Sensor applications

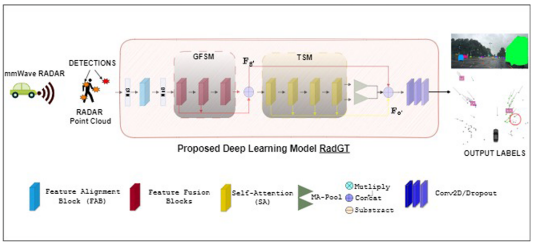
RadGT: Graph and Transformer-Based Automotive Radar Point Cloud Segmentation

Rasim A. Sevimli[1], Murat Üçüncü[1], and Aykut Koç[2,3,*]

[1]Department of Electrical and Electronics Engineering, Baskent University, 06810 Ankara, Türkiye
[2]Department of Electrical and Electronics Engineering, Bilkent University, 06800 Ankara, Türkiye
[3]UMRAM, Bilkent University, 06800 Ankara, Türkiye
*Senior Member, IEEE

Manuscript received 3 September 2023; revised 3 October 2023; accepted 21 October 2023. Date of publication 25 October 2023; date of current version 7 November 2023.

Abstract—The need for visual perception systems providing situational awareness to autonomous vehicles has grown significantly. While traditional deep neural networks are effective for solving 2-D Euclidean problems, point cloud analysis, particularly for radar data, contains unique challenges because of the irregular geometry of point clouds. This letter proposes a novel transformer-based architecture for radar point clouds adapted to the graph signal processing (GSP) framework, designed to handle non-Euclidean and irregular signal structures. We provide experimental results by using well-established benchmarks on the nuScenes and RadarScenes datasets to validate our proposed method.



Index Terms—Sensor applications, automotive RADAR, graph signal processing (GSP), point cloud processing, segmentation, transformers.

I. INTRODUCTION

The need for enhanced perception capabilities for autonomous driving technology has become crucial [1], [2]. Previous research extensively investigated the classification and segmentation of point cloud data, specifically focusing on light detection and ranging (LiDAR) technology [3]. The use of automotive radars has increased significantly in the automobile sector, in line with the developments in millimeter-wave radar technology [4]. Therefore, automotive radar has become an alternative to camera and LiDAR sensors for object recognition and scene interpretation.

Recent research indicated that point clouds can be effectively processed by four distinct models: Multiview-based and volumetric-based models, point-based models, graph-based models, and transformer-based models. Advances in deep neural networks have demonstrated that the point cloud interaction of irregular fields can be exploited in deep graph representation. These developments have paved the way for significant advances in graph-based signal processing [5].

The transformer architecture has succeeded significantly in various fields, such as natural language processing [6] and image processing. The transformer's strength lies in its ability to effectively capture relationships of distant connections and enable efficient parallel processing [7].

This letter presents RadGT: a transformer-derived model built upon the graph signal processing (GSP) structure. The objective is to accurately and efficiently categorize multiclass automotive radar detections. RadGT utilizes important radar characteristics, such as spatial coordinates, speed, radar cross section (RCS), and GSP-based data. While existing literature have introduced various techniques, including deep learning approaches, for processing RADAR point clouds, these methods primarily excel in 2-D Euclidean domains. RADAR data, conversely, are collected on irregular grids, making it better suited for GSP methods. Our method effectively captures the dependencies among points within local neighborhoods while assigning distinct weights to neighboring points and learning comprehensive global features. Consequently, our primary motivation is to explore and formulate a GSP and transformer-based methodology that can effectively handle the inherently irregular nature of RADAR point clouds. This method allows for segmentation using unprocessed point cloud data, eliminating the need for preceding clustering algorithms or preprocessing methods. We employed RadarScenes [8] and nuScenes [9] datasets in our experiments to compare the RadGT to the most recent baseline approaches. To conclude, we present RadGT, a model designed for automotive radar sensors that utilizes point cloud irregularities to incorporate local and global features. Besides, we have developed a novel feature extraction component that enhances feature vectors by using GSP. Finally, we proved the efficiency and the improved performance of our proposed method RadGT, using RadarScenes and nuScenes datasets.

