CNN Based Road User Detection Using the 3D Radar Cube

Palffy, A.; Dong, Jiaao; Kooij, J. F. P.; Gavrila, D. M.

DOI
10.1109/LRA.2020.2967272

Publication date
2020

Document Version
Final published version

Published in
IEEE Robotics and Automation Letters

CNN Based Road User Detection Using the 3D Radar Cube

Andras Palffy, Jiaao Dong, Julian F. P. Kooij, and Dariu M. Gavrila

***Abstract*—This letter presents a novel radar based, single-frame, multi-class detection method for moving road users (*pedestrian, cyclist, car*), which utilizes low-level radar cube data. The method provides class information both on the radar target- and object-level. Radar targets are classified individually after extending the target features with a cropped block of the 3D radar cube around their positions, thereby capturing the motion of moving parts in the local velocity distribution. A Convolutional Neural Network (CNN) is proposed for this classification step. Afterwards, object proposals are generated with a clustering step, which not only considers the radar targets' positions and velocities, but their calculated class scores as well. In experiments on a real-life dataset we demonstrate that our method outperforms the state-of-the-art methods both target- and object-wise by reaching an average of 0.70 (baseline: 0.68) target-wise and 0.56 (baseline: 0.48) object-wise F1 score. Furthermore, we examine the importance of the used features in an ablation study.**

***Index Terms*—Object detection, segmentation and categorization, sensor fusion, deep learning in robotics and automation.**

I. Introduction

RADARS are attractive sensors for intelligent vehicles as they are relatively robust to weather and lighting conditions (e.g. rain, snow, darkness) compared to camera and LIDAR sensors. Radars also have excellent range sensitivity and can measure radial object velocities directly using the Doppler effect. Thus, they are widely used in applications such as adaptive cruise control and pre-crash safety.

Commercially available radars output a point-cloud of reflections called *radar targets* in every frame (sweep). Each radar target has the following features: range r and azimuth α, radar cross section RCS (i.e. reflectivity), and the object's radial speed v_r relative to the ego-vehicle. We will call these features *target-level*. Since a single reflection does not convey enough information to segment and classify an entire object, many radar based road user detection methods (e.g. [1]–[3]) first cluster radar targets by their target-level features. Clusters are then classified as a whole based on derived statistical features (e.g. mean, variance of r, v_r, RCS of contained radar targets), and the same class label is assigned to all radar targets in the cluster. Object segmentation and classification performance in such pipeline depend on the success of the initial clustering step.

Various methods [4]–[6] instead explore using the *low-level radar cube* extracted from an earlier signal processing stage of the radar. The radar cube is a 3D data matrix with axes corresponding to range, azimuth, and velocity (also called Doppler), and a cell's value represents the measured radar reflectivity in that range/azimuth/Doppler bin. In contrast to the target-level data, the radar cube provides the complete speed distribution (i.e. Doppler vector) at multiple 2D range-azimuth locations. Such distributions can capture modulations of an object's main velocity caused by its moving parts, e.g. swinging limbs or rotating wheels, and were shown to be a valuable feature for object classification [4], [5]. Commonly radar cube features are computed by first generating 2D range-azimuth or range-Doppler projections, or by aggregating the projected Doppler axis over time into a Doppler-time image [6], [7]. We will call features derived from the 3D cube or its projections *low-level*. A downside of such low-level radar data is the lower range and azimuth resolution than the radar targets, and that radar phase ambiguity is not yet addressed, since no advanced range interpolation and direction-of-arrival estimation has taken place.

In this letter we propose a radar based, multi-class moving road user detection method, which exploits *both* expert knowledge at the target-level (accurate 2D location, addressed phase ambiguity), and low-level information from the *full* 3D radar cube rather than a 2D projection. Importantly, the inclusion of low-level data enables classification of individual radar targets before any object clustering; the latter step can benefit from the obtained class scores. At the core of our method is a Convolutional Neural Network (CNN) called Radar Target Classification Network, or *RTCnet* for short. See Fig. 1 for an overview of our method's inputs (radar targets and cube) and outputs (classified targets and object proposals).

Our method can provide class information on both radar target-level and object-level. Target-level class labels are valuable for sensor fusion operating on intermediate-level, i.e. handling multiple measurements per object [8], [9]. Our target-level classification is more robust than cluster-wise classification where the initial clustering step must manage to separate radar targets from different objects, and keep those coming from the same object together, see Fig. 2. Our object-level class information provides instances that are both segmented and

Manuscript received September 10, 2019; accepted January 9, 2020. Date of publication January 17, 2020; date of current version January 31, 2020. This letter was recommended for publication by Associate Editor E. Ricci and Editor E. Marchand upon evaluation of the reviewers' comments. This work was supported in part by the Dutch Science Foundation NWO-TTW, within the SafeVRU under Project 14667. Andras Palffy was also funded by the Tempus Public Foundation by means of a Hungarian Eotvos State Scholarship. *(Corresponding author: Andras Palffy.)*

The authors are with the Intelligent Vehicles Group, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: a.palffy@tudelft.nl; jiaao7464@163.com; j.f.p.kooij@tudelft.nl; d.m.gavrila@tudelft.nl).

