

# Real-Time and Robust 3D Object Detection Within Road-Side LiDARs Using Domain Adaptation

Walter Zimmer<sup>1</sup> , Marcus Grabler<sup>1,2</sup>  and Alois Knoll<sup>1</sup> 

**Abstract**—This work aims to address the challenges in domain adaptation of 3D object detection using infrastructure LiDARs. We design a model *DASE-ProPillars* that can detect vehicles in infrastructure-based LiDARs in real-time. Our model uses *PointPillars* as the baseline model with additional modules to improve the 3D detection performance. To prove the effectiveness of our proposed modules in *DASE-ProPillars*, we train and evaluate the model on two datasets, the open source *A9-Dataset* and a semi-synthetic infrastructure dataset created within the *Regensburg Next* project. We do several sets of experiments for each module in the *DASE-ProPillars* detector that show that our model outperforms the *SE-ProPillars* baseline on the real A9 test set and a semi-synthetic A9 test set, while maintaining an inference speed of 45 Hz (22 ms). We apply domain adaptation from the semi-synthetic *A9-Dataset* to the semi-synthetic dataset from the *Regensburg Next* project by applying transfer learning and achieve a 3D *mAP*@0.25 of 93.49% on the Car class of the target test set using 40 recall positions.

## I. INTRODUCTION

High quality and balanced data is crucial to achieve high accuracy in deep learning applications. The creation of labeled datasets of infrastructure mounted LiDAR is a very difficult task. Considering the high labor cost of manual labeling for 3D bounding boxes in LiDAR point clouds, we need to find a solution to deal with small datasets. Publicly available LiDAR datasets were recorded and labeled from a driver perspective which makes it difficult to apply these trained detectors on infrastructure-based LiDARs. The focus of this work lies in the area of domain adaptation to tackle the domain shift problem. How can a neural network that was trained in one operational design domain (ODD), e.g. an urban area like in the *A9-Dataset* [4], be adapted to a slightly different domain, e.g. an intersection in a different city with different LiDAR sensors and mounting positions? This process is known as transfer learning – training a model on a large dataset (source domain) and fine-tuning it on a smaller dataset (target domain). Another challenge is real-time 3D object detection on infrastructure-based LiDARs, i.e. to detect objects with a high enough frame rate to prevent accidents. This highly depends on the LiDAR type and the number of points per scan that need to be processed. The final challenge this work is dealing with is a robust 3D detection of

\*This work was supported by the Federal Ministry of Transport and Digital Infrastructure, Germany as part of the research project Providentia++

<sup>1</sup>The authors are with the Informatics Faculty, Technical University of Munich (TUM), 85748 Garching-Hochbrueck, Germany [walter.zimmer@cs.tum.edu](mailto:walter.zimmer@cs.tum.edu), [{grabler,knoll}@in.tum.de](mailto:{grabler,knoll}@in.tum.de)

<sup>2</sup>Autonomous Reply, Riesstraße 22, 80992 Munich, Germany [m.grabler@reply.de](mailto:m.grabler@reply.de)

all traffic participants. Detecting small and occluded objects at different weather conditions and rare traffic scenarios is a highly important research area to increase safety of automated vehicles.

The first objective is to create a large infrastructure dataset with sufficient labeled point cloud frames. This dataset should be balanced in terms of object classes and contain a high variety so that objects can be detected in different scenarios and different environment conditions. Another objective is to analyze whether transfer learning from a larger road-side LiDAR dataset, such as the recently released *A9-Dataset*, can improve the model performance on other infrastructure datasets. The first batch of the published *A9-Dataset* includes 459 manually labeled point cloud frames and contains 3104 labeled 3D objects. In this work the aim is to design a single-stage 3D object detector, train it in one domain and fine-tune it on a different domain. An intersection, part of the *Providentia++ project* [15], [16], [3], is equipped with five LiDAR sensors (see Figure 2). The goal is to reach an inference rate of at least 25 Hz, which leads to a maximum inference time of 40 ms per point cloud frame. In terms of accuracy the target is to achieve at least 90% *mAP* on the Car class of the test set. In the end the designed model needs to reach a reasonable trade-off between the inference time and the model performance. The final target is to evaluate the model performance and speed on both dataset.

This work provides a domain adaptation solution for the single LiDAR detection task. The main contributions in this work are summarized as follows:

- We propose a single-stage LiDAR-only detector based on *PointPillars*. We introduce five extensions to improve *PointPillars* and evaluate the performance of the designed model on the test set of the *A9-Dataset* [4] and a dataset from the *Regensburg Next* project [28].
- We create a semi-synthetic dataset, called *proSynthSemi*, with 6k labeled LiDAR point cloud frames using the CARLA simulator [6] and train our model on that.
- Experiments show that our *DASE-ProPillars* model outperforms the *SE-ProPillars* [31] model by 30.56% 3D *mAP* on the A9 test set (Car class), while the inference speed is maintained at 45 Hz (22 ms).
- We perform transfer learning from the *A9-Dataset* to the dataset from the *Regensburg Next* project, and achieve a 3D *mAP*@0.25 of 93.49% on the Car class of the target test set using 40 recall positions.

