel Transformer Module to learn features at a global scale. This improves the network's capabilities in detecting large 'stuffs' [3] (amorphous background regions) in the point cloud.

Unlike previous works that rely on a simple linear classifier or a convolutional layer to segment the point cloud from point features, we are keen to extract class features on a global scale by examining every point in the point cloud. As such, we designed Mask Point Transformer (MPT). MPT

[1]Zhe Jun Tang is with SenseTime Research and with School of Computer Science and Engineering, Nanyang Technological University, Singapore zhejun001@e.ntu.edu.sg

[2]Tat-Jen Cham with the School of Computer Science and Nanyang Technological University, Singapore astjcham@ntu.edu.sg

is simple and highly flexible which can replace the linear classifier head in most networks. It consists of a combination of cross attention and self attention blocks. In MPT, we define K class numbers of 'mask tokens'. These mask tokens query from the keys and values of all point features via cross attention. Hence, every point in the point cloud is attended to. By querying from the point feature key and value sets, MPT seeks to extract relevant point features specific to the individual mask tokens. By this approach, we attend to every point without suffering from the huge quadratic computation involved in point to point attention of transformers. After which, MPT adopts a sequential series of self attention blocks to process these mask tokens. In particular, applying self attention on mask tokens allows learning of inter-class level features to compute an overall scenic understanding of the point cloud.

Combining the backbone network with MPT forms Mask Point Transformer Net (MPT-Net). We evaluate MPT-Net on SemanticKitti, a large scale outdoor dataset. Experiments show that MPT-Net achieves a 3.8% performance improvement over the baseline SPVCNN network and is effective in detecting 'stuffs' in point cloud. Following, we conduct a detailed ablation study on the different components of MPT-Net.

The contribution of this work is threefold: (1) We design an encoder network to learn global and local features of the point cloud with sparse point convolutional blocks and a self attention block with a transformer encoder. (2) We design a Mask Point Transformer module to attend to every point in the point cloud and extract specific point features related to the respective classes, while averting massive quadratic computation in transformers using mask tokens. (3) The proposed MPT-Net achieves significant performance gain over the baseline in point cloud semantic segmentation on the SemanticKITTI dataset and performs well on nuScenes. To our knowledge, this is the first work on applying masked token classes on point cloud.

II. Related Work

A. Point Based Methods

In Point Based architectures, the common approach is to directly feed raw points into the network without a structured or ordered representation. PointNet [4], consists of several Multi Layer Perceptron (MLP) layer, directly takes in raw points for point cloud learning. To address the unstructured and unordered-ness of point cloud, a max pooling symmetric function is embedded in the network. However, this network is unable to model local geometry in point clouds and the relations between points. PointNet++ [5] builds upon [4] by dividing the point cloud into different regions using a hierarchical approach. Furthest point sampling (FPS) divides the point cloud and an individual mini PointNet processes these sub regions before upsampling to form an entire point cloud representation. PointSift [6] adopts an orientation encoding and scale awareness to further encode local information by stacking several orientation-encoding units. To leverage on spatially-local correlation in the data, PointCNN [7] applies a convolutional layer by weighing individual point features before permuting the points into a latent and canonical order. Furthermore, PointWeb [8] applies a series of locally linked web to encode region information using a push-pulling feature approach. As former approaches such as [4] [5] are unable to scale to large point clouds, that often consist over 100,000 points and are bottlenecked by the FPS operation, RandLA-Net [9] uses a more memory efficient random sampling method and local feature aggregation module which consists of attention modules to progressively increase the receptive field of 3D points.

B. Projection Methods

Compared to Point Based architectures, projection based methods aim to provide a regular grid like structure for point clouds. This provides several advantages over point methods methods as the two main characteristics of unorderedness and unstructured are addressed. PolarNet [10] transforms point clouds in cartesian coordinates into polar bird eye view (BEV) coordinates and then apply a 2D ring CNN. This method provides a natural way to process points from the lidar.RangeNet++ [11] follows a more traditional image based approach by applying a spherical projection of the point cloud onto a 2D plane and use a DarkNet53 [12] backbone network. However by projecting the 3D points directly onto a 2D plane causes ambiguities and discretization errors. SqueezeSegV2 [13] improves upon [14] by introducing a Context Aggregation Module to mitigate the effects of dropout noise. Beyond this, SqueezeSegV2 also utilizes a synthetic GTA-Lidar data to pretrain the network before applying to real world datasets via domain adaptation. Furthermore, SqueezeSegV3 [15] included a Spatially-Adaptive Convolution to apply adaptively apply different filters to different regions due to features only active in different point cloud regions. Similarly, SalsaNext [16] projects the points onto a BEV and Spherical Front View (SFV) and use a 2D CNN backbone to process point cloud segmentation. Unlike aforementioned methods, Cylinder3D [1] transforms the point cloud into a cylinder shape format and apply a 3D CNN to retain the 3D nature of the points. Although it is highly successful in the task of point cloud segmentation, the added 3D CNN comes with a high computational cost. An increase in the number of voxels causes a cubic increase in computational cost. Nevertheless, these projection based methods suffer from a similar issue - A high resolution point cloud rasterized into voxels results in a spatial loss of information. A small object in the point cloud is being represented by a few voxels results in a high difficulty for the model to differentiate objects.