Digital Object Identifier 10.1109/LRA.2020.2967272

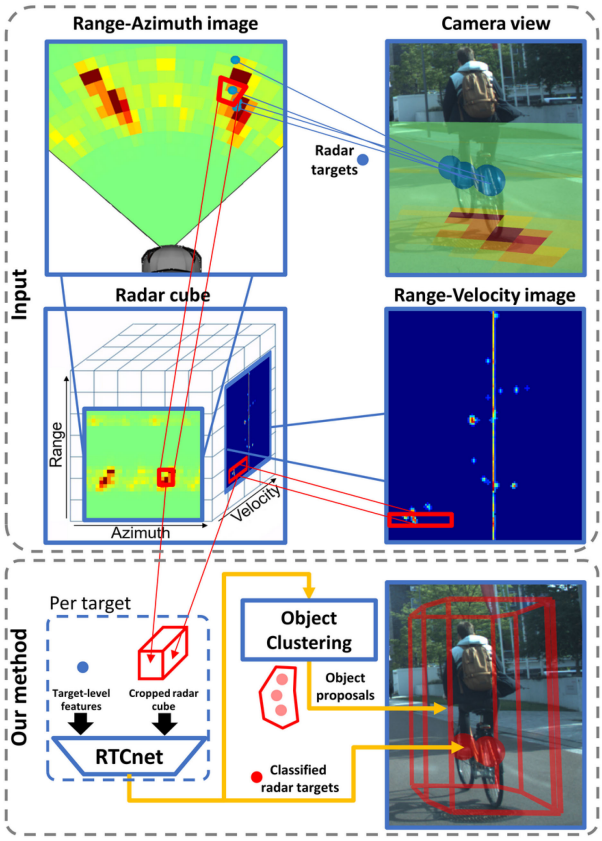


Fig. 1. Inputs (radar cube and radar targets, top), main processing blocks (RTCnet and object clustering, bottom left), and outputs (classified radar targets and object proposals, bottom right) of our proposed method. Classified radar targets are shown as colored spheres at the sensor's height. Object proposals are visualized by a convex hull around the clustered targets on the ground plane and at 2 m.

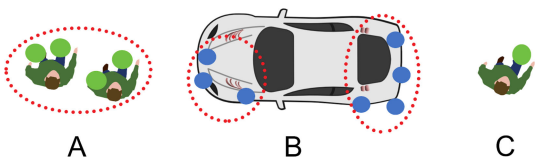


Fig. 2. Challenging cases for cluster-wise classification methods. A: Objects may be clustered together (red circle). B: Large objects may be split up into several clusters. C: Object with only one reflection. Radar targets are shown as dots, colored green/blue for pedestrian/car ground truth class.

classified (object detection), which is valuable for high-level (i.e. late) sensor fusion. While traditional methods must perform clustering with a single set of parameters for all classes, our approach enables use of class-specific clustering parameters (e.g. larger object radius for cars).

II. Related Work

Some previous work on radar in automotive setting has dealt with *static* environments. E.g. [12] shows preliminary results of a neural network based method in a static experimental setup, which creates accurate target-level information from the radar cube. [13] creates an occupancy grid with low-level data. Static object classification (e.g. parked cars, traffic signs) has been shown with target-level [14] and with low-level data [15]. We will focus only on methods addressing *moving* road users.

Many road user detection methods start by clustering the radar targets into a set of object proposals. In [1], radar targets are first clustered into objects by DBSCAN [16]. Then, several cluster-wise features are extracted, e.g. the variance/mean of v_r and r. The performance of various classifiers (Random Forest, Support Vector Machine (SVM), 1-layer Neural Network, etc.) were compared in a single-class (pedestrian) detection task. [2] also uses clusters calculated by DBSCAN as the base of a multi-class (*car, pedestrian, group of pedestrians, cyclist, truck*) detection, but extract different features, e.g. deviation and spread of α. Afterwards, Long Short-Term Memory (LSTM) and Random Forest classifiers were compared for the classification step. Falsely merged clusters (Fig. 2A) were corrected manually to focus on the classification task itself. The same authors showed a method [17] to incorporate a priori knowledge about the data into the clustering. [18] also aims to improve the clustering with a multi-stage approach. [3] follows the work of [2] for clustering and classification, but tests and ranks further cluster-wise features in a backward elimination study.

While clustering based methods are widely used, it is often noted (e.g. [11], [17]) that the clustering step is error-prone. Objects can be mistakenly merged (Fig. 2A) or split apart (Fig. 2B). Finding suitable parameters (e.g. radius and minimum number of points for DBSCAN) is challenging as the same parameters must be used for all classes, although they have significantly different spatial extension and velocity profiles. E.g. a larger radius is beneficial for cars, but could falsely merge pedestrians and cyclists. Another challenge of clustering based methods is that small objects may not have enough reflections (Fig. 2C) to extract meaningful statistical features, e.g. variance. E.g. both [1] and [2] have DBSCAN's minimum number of points to form a cluster ($MinPoints$) larger than one, which means that single standing points are thrown away.

To address these challenges, there is a trend to classify each target individually instead of in clusters. Encouraged by the results achieved with semantic segmentation networks on point-clouds from LIDAR or stereo camera setups, e.g. Pointnet++ [19], researchers have tried to apply the same techniques to radar data. However, the output of a single radar sweep is too sparse. To overcome this, they used multiple frames [11] or multiple radar sensors [20].

Low-level radar data has been used for road user classification, especially for pedestrians. E.g. a walking pedestrian's Doppler-time image contains a characteristic walking gait pattern [4], [5]. This is beneficial to exploit if the radar sensor is stationary, e.g. in surveillance applications [21], [22], [7]. Doppler-time features were also used in automotive setups. [6] applies a CNN-LSTM network on Range-Doppler and Doppler-Time spectrograms of 0.5–2 seconds to classify *pedestrian, group of pedestrians, car*, and *cyclist* classes. [10] pointed out that a long multi-frame observation period is not viable for urban driving, and proposed a single-frame usage of low-level data. Their method still generates object proposals with DBSCAN similar to [1], [2], but extracts for each cluster the corresponding area in a 2D Range-Doppler image, which is then classified

