Spatial and Temporal Awareness Network for Semantic Segmentation on Automotive Radar Point Cloud

Ziwei Zhang, Jun Liu, *Senior Member, IEEE*, and Guangfeng Jiang

***Abstract*—Radar sensors are vital for autonomous driving due to their consistent and dependable performance, even in challenging weather conditions. Semantic segmentation of moving objects in sparse radar point clouds is an emerging task that contributes to improving the safety of autonomous driving. However, the methods still need to be explored since radar points in driving scenes are distributed sparsely and irregularly. Typical methods address the sparsity by aggregating multi-scan data to generate dense point clouds, where the temporal correlation is ignored. Inputting consecutive frames of point clouds can preserve temporal information, but the irregular distribution makes it difficult to achieve interframe communications. In this article, we propose a scheme to process points of multi-scan data into a single frame with the availability of temporal features. Our novel network, called Spatial and Temporal Awareness Network (STA-Net), enables points at different times to interact and establish their spatiotemporal connections to comprehend the surroundings of these points. Furthermore, we design a shallow feature extraction method based on the radar measurements of points, which enhances the representation of local features. To further improve the capability of the network to distinguish between various moving objects, we also introduce a prompt layer inspired by prompt-based learning. This layer instructs the network to generate a discriminative representation for each type of moving object. Our experiments demonstrate that our network achieves state-of-the-art performance compared to other methods designed on the RadarScenes dataset. In particular, it shows a remarkable ability to segment small objects such as pedestrians.**

***Index Terms*—Autonomous driving, radar point cloud, deep learning, semantic segmentation.**

I. Introduction

AUTONOMOUS driving relies on various modalities of sensors, such as LiDARs, cameras, and radars, to accurately perceive and analyze the driving conditions. By integrating multiple modal information from sensors, self-driving cars enhance their detection capability to detect and respond to different environmental conditions and obstacles. As some sensor types may encounter malfunctions or interference in specific scenarios, data redundancy from multiple modal sensors is necessary to make the sensor suite trustworthy [1]. For example, LiDAR sensors and cameras fail to capture precise surroundings in adverse weather, such as rain and snow [2]. In such situations, radar sensors can still provide valuable information. Apart from being effective in all weather conditions, they are able to detect distance, speed, and the presence of nearby objects. Therefore, radar sensors play a crucial role in making reliable and safe decisions in autonomous driving [3], [4], [5], [6], [7], [8].

With the development of large-scale 3D and 4D radar datasets, radar-based deep learning algorithms have made progress across a range of tasks, including object detection [9], [10], [11], [12], odometry and mapping [8], [13], semantic segmentation [14], [15], [16], and scene flow estimation [17], [18]. Among these, semantic segmentation on point clouds is an emerging task that assigns each point with a class label or a probability vector, distinguishing moving objects from static backgrounds. While most semantic segmentation methods are developed for LiDAR point clouds, there is a growing interest in performing semantic segmentation on automotive radar point clouds. Radar sensors provide two unique measurements, velocity and Radar Cross Section (RCS), which can offer valuable insights for semantic segmentation. However, radar point clouds exhibit greater irregularity and sparsity when compared to LiDAR point clouds. This poses new challenges for existing models, which are typically tailored to LiDAR point clouds. Usually, point-based networks are employed to address the irregularity, as they can directly operate on points while preserving the geometric structure and maintaining permutation invariance [19]. To tackle the sparsity issue, it is a popular way to aggregate multiple scans of radar data as single-scan input to enrich the input information and preserve prediction efficiency. As illustrated in Fig. 1, the aggregated data usually exhibits an intrinsic temporal correlation between successive scans. However, this correlation is usually dropped in traditional methods.

Therefore, we present our new approach for processing multi-scan point clouds, which regards all data as a single frame but provides temporal characteristic for each point. With this processing method, we can take advantage of the efficiency of the single-scan processing methods while facilitating communication between points collected at different times within a single frame. To accomplish the communication, we design

Manuscript received 16 November 2023; revised 11 December 2023; accepted 15 December 2023. Date of publication 28 December 2023; date of current version 29 April 2024. This work was supported in part by the Youth Innovation Promotion Association CAS under Grant CX2100060053, and in part by the Natural Science Foundation of Anhui Province under Grant 2208085J17. *(Corresponding author: Jun Liu.)*

The authors are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China (e-mail: zziwei@mail.ustc.edu.cn; junliu@ustc.edu.cn; jgf1998@mail.ustc.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIV.2023.3347692.

Digital Object Identifier 10.1109/TIV.2023.3347692

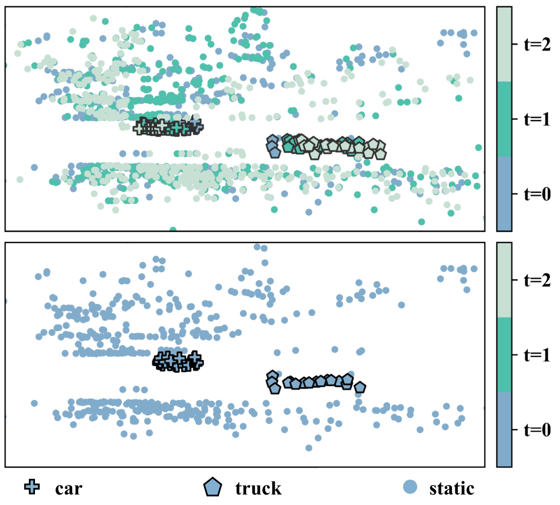

Fig. 1. Illustration of the strong correlation between successive scans. The image above is the presentation of multi-scan data, and the image below shows the points at the first moment in the above point cloud.

a spatiotemporal mechanism that enables points to perceive their surroundings from both temporal and spatial perspectives. Beyond that, we notice that most of the previous methods focus on extracting deep features from the neighborhoods of points. The velocity and RCS have not been fully appreciated for their ability to represent the neighborhood. Therefore, we design the Local Perception Module (LPM) to exploit the representation of local areas from radar measurements. Our STA-Net learns the semantics of points with an encoder-decoder structure. We introduce the Neighborhood Feature Extraction Block (NFEB) to encode multi-level semantics of points. Within this block, the spatiotemporal relationships between points and their respective surroundings are captured through the Spatiotemporal Attention Module (SAM), while shallow local radar features are extracted using the LPM. These features from the two modules are further combined to enhance the representation of neighborhoods of points. Furthermore, we propose a Prompt Layer (PL) that is inspired by prompt-based learning in the field of Natural Language Processing (NLP) to refine the class features of the local areas.

In summary, our work makes the following contributions:

1) We propose STA-Net, which efficiently facilitates inter-scan communications among points, and achieves state-of-the-art performance on multi-scan radar semantic segmentation.
2) We design the novel NFEB to generate multi-level neighborhood features, and the PL to enhance the discrimination of the features. We conduct a series of ablation studies, which validate the effectiveness of these key components.
3) We evaluate both the effectiveness and efficiency of our proposed method in experiments, and our method achieves state-of-the-art results on RadarScenes.

II. Related Work

A. Radar Datasets

1) 3D Radar Datasets: There are a few publicly available datasets for automotive radar perception [20]. They vary in the scale of data and data formats. The large-scale dataset nuScenes [21] is popular for its full sensor suite perception containing cameras, LiDARs, radars, etc. However, it offers an extremely sparse radar point cloud, which is insufficient for radar-only tasks. The RadarScenes dataset [22] provides much denser point clouds than the nuScenes dataset, with 158 different scenarios. It has pointwise annotations and contains 11 classes of moving road users.