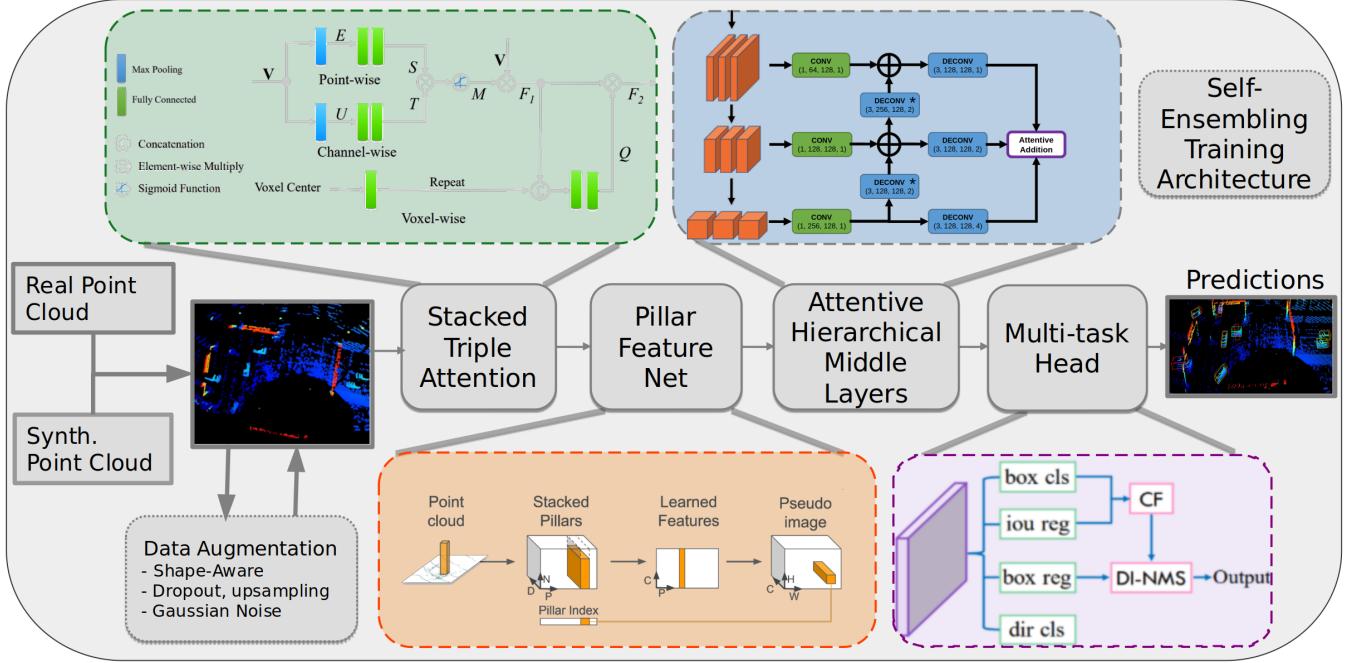


Fig. 1. Overview architecture of DASE-ProPillars, a LiDAR-only single-stage pillar-based 3D object detector. The detector is based on PointPillars [17], with the following five additional extensions. 1) Data Augmentation (Shape-Aware [40], Dropout, upsampling and Gaussian Noise). 2) Stacked Triple Attention Mechanism [20]. 3) Attentive Hierarchical Middle Layers. 4) Multi-task detection head. 5) Self-Ensembling Training Architecture [40]. The stacked triple attention module extracts features from the semi-synthetic point cloud using the triple attention mechanism, including channel-wise, point-wise, and voxel-wise attention to enhance the learned features. The pillar feature net turns point-wise features into pillar features and scatters the pillar features into a pseudo image. The hierarchical middle layers perform 2D convolution operations on the pseudo image. Hierarchical feature maps are concatenated with attentive addition. Finally, the multi-task head is used for the final prediction, that includes an IoU prediction to alleviate the misalignment between the localization accuracy and classification confidence. In addition, we introduce two new training techniques: the shape-aware data augmentation module and the self-ensembling teacher and student training framework.

## II. RELATED WORK

In the first part we compare one-stage and two stage methods, types of point cloud representations, single- and multi-frame approaches, supervised and unsupervised as well as center and anchor-based methods. In the second part we also analyze the increasing importance of data augmentation and the generation of synthetic point clouds for 3D object detection to solve the domain shift problem.

### A. 3D Object Detection Models

According to the form of feature representation, LiDAR-only 3D object detectors can be divided into four main streams, i.e. point-based, voxel-based, range-view based and multi-view-based methods [42]. In point-based methods, features maintain the form of point-wise features, either by a sampled subset or derived virtual points. PointRCNN [26] uses a PointNet++ backbone [25] to extract point-wise features from the raw point cloud, and performs foreground segmentation. Then for each foreground point, it generates a 3D proposal followed by a point cloud ROI pooling and a canonical transformation-based bounding box refinement process. Point-based methods usually have to deal with a huge amount of point-wise features, which leads to a relatively lower inference speed. To accelerate point-based methods, 3DSSD [37] introduces feature farthest-point-sampling (F-FPS), which computes the feature distance for sampling,

instead of Euclidean distance in traditional distance farthest-point-sampling (D-FPS). The inference speed of 3DSSD is competitive with voxel-based methods.