II. RELATED WORK

Researchers are adapting deep learning algorithms originally designed for LiDAR data to evaluate radar data [10], [11], [12], leveraging the synergistic potential of these technologies to enhance the identification and categorization of objects in self-driving contexts. Multiview models employ image-based rendering and view-pooling methods to gather information from various perspectives. However, they face difficulties due to higher computational requirements at greater resolutions and the loss of important details at lower resolutions. Point-based

Corresponding Author: Aykut Koç (e-mail: aykut.koc@bilkent.edu.tr).
Associate Editor: M. K. Shukla.
Data is available online at https://github.com/koc-lab/RadGT.
Digital Object Identifier 10.1109/LSENS.2023.3327593

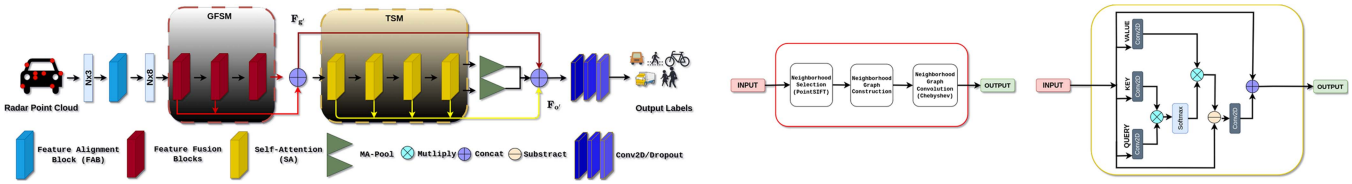


Fig. 1. Proposed RadGT architecture.

models seek to extract features from an unstructured set of points directly. However, the sampling and point grouping steps introduce extra computational burdens when working with extensive point clouds. Graph-based models utilize the inherent graph-like structure in point clouds to capture spatial connections among points. Svenningsson et al. [10] introduced Radar-PointGNN by adapting the Point-GNN model [13] for radar data. Dynamic graph convolutional neural networks (DGCNN) [14] captures local geometric relationships for feature extraction and classification, while PointNGCNN [15] combines PointNet++ and graph neural network (GNNs) to improve context modeling. Sevimli et al. [11] extended the study to graph-based radar versions of DGCNN and PointNGCNN. Transformer-based models extract features, uncover relationships, and learn semantics in point cloud processing, especially in LiDAR. Point cloud transformer [16] innovatively extracts local features using intra-domain self-attention (SA). Zhou et al. [17] addressed spatial relationships using a graph and transformer-based framework along with other studies exploring transformer-based methods for LiDAR.

III. RADGT

RadGT is a transformer-based model that leverages GSP techniques for radar point cloud segmentation. RadGT effectively learns local and global features from input data by extracting specific features derived by using GSP and geometric coordinates. RadGT enhances the representation of features and subsequently improves the efficacy of segmenting radar point clouds. Fig. 1 depicts the model's architecture for the proposed method RadGT. RadGT comprises three subsequent modules: The feature alignment block (FAB), the graph feature space module (GFSM), and the transformer-based segmentation module (TSM). RadGT initially constructs graph representations from input point clouds, which the FAB then employs. The FAB is introduced to produce enhanced point cloud data and derive features from the graphs. In GFSM, we employ feature fusion blocks (FFBs), which play a crucial role in overcoming the challenge of integrating multiscale and multilevel features in point cloud analysis, as described in [15]. GFSM allows our model to capture fine-grained details and contextual relationships in the input point cloud data. The next main block is the TSM containing SA blocks. The initial output of the GFSM is the input to the first SA block. The SA blocks allow the model to assign varying degrees of importance to different parts of the input data, enhancing its ability to understand complex relationships and make more informed decisions.