To better achieve automotive perception only based on radar data, high-resolution datasets are published. Zendar [23] is a high-resolution dataset that uses a synthetic aperture radar for moving vehicle detection to enhance imaging resolution. However, it only has a small number of scenarios. The RADIATE [24], Oxford Radar RobotCar [25], and MulRan [26] contain dense radar images utilizing spinning radar sensors. However, this type of radar sensor is not commonly used for automotive applications and is short of Doppler information. The CARRADA [27], CRUW [28], and RADDet [29] datasets also provide range-azimuth maps with annotations for object detection. However, these datasets lack full point cloud information.

Several datasets present radar data measured under adverse weather conditions, which provide a foundation for developing robust perception algorithms. The Dense dataset [30] uses long-range sensors and captures data in various natural weather conditions. Unfortunately, it has poor resolution and sparse radar targets due to the limited field of view. Although the RADIATE [24] collects data in different driving scenarios and weather conditions, it only labels road users with bounding boxes unsuitable for semantic segmentation.

2) 4D Radar Datasets: The introduction of new-generation 4D radar [31], [32], [33], [34] mitigates the limitations of conventional automotive radar, offering improved resolution and elevation measurement capabilities. The Astyx dataset is the first publicly available 4D radar dataset for object detection but only provides 545 annotated frames [31]. The VOD dataset contains 8693 frames of synchronized and calibrated LiDAR, camera, and 4D radar data acquired in various traffic [32], and it includes annotations for 3D bounding boxes.

Similarly, the Tj4DRadSet dataset contains a total of 7757 synchronized frames under rich driving scenarios, which provides 3D bounding boxes and trajectory IDs [33]. The proposed K-Radar is the first large-scale dataset that provides 4D radar tensor on diverse conditions along with 3D bounding box labels [34]. Unfortunately, these 4D datasets lack pointwise annotations and are unsuitable for semantic segmentation.

B. Semantic Segmentation on Point Cloud

Current mainstream methods for semantic segmentation can be categorized into four groups: voxel-based [35], [36], [37], projection-based [38], [39], [40], point-based [14], [15], [16], [41], and fusion-based methods [42], [43]. However, most of the voxel-based and projection-based methods are designed for LiDAR data and inherently introduce discretization errors. Moreover, these methods become less effective when applied to radar data due to its limited information and a lower resolution than LiDAR data. The fusion methods, where a combination of voxel and point-based method, or projection and point-based method is used to extract features, impose significant demands on memory and computational resources. Therefore, we mainly review point-based methods here.

For sparse radar point clouds, point-based networks can keep the geometric structure of the points and process irregular data directly. PointNet++ [41] is a hierarchical spatial structure that processes local areas of point clouds sampled progressively and extracts local features similar to the convolutional neural networks CNNs. Schumann et al. [15] first perform semantic segmentation on automotive radar data utilizing PointNet++. They enhance the representation of radar point clouds by aggregating radar data in multiple scans to generate a denser point cloud. However, the time information is usually discarded, and the strong correlation among consecutive scans is not considered. To resolve this, Schumann et al. [16] propose a recurrent architecture, which embeds a memory abstraction module to fuse the input point cloud with previously memorized input. While the memory abstraction module establishes the connection of points between different scans, it may be inconvenient to locate the positions of points in previous scans due to the displacement caused by the movement of the ego-car. In contrast, Zeller et al. investigate the single-scan method [14], which is appropriate for real-time applications that require low latency. However, the single-scan method may struggle with limited information carried by the point clouds due to the low resolution of radar sensors. The situation would be worse if objects have unfavorable aspect angles in the specific scan.

Inspired by the achievement of transformers [44], [45] in computer vision tasks, point-based networks have successfully incorporated self-attention mechanisms. Zhao et al. [46] design point transformer layers to aggregate local features and improve the performance of large-scale semantic scene segmentation. Zellar et al. [14] adapt the Point Transformer by replacing the point transformer layer with the Gaussian transformer layer, which enhances the feature extraction. Due to the single-scan input, these methods primarily focus on the exploration of spatial relations. However, it has been observed that multi-scan methods [15], [16] exhibit superior performance when it comes to segmenting smaller objects, owing to the strong correlation present in the input data.

To further exploit this inherent correlation, we draw inspiration from spatiotemporal attention methods [47], [48] which capture relations between patches of neighbor frames. We propose a spatiotemporal attention mechanism for radar point clouds, where points of different scans within neighborhoods can directly interact with each other. Instead of maintaining each scan separately in memory to keep time information, we employ a temporal feature to retain the temporal awareness and integrate them into a single frame, thus enhancing the efficiency of our method.

C. Prompt Learning

In recent years, prompt-based learning techniques [49], [50], [51] have emerged in the field of NLP. The main idea of designing prompt functions is to learn specific representations for downstream tasks in a transfer learning model [47]. The introduction of prompts in [52] instructs the frozen model to make predictions for downstream tasks by only learning a small number of parameters of prompts. In [51], a promptable model for image segmentation is introduced, which enables powerful generalization to a range of downstream tasks. The point clouds contain moving objects under a range of scenarios, depending on the weather, road conditions, and traffic at the time of detection. Therefore, we borrow the idea from the prompt-based methods to instruct our network that can be applied in diverse scenarios to generate robust class-specific features for moving objects.

III. Our Approach

A. Network Architecture

In the preprocessing stage, multiple scans of point clouds are first processed to derive temporal features and merged into a single frame as the input point cloud $\mathcal{P}_0$ with N_0 points. As shown in Fig. 2, the encoder consists of the Radar Feature Encoder (RFE), two NFEB, and a PL. The input point cloud $\mathcal{P}_0$ is first fed into the RFE, where each point of $\mathcal{P}_0$ obtains the input feature based on its radar measurements.

To extract the local features of points from their neighborhoods, we follow the PointNet++ [41] to downsample the point clouds and generate partitions, i.e., neighborhoods, around the sampled points. We use two sequential NFEBs, each of which performs downsampling, neighborhood generation, and neighborhood feature extraction. The n-th NFEB (where $n = 1, 2$) takes the point cloud $\mathcal{P}_{n-1}$ as input and outputs downsampled point cloud $\mathcal{P}_n$ with their neighborhood features.

Since $\mathcal{P}_0$ is composed of multiple scans of point clouds, each point is assigned a temporal feature based on its scan moment. To access information from points of preceding scans, we sample N_1 points from the last scanning moment in the $\mathcal{P}_0$, i.e., the points with the largest temporal value relative to the rest points, until the desired number of sampling points is reached. This allows the sampled points to utilize points collected in previous moments as their neighbors. As the point clouds from different scans are highly similar due to the short time interval, our sampling method is more efficient in generating a representative sampled point cloud compared to the farthest sampling method [41].