SECOND [34] proposes to use a sparse convolutional middle extractor [8], [19], which greatly speeds up the inference time. In PointPillars [17], the point cloud is divided into pillars (vertical columns), which are special voxels without partition along the z-direction. The feature map of pillars can be treated as a pseudo-image, and therefore the expensive 3D convolution is replaced by 2D convolution. PointPillars achieves the fastest speed with the help of TensorRT acceleration. SA-SSD [9] adds a detachable auxiliary network to the sparse convolutional middle layers to predict a point-wise foreground segmentation and a center estimation task, which can provide additional point-level supervision. SA-SSD also proposes a part-sensitive warping (PS-Warp) operation as an extra detection head. It can alleviate the misalignment between the predicted bounding boxes and classification confidence maps, since they are generated by two different convolutional layers in the detection head. CIA-SSD [39] also notices the misalignment issue. It designs an IoU-aware confidence rectification module, using an additional convolutional layer in the detection head to make IoU predictions. The predicted IoU value is used to rectify the classification score. By introducing only one additional convolutional layer, CIA-SSD is more lightweight than SA-



Fig. 2. 1) Birds-eye view (BEV) of the intersection. 2) 3D reconstruction of the intersection. 3) Intersection modelled in the CARLA simulator. 4) Synthetic point cloud frames with labeled ground truth vehicles extracted from the CARLA simulator.

SSD. SE-SSD [40] proposes a self-ensembling post-training framework, where a pre-trained teacher model produces predictions that serve as soft targets in addition to the hard targets from the label. These predictions are matched with student's predictions by their IoU and supervised by the consistency loss. Soft targets are closer to the predictions from the student model and therefore can help the student model to fine-tune its predictions. The Orientation-Aware Distance-IoU Loss is proposed to replace the traditional smooth- $L_1$  loss of bounding box regression in the post training, in order to provide a fresh supervisory signal. SE-SSD also designs a shape-aware data augmentation module to improve the generalization ability of the student model.

SE-SSD [40] proposes a one-stage approach and adopts a single-stage detection architecture consisting of a teacher and student model. These two models share the same structure that resembles the CIA-SSD [39] architecture. The teacher model makes predictions based on the original point cloud, these are transformed into "soft targets" used to train the student. The student model makes predictions based on especially augmented data and is trained both on the hard and soft targets via a Consistency Loss as well as a new Orientation Aware Distance-IoU loss (ODIoU). This loss emphasizes the orientation of the bounding boxes as well as the alignment of the center points. The authors propose a new Shape-Aware Data Augmentation scheme that performs dropout, swapping and sparsification of points. This data augmentation is applied to the point cloud data the student model is trained on. In this way, by using both a teacher and a student single-stage object detector, the framework can boost the precision of the detector significantly without incurring extra computation during the inference. At the same time, it also inherits the advantages of one-stage methods, namely the speed.

A recent two-stage method with state-of-the-art performance in 3D object detection is Pyramid R-CNN [21] that concentrates on handling the sparsity and non-uniform distribution of point clouds. The authors propose a novel second-stage module (called pyramid ROI-head), that adaptively extracts the features from the sparse points of interest. Utilizing features only inside the ROIs performs well in 2D detection models mainly for two reasons. First, the input feature map is dense and second, the collected pixels have large receptive fields. However, in 3D models, the points of interest are sparse and non-uniformly distributed inside the

ROIs. Therefore, accurately inferring the sizes and categories of objects becomes hard when collecting the features of few individual points and not gathering enough information from neighbours. Pyramid ROI effectively solves this problem by constructing a pyramid grid structure that contains the ROI-grid points both inside and outside ROIs, so that both fine-grained shape structures for accurate box refinement as well as large context information for identifying incomplete objects can be captured by grid points inside ROIs and outside ROIs, respectively. The authors of this paper performed experiments on both the KITTI Dataset and the Waymo Open Dataset [27]. They showed that Pyramid R-CNN outperforms the state-of-the-art 3D detection models on these two datasets. On the Waymo Open Dataset their Pyramid-PV model achieves 81.77% L1 mAP which ranks second on the Waymo vehicle detection leader board and surpasses all the LiDAR-only approaches.

As conventional 3D convolutional backbones in voxel-based 3D detectors are not able to efficiently capture extensive context information, Voxel Transformer (VoTr) [22] is proposed to resolve this issue by introducing a voxel-based transformer backbone for 3D object detection from point clouds. It consists of a series of sparse voxel modules, which can extract features at empty locations and thus are responsible for the downsampling of the voxel-grids and submanifold voxel modules, which perform multi-head self attention strictly on non-empty voxels to keep the original 3D structure with increasing the receptive fields. The attention mechanism is split up into two components, called local attention and dilated attention. The local attention approach focuses on the neighboring region to preserve detailed information and the dilated attention obtains a large attention range with only a few attending voxels by gradually increasing the search step. VoTr can be applied to both, single-stage and two-stage detectors and comparing with backbone of the respective module, the results on the KITTI test set regarding AP calculated by 40 recall positions for the class car are increased slightly.

Signal miss, external and self occlusion often cause shape misses for disordered point clouds. Behind the Curtain Detector (BtcDet) [33] deals with this problem by learning the object shape priors and estimate the complete object shapes including the partially occluded points. Therefore, points are filled into the labeled bounding boxes and using the recovered shape miss, the detection results are improved. For

the training process, the complete object shapes are approximated using the ground truth labels of the corresponding objects. For cars and cyclist, the objects points are mirrored against the middle section plane of the bounding box and a heuristic  $H(A,B)$  determines if a source object B covers most parts of a target object A. The heuristic also provides points that can fill the target object's shape miss. The detection pipeline of BtcDet is built up as followed: First, the regions of occlusion and signal miss have to be identified after the spherical voxelization for the point cloud. Then, a shape occupancy network estimates the probability of object shape occupancy using the created training targets, which consists of the approximated complete object shapes. The extracted point cloud 3D features are then sent to a region proposal network to generate 3D proposals, which are refined in the last step, the proposal refinement step.