C. Fusion Based Methods

To tackle the deficiency of projection based methods which results in information loss caused by discretizing a high resolution point cloud into a finite number of voxels, fusion based methods are developed. In Point-Voxel CNN [17], two branches are developed - one for individual points and another for voxels. In the voxel branch, points are

first rasterized into voxels with individual voxel features computed by averaging features of all points belonging to the same voxels. These voxels are then processed by a 3D convolution kernel.In the point branch, a multi-layer perceptron layer takes in raw points and output point wise features. Subsequently, voxel features are devoxelized and mapped back to individual points using trilinear interpolation. Finally, features processed from the voxels and point branched are merged, retaining both high resolution point-wise feature and regularly structured voxel features. Voxelizing point cloud into voxels often result in empty voxels as these voxels do not contain any points due to the irregular and sparse nature of point cloud. To avoid unnecessary convolution operations on empty voxels, SPVCNN [2] is developed. The key difference between PVCNN and SPVCNN is the use of sparse convolution. Only active input sites are processed with a 3D convolution kernel which speeds up computation. An iterative approach to voxelize and devoxelize points is attempted in DRI-Net [18]. Points are first voxelized and processed with convolution. The voxels features are then devoxelized into point features and processed with a point branch. This is done for 3 consecutive times. Besides the two point and voxel branches seen in previous networks, RPV-Net [19] added a third branch. This branch includes the spherical projection of 3D points onto a 2D plane seen in the previous section; projection methods. With a hash code, features from the 3 branches are exchanged at every stage before merging back to point level features and then processed with a MLP layer for dense prediction.

D. Attention Based Methods

In RandLA-Net [9] attentive pooling is applied to select important features from a set of neighbouring point. This is required is propagate useful features from shallow to deeper layer. Other than outdoor driving scene datasets, attention methods are employed in smaller scale datasets like S3DIS. Point Transformer [20] applies self attention on individual points in the downsampling and upsampling. However due to the quadratic computation involved in the transformer encoder module, it is difficult to scale to large outdoor driving scene datasets. Other transformer based methods such as YOGO [21] groups the point cloud into smaller subset once before applying self attention. In later stages, cross attention is between the overall point cloud and the grouped subsets.

III. Mask Point Transformer Network

An outdoor driving scene dataset consists of points spanning more than hundreds of thousands of points. In addition, given the sparsity of points in the dataset, it is difficult to extract spatial information unlike an indoor dataset with regular and fixed number of points. Rather than relying on heuristic sampling methods like Further Point Sampling or Inverse Density Important Sampling, Mask Point Transformer Network relies on a fixed voxelization method to provide a fixed regular grid structure for processing. To extract features from the voxelized point cloud, we utilise an encoder branch based on Sparse Point Voxel Convolution Neural Network and a self attention block on voxel scale. For the decoder, we apply a Mask Point Transformer Segmentation Head (MPT-H) for class based segmentation.

Our general network architecture is illustrated in Figure 2.

A. Encoder

MPT-Net backbone consists two branches, point and voxel branch, in the encoder segment. Given that the input point cloud is unstructured and unordered, it is useful to map points to voxels to provide a regular 3D grid to extract spatial information. Hence, we rely on the voxelization method based on SPVCNN. An input point cloud $P = [p_1, ..., p_N] \in \mathbb{R}^{N \times d_{in}}$ contains N number of points. Each point $p_n = (x, y, z, i)$ consists of positional information of the point in Cartesian coordinate system with i representing the intensity of the reflected laser. For every p_n, there exists l_n that represents the class label.

We voxelize P into a sparse tensor $S = [s_1, ..., s_k] \in \mathbb{R}^{K \times d_{in}}$ where $s_k = (v_k, f_k)$. v_k represents the voxel coordinate in 3D space and f_k equates to voxel feature. Each point p_n is gathered into v_k voxel based on a gather operation $v_k = (x, y, z)/r$ with r set as a voxel resolution of $0.05m$. f_k is the averaged feature of the points containing in the same voxel. The inverse devoxelization operation to convert S to P is based on tri-linear interpolation where each point p_n feature is interpolate based on the neighbouring voxels. We encourage readers to refer to SPVCNN [2] for more details. The input sparse tensor S is processed by a downsampling only decoder based on Sparse Point Voxel Convolution as shown in Figure 2.