In the second NFEB, we design a uniform sampling method to ensure the diversity of RCS and velocity values at the sampling points. We predetermined the number of RCS and velocity ranges for the $\mathcal{P}_1$ as 6 in our work. Then, we sort the points according to velocity and RCS. Following each sorting, we evenly split the points into subsets in descending order. Each subset covers a specific range of values, and we sample the same number of points in each range. Note that the number of sampled points under each sorting criterion is half of the total sampling number N_2. In this way, we make the points with various velocities and RCS be focused and learned. The neighborhoods for the sampled points of $\mathcal{P}_n$ are created by searching for their K_n nearest points in the metric space using the K-Nearest Neighbors (KNN) algorithm [41]. The metric chosen for the first NFEB is the Euclidean distance metric,

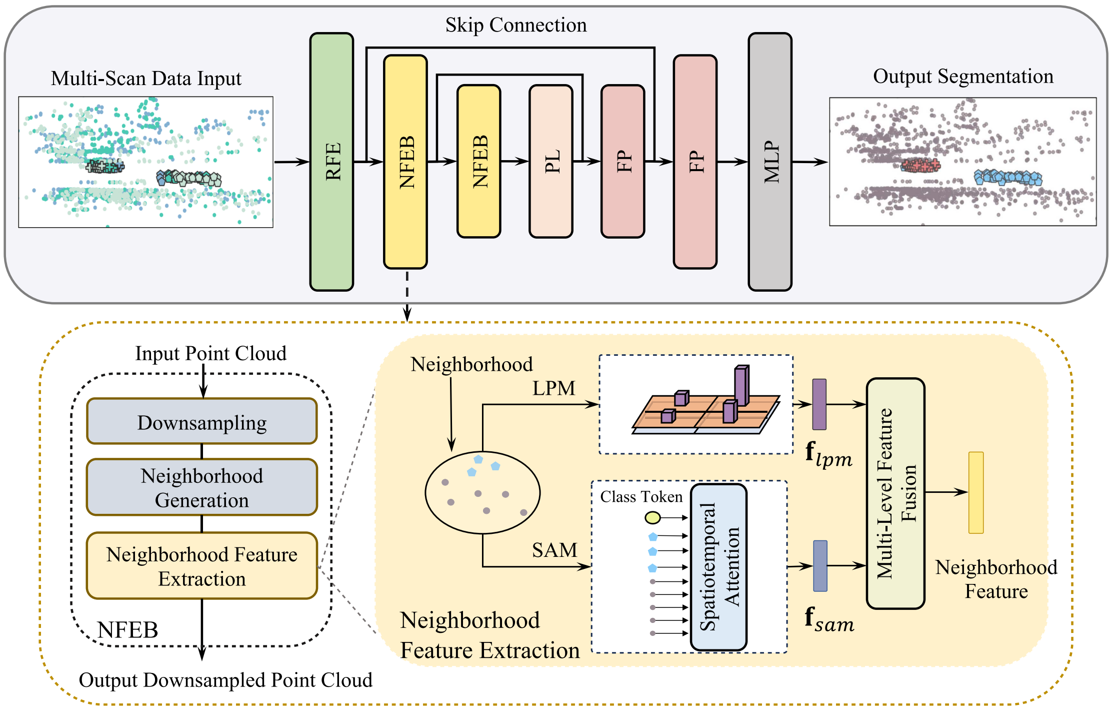

Fig. 2. Overview of the STA-Net. The network takes multi-scan point cloud input with time information. The point cloud is first encoded by the RFE, followed by two NFEBs. Before inputting into the NFEB, the point cloud is downsampled and divided into overlapping local areas. The NFEB consists of two branches: the LPM and the SAM, to extract neighborhood features. The local feature $\mathbf{f}_{lpm}$ and the spatiotemporal feature $\mathbf{f}_{sam}$ are fused in the multi-level feature fusion layer. The PL refines the output feature from the second NFEB. Latent features are propagated through two FP layers to the input point cloud. The MLP outputs class probability for each point.

and the second NFEB employs feature distance measured by similarity. Each NFEB contains two modules: the SAM and the LPM, extracting neighborhood features in parallel. The SAM facilitates interactions between neighbor points within each neighborhood using spatiotemporal attention operations, while the LPM focuses on leveraging radar measurements of neighbor points to generate local radar features.

As the point cloud is successively downsampled, the receptive field of the sampled points is larger, and hence the output features are deeper. The PL refines the neighborhood feature captured by the second NFEB to make them class-specific. Since the semantic segmentation task requires predicting features for each point, we utilize two Feature Propagation (FP) layers in the decoder to progressively propagate semantic features from $\mathcal{P}_2$ to the initial point cloud $\mathcal{P}_0$. Following [15], we perform feature interpolation for points within the denser point cloud through each FP layer. In this process, we calculate the interpolated feature for each output point by averaging the features of three nearest points in the input point cloud, using the same distance metric as the corresponding downsampling layer in the encoder. These interpolated features are then concatenated with their corresponding features in the encoder using a skip connection and merged into a multi-scale feature. We iterate the propagation process until we obtain semantics involving three resolution levels for each point in the input point cloud $\mathcal{P}_0$. Finally, these resulting semantics are input into a Multi-Layer Perception (MLP) as the classification head to produce the segmentation results.

B. Radar Feature Encoder

Radar sensors provide a range of pointwise measurements, including spatial coordinates (x, y), the ego-motion compensated Doppler velocity v, and the RCS σ. We denote each point as a $c = 4$ dimensional vector $\mathbf{p} = (x, y, v, \sigma)$. Since the input is composed of multi-scan data, the spatial coordinates of the points are calculated according to the coordinate system at the first scan. Additionally, the range r and azimuth a of each point with respect to the ego-vehicle are measured during the movement of the ego-vehicle. We define the polar coordinates of each point at the time it was collected as $\hat{\mathbf{p}} = (r, a)$. Note that the polar coordinates are also transformed to the global coordinate system. To leverage the time information in the multi-scan data, we assign each point with a temporal feature t whose value is determined by the temporal order of the scans. If this point is obtained in the first scan, the value of the temporal feature is 0, i.e., $t = 0$. For each point, we generate features by encoding its radar measurements along with the temporal features through the Radar Feature Encoder (RFE):

$$\mathbf{x} = \delta(\mathbf{p}, t) \tag{1}$$

where $\delta : \mathbb{R}^5 \rightarrow \mathbb{R}^d$ is the 1D convolutional layer, and $\mathbf{x} \in \mathbb{R}^d$ is the output feature of point $\mathbf{p}$ from RFE.

C. Spatiotemporal Attention Module

To enhance the understanding of the surroundings of the radar points, we design an attention mechanism that explores

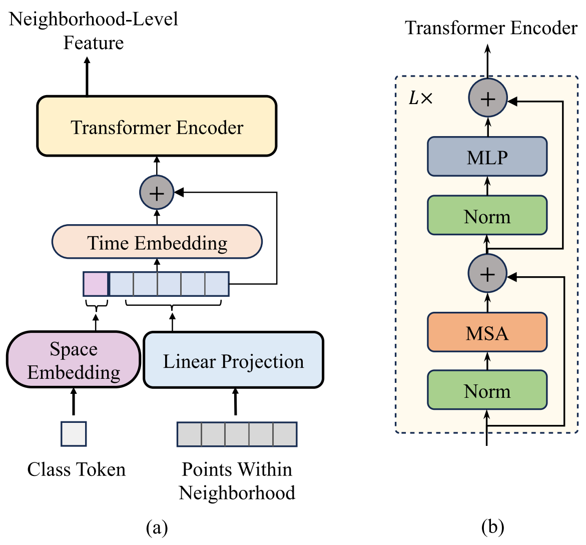

Fig. 3. Illustration of how our spatiotemporal attention works and the structure of the SAM in (a). (b) shows the inner structure of the Transformer Encoder in (a).

relationships among local points in both space and time. The SAM takes the structure of the Vision Transformer (ViT) as a basis, which is a stack of L encoding blocks. Each encoding block consists of two layer-normalization (LN) layers, a spatiotemporal attention layer, and an MLP layer. We illustrate the spatiotemporal attention as well as the structure of SAM in Fig. 3.