A. Preprocessing

The input radar point cloud $\mathbf{P} = \mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N$, is described as follows: Each point $\mathbf{p}_i$ within the set consists of a 3-D Cartesian coordinate vector $\mathbf{c}_i \in \mathbb{R}^3$, represented as $\mathbf{c}_i = (\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i)$, and an associated state vector $\mathbf{s}_i \in \mathbb{R}^k$, where k represents the dimensionality of the supplementary radar point cloud attributes. The preprocessing step involves capturing the inherent patterns and dependencies present in the graph structure. A graph can be defined as $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ to convert the point cloud data into a problem based on graphs. In this context, $\mathcal{V}$ represents a finite set of N vertices. $\mathcal{E}$ stands for the connection existing between these vertices, and $\mathbf{W}$ symbolizes the matrix that contains the weights associated with these connections. In this letter, the edge set $\mathcal{E}$ is formed as $\mathcal{E} = \{(i, j) | \ \|\mathbf{c}_i - \mathbf{c}_j\| < R\}$ Here, R represents the measurement of the radius in meters. Gaussian function is used to establish the weights $\mathbf{W}$ representing the connection strengths between vertices i and j: $\mathbf{W}_{ij} = \exp(\frac{-d(i,j)^2}{2\sigma^2})$, if $(i, j) \in \mathcal{E}$.

Let us assume a graph signal $\mathbf{f} \in \mathbb{C}^N$ that is a mapping from $\mathcal{V}$ to $\mathbb{C}^N$ such that $\mathbf{f} : \mathcal{V} \to \mathbb{C}^N$ and $v_n \to f_n$. In GSP context, a graph signal can be understood as a discrete representation of a continuous signal, which is established on the graph's vertices. We model RCS and velocity as graph signals in our setting and utilize the graph Fourier transform (GFT) to examine the spectral domain attributes, allowing for the analysis of frequency constituents and extracting pertinent features. The procedure of eigendecomposition of the graph Laplacian is used to examine the graph signal in the spectral domain. For this purpose, a graph Laplacian matrix $\mathbf{L}$ is given by $\mathbf{L} = \mathbf{D} - \mathbf{W}$. Here, $\mathbf{D}$ represents a diagonal matrix that corresponds to the nodes' degree in an undirected graph, computed as $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. In essence, $\mathbf{L}$ is real, symmetric, and positive semidefinite matrix; therefore, all orthonormal eigenvectors of $\mathbf{L}$, i.e., $\{\mathbf{e}_l\}_{l=0,1\ldots N-1}$ have nonnegative real eigenvalues. It is possible to order the eigenvalues of matrix $\mathbf{L}$ in a way such that they are in increasing sequence as $0 = \lambda_0 \leqslant \lambda_1 \leqslant \cdots \leqslant \lambda_{N-1}$. Here, N denotes the overall count of nodes present within the graph. In the realm of GSP, these eigenvalues are commonly understood as representing frequencies. Consequently, the GFT of a graph signal is calculated as follows:

$$\mathcal{GF}[\mathbf{f}](\lambda_l) = \widehat{\mathbf{f}}(\lambda_l) = < \mathbf{f}, \mathbf{e}_l > = \sum_{i=0}^{N-1} f(i) e_l^*(i) \tag{1}$$

where $\widehat{\mathbf{f}}(\lambda_l)$ is the GFT of the graph signal $\mathbf{f}$, and $e_l^*(i)$ is the orthonormal eigenvectors.

As a result, extracting characteristics from graphs hinges on conducting the GFT on RCS and velocity information. Moreover, the feature extraction procedure encompasses the node connectivity count, $\deg(\mathbf{p}_i)$, spatial coordinates, and RCS and velocity values. Subsequently, the proposed feature vector in FAB, denoted as $\mathbf{F}_{e,i} \in \mathbb{R}^{N \times 8}$, is formulated as follows:

$$\mathbf{F}_{e,i} = \{c_i, \sigma_i, v_i, \deg(\mathbf{p}_i), \mathcal{GF}[\sigma_i], \mathcal{GF}[v_i]\} \tag{2}$$

where c_i represents the spatial coordinates, σ_i represents the RCS, v_i represents the velocity measured by the sensor, and $\deg(\mathbf{p}_i)$ indicates the count of edges that are linked to $\mathbf{p}_i$ within a 1-m distance. $\mathcal{GF}[\sigma_i]$ stands for the GFT of the RCS, while $\mathcal{GF}[v_i]$ corresponds to the GFT of the velocity. Through the extraction of features from the raw radar point cloud, we are able to transform the input point cloud into a higher dimensional representation denoted as $\mathbf{F}_{e,i}$. Consequently, we generate the 8-D proposed feature vectors within the FAB to serve as a basis for the GFSM.