1) Context Block: To extract voxel features, we pass the network through a Context Block (CB). CB contains three parallel convolution branch of different kernel size of small, medium, and large. Each branch extracts different features based on the respective kernel receptive field size. CB greatly enhances spatial feature extraction. We then concatenate the features from the three different branches and devoxelize the feature map to project features back to point form.

2) Local Feature Extractor: The main encoder branch consists of 4 down-sampling convolutional stages. Each stage contains a pooling stage and two residual block. In the pooling stage, we apply a 2 by 2 by 2 convolution process with stride 2 to down-sample the feature maps at voxel level. After the down-sampling operation, we apply two residual blocks to further extract voxel features. At the point feature scale, we apply an MLP layer to pass point features from the previous scale to the downsampled scale. We then merge point features and voxel features by devoxelization and an addition operation.

3) Global Feature Extractor: Feature extraction based on 3D voxel convolution results in only local features being extracted based on the convolution kernels lacks global context understanding. Yet global context understanding of the point cloud is essential. For instance, in an outdoor driving scene, the positions of 'stuff' like vegetation, fence, and, buildings requires a global level understanding. An

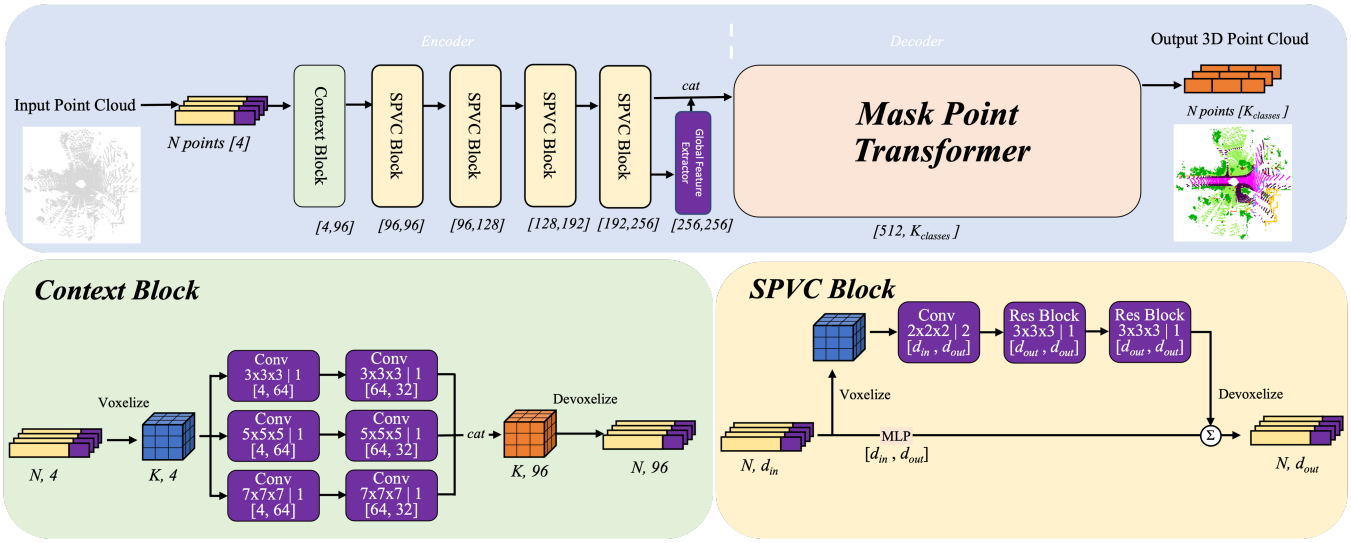

Fig. 2. (1) Top: Overall architecture of MPT-Net. The input point cloud is first fed into a context block to extract voxel-wise features based on the local kernel size and passed into a series of sparse point voxel convolution blocks and a self attention block before the Mask Point Transformer module. (2) Bottom left: Input point cloud is first voxelized before applying a 3D CNN (Kernel Size, Stride). (3) Bottom right: Sparse Point Convolution block is applied to both extract voxel-wise and point-wise features.

understanding that road occupying a fixed width would allow the network to focus on other flat surfaces like vegetation and terrain in the overall scene. We also note that fixed 'stuff' like buildings and fence often appear next to each other and near roads.

To enhance global feature understanding of the overall scene, we adopt a transformer encoder to attend to voxel features only. We placed this transformer encoder after the last SPVC block, with an added downsampling stage, to reduce the total features passed into self attention layer. Only non-empty voxels are used to compute the self attention features. Each voxel feature is encoded with the voxel positional information. By allowing self attention on voxel features, global contextual features are learned. The global feature extract proves important in extracting placement of large components in the point cloud like 'stuff' and has improved performance as shown in the ablation studies.