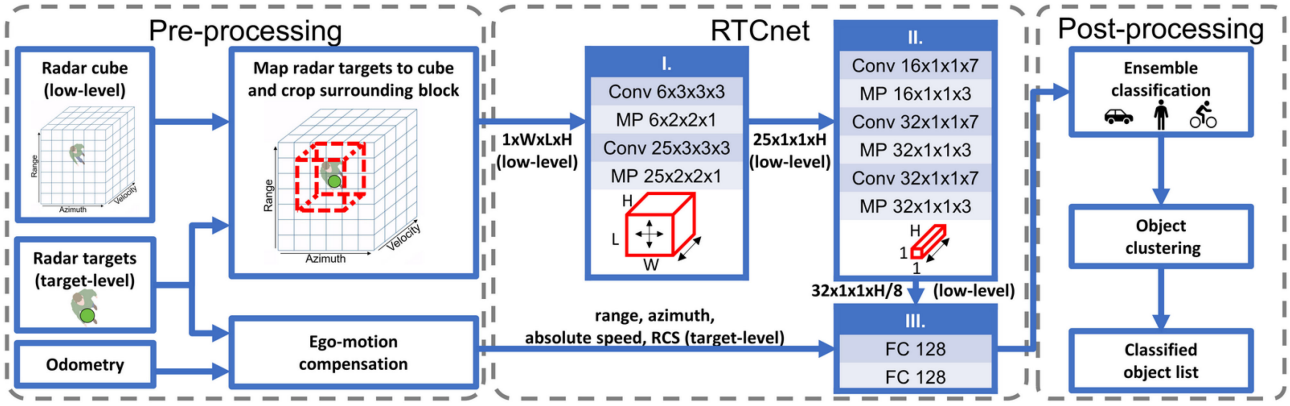


Fig. 3. Our pipeline. A block around each radar target is cropped from radar cube. *RTCnet* has three parts. I. encodes range and azimuth dimensions. II. extracts class information from the speed distribution. III. provides scores based on II. and target-level features. Ensembling assigns a class label to each radar target. The class-specific clustering provides object proposals.

TABLE I
OVERVIEW OF THE MOST CLOSELY-RELATED METHODS

Method	Basis	Features	Classes	Time window
Prophet [1] †	clusters	target	single	1 frame (50 ms)
Schumann [2] †	clusters	target	multi	2 frames (150 ms)
Prophet [10]	clusters	both	single	1 frame
Schumann [11]	targets	target	multi	0.5 s
Angelov [6]	targets	low	multi	0.5-2 s
RTCnet (ours)	*targets*	*both*	*multi*	*1 frame (75 ms)*

†: marks methods selected as baselines.

using conventional computer vision. In [23], the full radar cube is used as a multi-channel image input to a CNN network to classify *cars, pedestrians*, and *cyclists*. The study only addresses a single-object classification task, i.e. location is not fetched.

In conclusion, the topic of radar based road user detection was extensively researched. Table I gives an overview of the most relevant methods with their basis of the classification (cluster-wise or target-wise), the level of features (target or low), the number of classified classes, and the required time window to collect suitable amount of data. None of the found methods avoids error-prone clustering for classification and operates with a low latency for urban driving (i.e. one or two radar sweeps $(75-150$ ms)) at the same time.

Our main contributions are as follows. 1) We propose a radar based, single-frame, multi-class (*pedestrian, cyclist, car*) moving road user detection method, which exploits both target-level and low-level radar data by a specially designed CNN. The method provides both classified radar targets and object proposals by a class-specific clustering. 2) We show on a large-scale, real-world dataset that our method is able to detect road users with higher than state-of-the-art performance both in target-wise (target classification) and object-wise (object detection) metrics using only a single frame of radar data.

III. PROPOSED METHOD

In this research, we combine the advantages of target-level (accurate range and azimuth estimation) and low-level data (more information in speed domain) by mapping the radar targets into the radar cube and cropping a smaller block around it in all three dimensions (Subsection III-A). *RTCnet* classifies each target individually based on the fused low-level and target-level data. The network consists of three parts (Subsection III-B). The first encodes the data in spatial domains (range, azimuth) and grasps the surroundings' Doppler distribution. The second is applied on this output to extract class information from the distribution of speed. Finally, the third part provides classifications scores by two fully connected layers (FC). The output is either multi-class (one score for each class) or binary. In the latter case, an ensemble voting (Subsection III-C) step combines the result of several binary classifiers similarly to [24]. A class-specific clustering step (i.e. the radar targets' predicted class information is used) generates an object list output (subsection III-D). See Fig. 3 for an overview of our method. The software of our pipeline is available on our website.[1]

A. Pre-Processing

First, a single frame of radar targets and a single frame of the radar cube (low-level data) is fetched. Each radar target's speed is compensated for ego-motion similarly to [2]. As we only address moving road users, radar targets with low compensated (absolute) velocity are considered as static and are filtered out. Then, corresponding target-level and low-level radar data are connected. That is, we look up each remaining dynamic radar target's corresponding range/azimuth/Doppler bins, i.e. a grid cell in the radar cube based on their reported range, azimuth and (relative) velocity (r, α, v_r). Afterwards, a 3D block of the radar cube is cropped around each radar target's grid cell with radius in range/azimuth/Doppler dimensions (L, W, H). See "Pre-Processing" part on Fig. 3.

B. Network

RTCnet consists of three modules as seen on Fig. 3.

1) Down-Sample Range and Azimuth Dimensions: The first part's aim is to encode the radar target's spatial neighborhood's

[1][Online]. Available: https://github.com/tudelft-iv/RTCnet

Doppler distribution into a tensor without extension in range or azimuth. In other words, it transforms the $1 \times W \times L \times H$ sized data to a $C \times 1 \times 1 \times H$ sized tensor (sizes are given as $Channel \times Azimuth \times Range \times Doppler$), where C was chosen as 25. To do this, it contains two 3D convolutions (Conv) with the kernel sizes of $6 \times 3 \times 3 \times 3$ and $25 \times 3 \times 3 \times 3$ (padding is 1). Both convolutional layers are followed by a maxpool (MP) layer with the kernel sizes of $6 \times 2 \times 2 \times 1$ and $25 \times 2 \times 2 \times 1$ with 0 padding to down-sample in the spatial dimensions.