The SAM takes as input the neighborhood $\mathcal{P}^{neigh} = \{\mathbf{p}_j | j = 1, \ldots, K\}$ of each centroid point $\mathbf{p}$ and the corresponding feature set $\mathcal{X} = \{\mathbf{x}_j | j = 1, \ldots, K\}$, where $\mathbf{p}_j \in \mathbb{R}^c$ is the neighboring point of $\mathbf{p}$, c is the number of input measurements, and $\mathbf{x}_j \in \mathbb{R}^d$ is the d-dimensional input feature of the j-th neighboring point $\mathbf{p}_j$. The neighboring feature $\mathbf{x}_j^{neigh}$ of each neighboring point $\mathbf{p}_j$ is calculated as:

$$\mathbf{x}_j^{neigh} = [\mathbf{x}_j; \mathbf{p}_j] \tag{2}$$

where $\mathbf{x}_j^{neigh} \in \mathbb{R}^D$ is a feature of dimension $D = c + d$, and $[;]$ denotes the concatenation operation.

1) Class Token: Following the ViT [45] that utilizes a standard Transformer encoder [44], we extend the class token in the ViT, typically serving as image representation in image classification tasks, to represent neighborhoods in point clouds.

As our goal is to extract the neighborhood feature of $\mathbf{p}$ by leveraging the information of neighboring points, the class token can aggregate features of points in the same class with $\mathbf{p}$ while excluding those from different classes to avoid interference. Moreover, this mechanism is suitable for various cases of neighborhoods, which can contain any number of points with diverse temporal features.

2) Space Embedding of Class Token: We use class tokens to integrate information across the neighborhoods of centroid points. The motivation for designing space embedding for the class token is to make it neighborhood-level representative. Instead of directly training the representation of the $\mathbf{x}_0^{neigh}$, we add a space embedding layer to learn the spatial patterns of the neighborhoods. To achieve this, we input the embedding layer with encoded neighborhood information, which is represented by the differences between the centroid point and its neighboring points. The point differences are computed by:

$$\mathbf{r}_j^s = \mathbf{p}_j - \mathbf{p} \tag{3}$$

where $\mathbf{r}_j^s$ represent the differences between the centroid point $\mathbf{p}$ and their neighboring points $\mathbf{p}_j$ in coordinates, velocities, and RCS. While the coordinates of neighboring points relative to the centroid point can reflect their relative positions and the structure of local regions, the relative velocities as well as RCS can indicate the motion of the centroid point.

We concatenate the point differences into a matrix $\mathbf{r}^s = [\mathbf{r}_1^s; \ldots; \mathbf{r}_k^s]$. The matrix is encode by a convolution layer $\mathrm{Conv} : \mathbb{R}^{K \times c} \to \mathbb{R}^D$:

$$\hat{\mathbf{r}}^s = \mathrm{Conv}(\mathbf{r}^s) \tag{4}$$

where $\hat{\mathbf{r}}^s$ is the input vector of dimension D.

Then we obtain the space embedding $\mathbf{e}^s$ for the class token by:

$$\mathbf{e}^s = E_s(\hat{\mathbf{r}}^s) \tag{5}$$

where $E_s : \mathbb{R}^D \to \mathbb{R}^D$ is the space embedding function to generate neighborhood representation that indicates the neighborhood type.

3) Time Encoding: In order to improve the efficiency of point aggregation, position encoding is often applied in self-attention methods since the input sequence lacks positional information. Instead of encoding their relative spatial positions, our research investigates the possibility of encoding time relevance to enable the incorporation and communication of multi-scan data. To encode the input sequence with its temporal information, we first calculate the relative time r_j^t of points in the input sequence by:

$$r_j^t = t_j - t_0 + 1 \tag{6}$$

where t_j is the temporal feature we attached to $\mathbf{p}_j$, and $t_0 = 0$ indicates the fixed temporal feature assigned to $\mathbf{x}_0^{neigh}$. The temporal feature of the class token is independent of those of all other points, which enables the class token to capture the evolution of the entire local area over time. Then we use the trainable 1D embedding function $E_t : \mathbb{R} \to \mathbb{R}^D$ applied to r_j^t and $r_0^t = 1$ to generate time embeddings as:

$$\mathbf{e}_j^t = E_t(r_j^t) \tag{7}$$

where $\mathbf{e}_j^t \in \mathbb{R}^D$ is the time embedding for $\mathbf{x}_j^{neigh}$.

4) Spatiotemporal Attention Layer: Before feeding into the $L = 0$ encoding blocks, the input features are added with the space embeddings and time embeddings:

$$\mathbf{x}_0^{neigh} = \mathbf{e}^s \tag{8}$$

$$\hat{\mathbf{x}}_j^{neigh} = \mathbf{x}_j^{neigh} + \mathbf{e}_j^t \tag{9}$$

where $\hat{\mathbf{x}}_j^{neigh}$ for $j = 0, 1, \ldots, K$ is the j th embedded feature, and the class token is denoted as $\mathbf{x}_0^{neigh} \in \mathbb{R}^D$. The class token

$\mathbf{x}_0^{neigh}$ is concatenated with all $\mathbf{x}_j^{neigh}$ to form an input sequence of length $K+1$:

$$\mathbf{z} = \left[\hat{\mathbf{x}}_0^{neigh}; \hat{\mathbf{x}}_1^{neigh}; \cdots; \hat{\mathbf{x}}_K^{neigh}\right] \tag{10}$$

where $\mathbf{z} \in \mathbb{R}^{(k+1)\times D}$ is the input sequence to the spatiotemporal attention.

At each block $l, l = 1, \ldots, L$, it conducts the following operations:

$$\mathbf{z}^{l-1} = \text{MSA}(\text{LN}(\mathbf{z}^{l-1})) + \mathbf{z}^{l-1} \tag{11}$$

$$\mathbf{z}^{l} = \text{MLP}\left(\text{LN}\left(\mathbf{z}^{l-1}\right)\right) \tag{12}$$

where MSA contains h heads to operate self-attention [44] in parallel, and LN denotes LayerNorm [53]. These heads learn the relations of the input data from different aspects and output features with dimension D_h, which is set to D/n. The MSA concatenates the outputs of each head as its attention output.

After L encoding blocks, we obtain the output features $\mathbf{z}_j^L$ for each input element. When dealing with sparse radar point clouds, the selected neighbors may not belong to the same class. Therefore, the common way to aggregate all the neighboring features into a single neighborhood-level representation may introduce irrelevant information from different classes, resulting in the ambiguity of the final class-specific feature, which is particularly problematic for small objects. To address this problem, we use the feature of the class token to represent the entire neighborhood without the need to design proper integration strategies. The output neighborhood feature $\mathbf{f}_{sam}$ of the SAM is obtained by:

$$\mathbf{f}_{sam} = \mathbf{z}_0^L \tag{13}$$

where $\mathbf{z}_0^L$ is the output feature of the class token from the L-th encoding block.