B. Graph Feature Space Module (GFSM)

The module containing three FFBs takes high-dimensional feature vectors, represented as F_e, as input. The FFB architecture is depicted with red rectangles in Fig. 1. At first, we choose the k-nearest neighbors for each center and then convert them into the center's local coordinates. It is essential to note some nuances in this process. GFSM adopts the PointSIFT algorithm [18] to use information from eight different orientations. PointSIFT successfully selects neighborhood points, allowing for the extraction of comprehensive neighborhood features, which are then treated as graph signals. Another notable improvement over a similar method, i.e., *EdgeConv*, [14] is the introduction of a graph filter, which enables us to gather extra geometric information from the surrounding area. The features for each data center are generated by combining the Laplacian and feature matrices after graph filters.

Graph convolution operations extract information from local neighborhoods as the data format is not regular. When contrasting graph-based spectral domain computation with traditional signal processing, the utilization of the Laplacian matrix resembles the application of the Fourier basis. The calculation of spectral filtering for the graph signal $\mathbf{u}$ can be determined by $\mathbf{u}' = \mathbf{U}\boldsymbol{\Theta}'\mathbf{U}^T\mathbf{u}$, where the output $\mathbf{u}'$ represents the filtered graph signal, $\boldsymbol{\Theta}'$ is a diagonal learnable matrix, and $\mathbf{U}$ is the eigenvector of the Laplacian matrix $\mathbf{L}$. The matrix $\boldsymbol{\Theta}'$ is expressed as a function of the eigenvector Λ obtained from the graph Laplacian matrix $\mathbf{L}$. As a result, $\mathbf{u}'$ can be rewritten by using Chebyshev polynomials approximation as

$$\mathbf{u}' = \mathbf{U}\boldsymbol{\Theta}'(\Lambda)\mathbf{U}^T = \sum_{k=0}^{P-1} \varphi_k \mathbf{U}\mathrm{T}_k(\overline{\boldsymbol{\Lambda}})\mathbf{U}^T\mathbf{u} = \sum_{k=0}^{P-1} \varphi_k \mathrm{T}_k(\overline{\mathbf{L}})\mathbf{u} \quad (3)$$

where $\overline{\mathbf{L}} = 2\mathbf{L}/\lambda_{\max} - \mathbf{I}$, $\varphi_0, \ldots, \varphi_{P-1}$ are system parameters with the order of the polynomials P, and $\mathrm{T}_k(\overline{\mathbf{L}})$ represents the Chebyshev polynomial with a degree of k. Then, $\mathbf{y}$ is calculated as $\mathbf{y} =$ rectified linear unit (ReLU) $(\mathbf{u}'\Omega + \mathbf{b})$, where $\mathbf{b} \in \mathbb{R}^{N\times F_g}$ is the bias term, Ω denotes weight matrix, $\mathbf{y} \in \mathbb{R}^{N\times F_g}$ is the last output, and ReLU is the activation function. Here, $\mathbf{b}$ and Ω are trainable parameters.

C. Transformer Module (TSM)

In order to apply the transformer architecture to tasks involving point clouds, the fundamental approach involves in considering the complete point cloud as a sentence, where individual points play a role similar to words, such as in the original transformer introduced to process natural language. To this end, we construct TSM with the SA blocks at its core, as shown in Fig. 1. TSM aims to capture long-range dependencies and global context information. The SA block can also compute relationships between points independently of their positions regardless of their specific order. The initial input to the first SA block is obtained by utilizing the concatenated output of the GFSM, $\mathbf{F}_{g'}$. The three primary elements *Queries* ($\mathbf{Q}$), *Keys* ($\mathbf{K}$), and *Values* ($\mathbf{V}$) of the SA mechanism of the transformer are used to compute attention scores capturing the relationships among the elements of the input sequence [7]. Three distinct matrices are created by performing linear transformations on the original input, $\mathbf{F}_{g'}$. The generation of attention weights involves multiplying the query $\mathbf{Q}$ and key $\mathbf{K}$ vectors, resulting in a matrix depicting the significance of every pair of particles. After normalization, these attention weights reflect the weighted relevance of each particle pair. Subsequently, the SA mechanism is achieved by taking the product of the attention weights and the $\mathbf{V}$ matrix. This produces a weighted composition of the values. To facilitate these operations, Conv2D layers are employed for linear transformations as $\mathbf{Q}, \mathbf{K}, \mathbf{V} = \mathbf{F}_{g'}.(\widetilde{\mathbf{W}}_\mathbf{q}, \widetilde{\mathbf{W}}_\mathbf{k}, \widetilde{\mathbf{W}}_\mathbf{v})$, where $d_{\text{in}} = d_{\text{out}}/4$, $\mathbf{Q}, \mathbf{K} \in \mathbb{R}^{N\times d_{\text{in}}}$ and $\mathbf{V} \in \mathbb{R}^{N\times d_{\text{out}}}$.