### B. Domain Adaptation

Domain adaptation is a type of transfer learning and aims to transfer knowledge from a source domain, for which annotated data is available to a target domain, for which no or only less annotated data is available. Semi-supervised domain adaptation uses a few labeled examples from the target domain to learn a target model and unsupervised domain adaptation exploits only the labeled data from the source domain without having any annotated target domain data [32]. The domain adaptation methods can be divided into four different approaches, which are either data-driven, such as domain-invariant data representation, domain mapping, and normalization statistics or model-driven like the domain-invariant feature learning [29]. With the use of these domain adaptation methods, the gap between the source and the target domain should be mitigated [14].

Wang et al [30] propose a dataset-to-dataset, semi-supervised domain adaptation for 3D object detection and provide a baseline for 3D object detection adaptation across countries using normalization statistics domain adaptation methods. As only annotated datasets are considered, few-shot fine-tuning enables to increase the accuracy by selecting 10 labeled scenes from the target domain, which are added to the source domain during training. As the car sizes vary in several countries, the object sizes of the source domain differ from the objects sizes of the target domain. Therefore, the already trained object detector is modified that the predicted box sizes can better match the previously determined target statistics. To fit the adjusted box size, the corresponding labels are scaled up or shrunked down and new point clouds with the associated labels are whose care sizes are much similar to the target domain data are generated. This step is called statistical normalization.

ST3D [36] provides a self-training pipeline for unsupervised domain adaptation on 3D object detection from point clouds, where no annotated data in the target domain is available. As already mentioned in [30] the sizes of the objects vary in different datasets due to, e.g. geographical location in which the data were recorded. For this reason, ST3D proposes pre-training of the 3D detector on source

domain with random object scaling (ROS) strategy to mitigate the negative effects of source domain bias. Using the 3D object detector, pseudo labels for the unlabeled target data are generated. The quality-aware triplet memory bank (QTMB) modules parses the object predictions to the pseudo labels, however the negative impacts on pseudo labeled objects lead to noisy supervisory information and instability for self-training. The memory bank updates the pseudo labels that also serve as labels for subsequent model training. The curriculum data augmentation (CDA) module allows to generate gradually increasingly diverse and potentially hard examples to improve the model. This enables the model to learn from challenging samples while making the examples more difficult during training.

## III. DATA GENERATION

### A. Semi Synthetic Data Generation

We created a semi-synthetic dataset (*proSynthSemi*) with 6000 point cloud frames using the CARLA simulator and train our *DASE-ProPillars* model on it. Figure 2 shows an example of an intersection with generated traffic in the CARLA simulator. A simulated LiDAR sensor represents a real Ouster OS1-64 (gen. 2) LiDAR sensor with 64 channels and a range of 120 m. In the simulation a noise model is used with a standard deviation of 0.1 to disturb each point along the vector of its raycast. The LiDAR emits 1.31 million points per second and runs at 10 Hz (131,000 points per frame). The extracted point clouds were stored in .pcd (ASCII) files, whereas the labels were stored in .json files according to the OpenLABEL standard [1]. To get more realistic point clouds, the points of the objects in the simulated point clouds are extracted and included into a background of a point cloud captured by a real Ouster OS1-64 (gen.2) LiDAR, which does not contain any objects. Before the simulated object points are inserted into the real background point set, those points in the real point cloud are cut out, which are inside the simulated objects or below, respectively the points of the ground plane. As the height profiles of simulated and real data do not coincide, the z-coordinate of the objects must be adjusted in order to place the objects on the ground plane. The RANSAC algorithm [5] can be used to determine the ground plane of the real point cloud and thus, calculating the corresponding height profile. Subsequently, a pipeline is applied to the background points using both gaussian noise and drop out of points in order to get more variance in the point clouds. This gives the advantage to get more realistic point clouds and having labeled objects from the simulation.

### B. Real Data Generation

The first step is to record point cloud data from roadside LiDARs. These recordings should cover a large variety of different scenarios, e.g. traffic jam, overtaking, lane change, lane merge, tail-gate events, accidents, etc.. In the next step (data selection), scenarios of high importance are selected. The point clouds are then converted into the right format

(.pcd) and the ground is removed using the *RANSAC* algorithm. Dynamic objects are then labeled using a custom 3D point cloud labeling tool called *proAnno*, that is extended version of the open source 3D bounding box annotation tool *3D-BAT* [43]. 50 LiDAR frames were manually labeled to fine-tune the detector on real data.

#### IV. APPROACH

We design a real-time LiDAR-only 3D object detector (*DASE-ProPillars*) that could be applied to real-world scenarios to solve the domain shift problem. The architecture of the designed *DASE-ProPillars* model is shown in Figure 1. We describe the each part of the detector in the following sections.

**Normalization.** As the bounding boxes of the classes differ in the semi-synthetic and the A9 dataset, the boxes are normalized to an average size of the bounding boxes for each class. Since the manually labeled data can often result in incorrect sizes of the length, width and height of the bounding box, the normalized bounding boxes of the synthetic labels are used instead. For the synthetic data, the exact dimensions can be extracted directly from the simulation. With the help of normalization, the distinction between the individual classes can be improved, which is useful, e.g., for the classes "van" and "car". In addition, the normalized sizes are used in a domain adaptation to adjust the source domain data to the target domain data.