D. Local Perception Module

While the SAM captures local patterns of each neighborhood based on their input features, the LPM integrated into the NFEB takes the radar measurement of the neighborhood as input to enhance the representation. The LPM leverages radar features of local points to represent the spatial distributions and motion states of the local area. We define a 2D grid, where the R and A axes represent the range and azimuth, respectively. The grid has 16 bins $b_{\mu,\nu}$, $\mu, \nu = 0, \ldots, 3$, arranged in a four-by-four pattern. The range and azimuth ranges of local point sets are divided evenly. Each bin represents an interval with a range size of r_u and an azimuth size of a_u. Then we calculate the indices μ and ν along the R and A axes, respectively, to determine the bin in which each neighbor point falls:

$$\mu = \left\lfloor \frac{r - r_{\min}}{r_u} \right\rfloor \tag{14}$$

$$\nu = \left\lfloor \frac{a - a_{\min}}{a_u} \right\rfloor \tag{15}$$

where $r_{\min}$ and $a_{\min}$ represent the minimum values of range and azimuth, respectively, among all the points in the neighborhood.

To characterize the local areas with the radar measurements, we extract local radar features $\mathbf{g}_{\mu,\nu} \in \mathbb{R}^3$ for each bin which contains $s_{\mu,\nu}$ points. The first dimension is the number of points in the bin. Then we apply max pooling operators on each bin to calculate the maximum velocity and the RCS of points. These operations can be formulated as follows:

$$\mathbf{g}_{\mu,\nu}[0] = s_{\mu,\nu} \tag{16}$$

$$\mathbf{g}_{\mu,\nu}[1] = \max_{s_n=1,\ldots,s_{\mu,\nu}} v_{s_n} \tag{17}$$

$$\mathbf{g}_{\mu,\nu}[2] = \max_{s_n=1,\ldots,s_{\mu,\nu}} \sigma_{s_n} \tag{18}$$

Once we extract bin features $\mathbf{g}_{\mu,\nu}$, we employ a small convolutional neural network γ to capture the overall feature of the entire grid, and then flatten it to a fixed-dimension vector $\mathbf{f}_{lpm}$:

$$\mathbf{f}_{lpm} = \gamma(\mathbf{G}) \tag{19}$$

where $\mathbf{G} \in \mathbb{R}^{4\times4\times3}$ is the feature matrix with elements $\mathbf{G}[\mu,\nu] = \mathbf{g}_{\mu,\nu}$.

For each neighborhood in the NFEB, the local radar feature $\mathbf{f}_{lpm}$ is fused with the deeper feature $\mathbf{f}_{sam}$ output from the SAM by:

$$\mathbf{f}_{neigh} = \text{MLP}([\mathbf{f}_{lpm}; \mathbf{f}_{sam}]) \tag{20}$$

where MLP consists of linear layers and an activation function ReLU attached to the first layer. The neighborhood-level features $\mathbf{f}_{neigh}$ are output as the point features of the downsampled point cloud.

E. Prompt Layer

The motivation to introduce a PL is to identify class-specific representations in the latent space for each moving object, which refines the neighborhood features $\mathbf{f}_{neigh}$ from the second NFEB. We propose a prompt pool consisting of l learnable prompts i.e., $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_l\}$, where each element in $\mathbf{H}$ is a learnable feature with the same dimensional size D as $\mathbf{f}_{neigh}$. Each prompt represents a kind of object and is associated with a key $\mathbf{k}_\omega \in \mathbb{R}^D, \omega = 1, \ldots, l$. The keys are learnable features during the training process. Similar to the scaled dot-product attention [44], we introduce the query $\mathbf{q}$ of $\mathbf{f}_{neigh}$, which is obtained by $\mathbf{q} = \mathbf{f}_{neigh}$ for simplicity. The $\mathbf{q}$ is matched with a prompt key $\mathbf{k}_\omega$ by calculating their similarity:

$$\xi = \arg\max_{\omega} \rho(\mathbf{q}, \mathbf{k}_\omega) \tag{21}$$

where $\rho : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$ is the cosine similarity function and ξ is the best-matched index of the key. As shown in Fig. 4, we assign the prompt h_ξ with the largest similarity value to obtain the class feature $\mathbf{f}_{neigh} = h_\xi$ for the neighborhood. This allows us to achieve refining features for discrimination without the need to make clear boundaries. The prompts are updated during the training stage to be generalizable for each kind of object in different scenarios.

F. Loss Design

There is an imbalance problem due to radar reflections from the static environments accounting for more than 90% of the whole dataset [22]. This leads to the static class contributing to

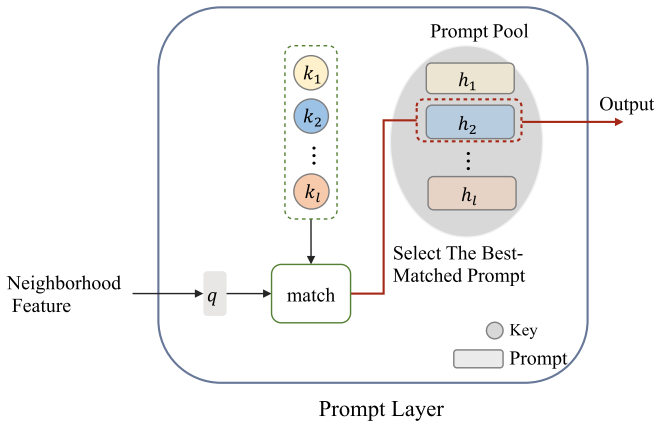

Fig. 4. Illustration of the proposed PL. The query of the input looks up the best match key by calculating their similarities. The prompt with the largest similarity value is selected as the output prompt, which is indicated with the red rectangle.

the majority of the training loss and dominating the gradients. The imbalance can be addressed by introducing a weight vector to scale the loss of each class individually. The weights for different classes are usually set empirically and the weights for classes with vast samplers tend to be a small value.

However, the optimal choice for the weight vector may vary greatly, depending on the distribution of the training data, and thus cannot be generalized to all tasks. Instead of assigning the importance of each class, we differentiate the classes based on their difficulty in training. We choose focal loss [54], which is a reshaped cross entropy function. It is added with a modulating factor $(1 - y_t)^\gamma$, where y_t is the probability of belonging to the ground true label, determined by the trained model. We formulate the loss function for semantic segmentation as follows:

$$\mathcal{L}_{focal} = \sum_{i=1}^{N_0} (1 - y_t^i)^\gamma \log y_t^i \tag{22}$$

where γ is a tunable parameter, smoothly adjusting the rate at which easy samples are down-weighted. Following [54], we use $\gamma = 2$ in our experiments.

In the PL, prompts for classes learn to represent the class-specific feature. To encourage the network to generate discriminative features with the instruction of prompts, we model the prompt learning process by an additional loss function:

$$\mathcal{L}_{prompt} = \sum_{i=1}^{N_2} (1 - \rho((\mathbf{q}^i), \mathbf{k}_\xi^i)) \tag{23}$$

where the similarity function ρ is used to evaluate the choice for the prompt. Since the PL refines the output features of points in $\mathcal{P}_2$ from the second NFEB, only N_2 points contribute to the prompt loss. When the query for point i matches the prompt of index ξ, the similarity value should be high and bring a small loss. We optimize our model by minimizing the overall training loss:

$$\mathcal{L} = \mathcal{L}_{focal} + \alpha \mathcal{L}_{prompt} \tag{24}$$

where α is set to 1, determining the proportion of each term contributed to the final loss.