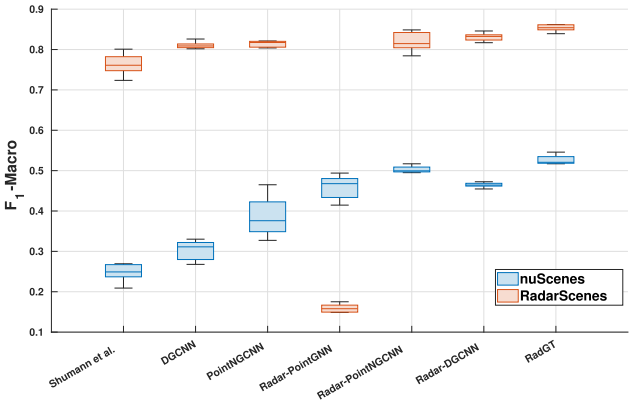


Fig. 2. F_1-Macro scores obtained using fivefold cross-validation.

The trainable linear coefficients $\widetilde{\mathbf{W}}_\mathbf{q}$, $\widetilde{\mathbf{W}}_\mathbf{k}$, and $\widetilde{\mathbf{W}}_\mathbf{v}$ are the matrices, which can be adjusted during the training process. The attention weights of $\mathbf{A}$ are computed by performing matrix multiplication between $\mathbf{Q}$ and $\mathbf{K}^T$ as $\mathbf{A} = \mathrm{Softmax}(\mathbf{Q}\mathbf{K}^T)/N$, where N is the dimension of the matrix $\mathbf{A}$. Then, the SA output features $\mathbf{F}_{\mathbf{sa}}$ is obtained as $\mathbf{F}_{\mathbf{sa}} = \mathbf{A}\mathbf{V}$,where $\mathbf{F}_{\mathbf{sa}} \in \mathbb{R}^{N\times d_{\text{out}}}$. While the initial transformer design employs $\mathbf{F}_{\mathbf{sa}}$ for obtaining individual object-specific absolute attention, Guo et al. [16] investigated that incorporating offset-attention (OA) might lead to enhanced classification performance. The OA layer computes the disparity or variation between the SA characteristics. The Conv2D processes the discrepancy between $\mathbf{F}_{g'}$ and $\mathbf{F}_{\mathbf{sa}}$, which is referred to as OA, and then, the result is added to the initial inputs. Finally, the output is obtained as $\mathbf{F}_\mathbf{o} = \mathrm{Conv2D}(\mathbf{F}_{g'} - \mathbf{F}_{\mathbf{sa}}) + \mathbf{F}_{g'}$. To construct the overall model, the four SA layers are merged by concatenation along the feature dimension. In addition, we perform separate calculations for Max-(MP) and Average-(AP) Pooling. These results are then combined with the output of the final SA block, referred to as MA-Pool in the diagram. At the end, all four outputs including $\mathbf{F}_{o'}$, $\mathbf{F}_{g'}$, and MA-Pool results are concatenated. Then, the output of concatenation is followed by convolution and dropout layers.

IV. EXPERIMENTS

We evaluated RadGT on common 3-D mm-Wave benchmarks: nuScenes [9] and RadarScenes [8]. Adhering to established practices [10], [11], [19], we partition the training samples with 60%, 20%, and 20% training, validation, and test partitionings, respectively. The nuScenes dataset is categorized as *Vehicle*, *Pedestrian*, *Cycle*, and *None*. The RadarScenes dataset encompasses six object classes: *Vehicle*, *Pedestrian*, *Truck*, *Cycle*, *Pedestrian-Group*, and *None*.