B. Decoder

1) Mask Point Transformer: The aim of the decoder branch, illustrated in Figure 3, is to decode the encoded point cloud $P_{enc} \in \mathbb{R}^{N \times D_{enc}}$ to segmented point cloud $Y_{pred} \in \mathbb{R}^{N \times K}$ where K is the number of classes. The decoder learns to map point-level encoding into point-level class. This is done via a Mask Point Transformer module which is mixed attention based architecture. It consists of a set of learnable class specific tokens that attends to every point on a global scale. The encoded point features passed into MPT are from the concatenated point features extracted from the Local Feature Extractor and the Global Feature Extractor branches.

We randomly initialize K learnable class tokens $C = [cls_1, ..., cls_k] \in \mathbb{R}^{K \times D_{dec}}$ and assign each cls_k to a single semantic class. Shown in Figure 3, the class tokens learns class specific features via multi-head cross attention. A linear layer first generates the queries from the class tokens such that $C_q \in \mathbb{R}^{K \times D_{dec}}$. The keys P_k and values P_v are obtained from the encoded point cloud P_{enc} in the same way. The dimensions of D_{dec} and D_{enc} are set equal. Here, we allow class tokens to query from the entire encoded point cloud. The multi-head cross attention between the class tokens and the encoded point cloud is computed as follows:

$$MCA(C,\ P) = softmax\left(\frac{C_q \cdot {P_k}^{\mathrm{T}}}{\sqrt{d_{dec}}}\right) \cdot P_v \tag{1}$$

We set the dimensions of individual attention head to 64. Subsequently, a transformer based multi-head self attention operation is applied solely on the class tokens.

$$MSA(C,\ C) = softmax\left(\frac{C_q \cdot {C_k}^{\mathrm{T}}}{\sqrt{d_{dec}}}\right) \cdot C_v \tag{2}$$

This allows class features to attend to each other providing class contextual understanding. We repeat the above process for a total of L times. In each L_i process, we apply L_{ca} cross attention layers followed by series of self L_{sa} attention layer on the mask tokens.

Finally, a set of N masked points $Y_{pred} = [y_1, ..., y_N]$ is obtained by:

$$Y_{pred} = P_{enc} \cdot C^{\mathrm{T}} \tag{3}$$

A softmax function is applied to Y_{pred} to obtained point wise class score.

The attention maps of individual class tokens is depicted in Figure 1 where we show how each class token attends to every point of the point cloud. The advantage of MPT lies in its simplicity for implementation which can replace most segmentation heads in other networks with high performance gains. It provides the benefit of attending to every point without the quadratic computation involved in methods like Point Transformer. Our Mask Point Transformer is largely

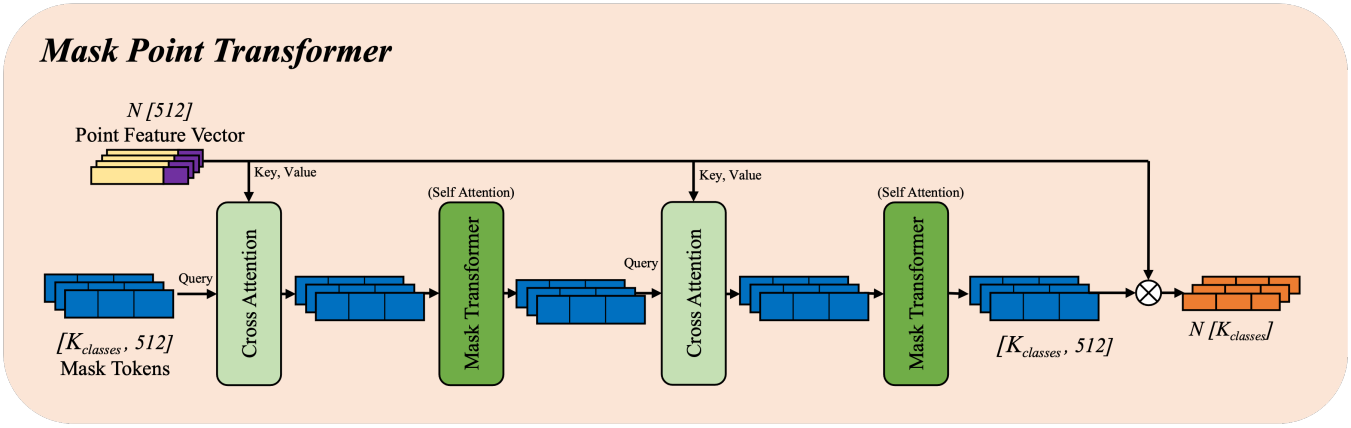

Fig. 3. Layout of Mask Point Transformer Head. We define K number of mask tokens which correspond to the number of classes within the dataset. In this process, the mask tokens first queries from the entire point cloud using cross attention. Subsequently, self attention is applied to the set of mask tokens. We repeat the sequences of cross and self attention layers for a total of 2 times before multiplying the tokens and the point features for class scores.