2) Process Doppler Dimension: The second part of the network operates on the output of the first which is a $25 \times 1 \times 1 \times H$ sized tensor. The aim of this module is to extract class information from the speed distribution around the target. To do this, we use three 1D convolutions along the Doppler dimension with the kernel size of 7 and output channel sizes of $16, 32, 32$. Each convolution is followed by a maxpool layer with the kernel size of 3 and stride of 2, which halves the length of the input. The output of the this module is a $32 \times 1 \times 1 \times H/8$ block.

3) Score Calculation: The output of the second module is flattened and concatenated to the target-level features (r, α, v_r, RCS), and fed into the third one. We use two fully connected layers with 128 nodes each to provide scores. The output layer has either four nodes (one for each class) for multi-class classification or two for binary tasks. In the latter case, ensemble voting is applied, see next subsection.

C. Ensemble Classifying

With four output nodes, it is possible to train the third module to perform multi-class classification directly. We also implemented an ensemble voting system of binary classifiers (networks with two output nodes). That is, aside training a single, multi-class network, we followed [24] and trained One-vs-All (OvA) and One-vs-One (OvO) binary classifiers for each class (e.g. car-vs-all) and pair of classes (e.g. car-vs-cyclist), 10 in total. The final prediction scores depend on the voting of all the binary models. OvO scores are weighted by the summation of the corresponding OvA scores to achieve a more balanced result. Although we experimented with ensembling multi-class classifiers trained on bootstrapped training data as well, it yielded worse results.

D. Object Clustering

The output of the network (or voting) is a predicted class label for each target individually. To obtain proposals for object detection, we cluster the classified radar targets with DBSCAN incorporating the predicted class information, i.e. radar targets with *bike/pedestrian/car* predicted labels are clustered in separate steps. As metric, we used a spatial threshold γ_{xy} on the Euclidean distance in the x, y space (2D Cartesian spatial position), and a separate speed threshold γ_v in velocity dimension (*Prophet* [1], [18], [25]). The advantage of clustering each class separately is that no universal parameter set is needed for DBSCAN. Instead, we can use different parameters for each class, e.g. larger radius for cars and small ones for pedestrians (Fig. 2A and B). Furthermore, swapping the clustering and classification step makes it possible to consider objects with a single reflection, e.g. setting $MinPoints$ to one for pedestrian labeled radar targets (Fig. 2C). A possible drawback is that if a subset of an object's reflections are misclassified (e.g. a car with multiple targets, most labeled *car* and some as *cyclist*), the falsely classified targets (i.e. the *cyclist* ones) will be mistakenly clustered into a separate object. To address this, we perform a filtering on the produced object proposals, calculating their spatial, (radial) velocity, and class score distribution distances (scores are handled as 4D vector, and we take their Euclidean distance after normalization). If two clusters have different classes and are close enough in all dimensions (cf. parameters in Sec. V-B), we merge the smaller class to the larger (i.e. pedestrians to cyclists and cars, cyclists to cars) given that the cluster from the larger class has more radar targets.

TABLE II
NUMBER OF INSTANCES FROM EACH CLASS IN OUR TRAINING SET. MANY ROAD USERS HAVE ONLY ONE RADAR REFLECTION, WHICH IS NOT ENOUGH TO EXTRACT MEANINGFUL STATISTICAL FEATURES

	Pedestrians	Bikers	Cars
Number of instances	31300	15290	9362
Number of radar targets	63814	45804	30906
Avg. number of radar targets per instance	2.04	3.00	3.30
Instances with only one radar target	12990	3526	2878
Ratio of instances with one radar target	41.5%	18.8%	37.6%

IV. DATASET

Our real-world dataset contains $\sim$1 hour of driving in urban environment with our demonstrator vehicle [26]. We recorded both the target-level and low-level output of our radar, a Continental 400 series mounted behind the front bumper. We also recorded the output of a stereo camera (1936×1216 px) mounted on the wind-shield, and the ego-vehicle's odometry (filtered location and ego-speed).

Annotation was fetched automatically from the camera sensor using the Single Shot Multibox Detector (SSD) [27] trained on the EuroCity Persons dataset [28]. Distance is estimated by projecting each bounding box into the stereo point-cloud computed by the Semi-Global Matching algorithm (SGM) [29], and taking the median distance of the points inside each. In a second iteration, we manually corrected mislabeled ground truth, e.g. cyclist annotated as pedestrian. The training set contains more than $30/15/9 \times 10^3$ pedestrian/cyclist/car instances respectively (one object may appear on several frames), see Table II. Fig. 7 shows the distribution of radar targets in the training set distance-wise. To further extend our training dataset, we augmented the data by mirroring the radar frames and adding a zero-mean, 0.05 std Gaussian noise to the normalized r and v_r features. Training and testing sets are from two independent driving (33 and 31 minutes long) which took place on different days and routes. Validation set is a 10% split of training dataset after shuffling.

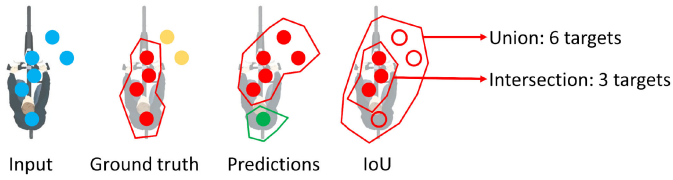


Fig. 4. Object-level metric. Intersection and Union are defined by number of radar targets. $\frac{Intesection}{Union} \geq 0.5$ counts as a true positive. In this example, there is a true positive cyclist and a false positive pedestrian detection.