The GSFM of the proposed RadGT uses three FFBs, each with predetermined input–output dimensions: (8, 64), (64, 64), and (64, 128). Likewise, the TSM includes four SA blocks with consistent input–output dimensions of (128, 128). RadGT is trained for 50 epochs on both datasets. This is done by using a batch size of four, spread across four Nvidia GeForce GTX 3060 GPUs. The training employs a learning rate of 0.0001 by using the Adam optimizer. We apply a pair of commonly utilized data augmentation methods, namely, introducing random noise and scaling, to address the overfitting issue.

We evaluate RadGT's performance in comparison to various cutting-edge methods: Shumann et al. [19], DGCNN [14], PointNGCNN [15], RADAR-PointGNN [10], RADAR-PointNGCNN [11], and RADAR-DGCNN [11]. Results are reported using the macro-averaged F_1 score (F_1-Macro) with fivefold cross-validation in Fig. 2.

The results presented in Fig. 2 illustrate that the RadGT method surpasses all the comparison techniques on both benchmark datasets.

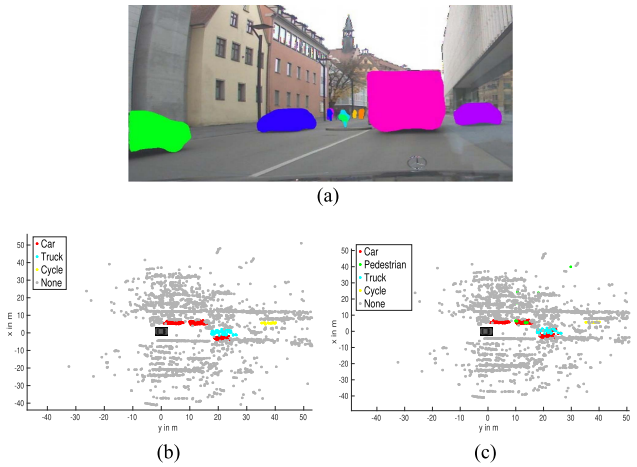


Fig. 3. Sample scene camera image with groundtruth RadGT result. (a) Sample scene from the RadarScenes. (b) Groundtruth point cloud. (c) Results of the RadGT.

TABLE 1. Accuracy (%) Results of Ablation Studies on RadarScenes

Exp	GFSM	TSM	Vehicle	Pedestrian	Cycle
1	✗	✗	88.8 ± 0.2	55.8 ± 1.4	91.7 ± 1.2
2	✓	✗	87.2 ± 1.1	54.3 ± 1.0	87.8 ± 1.3
3	✗	✓	88.9 ± 1.2	55.7 ± 1.1	93.2 ± 1.0
4	✓	✓	89.4 ± 0.3	57.7 ± 0.4	93.7 ± 0.6

Our model demonstrates an improvement of 2% in terms of F_1-Macro compared with the RADAR-PointNGCNN [11] when evaluated on the nuScenes dataset. In the case of the RadarScenes dataset, RadGT also achieves the best performance with 1% increase compared with RADAR-DGCNN. The standard deviation of results in our fivefold cross-validation experiments on both datasets is also relatively lower.

The proposed RadGT's better performance over the RADAR-PointNGCNN, which is the most recent cutting-edge approach, can be attributed to RadGT's ability to extract neighborhood characteristics and spatial distribution information, leading to enhanced efficiency compared with the EdgeConv-based RADAR-DGCNN.

To illustrate the application of the RadGT, we present a sample camera image, along with its groundtruth point cloud. In addition, we showcase the results obtained from applying this method to the RadarScenes dataset in Fig. 3. Furthermore, Fig. 3(a) displays three cars marked in green, blue, and purple, one truck, and one cycle, while Fig. 3(b) illustrates the accurate original point cloud data for this specific scenario. Fig. 3(a) depicts instance labels sourced from the original RadarScens dataset, annotated by the dataset's creator, rather than RadGT. The objects in the resulting point cloud remain discernible despite a few false positive points, as shown in Fig. 3(c).