inspired by the Segmenter [22] and Perceiver [23]. Unlike the Segmemter which is an architecture only used in image domains, our method relies on a hybrid CNN and transformer based approach and we maintain two distinct feature embeddings; one for points and another for class tokens. It is highlighted in the Segmenter the size of patch embedding is critical in the overall segmentation result where smaller patch size relates to better results. Our attempt in mixing a large voxelized point cloud and mask tokens like the Segmenter results in poor performance as large voxelization causes small objects in the point cloud to be missed. We note that Max-DeepLab [24] which utilises a hybrid transformer, a CNN based approach, only operates in the image domain on a patch based level which is highly different from the point cloud domain. None of the above methods operate on a fine scale e.g pixel level. Our approach is nevertheless different in that our method operates on a fine scale, point level, features whereas their method rely on augmenting latent features at several layers passing the augmented features back to the convolutional layers. Max-Deeplab also maintains a dual path information exchange between CNN and transformer for images whereas our network maintains a singular path and rely on transformer to perform point feature decoding; features learned using the attention mechanism does not augment features from SPVC blocks. In the Perceiver, a general architecture used for images, audio, and point cloud, the network only performs point cloud classification task on small datasets and relies on a MLP head and only used in classification tasks. With the Perceiver being similar to the Segmenter, both are end-to-end transformer based network which is different from our single path hybrid approach. Besides, the Perceiver utilises a latent array to learn general point cloud features whereas our approach focuses on class specific features ($K_{class} << d_{latent}$) with mask tokens. To our knowledge, this is the first work which applies mask tokens on a full scale point cloud to extract class level features for segmentation.

IV. Experiments

We report the detailed experimental setups and the respective results evaluated on two outdoor large scale datasets; SemanticKITTI and nuScenes. Following, we conduct extensive ablation studies to examine the effectiveness of the individual network components.

A. Dataset and Metric

Dataset:SemanticKITTI. SemanticKITTI [29] is an outdoor large scale driving-scene point cloud dataset. With this dataset, we evaluate MPT-Net with the point cloud semantic segmentation task. This dataset is derived from the KITTI Vision Odometry benchmark which is obtained from a Velodyne-HDLE64 LiDAR equipped on a car driving on the streets of Germany. Sequences 00 to 10 are used as training set, with the exclusion of sequence 08 used for validation. Sequences 11-21 are used as test set. Each scan contains a maximum of 28 classes and about 120,000 points. Of the 28, only 19 classes are utilized for the official evaluation task. The 19 classes are shown in Table 1.

Dataset: nuScenes. The nuScenes [30] dataset is recently released with point annotated semantic segmented classes. The dataset contains 1000 scenes with 850 scenes split for training and 150 scenes for validation. It contains an outdoor driving scene which data is collected in Boston and Singapore. nuScenes is collected using a Velodyne HDL-32E sensor. Each point in the dataset is marked as one of 32 unique classes. However, only 16 classes are used for official evaluation. As nuScenes contains driving scenes of two different countries, it is similarly challenging. Moreover, compared to SemanticKITTI, it is less dense due to the difference in sensor used.

Metric. We utilize the mean Intersection over Union (mIoU) as an evaluation metric to assess the performance of our method. mIoU is calculated as

$$mIoU = \frac{1}{C}\sum_{c=1}^{C}\left(\frac{TPc}{TP_c + FP_c + FNc}\right) \tag{4}$$

TABLE I

SEMANTIC SEGMENTATION IOU % RESULTS ON SEMANTICKITTI TEST DATASET. THE METHODS ARE GROUPED ACCORDING TO: POINT-BASED NETWORKS, RANGE VIEW NETWORKS, BIRD'S EYE VIEW NETWORK, AND VOXELIZATION BASED METHODS.