V. Experiments

We compared our proposed method, *RTCnet* with binary bagging (from now on, referred to as *RTCnet*) to two baselines in two experiments to examine their radar target classification and object detection capabilities.

In the first experiment, we examined their performance in classification task, using a target-wise metric, i.e. a true positive is a correctly classified target [11]. For cluster-wise methods (the baselines) the predicted label of a cluster is assigned to each radar target inside following [11]. Furthermore, we also performed an ablation study to see how different features benefit our method in this classification (adaptation in brackets). *RTCnet (no ensemble)* is a single, multi-class network to see if ensembling is beneficial. *RTCnet (no RCS)* is identical to *RTCnet*, but the RCS target-level feature is removed to examine its importance. Similarly, in *RTCnet (no speed)* the absolute speed of the targets is unknown to the networks, only the relative speed distribution (in the low-level data) is given. Finally, *RTCnet (no low-level)* is a significantly modified version as it only uses target-level features. That is, the first and second convolutional parts are skipped, and the radar targets are fed to the third fully connected part directly. Note that in contrast to *RTCnet (no speed)*, *RTCnet (no low-level)* has access to the absolute speed of the target, but lacks the relative speed distribution. Object clustering is skipped in the first experiment.

In the second experiment, we compare the methods in object detection task, examining our whole pipeline, including the object clustering step. Predictions and annotations are compared by their intersection and union calculated in number of targets, see Fig. 4. A true positive is a prediction which has an Intersection Over Union (IoU) bigger than or equal to 0.5 with an annotated object. Further detections of the same ground truth object count as false positives.

All presented results were measured on moving radar targets to focus on moving road users.

A. Baselines

We selected *Schumann* [2] as baseline because it is the only multi-object, multi-class detection method found with small latency, see Table I. As no other research handled multiple classes, we selected *Prophet* [1] as our second baseline, which is a single-class pedestrian detector, but the negative training and testing set contained cars, dogs, and cyclists. We re-implemented their full pipeline (DBSCAN clustering and cluster-wise classification) and trained their algorithms with our training set.

TABLE III
Optimized DBSCAN Parameters for the Two Baselines, and For Our Class-Specific Clustering for Each Class

Method	γ_{xy}	γ_v	$MinPoints$	v_{min}
Prophet [1]	1.2 m	1.3 m/s	2	0.4 m/s
Schumann [2]	1.3 m	1.4 m/s	2	0.4 m/s
Class-specific (peds.)	0.5 m	2.0 m/s	1	–
Class-specific (cyclists)	1.6 m	1.5 m/s	2	–
Class-specific (cars)	4.0 m	1.0 m/s	3	–

Optimal DBSCAN parameters are sensor specific (depending on density, resolution, etc.), thus we optimized the threshold in spatial dimensions γ_{xy} (0.5 m $-$ 1.5 m, step size 0.1 m) and the threshold in velocity γ_v (0.5 $-$ 1.5 m/s, step size 0.1 m/s) on our validation set for both baselines independently. We used the same metric as in our object clustering. Both baselines have features describing the number of static radar targets in the cluster. We also searched for an optimal speed threshold v_{min} (0 $-$ 0.5 m/s, step size 0.1 m/s) for both to define these static radar targets. All reported results for baselines were reached by using their optimal settings, see Table III. $MinPoints$ was set to two as in *Prophet* [1] (increasing it further would exclude almost all pedestrians, see Table II). In *Schumann* [2] the authors used manually corrected clusters (i.e. separating objects falsely merged by DBSCAN) to focus on the classification. We did not correct them to examine real-life application possibilities. We implemented a Random Forest classifier with 50 trees for both baselines, as *Prophet* [1] reported it to be the best for their features. *Schumann* [2] also tested LSTM, but used several frames aggregated as input.

B. Implementation

We set $L = W = 5$, $H = 32$ as the size of the cropped block. Speed threshold to filter out static objects is a sensor specific parameter and was set to 0.3 m/s based on empirical evidence. Table III shows the DBSCAN parameters for both baselines and for our class-specific clustering step. The thresholds to merge clusters during object clustering were set to 1 m spatially, 0.6 for scores, 2 m/s for pedestrian to cyclist, and 1.2 m/s for pedestrian/cyclist to car merges.

We normalized the data to be zero-mean and have a standard deviation of 1 feature-wise for r, α, v_r, RCS, and for the whole radar cube. At inference values calculated from training data are used. We used PyTorch [30] for training with a cross-entropy loss (after softmax) in 10 training epochs. Inference time is ~0.04 s on a high-end PC (Nvidia TITAN V GPU, Intel Xeon E5-1650 CPU, 64 GB RAM), including all moving radar targets, the 10 binary classifiers and the ensembling.

C. Results

1) Target Classification: We present the results of the target classification experiment in Table IV. Target-wise F1 scores for all classes and their macro-average are given for each method. *RTCnet* outperformed the two cluster-wise baselines reaching an average F1 score of 0.70. *Schumann* [2] has slightly better results on *cyclists* than *RTCnet* (0.68 vs 0.67), but performed

(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 5. Examples of correctly classified radar targets by *RTCnet*, projected to image plane. Radar targets with pedestrian/cyclist/car labels are marked by green/red/blue. Static objects and the class *other* are not shown.