**Voxelization.** We devide the raw point cloud into vertical pillars before feeding them into a neural network. These are special voxels that are not splitted along the vertical axis. Pillars have several advantages over voxels. A pillar-based backbone is faster than a voxel-based backbone due to fewer grid cells. Time consuming 3D convolutional middle layers are also being eliminated and instead 2D convolutions are being used. We also do not need to manually tune the bin size along the z-direction hyperparameter. If a pillar contains more points than specified in the threshold, then the points are being subsampled to the threshold using farthest point sampling [7]. If a pillar contains fewer points than the threshold, then it is padded with zeros to make the dimensions consistent. Due to the sparsity issue most of the pillars are empty. We record the coordinates of non-empty pillars according to the pillar's center their index. Empty pillars are not being considered during the feature extraction until all pillars are being scattered back to a pseudo image for 2D convolution.

**Stacked Triple Attention.** The *Stacked Triple Attention* module is used for a more robust and discriminative feature representation. Originally introduced in *TANet* [20] by Liu et. al., the stacked triple attention module enhances the learning of hard to detected objects and deals better with noise points. This method can be applied on both, voxel and pillar-based point clouds. The attention mechanism in this module follows the Squeeze-and-Excitation pattern [11]. If channel-wise attention is applied to an input tensor with shape  $(H \times W \times C)$ , then first a global pooling operation (max pooling) is used to pool the tensor to shape  $(1 \times 1 \times C)$ , called

squeeze operation. Then, two fully connected (FC) layers are applied to the squeezed tensor attention score, called excitation operation. Between the two FC layers, the feature dimension is reduced and then recovered with a reduction ratio which forms a bottleneck structure. After that, a sigmoid function is applied to get the attention score. Finally, the  $(1 \times 1 \times C)$  tensor is multiplied element-wise to get the original  $(H \times W \times C)$  feature.

The input to the module is a  $(P \times N \times C)$  tensor, where  $P$  is the number of non-empty pillars,  $N$  is the maximum number of points in each pillar, and  $C$  is the dimension of the input point-wise feature. At the beginning, we have a 9-dimensional ( $C = 9$ ) feature vector  $(x, y, z, r, x_c, y_c, z_c, x_p, y_p)$ , where  $x, y, z$  are the coordinates of the point,  $r$  is the intensity,  $x_c, y_c, z_c$  are the distance to the arithmetic mean of all points inside the pillar,  $x_p, y_p$  are the location of the pillar from the pillars center. The triple attention (TA) module extracts features inside each pillar, using point-wise, channel-wise and voxel-wise attention. All three attention scores are combined to form the final output feature. To further exploit the multi-level feature attention, two triple attention modules are stacked with a structure similar to the skip connections in ResNet [10]. The first module takes the raw point cloud 9-dim features as input, while the second one works on the extracted high dimensional features. For each TA module the input is concatenated or summed to the output to fuse more feature information. Each TA module is followed by a fully connected layer to increase the feature dimension. Inside the TA modules, the attention mechanism only re-weights the features, but does not increase their dimensions.

**Pillar Feature Net.** We choose *PointPillars* [17] as our baseline and improve its 3D detection performance at the expense of inference time. The inference speed of *PointPillars* is 42 Hz without the acceleration of *TensorRT*. Since there is a trade-off between speed and accuracy, we can further boost the accuracy by incorporating additional modules without sacrificing the inference speed too much. The pillar feature net (PFN) shown in Fig. 1 takes pillars as input, extracts pillar features, and scatters pillars back to a pseudo image for 2D convolution operations in the middle layers. The pillar feature net acts as an additional feature extractor to the stacked triple attention module. The point-wise pillar-organized features from the stacked TA module with shape  $(P \times N \times C)$  are fed to a set of PFN layers. Each PFN layer is simplified PointNet [24], which consists of a linear layer, Batch-Norm [12], ReLU [23], and max pooling. The max-pooled features are concatenated back to the ReLU's output to keep the point-wise feature dimension inside each pillar, until the last FPN layer. The last FPN layer makes the final max pooling and outputs a  $(P \times C)$  feature as the pillar feature. Pillar features are then scattered back to the original pillar location, forming a  $(C \times H \times W)$  pseudo image, where  $H$  and  $W$  are the height and width of the pillar grid. Here the location of empty pillars is padded with zeros.

**Attentive Hierarchical Middle Layers.** We exchange the default backbone of *PointPillars* with an *Attentive Hier-*

*chical Backbone* to perform 2D convolution on the pseudo image from the pillar feature net. In the first stage, the spatial resolution of the pseudo image is gradually downsampled by three groups of convolutions. Each group contains three convolutional layers, where the first one has a stride of two for downsampling, and the two subsequent layers act only for feature extraction. After downsampling, deconvolution operations are applied to recover the spatial resolution. Deconvolutional layers (marked with an asterisk) recover the size of feature maps with stride 2 and element-wise add them to upper branches. The remaining three deconvolutional layers make all three branches have the same size (half of the original feature map). Then the final three feature maps are combined by an attentive addition to fuse both, spatial and semantic features. The attentive addition uses the plain attention mechanism. All three feature maps are being passed through a convolutional operation and are channel-wise concatenated as attention scores. The softmax function generates the attention distribution and feature maps are multiplied with the corresponding distribution weight. The element-wise addition in the end gives the final attention output, a  $(C \times H/2 \times W/2)$  feature map.