Method	Mean IoU	Car	Bicycle	Motorcycle	Truck	Other-vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other-ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic-sign
RandLA-Net [9]	53.9	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.7
KPConv [25]	58.8	96.0	32.0	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	90.5	64.2	84.8	69.2	69.1	56.4	47.4
RangeNet++ [11]	52.2	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1	64.6	47.9	55.9
Salsanext [16]	59.5	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1
KPRNet [26]	63.1	95.5	54.1	47.9	23.6	42.6	65.9	65.0	16.5	**93.2**	**73.9**	**80.6**	30.2	91.7	68.4	85.7	69.8	71.2	58.7	64.1
SqueezeSegv3 [15]	55.9	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7	65.4	49.6	58.9
PolarNet [10]	54.3	93.8	40.3	30.1	22.9	28.5	43.2	40.2	5.6	90.8	61.7	74.4	21.7	90.0	61.3	84.0	65.5	67.8	51.8	57.5
TornadoNet [27]	63.1	94.2	55.7	48.1	40.0	38.2	63.6	60.1	34.9	89.7	66.3	74.5	28.7	91.3	65.6	85.6	67.0	**71.5**	58.0	65.9
SPVCNN [2]	64.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SPVNAS [2]	67.0	**97.2**	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	**86.1**	73.4	71.0	**64.3**	67.3
DRI-Net [18]	67.5	96.9	57.0	56.0	43.3	54.5	69.4	75.1	58.9	90.7	65.0	75.2	26.2	91.5	67.3	85.2	72.6	68.8	63.5	66.0
Cylinder3D [1]	67.8	97.1	**67.6**	64.0	**59.0**	58.6	73.9	67.9	36.0	91.4	65.1	75.5	32.3	91.0	66.5	85.4	71.8	68.5	62.6	65.6
AF2S3 [28]	**69.7**	94.5	65.4	**86.8**	39.2	41.1	**80.7**	**80.4**	**74.3**	91.3	68.8	72.5	**53.5**	87.9	63.2	70.2	68.5	53.7	61.5	**71.0**
MPT-Net (Ours)[a]	68.2	97.0	49.8	51.6	51.1	**59.2**	67.9	76.8	69.9	90.4	63.4	74.4	27.7	**92.2**	**68.5**	86.0	**73.4**	70.5	62.7	63.5

[a]We compare across architectures that do not utilise any tricks. Bold font denotes top in compared categories. Blue text denotes second in place.

TABLE II

SEMANTIC SEGMENTATION IOU % RESULTS ON NUSCENES VALIDATION DATASET BASED ON VOXELIZATION.

Methods	mIoU	Barrier	Bicycle	Bus	Car	Construction	Motorcycle	Pedestrian	Traffic-cone	Trailer	Truck	Driveable	Other	Sidewalk	Terrain	Manmade	Vegetation
AF2S3 [28]	62.2	60.3	12.6	82.3	80.0	20.1	62.0	59.0	49.0	42.2	67.4	94.2	68.0	64.1	68.6	82.9	82.4
Cylinder3D [1]	**76.1**	**76.4**	40.3	91.2	**93.8**	**51.3**	78.0	**78.9**	**64.9**	62.1	**84.4**	**96.8**	71.6	**76.4**	**75.4**	**90.5**	87.4
MPT-Net (Ours)	75.7	75.3	**42.2**	**92.3**	87.9	47.5	**83.8**	76.0	60.6	**68.2**	82.3	96.4	**72.7**	74.3	74.2	89.4	**87.4**

TP, FP, FN represent the number of True Positive, False Positive, and False Negative of classified points. C refers to the number of classes in the dataset.

B. Training Details

We optimize our model with SGD Optimizer with momentum of 0.9 trained at a initial learning rate of 0.08 and a weight decay of 0.0005. We set $L = 2$ process with $L_{ca} = 1$ cross attention layer and $L_{sa} = 4$ self attention layers in the Mask Point Transformer Module which yield the best results. A further increase in self attention and cross attention layers yield higher computational cost and does not benefit the segmentation results. Subsequently, we decrease the learning by half every 5 epochs for a total of 40 epochs for the SemanticKITTI dataset and 60 epochs for nuScenes. Experiments are conducted 8 Nvidia Tesla V100 GPUs and trained for 1.5 days. We set the number of attention heads in MPT as 8, each head dimension as 64 with 4 layers of self attention on the mask tokens.

The objective of our network is point wise loss which is the sum of Cross Entropy, for point wise accuracy maximization, and Lovasz Loss [31] ,for mIoU maximization, with equal weightage.

During training, we adopt the commonly used data augmentation techniques by randomly rotating the point cloud along the $z-axis$ and scale each point between a range of 0.95, 1.05. We emphasize that unlike RPV-Net [19] we do not employ any further data augmentation techniques nor manually craft additional data for training (e.g. Instance CutMix).

$$L_{total}(y,\hat{y}) = L_{ce}(y,\hat{y}) + L_{lov}(y,\hat{y}) \tag{5}$$

C. Semantic Segmentation Benchmarks

SemanticKITTI official test benchmark. We compare the results of MPT-Net and existing state-of-the-art methods on the SemanticKITTI test set and report the results shown in Table I. MPT-Net is competitive with prior methods and exceed the previous SOTAs on a few categories. Figures reported in the table are retrieved from the published papers. Our MPT-Net outperforms the baseline approach (SPVCNN) by a large margin of 3.8%. MPT-Net achieves top competitive results on 4 categories, despite achieving the second best overall mIoU score in training without any tricks. MPT-Net also ranks comparable with the top performers on 3 other classes and in total, matches AF2S3-Net. Notably, MPT-Net performs well on detecting 'stuffs' over AFS2S3-Net which focuses on things. Perceiving stuff well is highly critical for understanding the surrounding environment which would be useful for mapping technologies.