TABLE IV
TARGET-WISE F1 SCORES PER CLASS (BEST IN BOLD). *RTCNET* OUTPERFORMS THE BASELINES ON AVERAGE. THE ABLATION STUDY SHOWS BENEFITS OF ENSEMBLING AND USING LOW-LEVEL DATA

Method	Pedestrian	Cyclist	Car	Other	Avg.
Prophet [1]	0.61	0.58	0.34	0.91	0.61
Schumann [2]	0.67	**0.68**	0.46	**0.92**	0.68
RTCnet (no low-level)	0.56	0.63	0.33	0.90	0.61
RTCnet (no speed)	0.66	0.63	0.36	0.91	0.64
RTCnet (no RCS)	**0.71**	0.66	0.48	0.91	0.69
RTCnet (no ensemble)	0.67	0.65	0.47	0.89	0.67
RTCnet	**0.71**	0.67	**0.50**	**0.92**	**0.70**

(a) (b) (c) (d)

Fig. 6. Examples of radar targets misclassified by *RTCnet*, caused by: flat surfaces acting as mirrors and creating ghost targets (a), unusual vehicles (b), partial misclassification of an objects' reflections (c), and strong reflections nearby (d).

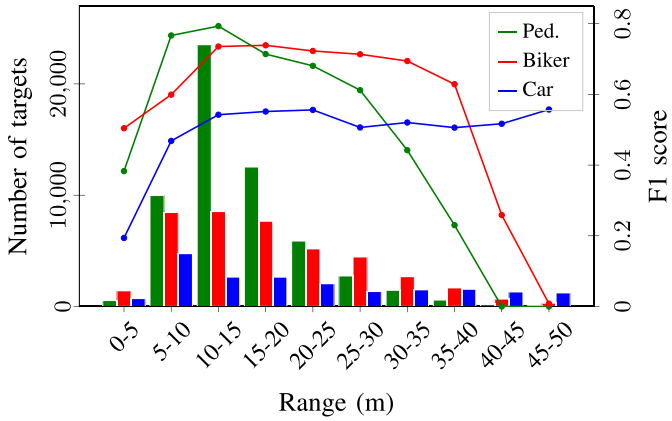


Fig. 7. Target-wise F1 scores (lines) and number of targets in training set (bars) in function of distance from ego-vehicle.

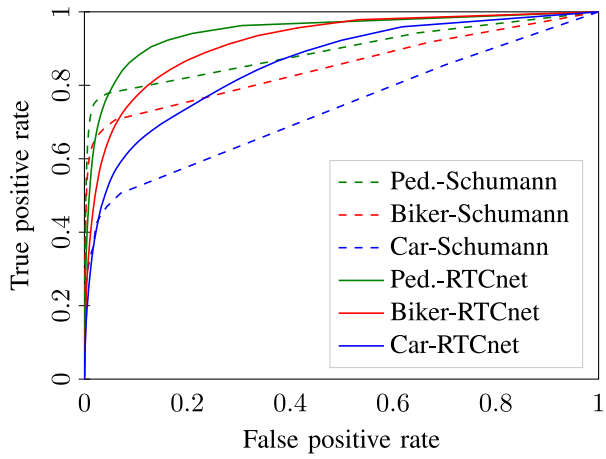


Fig. 8. ROC curves of road user classes by our method and *Schumann* [2]. Each curve is calculated by changing the decision threshold of a One-vs-All binary classifier.

significantly worse on *pedestrians* (0.67 vs 0.71) and *cars* (0.46. vs 0.50). The ablation study showed that removing each feature yields worse results than the complete pipeline, but the one without reflectivity information (*RTCnet (no RCS)*) comes close with an average of 0.69. Removing the low-level features (*RTCnet (no low-level)*) decreased the performance significantly to an average of 0.61. The multi-class (single) network *RTCnet (no ensemble)* outperforms the baselines on the *car* class, but performs worse on *cyclists*. Ensemble voting brings significant improvement on all classes. Example of correct and incorrect target classifications are shown on Fig. 5 and 6 for all road user classes. On Fig. 7 we show how the classification performance (target-wise F1 score) changes over distance (with 5 m bins) for each class, along with the number of radar targets in the training set. Although most annotation fall into the $5-20$ m range, the network performs reasonably beyond that distance, especially for the larger objects (*cyclist, car*). We trained One-vs-All classifiers both for *RTCnet* and *Schumann* [2] for each road user class, and plotted their performance on receiver operating characteristic (ROC) curves on Fig. 8. The varied threshold is cluster-wise for *Schumann* [2] and target-wise for *RTCnet*. Our method has a larger area under the curve of all classes.

TABLE V
F1 SCORES OBJECT-WISE (BEST SCORE IN BOLD). *RTCNET* OUTPERFORMS THE BASELINES ON AVERAGE

	Pedestrian	Cyclist	Car	Avg.
Prophet [1]	0.48	0.50	0.23	0.40
Schumann [2]	0.54	**0.60**	0.31	0.48
RTCnet (ours)	**0.61**	0.59	**0.47**	**0.56**

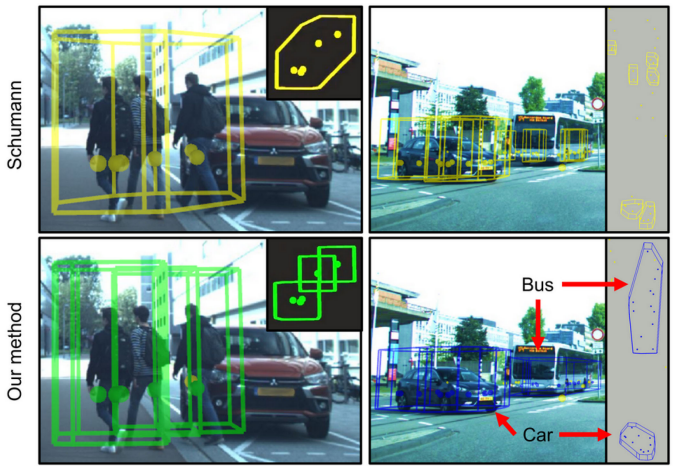
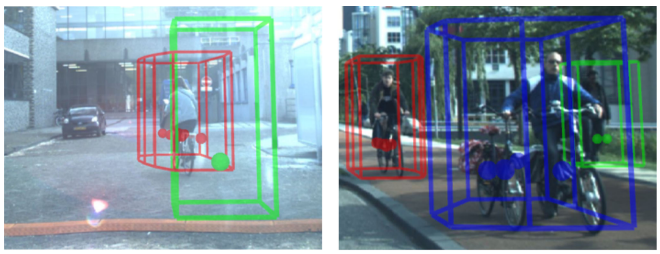


Fig. 9. Challenging cases for clustering, camera and top view. DBSCAN falsely split the car and the bus but merged the pedestrians into a single cluster, making *Schumann* [2] (top) fail. Our method (bottom) managed to classify the radar targets and cluster them correctly using class-specific parameters. Yellow marks *other* class.