IV. Experiments

A. Experimental Setting

a) Dataset: We use RadarScenes dataset to evaluate the effectiveness of our method. It contains over 4 hours of driving and a total of 158 sequences in various scenarios. The points are labeled into six categories: car, pedestrian (ped), pedestrian group (ped.group), bike, truck, and static. Following [14], we train our model with 130 recommended sequences and split the rest of the data into a validation set and a test set in the same way for a fair comparison.

b) Implementation details: We implement our network in PyTorch [55]. We collect successive radar scans in 500 ms as one input frame and transform these points to the same car coordinate system as the first scan used. We keep 3072 points in each input radar cloud to simplify the input processing, as the number of points in each scan. In the training stage, if the number of collected points is less than the required number, we use the method in [15] to extend the input points by resampling the first point as many times as the number of vacancies. Otherwise, the points belonging to the static class are randomly discarded. We employ the DBSCAN algorithm [56] to generate pseudo-labels for points. Each point is represented by its RCS and velocity values as its feature. The cluster containing the largest number of points is labeled as the static class. We record the number of valid points in each frame. These resampled points do not contribute to the training or the final assessment.

We adopt the SGD optimizer with a momentum of 0.9 to train the network. The learning rate is set to 0.1 at the beginning of the training, and gradually decreases linearly. We train for 100 epochs with a batch size of 16 on 3 NVIDIA GeForce RTX 3090 GPUs. The number of MSA layers L in the SAM is set to 4. For the number of neighbor points in each NFEB, we set 30 for the first block and 15 for the second.

B. Comparison With State-of-The-Art Methods

a) Evaluation metrics: To evaluate the performance, we employ standard metrics, the macro-averaged F_1 scores suggested by [22] as well as Mean Intersection-over-Union (mIoU) which is commonly used in semantic segmentation tasks. We also report the F_1 and the IoU for each class to show the capability of different methods to segment individual classes.

b) Comparison methods: We compare to the state-of-the-art methods: the adapted PointNet++ [15], which we call RadarPNv1 following [14], model in [16] (called RadarPNv2), Point Transformer [46] and Gaussian Radar Transformer [14]. We implement Point Transformer with available code. The RadarPNv1 and Gaussian Radar Transformer are also reproduced. To facilitate a fair comparison, we refer to the implementation details in the articles for parameter and training settings. The results of RadarPNv2 are referred to as those published in [14].

c) Analysis: Our network demonstrates state-of-the-art performance in terms of both macro F_1 score and mIoU, as indicated in Tables I and II. For a comparison of individual classes, our method obtains the best segmentation results for car, pedestrian, and pedestrian groups. It is worth noticing that the F_1 scores

TABLE I
SEMANTIC SEGMENTATION RESULTS OF MOVING OBJECTS ON THE RADARSCENES TEST SET IN TERMS OF F_1

Method	Type	F_1						macro F_1
		Car	Ped	Ped.group	Bike	Truck	Static	
Point Transformer [46]	single-scan	77.2	45.5	77.4	79.8	73.1	99.4	75.4
Gaussian Radar Transformer [14]	single-scan	77.9	48.2	76.0	72.1	72.7	99.4	74.4
RadarPNv1 [15]	multi-scan	81.3	50.8	73.0	**84.7**	62.5	99.7	75.3
RadarPNv2 [16]	multi-scan	77.9	55.9	73.8	67.5	**75.8**	98.7	75.0
STA-Net(Ours)	multi-scan	**86.3**	**60.2**	**90.1**	80.3	70.6	**99.9**	**81.2**

TABLE II
SEMANTIC SEGMENTATION RESULTS OF MOVING OBJECTS ON THE RADARSCENES TEST SET IN TERMS OF MIOU

Method	Type	IoU						mIoU
		Car	Ped	Ped.group	Bike	Truck	Static	
Point Transformer [46]	single-scan	63.0	29.5	63.1	66.4	57.6	98.7	63.0
Gaussian Radar Transformer [14]	single-scan	63.7	31.7	61.3	56.4	57.1	98.9	61.5
RadarPNv1 [15]	multi-scan	67.0	27.3	64.3	**70.1**	51.6	99.3	63.2
RadarPNv2 [16]	multi-scan	63.8	38.8	58.5	51.0	**61.0**	98.7	61.9
STA-Net(Ours)	multi-scan	**75.9**	**43.1**	**82.0**	67.1	54.6	**99.8**	**70.4**

TABLE III
ABLATION STUDY OF STA-NET

RFE	TE	SE	LPM	PL	F1	mIoU
					61.92	48.53
	✓	✓	✓	✓	75.86	63.4
✓		✓	✓	✓	78.13	66.0
✓	✓		✓	✓	79.85	68.2
✓	✓	✓		✓	78.79	66.9
✓	✓	✓	✓		79.28	67.5
✓	✓	✓	✓	✓	81.23	70.4

and the IoUs for pedestrian and pedestrian group classes are both the highest among all the methods. Those objects are regarded as small objects due to the limited reflections, typically one or two per frame. Multi-scan methods can improve this by accumulating reflections from small objects over consecutive scans. However, this approach is less effective in classifying pedestrian groups. Our method excels in effectively segmenting the two types of small objects without confusing them as shown in Fig. 5. This demonstrates our ability to leverage temporal information embedded in multiple scans to distinguish objects. In comparison to the multi-scan baseline method RadarPNv1, we achieve superior performance in five out of six classes, with the exception of the bike class. We attribute this to the limited number of bike samples, causing performance variations among different multi-scan classifiers. Additionally, our method outperforms the single-scan methods listed in the table, except for the truck class. This may be primarily due to the narrow-shape distribution of trucks in a multi-scan setting, posing a challenge to cluster all points from trucks using Euclidean distance.

C. Ablation Study

In this section, we conduct ablation studies to verify the improvements of each individual module in our network. The results are presented in Table III, which shows that each core design is critical to improving the overall performance.

a) RFE: The first ablation study substitutes the RFE layer with a fully connected layer, followed by a layer norm and a LeakyReLU. We keep the rest of the network unchanged. The performance drops significantly, which indicates that encoding the input feature is vital for the learning of subsequent layers. Our RFE layer can effectively encode input points with their radar measurements, thus enhancing the overall network performance.

b) Time encoding (TE): To quantify the benefit of exploring temporal information, we conduct an experiment in which we remove the temporal feature and replace the time embedding with absolute position embedding, which is a common scheme in the ViT. Without the inclusion of time information, we observe decreases in F_1 score and mIoU by 3.1% and 4.4%, respectively.

c) Space encoding (SE): We also investigate the effectiveness of the initialization of class tokens. We remove the encoding layers added to the class token. The class token is replaced with a learnable vector that is randomly initialized in the first training epoch and is updated by backpropagation. The removal of the SE results in decreases in both F_1 and mIoU, indicating that encoding the relative positions within local areas is helpful in distinguishing moving objects from static backgrounds.

d) LPM: To evaluate the impact of LPM on improving the representations of neighborhood features, we conduct an experiment where we disable LPM and only use the spatiotemporal feature $\mathbf{f}_{sam}$ that outputs from the SAM. As the row 5 in Table III shows, the removal of LPM results in a degradation in overall performance.

e) PL: To study the effectiveness of the prompt layer, we evaluate the performance of our network without employing the PL. As anticipated, the F_1 score and mIoU show a drop, which reveals that the proposed PL can make class features more discriminative.