**Multi-task Head.** The multi-task head outputs the final class (based on a confidence score), the 3D box position  $(x, y, z)$ , dimensions  $(l, w, h)$ , rotation  $(\theta)$  and the direction of the detected object. The direction (front/back) is being classified to solve the problem that the sine-error loss [35] cannot distinguish flipped boxes. Four convolutional layers operate on the feature map separately. One of the four heads is the IoU prediction head that predicts an IoU between the ground truth bounding box and the predicted box. It was introduced in CIA-SSD [39] to deal with the misalignment between the predicted bounding boxes and corresponding classification confidence maps. The misalignment is mainly because these two predictions are from different convolutional layers. Based on this IoU prediction, we use the confidence function (CF) to correct the confidence map and use the distance-variant IoU-weighted NMS (DI-NMS) module post-process the predicted bounding boxes. The distance-variant IoU-weighted NMS is designed to deal with long-distance predictions, to better align far bounding boxes with ground truths, and to reduce false-positive predictions. If the predicted box is close to the origin of perspective, we give higher weights to those box predictions with high IoU. If the predicted box is far, we give relatively uniform weights, to get a more smooth final box.

**Data Augmentation.** Data augmentation has proven to be an efficient way to better exploit the training dataset and help the model to be more generalized. We use the shape-aware data augmentation method proposed by SE-SSD [40]. This module simplifies the handling of partial occlusions, sparsity and different shapes of objects in the same class. Some traditional augmentation methods are also applied before the shape-aware augmentation, e.g. rotation, flipping, and scaling. For the generation of semi-synthetic data, several data augmentation techniques are also implemented to increase the variance of the point clouds. Therefore, in every

second frame, 0-20% of all points are dropped out and gaussian noise with a sigma of 0.2 is added to 20-40% of all points. These techniques increase the variance of point clouds and provide more robust and diverse data. Data augmentation plays an important role for domain adaptation methods, as it is used by several methods [36], [38],[13]. It is noticeable, that the point density of the target domain is more important than the point density of the source domain [36]. Subsequently, cropping and drop out of points, respectively point cloud upsampling is an important step to adjust the number of points of the source set to the target set. Statistics for both, the source and the target domain dataset, are calculated with respect to the number of points of the total point cloud and, if annotated data is available for the target domain data, the average number of points per object. Then, the data augmentation techniques drop out and upsampling are used to match the source domain dataset to the target domain dataset. To better illustrate this effect, a domain adaptation is applied in section V from the synthetic A9 dataset (source domain) to the dataset of the *Regensburg Next* project (target domain). Note that the number of points for the target domain set is reduced by a factor of 2.72 compared to the source domain set, whereas the average number of points for the class "car" is increased by a factor of 1.83 compared to the source dataset. Due to the different LiDARs used for each project, there is more overlap between the four permanently installed Blickfeld Cube 1 LiDARs in the *Regensburg Next* project. This sensor-to-sensor domain adaptation is considered in more detail in section V.

**Self-Ensembling Training Framework.** In addition, we introduce the self-ensembling training framework [40] to do a post training: We first train the model without self-ensembling, and then we take the pre-trained model as a teacher model to train the student model that has the same network structure. The predictions of the teacher model can be used as soft supervision. Combined with the hard supervision from the ground truth, we can provide more information to the student model. The student model and the teacher model are initialized with the same pre-trained parameters. The overall loss for training the student model consists of:

$$\mathcal{L}_{student} = \mathcal{L}_{cls}^s + \omega_1 \mathcal{L}_{OD-IoU}^s + \omega_2 \mathcal{L}_{dir}^s + \mu_t \mathcal{L}_{consist}, \quad (1)$$

where  $\mathcal{L}_{cls}^s$  is the focal loss [18] for box classification,  $\mathcal{L}_{OD-IoU}^s$  is the OD-IoU loss for bounding box regression,  $\mathcal{L}_{dir}^s$  is the cross-entropy loss for direction classification,  $\mathcal{L}_{consist}$  is the consistency loss, that is a sum of the bounding box loss and the classification loss,  $\omega_1$ ,  $\omega_2$ ,  $\lambda$  and  $\mu_t$  are weights of losses.

During the post-training, the parameters of the teacher model are updated based on the parameters of the student model using the exponential moving average (EMA) strategy.

## V. EVALUATION

To prove the effectiveness of our proposed modules in *DASE-ProPillars*, we evaluate the model on the *A9-Dataset* which is a novel infrastructure dataset in the autonomous

driving domain with labeled road-side sensor data. Furthermore, we fine-tune and evaluate the model on a second infrastructure dataset from the *Regensburg Next* project with road-side LiDAR sensors. The training and evaluation was performed on 3 x NVIDIA GeForce RTX 3090 GPUs.

#### A. A9-Dataset

We use the *DASE-ProPillars* as our model, train on the training set of the A9 dataset for 80 epochs, and evaluate on the test set. We report our result using 3D and BEV under IoU threshold 0.5 and 0.25 like in [41]. The confidence score is limited with threshold 0.1, but we again decrease the NMS IoU threshold to 0.2, exactly the same setting as the convention in nuScenes [2]. The LiDAR frames of the second batch of the A9 dataset are labeled on the intersection. We test the case of using 0.1 as the NMS threshold. The result is shown in Table I.

Metric	3D mAP		BEV mAP	
IoU threshold	0.5	0.25	0.5	0.25
SE-ProPillars	30.13	50.09	40.21	51.53
DASE-ProPillars (Ours)	<b>54.38</b>	<b>80.65</b>	<b>55.10</b>	<b>83.38</b>

TABLE I

3D OBJECT DETECTION RESULTS OF *DASE-ProPillars* ON THE A9 TEST SET. WE REPORT THE 3D AND BEV MAP OF CAR UNDER 0.5 AND 0.25 IOU THRESHOLD, WITH 40 RECALL POSITIONS.