(a) (b)

Fig. 10. Examples of correct and incorrect object detections of our method. A mis-classified radar target triggered a false positive pedestrian detection on (a). Bicycles moving side-by-side at the same speed are detected as a car on (b).

2) Object Detection: The results of our second experiment are shown in Table V. *RTCnet* reached slightly worse results on *cyclists* than *Schumann* [2] (0.59 vs 0.60), but significantly outperformed it on *pedestrians* (0.61 vs 0.54), *cars* (0.47 vs 0.31), and in average (0.56 vs 0.48). Fig. 9 shows how *Schumann* [2] and *RTCnet* handled two real-life cases from Fig. 2. Examples for both correct and incorrect object detections by *RTCnet* are shown on Fig. 10. A link to a video of our results can be found on our website.[2]

D. Discussion

Our method outperformed the baselines in target classification mainly due to two reasons. First, the classification does not depend on a clustering step. This decreases the impact of cases shown in Fig. 2 and allows to handle objects that contain a single radar target (a common occurrence, especially for pedestrians, see Table II). Second, we included low-level radar data, which brings information of the speed distribution around the radar target. To demonstrate that this inclusion is beneficial, we showed that only using target-level data and only the third module of the network (*RTCnet (no low-level)*) caused a significant drop in performance from 0.70 to 0.61 average F1 score. We examined the effect of removing absolute speed from the data too with *RTCnet (no speed)*. While the performance dropped, our network was still able to classify the radar targets by the relative speed distribution around them. The results of *RTCnet (no low-level)* and *RTCnet (no speed)* proves that the relative velocity distribution (i.e. the low-level radar data) indeed contains valuable class information. Interestingly, excluding RCS value did not have a significant impact on the performance. Based on our experiments, an ensemble of binary classifiers results in less inter-class miss-classifications than using a single multi-class network.

Note that even VRUs in occlusion (see Fig. 5(a), 5(b), 5(g)) are often classified correctly caused by the multi-path propagation of radar [8]. This, and its uniform performance in darkness/shadows/bright environments makes radar a useful complementary sensor for camera. Typical errors are shown in Fig. 6. Radar is easily reflected by flat surfaces (e.g. side of cars) acting like mirrors, creating *ghost targets*. E.g. in Fig. 6(a) our ego-vehicle was reflected creating several false positives. Fig. 6(b) is an example of hard to categorize road users. Many errors come from the confusion of *car* and *cyclist* caused by the similarity of their Doppler signature and reflectivity, see Fig. 6(c). Fig. 6(d) shows that a strong reflection nearby can mislead the classifier. Since our method does not throw away single targets in a clustering step, it has to deal with more noise reflections than a cluster-wise method. However, the results in *other* class suggest that it learned to ignore them.

The combination of our network and the clustering step outperformed the baseline methods in the object detection task. This is mainly because by swapping the clustering and classifying steps, classes can be clustered with different parameters. That is a significant advantage of our pipeline, as instead of finding a single set of clustering parameters to handle each class, we can tune them separately to fit each, see Table III. This is especially useful in *pedestrian* and *car* classes, which are smaller/larger than the optimal spatial radius $\gamma_{xy} = 1.2 - 1.3$ m found for the baselines. However, this radius fits bicycles well, which results in good performance on the *cyclists* class for *Schumann* [2] both on target-level and object-level. Fig. 9 shows two examples. DBSCAN falsely separated the car and the bus into several clusters, but merged the pedestrians into a single one using the optimized parameters, which caused *Schumann* [2] to fail. Our method managed to classify each radar target individually and cluster them correctly (i.e. keep the vehicles in a single cluster, but separate the pedestrians) using the class-specific clustering parameters. Although we used DBSCAN in this letter, we expect this advantage to stand using different types of clustering. On Fig. 10(a) we show a single mis-classified radar target, probably

[2][Online]. Available: http://intelligent-vehicles.org/publications/

reflected by the speed bump. The resulting false positive pedestrian detection is trade-off of setting $MinPoints$ to one for pedestrians. As mentioned, cyclists and cars are often confused. This is especially true if several cyclist ride side-by-side, see 10a, since their radar characteristics (extension, speed, reflectivity) are car-like. Both errors usually occur for a single frame only, and can be alleviated by a temporal filtering and tracking system.

VI. Conclusion

In this letter, we proposed a radar based, single-frame, multi-class road user detection method. It exploits class information in low-level radar data by applying a specially designed neural network to a cropped block of the radar cube around each radar target and the target-level features. A clustering step was introduced to create object proposals.

In extensive experiments on a real-life dataset we showed that the proposed method improves upon the baselines in target-wise classification by reaching an average F1 score of 0.70 (vs. 0.68 *Schumann* [2]). Furthermore, we demonstrated the importance of low-level features and ensembling in an ablation study. We showed that the proposed method outperforms the baselines overall in object-wise classification by yielding an average F1 score of 0.56 (vs. 0.48 *Schumann* [2]).

Future work may include a more advanced object clustering procedure, e.g. by training a separate head of the network to encode a distance metric for DBSCAN. Temporal integration and/or tracking of objects could further improve the method's performance and usability. Finally, extending the proposed framework to incorporate data from additional sensor modalities (e.g. camera, LiDAR) is worthwhile.