D. Runtime Evaluation

Table IV reports a summary of the model parameters, Floating Point Operations (FLOPs), and the average reference time of the selected models tested on the RadarScenes test set. Our model is highly efficient with the shortest reference time of 34.94 ms. Furthermore, in comparison to the Point Transformer and Gaussian Radar Transformer, our model exhibits much lower model size and FLOPs.

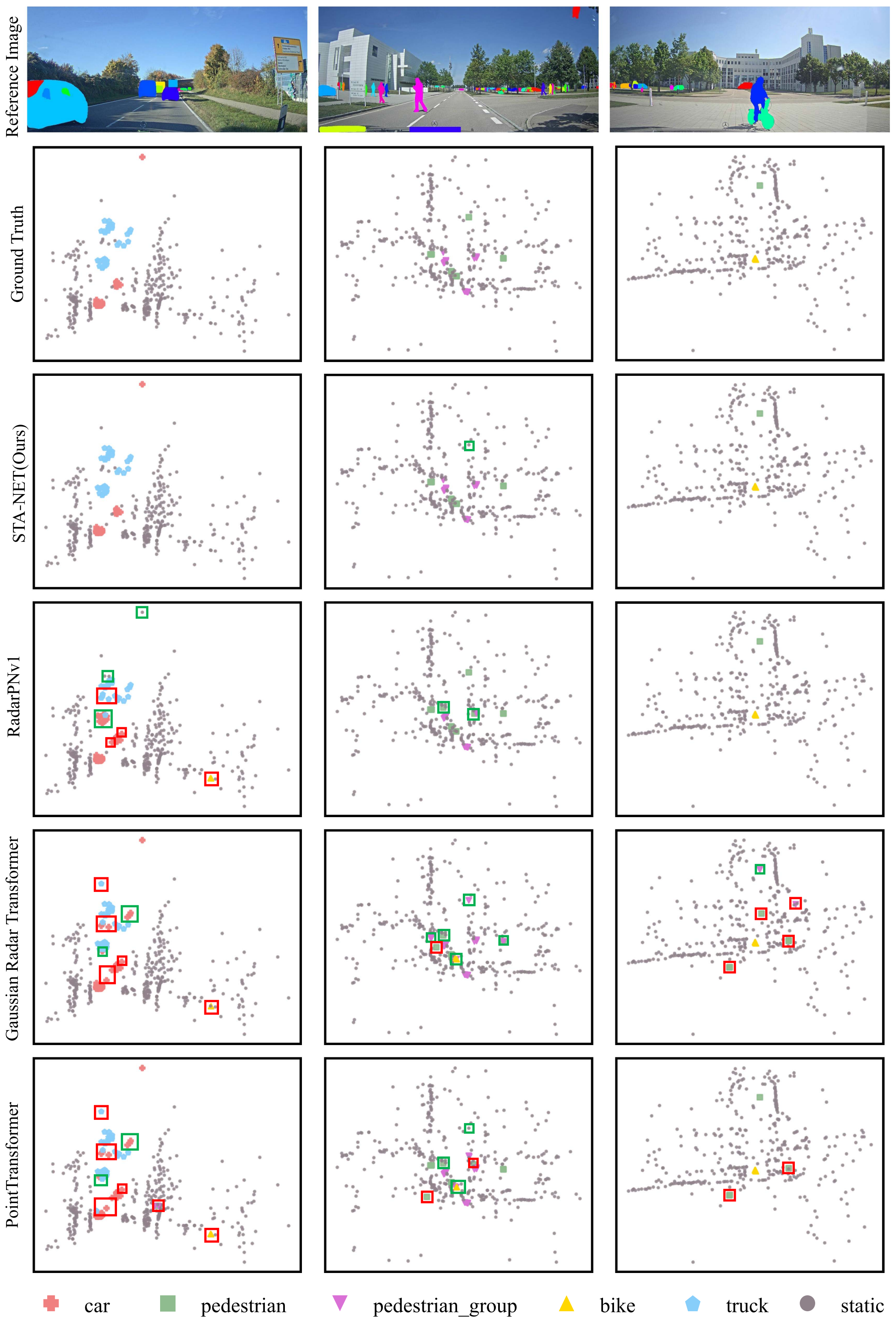

Fig. 5. Visualization of semantic segmentation results on the RadarScenes dataset. Each point is represented by a symbol, and the type of symbol represents the class of the point. The red rectangles indicate the background points that are misclassified as moving objects, while the green rectangles indicate that the types of the objects are incorrectly predicted.

TABLE IV
RUNTIME PERFORMANCE ON THE RADARSENES TEST SET

Method	Parameters (M)	FLOPs (G)	Time (ms)
Point Transformer [46]	91.67	19.39	58.09
Gaussian Radar Transformer [14]	108.74	19.30	112.93
RadarPNv1 [15]	**1.77**	**0.52**	278.07
STA-Net (Ours)	7.36	5.78	**34.94**

V. CONCLUSION

We presented a novel semantic segmentation approach to multi-scan radar point clouds in automotive scenarios. Instead of building complex communication mechanisms between scans, we enable multi-scan points to interact in an effective and simple way that operates spatio-temporal attention on local points with different time embeddings. This allows our network to successfully extract features of moving objects over a period of up to 500 ms and to be sensitive to small objects. We also demonstrate that radar measurements can be utilized to enhance the representation of local areas of the point clouds. The segmentation performance is further improved by our proposed PL, which makes the class features of points more distinguishing. Our approach achieved state-of-the-art performance on RadarScenes both in macro F_1 score and mIoU. In addition, our network has the fastest inference time compared to all other methods. We hope our work can provide inspiration and ideas for studying hidden temporal correlations in radar data and the design of radar perception networks.

REFERENCES

[1] Y. Zhang, A. Carballo, H. Yang, and K. Takeda, "Perception and sensing for autonomous vehicles under adverse weather conditions: A survey," *ISPRS J. Photogrammetry Remote Sens.*, vol. 196, pp. 146–177, 2023.

[2] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, and M. A. Kaafar, "The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 103–111, Jun. 2019.

[3] X. Cai, M. Giallorenzo, and K. Sarabandi, "Machine learning-based target classification for MMW radar in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 6, no. 4, pp. 678–689, Dec. 2021.

[4] A. Srivastav and S. Mandal, "Radars for autonomous driving: A review of deep learning methods and challenges," *IEEE Access*, vol. 11, pp. 97147–97168, 2023.

[5] S. Sun and Y. D. Zhang, "4D automotive radar sensing for autonomous vehicles: A sparsity-oriented approach," *IEEE J. Sel. Topics Signal Process.*, vol. 15, no. 4, pp. 879–891, Jun. 2021.

[6] J. Dickmann et al., "Automotive radar the key technology for autonomous driving: From detection and ranging to environmental understanding," in *Proc. IEEE Radar Conf.*, 2016, pp. 1–6.

[7] P. R. M. de Araujo, M. Elhabiby, S. Givigi, and A. Noureldin, "A novel method for land vehicle positioning: Invariant kalman filters and deep-learning-based radar speed estimation," *IEEE Trans. Intell. Veh.*, vol. 8, no. 9, pp. 4275–4286, Sep. 2023.

[8] S. Lu et al., "Efficient deep-learning 4D automotive radar odometry method," *IEEE Trans. Intell. Veh.*, early access, Sep. 01, 2023, doi: 10.1109/TIV.2023.3311102.

[9] L. Wang et al., "Interfusion: Interaction-based 4D radar and lidar fusion for 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 12247–12253.