nuScenes validation benchmark. We compare the validation results on nuScenes utilizing voxelization based methods. As the point cloud in nuScenes is less dense than SemanticKITTI, we set the voxelization size to 0.1m. MPT-Net outperforms AF2S3-Net by a huge margin. Overall, MPT-Net performs well in distinguishing large objects such as trailers and buses. MPT-net also excels in detecting rare

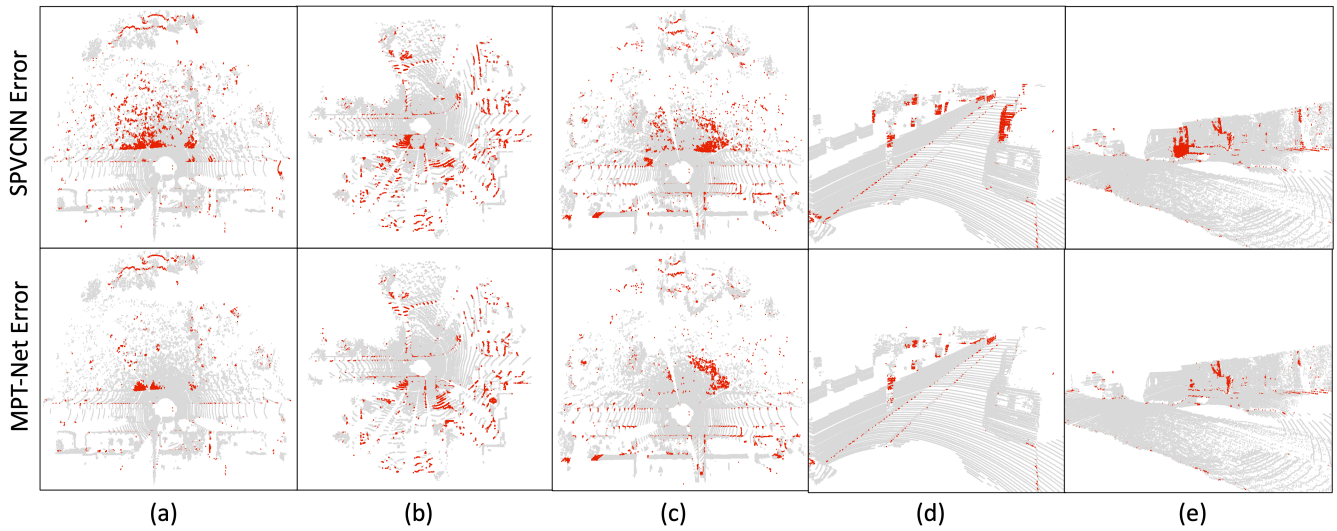

Fig. 4. (1) Top: Red highlighted points denote segmentation error for SPVCNN Network. (2) Bottom: Red highlighted points denote segmentation error for MPT-Net. Observe how MPT reduces sparse point errors that lie further away from the LiDAR.

classes like bicycle and motorcycle.

D. Ablation Studies

Effects of different components. We illustrate the effects of the different components in Table III. The removal of the transposed convolution layer has little effect on the overall mIoU score. We believe that the upsampling transposed convolution layer is not necessary due to devoxelization where point features are interpolated from surrounding voxels which has a similar upsampling effect and a MLP processed these point features in the SPVC Block. Adding a larger context block at the front allows better voxel features to be extracted due to the large receptive fields. As shown, just by replacing the linear classifier in SPVCNN with MPT boosts huge performance gain. Our MPT Head further refines features decoded from the last SPVC block where each point is being attended to by the individual mask tokens. Lastly, we experiment with adding another down convolution operation and passed these voxels into a transformer and concatenate the features from the last SPVC block before passing through the MPT Head. We observe that the combination of all proposed components achieves the best validation results on the SemanticKITTI validation dataset.

Effects of attention layers. We observed that setting a total $L = 2$ processes with $L_{ca} = 1$ and $L_{sa} = 4$ yields the optimal results in our experimental settings. The higher number of processes and respective attention layers yield better segmentation performance. An increase in the number of cross and self attention layers beyond the above-mentioned parameters causes a slight decrease in performance.

MPT Qualitative Results. We compare the baseline method SPVCNN with our network, MPT-Net shown in Figure 4. Red highlighted points in the figure denotes error in the segmentation tasks. As observed, MPT-Net outperforms the baseline in classifying class such as vegetation and trunk. The results are consistent in many scenes and MPT is still able to segment points that lie far off from the point cloud center.