References

[1] R. Prophet *et al.*, "Pedestrian classification with a 79 GHz automotive radar sensor," in *Proc. 19th Int. Radar Symp.*, 2018, pp. 1–6.

[2] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, "Comparison of random forest and long short-term memory network performances in classification tasks using radar," in *Proc. Sensor Data Fusion: Trends, Solutions, Appl.*, 2017, pp. 1–6.

[3] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, "Radar-based feature design and multiclass classification for road user recognition," in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 779–786.

[4] E. Schubert, M. Kunert, A. Frischen, and W. Menzel, "A multi-reflection-point target model for classification of pedestrians by automotive radar," in *Proc. 11th Eur. Radar Conf.*, 2014, pp. 181–184.

[5] E. Schubert, F. Meinl, M. Kunert, and W. Menzel, "High resolution automotive radar measurements of vulnerable road users - pedestrians & cyclists," in *Proc. IEEE MTT-S Int. Conf. Microw. for Intell. Mobility*, 2015, pp. 1–4.

[6] A. Angelov, A. Robertson, R. Murray-Smith, and F. Fioranelli, "Practical classification of different moving targets using automotive radar and deep neural networks," *IET Radar, Sonar Navigat.*, vol. 12, no. 10, pp. 1082–1089, 2018.

[7] J. Kwon and N. Kwak, "Human detection by neural networks using a low-cost short-range Doppler radar sensor," in *Proc. IEEE Radar Conf.*, 2017, pp. 0755–0760.

[8] A. Palffy, J. F. P. Kooij, and D. M. Gavrila, "Occlusion aware sensor fusion for early crossing pedestrian detection," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 1768–1774.

[9] K. Granström, M. Baum, and S. Reuter, "Extended object tracking: Introduction, overview, and applications," *J. Adv. Inf. Fusion*, vol. 12, no. 2, Dec. 2017. [Online]. Available: https://www.uni-goettingen.de/en/606544.html

[10] R. Prophet, M. Hoffmann, A. Ossowska, W. Malik, C. Sturm, and M. Vossiek, "Image-based pedestrian classification for 79 GHz automotive radar," in *Proc. 15th Eur. Radar Conf.*, 2018, pp. 75–78.

[11] 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.

[12] D. Brodeski, I. Bilik, and R. Giryes, "Deep radar detector," Apr. 2019, pp. 1–6, doi: 10.1109/RADAR.2019.8835792.

[13] R. Weston, S. Cen, P. Newman, and I. Posner, "Probably unknown: Deep inverse sensor modelling radar," in *Proc. Int. Conf. Robot. Autom.*, May 2019, pp. 5446–5452.

[14] J. Lombacher, M. Hahn, J. Dickmann, and C. Wöhler, "Potential of radar for static object classification using deep learning methods," in *Proc. IEEE MTT-S Int. Conf. Microw. for Intell. Mobility*, 2016, pp. 1–4.

[15] K. Patel, K. Rambach, T. Visentin, D. Rusev, M. Pfeiffer, and B. Yang, "Deep learning-based object classification on automotive radar spectra," in *Proc. IEEE Radar Conf.*, 2019, pp. 1–6.

[16] M. Ester, K. Hans-Peter, S. Jorg, and X. Xiaowei, "Density-based clustering algorithms for discovering clusters," *Comprehensive Chemometrics*, vol. 2, pp. 635–654, 2010.

[17] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, "Supervised clustering for radar applications: On the way to radar instance segmentation," in *Proc. IEEE MTT-S Int. Conf. Microw. for Intell. Mobility*, 2018, pp. 1–4.

[18] N. Scheiner, N. Appenrodt, and B. Sick, "A multi-stage clustering framework for automotive radar data," in *Proc. IEEE 22nd Intell. Transp. Syst. Conf.*, 2019, pp. 2060–2067.

[19] 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. 30*, 2017, pp. 5099–5108.

[20] A. Danzer, T. Griebel, M. Bach, and K. Dietmayer, "2D car detection in radar data with PointNets," 2019. [Online]. Available: http://arxiv.org/abs/1904.08414

[21] D. Tahmoush and J. Silvious, "Radar micro-Doppler for long range front-view gait recognition," in *Proc. IEEE 3rd Int. Conf. Biometrics Theory, Appl., Syst.*, 2009, pp. 1–6.

[22] S. Okumura, T. Sato, T. Sakamoto, and T. Sato, "Technique of tracking multiple pedestrians using monostatic ultra-wideband Doppler radar with adaptive Doppler spectrum estimation," in *Proc. Int. Symp. Antennas Propag.*, 2016, pp. 320–321.

[23] R. Perez, F. Schubert, R. Rasshofer, and E. Biebl, "Single-frame vulnerable road users classification with a 77 GHz FMCW radar sensor and a convolutional neural network," in *Proc. 19th Int. Radar Symp.*, 2018, pp. 1–10.

[24] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, "Radar-based road user classification and novelty detection with recurrent neural network ensembles," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 722–729.

[25] E. Schubert, F. Meinl, M. Kunert, and W. Menzel, "Clustering of high resolution automotive radar detections and subsequent feature extraction for classification of road users," in *Proc. 16th Int. Radar Symp.*, 2015, pp. 174–179.

[26] L. Ferranti *et al.*, "SafeVRU: A research platform for the interaction of self-driving vehicles with vulnerable road users," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 1660–1666.

[27] W. Liu *et al.*, "SSD: Single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016.

[28] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, "EuroCity persons: A novel benchmark for person detection in traffic scenes," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 41, no. 8, pp. 1844–1861, Aug. 2019.

[29] H. Hirschmüller, "Stereo processing by semi-global matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[30] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett Eds., New York, NY, USA: Curran Associates, Inc., 2019, pp. 8024–8035.