#### B. Regensburg Next Dataset

The results of the *DASE-ProPillars* model trained on the synthetic dataset are used to improve the results on a similar infrastructure dataset, from the *Regensburg Next* project [28], which consists of synthetic data including 3000 frames. The simulation environment *Gazebo* was used to randomly place objects in a predefined region of an intersection to generated annotated data. The LiDARs in the *A9-Dataset* and the semi-synthetic dataset are mounted on traffic sign gantry bridges, whereas the LiDAR sensors in the *Regensburg Next* project are mounted on road-side rooftops in a height of approximately seven meters above the ground plane. The recording of real point cloud data for the *Regensburg Next* project is planned for the future. We use *DASE-ProPillars* again as our model, train on the training set of the synthetic *A9-Dataset* for 80 epochs and fine-tune on the *Regensburg Next* dataset for 40 epochs, with a decrease of the learning rate of 50%. For comparison reasons we also trained our *DASE-ProPillars* model directly on the *Regensburg Next* dataset for 40 epochs using the same hyperparameters as above. The NMS IoU threshold is decreased to 0.2 for both models. As mentioned above, the average number of points per object is increased by a factor of 1.83 compared to the synthetic A9 dataset. Therefore, point upsampling was used, to match the number of average points per objects for both domains. Since the average number of points of the total point cloud in the target set is significantly lower (factor 2.72) compared to the source set, only points of the objects are upsampled. In order to reduce the difference in terms of total number of points of the respective point clouds

simultaneously, drop out is used to reduce the number of total points of the source set. Table II shows the impact of the several data augmentation techniques of a *DASE-ProPillars* model trained only on the source set and evaluated directly on the target set without the use of any fine-tuning. The Table shows, that the upsampling of the object points improved the 3D object detection, whereas the subsequent drop out had led to slight deterioration, contrary to expectations. Table III shows the results for the models trained only on the dataset of the *Regensburg Next* project compared with the model which was trained on the synthetic A9 dataset and finetuned on the target domain set.

Metric	3D mAP		BEV mAP	
IoU threshold	0.25	0.25	0.25	0.25
upsampling + drop out (0.5)	49.64	61.14		
drop out (0.25)	60.92	69.05		
original	61.72	72.83		
upsampling	<b>63.11</b>	<b>76.65</b>		

TABLE II

3D OBJECT DETECTION RESULTS OF *DASE-ProPillars* ON THE TEST SET OF THE *Regensburg Next* PROJECT. WE REPORT THE 3D AND BEV MAP OF CAR UNDER 0.25 IOU THRESHOLD, WITH 40 RECALL POSITIONS INCLUDING SEVERAL DATA AUGMENTATION TECHNIQUES

Metric	3D mAP		BEV mAP	
IoU threshold	0.5	0.25	0.5	0.25
DASE-Propillars	23.26	89.87	27.61	93.42
DASE-Propillars + fine-tuning	<b>33.73</b>	<b>93.49</b>	<b>38.81</b>	<b>94.79</b>

TABLE III

3D OBJECT DETECTION RESULTS OF *DASE-ProPillars* ON THE TEST SET OF *Regensburg Next* PROJECT. WE REPORT THE 3D AND BEV MAP OF CAR UNDER 0.5 AND 0.25 IOU THRESHOLD, WITH 40 RECALL POSITIONS.

## VI. CONCLUSIONS

In this work we presented our *DASE-ProPillars* 3D object detector that is an improved version of the *PointPillars* model. We show generalization ability of our model and make it more robust via domain adaptation. We replace the detection head with a more lightweight multi-task head. We add two training techniques to our baseline: the shape-aware data augmentation module and the self-ensembling training architecture. Sufficient data collection is key to train a model and achieve a good accuracy. We do several sets of experiments for each module to prove its accuracy and runtime performance.

To sum up, the *DASE-ProPillars* 3D object detector is a significant contribution within the area of LiDAR-based 3D perception to support self-driving vehicles and improve road traffic safety.

## ACKNOWLEDGMENT

This work was funded by the Federal Ministry of Transport and Digital Infrastructure, Germany as part of the *Providentia++* research project (Grant Number: 01MM19008A). The authors would like to express their gratitude to the funding agency and to numerous students who have contributed to the creation of the first batch of the *A9-Dataset*.