In Figure 4 (d) and (e), it is shown that MPT-Net is able to segment buildings and partially obscured 'things; like buildings and cars better.

Further studies on low level features From Table I, we observe that MPT-net fails to identify smaller objects such as bicycle and motorcycle which heavily affected the score of the overall mIoU. To verify whether this is due to the lack of access to low level features using U-Net structure like Cylinder3D and AF2S3-net, we conduct 2 experimental settings shown in Table IV. Even with a U-Net like structure

TABLE III
EFFECTS OF NETWORK COMPONENTS. TESTED ON SEMANTICKITTI VALIDATION SET.

Architecture	Optimized [a]	No TransConv	Context Blk	MPT-Head	Global-Fea-Extractor.	mIoU
Baseline (SPVCNN) [2]						63.7
Proposed Components	✓					66.2
	✓	✓				66.0
	✓	✓	✓			66.3
	✓	✓	✓	✓		67.9
	✓	✓	✓	✓	✓	**69.0**

[a]Optimized refers to training the network according to section IV-B

TABLE IV
ABLATION STUDY OF DIFFERENT MPT-NET CONFIGURATIONS

Model	mIoU
U-Net structure	66.9
Access all SPVC Block features	67.8
Our Model	**69.0**

or a direct point wise concatenation of features across all down-sampled SPVC blocks does not improve the overall mIoU score. Lastly, concatenating all point features from the individual SPVC blocks and pass into MPT-head to 'access all SPVC block features' does not improve performance as much as our final model.

V. Conclusion

In this paper, we proposed a simple attention mechanism, namely MPT, to attend to every point in the point cloud without the massive quadratic computation of points involved in transformers for point cloud semantic segmentation. MPT is designed to be easy and neat for implementation to replace most segmentation head in other networks. Specifically, generate a set of class tokens, query the encoded point cloud, and multiply the tokens with the point cloud for segmentation score. The benefit of using mask tokens and cross attention allows the network to examine the overall point cloud and extract class related features globally and with self attention on class tokens, it provides inter-class feature learning. The overall backbone in MPT-Net provides both local and global feature extraction. We conduct extensive experiments to illustrate the effectiveness of MPT-Net and achieved a huge performance gain over the baseline approach on two large scale outdoor point cloud datasets.

References

[1] H. Zhou, X. Zhu, X. Song, Y. Ma, Z. Wang, H. Li, and D. Lin, "Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation," *arXiv preprint arXiv:2008.01550*, 2020.

[2] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, "Searching efficient 3d architectures with sparse point-voxel convolution," in *European Conference on Computer Vision*. Springer, 2020, pp. 685–702.

[3] H. Caesar, J. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1209–1218.

[4] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[5] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.

[6] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "Pointsift: A sift-like network module for 3d point cloud semantic segmentation," *arXiv preprint arXiv:1807.00652*, 2018.

[7] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on $\mathcal{X}$-transformed points," 2018.

[8] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "Pointweb: Enhancing local neighborhood features for point cloud processing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5565–5573.

[9] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 108–11 117.

[10] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh, "Polarnet: An improved grid representation for online lidar point clouds semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9601–9610.

[11] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet++: Fast and accurate lidar semantic segmentation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4213–4220.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[13] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. 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)*. IEEE, 2019, pp. 4376–4382.

[14] B. Wu, A. Wan, X. Yue, and K. 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)*. IEEE, 2018, pp. 1887–1893.

[15] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *European Conference on Computer Vision*. Springer, 2020, pp. 1–19.

[16] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving," 2020.

[17] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel cnn for efficient 3d deep learning," *arXiv preprint arXiv:1907.03739*, 2019.

[18] M. Ye, S. Xu, T. Cao, and Q. Chen, "Drinet: A dual-representation iterative learning network for point cloud segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7447–7456.

[19] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, "Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation," *arXiv preprint arXiv:2103.12978*, 2021.

[20] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 259–16 268.

[21] C. Xu, B. Zhai, B. Wu, T. Li, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "You only group once: Efficient point-cloud processing with token representation and relation inference module," *arXiv preprint arXiv:2103.09975*, 2021.

[22] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," 2021.

[23] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira, "Perceiver: General perception with iterative attention," 2021.

[24] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Max-deeplab: End-to-end panoptic segmentation with mask transformers," 2021.

[25] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6411–6420.

[26] D. Kochanov, F. K. Nejadasl, and O. Booij, "Kprnet: Improving projection-based lidar semantic segmentation," 2020.

[27] M. Gerdzhev, R. Razani, E. Taghavi, and L. Bingbing, "Tornado-net: multiview total variation semantic segmentation with diamond inception module," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 9543–9549.

[28] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "(af)2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," 2021.

[29] 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.

[30] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[31] M. Berman, A. R. Triki, and M. B. Blaschko, "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," 2018.