[10] L. Zheng et al., "RCFusion: Fusing 4-D radar and camera with bird's-eye view features for 3-D object detection," *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 8503814.

[11] B. Yang, R. Guo, M. Liang, S. Casas, and R. Urtasun, "RadarNet: Exploiting radar for robust perception of dynamic objects," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 496–512.

[12] J. Liu et al., "Deep instance segmentation with automotive radar detection points," *IEEE Trans. Intell. Veh.*, vol. 8, no. 1, pp. 84–94, Jan. 2023.

[13] Y. Zhuang, B. Wang, J. Huai, and M. Li, "4D iRIOM: 4D imaging radar inertial odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 8, no. 6, pp. 3246–3253, Jun. 2023.

[14] M. Zeller, J. Behley, M. Heidingsfeld, and C. Stachniss, "Gaussian radar transformer for semantic segmentation in noisy radar data," *IEEE Robot. Automat. Lett.*, vol. 8, no. 1, pp. 344–351, Jan. 2023.

[15] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, "Semantic segmentation on radar point clouds," in *Proc. IEEE 21st Int. Conf. Inf. Fusion*, 2018, pp. 2179–2186.

[16] O. Schumann, J. Lombacher, M. Hahn, C. Wöhler, and J. Dickmann, "Scene understanding with automotive radar," *IEEE Trans. Intell. Veh.*, vol. 5, no. 2, pp. 188–203, Jun. 2020.

[17] F. Ding, Z. Pan, Y. Deng, J. Deng, and C. X. Lu, "Self-supervised scene flow estimation with 4-D automotive radar," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 8233–8240, Jul. 2022.

[18] F. Ding, A. Palffy, D. M. Gavrila, and C. X. Lu, "Hidden gems: 4D radar scene flow learning using cross-modal supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 9340–9349.

[19] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

[20] J. Karangwa, J. Liu, and Z. Zeng, "Vehicle detection for autonomous driving: A review of algorithms and datasets," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 11568–11594, Nov. 2023.

[21] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[22] O. Schumann et al., "Radarscenes: A real-world radar point cloud data set for automotive applications," in *Proc. IEEE 24th Int. Conf. Inf. Fusion*, 2021, pp. 1–8.

[23] M. Mostajabi, C. M. Wang, D. Ranjan, and G. Hsyu, "High resolution radar dataset for semi-supervised learning of dynamic objects," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 450–457.

[24] M. Sheeny, E. De Pellegrin, S. Mukherjee, A. Ahrabian, S. Wang, and A. Wallace, "RADIATE: A radar dataset for automotive perception in bad weather," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 1–7.

[25] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 6433–6438.

[26] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "MulRan: Multimodal range dataset for urban place recognition," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 6246–6253.

[27] A. Ouaknine, A. Newson, J. Rebut, F. Tupin, and P. Pérez, "Carrada dataset: Camera and automotive radar with range-angle-doppler annotations," in *Proc. IEEE 25th Int. Conf. Pattern Recognit.*, 2021, pp. 5068–5075.

[28] Y. Wang, Z. Jiang, X. Gao, J.-N. Hwang, G. Xing, and H. Liu, "Rodnet: Radar object detection using cross-modal supervision," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 504–513.

[29] A. Zhang, F. E. Nowruzi, and R. Laganiere, "Raddet: Range-azimuth-doppler based radar object detection for dynamic road users," in *Proc. IEEE 18th Conf. Robots Vis.*, 2021, pp. 95–102.

[30] M. Bijelic et al., "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11682–11692.

[31] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3D object detection," in *Proc. IEEE 16th Eur. Radar Conf.*, 2019, pp. 129–132.

[32] A. Palffy, E. Pool, S. Baratam, J. F. P. Kooij, and D. M. Gavrila, "Multi-class road user detection with 3+1D radar in the view-of-delft dataset," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4961–4968, Apr. 2022.

[33] L. Zheng et al., "TJ4DRadset: A 4D radar dataset for autonomous driving," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst.*, 2022, pp. 493–498.

[34] D.-H. Paek, S.-H. Kong, and K. T. Wijaya, "K-radar: 4D radar object detection for autonomous driving in various weather conditions," in *Proc. 36th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, 2022, pp. 3819–3829.

[35] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.

[36] Y. Ben-Shabat, M. Lindenbaum, and A. Fischer, "3DmFV: Three-dimensional point cloud classification in real-time using convolutional neural networks," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3145–3152, Oct. 2018.

[37] J. Park, C. Kim, S. Kim, and K. Jo, "PCSCNet: Fast 3D semantic segmentation of LiDAR point cloud for autonomous car using point convolution and sparse convolution network," *Expert Syst. Appl.*, vol. 212, 2023, Art. no. 118815.

[38] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++ : Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 4213–4220.

[39] C. Xu et al., "SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 1–19.

[40] Y. Zhang et al., "PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9598–9607.

[41] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet : Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[42] 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," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 12547–12556.

[43] 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," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16024–16033.

[44] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[45] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–22.

[46] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16259–16268.

[47] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, no. 1, pp. 5485–5551, 2020.

[48] Y. Zhao, Z. Li, X. Guo, and Y. Lu, "Alignment-guided temporal attention for video action recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 13627–13639.

[49] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, 2023.

[50] J. Gu et al., "A systematic survey of prompt engineering on vision-language foundation models," 2023, *arXiv:2307.12980*.

[51] A. Kirillov et al., "Segment anything," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.

[52] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 4582–4597.

[53] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[54] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[55] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[56] M. Ester et al., "A density-based algorithm for discovering clusters in large spatial databases with noise," *kdd*, vol. 96, no. 34, pp. 226–231, 1996.

Ziwei Zhang received the B.S. degree in electronic and information engineering from Xidian University, Xi'an, China, in 2021. She is currently a Graduate Student in information and communication engineering with the University of Science and Technology of China, Hefei, China. Her research interests include radar semantic segmentation, hyperspectral image processing, and machine learning.

Jun Liu (Senior Member, IEEE) received the B.S. degree in mathematics from the Wuhan University of Technology, Wuhan, China, in 2006, the M.S. degree in mathematics from Chinese Academy of Sciences, Beijing, China, in 2009, and the Ph.D. degree in electrical engineering from Xidian University, Xi'an, China, in 2012. From July 2012 to December 2012, he was a Postdoctoral Research Associate with the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA. From 2013 to 2014, he was a Postdoctoral Research Associate with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA. He is currently an Associate Professor with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China. He is the co-author of two books *Advances in Adaptive Radar Detection and Range Estimation* (Springer) in 2022 and *Adaptive Detection of Multichannel Signals Exploiting Persymmetry* (CRC Press) in 2023. His research interests include statistical signal processing, image processing, and machine learning. He is a Member of the Sensor Array and Multichannel (SAM) Technical Committee, IEEE Signal Processing Society. He is currently an Associate Editor for IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS, and a member of the Editorial Board of the *Signal Processing* (Elsevier). From February 2018 to February 2023, he was an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS. He was the recipient of the Best Paper Award from the IEEE WCSP 2021.

Guangfeng Jiang received the B.S. degree in communication engineering from the Shandong University, Jinan, China, in 2021. He is currently working toward the Ph.D. degree in information and communication engineering with the University of Science and Technology of China, Hefei, China. His research interests include autonomous driving and 3D computer vision.