## REFERENCES

- [1] “Asam e.v. openlabel v1.0.0 standardization project.” <https://www.asam.net/project-detail/asam-openlabel-v100/>.
- [2] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liang, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuscenes: A multimodal dataset for autonomous driving,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.
- [3] C. Creß and A. C. Knoll, “Intelligent transportation systems with the use of external infrastructure: A literature survey,” *arXiv preprint arXiv:2112.05615*, 2021.
- [4] C. Creß, W. Zimmer, L. Strand, M. Fortkord, S. Dai, V. Lakshminarasimhan, and A. Knoll, “A9-dataset: Multi-sensor infrastructure-based dataset for mobility research,” *arXiv preprint arXiv*, 2022.
- [5] K. G. Derpanis, “Overview of the ransac algorithm,” *Image Rochester NY*, vol. 4, no. 1, pp. 2–3, 2010.
- [6] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “Carla: An open urban driving simulator,” in *Conference on robot learning*. PMLR, 2017, pp. 1–16.
- [7] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, “The farthest point strategy for progressive image sampling,” *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, 1997.
- [8] B. Graham, M. Engelcke, and L. Van Der Maaten, “3d semantic segmentation with submanifold sparse convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9224–9232.
- [9] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, “Structure aware single-stage 3d object detection from point cloud,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 873–11 882.
- [10] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [11] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.
- [12] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International conference on machine learning*. PMLR, 2015, pp. 448–456.
- [13] M. Jaritz, T. Vu, R. de Charette, É. Wirbel, and P. Pérez, “xmuda: Cross-modal unsupervised domain adaptation for 3d semantic segmentation,” *CoRR*, vol. abs/1911.12676, 2019. [Online]. Available: <http://arxiv.org/abs/1911.12676>
- [14] D. Jia, A. Hermans, and B. Leibe, “Domain and modality gaps for lidar-based person detection on mobile robots,” *CoRR*, vol. abs/2106.11239, 2021. [Online]. Available: <https://arxiv.org/abs/2106.11239>
- [15] A. Krämer, C. Schöller, D. Gulati, V. Lakshminarasimhan, F. Kurz, D. Rosenbaum, C. Lenz, and A. Knoll, “Providentia-a large-scale sensor system for the assistance of autonomous vehicles and its evaluation,” *Journal of Field Robotics*, 2022.
- [16] V. Lakshminarasimhan and A. Knoll, “C-v2x resource deployment architecture based on moving network convoys,” in *2020 IEEE 91st vehicular technology conference (VTC2020-Spring)*. IEEE, 2020, pp. 1–6.
- [17] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “Pointpillars: Fast encoders for object detection from point clouds,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.
- [18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
- [19] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, “Sparse convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 806–814.
- [20] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, “Tanet: Robust 3d object detection from point clouds with triple attention,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 11 677–11 684.
- [21] J. Mao, M. Niu, H. Bai, X. Liang, H. Xu, and C. Xu, “Pyramid r-cnn: Towards better performance and adaptability for 3d object detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2723–2732.
- [22] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, and C. Xu, “Voxel transformer for 3d object detection,” 2021.
- [23] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Icm*, 2010.
- [24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [25] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++ deep hierarchical feature learning on point sets in a metric space,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5105–5114.
- [26] S. Shi, X. Wang, and H. Li, “Pointrenn: 3d object proposal generation and detection from point cloud,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 770–779.
- [27] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al., “Scalability in perception for autonomous driving: Waymo open dataset,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.
- [28] V. R. Torunsky, “Pilotprojekt mit Vorbildcharakter,” p. 1.
- [29] L. T. Triess, M. Dreissig, C. B. Rist, and J. M. Zöllner, “A survey on deep domain adaptation for lidar perception,” *CoRR*, vol. abs/2106.02377, 2021. [Online]. Available: <https://arxiv.org/abs/2106.02377>
- [30] Y. Wang, X. Chen, Y. You, L. E. Li, B. Hariharan, M. E. Campbell, K. Q. Weinberger, and W. Chao, “Train in germany, test in the USA: making 3d object detectors generalize,” *CoRR*, vol. abs/2005.08139, 2020. [Online]. Available: <https://arxiv.org/abs/2005.08139>
- [31] J. Wu, W. Zimmer, and A. Knoll, “Real-time lidar-based 3d object detection on the providentia++ test stretch using a single-stage architecture,” Master’s thesis, Technische Universität München, 2021, unpublished thesis.
- [32] A. Xiao, J. Huang, D. Guan, F. Zhan, and S. Lu, “Synlidar: Learning from synthetic lidar sequential point cloud for semantic segmentation,” *CoRR*, vol. abs/2107.05399, 2021. [Online]. Available: <https://arxiv.org/abs/2107.05399>
- [33] Q. Xu, Y. Zhong, and U. Neumann, “Behind the curtain: Learning occluded shapes for 3d object detection,” *CoRR*, vol. abs/2112.02205, 2021. [Online]. Available: <https://arxiv.org/abs/2112.02205>
- [34] Y. Yan, Y. Mao, and B. Li, “Second: Sparsely embedded convolutional detection,” *Sensors*, vol. 18, no. 10, p. 3337, 2018.
- [35] ———, “Second: Sparsely embedded convolutional detection,” *Sensors*, vol. 18, no. 10, p. 3337, 2018.
- [36] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, “ST3D: self-training for unsupervised domain adaptation on 3d object detection,” *CoRR*, vol. abs/2103.05346, 2021. [Online]. Available: <https://arxiv.org/abs/2103.05346>
- [37] Z. Yang, Y. Sun, S. Liu, and J. Jia, “3dssd: Point-based 3d single stage object detector,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 040–11 048.
- [38] L. Yi, B. Gong, and T. A. Funkhouser, “Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds,” *CoRR*, vol. abs/2007.08488, 2020. [Online]. Available: <https://arxiv.org/abs/2007.08488>
- [39] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, “Cia-ssd: Confident iou-aware single-stage object detector from point cloud,” *arXiv preprint arXiv:2012.03015*, 2020.
- [40] W. Zheng, W. Tang, L. Jiang, and C.-W. Fu, “Se-ssd: Self-ensembling single-stage object detector from point cloud,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 494–14 503.
- [41] X. Zhou, W. Zimmer, E. Erçelik, and A. Knoll, “Real-time lidar-based 3d object detection on the highway,” Master’s thesis, Technische Universität München, 2021, unpublished thesis.
- [42] W. Zimmer, E. Ercelik, X. Zhou, X. Jair Diaz Ortiz, and A. Knoll, “A survey of robust 3d object detection methods in point clouds,” *arXiv preprint arXiv:submit/4161670*, 2022.
- [43] W. Zimmer, A. Rangesh, and M. Trivedi, “3d bat: A semi-automatic, web-based 3d annotation toolbox for full-surround, multi-modal data streams,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1816–1821.