We conducted an ablation study by using fivefold cross-validation. Training and validation subsets are used to perform ablation experiments. As shown in Table 1, GFSM and TSM are employed to contrast the performance of the baseline and RadGT under varying usage scenarios. We used RADAR-DGCNN [11] as the baseline where neither GFSM nor TSM is present. Then, we activate all combinations of both modules with the baseline and achieve the optimal detection performance. We also performed statistical significance tests (two-sample t-test) with reference to the baseline model (Exp 1). The results in Exp 2 to Exp 4 in Table 1 with p-values less than 0.05 are underlined. The corresponding results confirm the effectiveness of both the GFSM and the TSM modules.

V. CONCLUSION

RadGT demonstrates its capability to extract comprehensive contextual details from individual object points, showcasing a dual proficiency in capturing intricate and extensive information. Integrating SA mechanisms alongside graph-based techniques empowers RadGT to discern and exploit semantic relationships intrinsic to the points. By effectively harnessing the advantages of local irregular geometric features and encompassing global contextual insights, our proposed approach, i.e., RadGT, surpasses conventional methodologies, thus opening new avenues for advanced point cloud analysis within the automotive radar domain. Possible future work can be the assessment of RadGT on 4-D mmWave radar and evaluating the proposed model on completely unseen datasets.

ACKNOWLEDGMENT

The work of Aykut Koç's was supported by BAGEP 2023 Award.

REFERENCES

[1] Y. Zhou, L. Liu, H. Zhao, M. López-Benítez, L. Yu, and Y. Yue, "Towards deep radar perception for autonomous driving: Datasets, methods, and challenges," *Sensors*, vol. 22, no. 11, 2022, Art. no. 4208.

[2] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, "Autonomous vehicles: Challenges, opportunities, and future implications for transportation policies," *J. Modern Transp.*, vol. 24, pp. 284–303, 2016.

[3] R. Zhang and S. Cao, "Real-time human motion behavior detection via CNN using mmWave radar," *IEEE Sens. Lett.*, vol. 3, no. 2, Feb. 2019, Art. no. 3500104.

[4] P. J. B. Morris and K. Hari, "Detection and localization of unmanned aircraft systems using millimeter-wave automotive radar sensors," *IEEE Sens. Lett.*, vol. 5, no. 6, Jun. 2021, Art. no. 6001304.

[5] L. Stanković, M. Daković, and E. Sejdić, "Introduction to graph signal processing," in *Vertex-Frequency Analysis of Graph Signals*, Cham, Switzerland: Springer, 2019, pp. 3–108.

[6] A. Y. Yıldız, E. Koç, and A. Koç, "Multivariate time series imputation with transformers," *IEEE Signal Process. Lett.*, vol. 29, pp. 2517–2521, 2022.

[7] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[8] 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.

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

[10] P. Svenningsson, F. Fioranelli, and A. Yarovoy, "Radar-pointGNN: Graph based object recognition for unstructured radar point-cloud data," in *Proc. IEEE Radar Conf.*, 2021, pp. 1–6.

[11] R. A. Sevimli, M. Üçüncü, and A. Koç, "Graph signal processing based object classification for automotive radar point clouds," *Digit. Signal Process.*, vol. 137, 2023, Art. no. 104045.

[12] Z. Yu et al., "A radar-based human activity recognition using a novel 3-D point cloud classifier," *IEEE Sensors J.*, vol. 22, no. 19, pp. 18218–18227, Oct. 2022.

[13] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1711–1719.

[14] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[15] Q. Lu, C. Chen, W. Xie, and Y. Luo, "PointngCNN: Deep convolutional networks on 3D point clouds with neighborhood graph filters," *Comput. Graph.*, vol. 86, pp. 42–51, 2020.

[16] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Comput. Vis. Media*, vol. 7, pp. 187–199, 2021.

[17] W. Zhou et al., "GTNet: Graph transformer network for 3D point cloud classification and semantic segmentation," 2023, *arXiv:2305.15213*.

[18] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "PointSift: A Sift-like network module for 3D point cloud semantic segmentation," 2018, *arXiv:1807.00652*.

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