GACE: Geometry Aware Confidence Enhancement for Black-box 3D Object Detectors on LiDAR-Data

David Schinagl[1,2] Georg Krispel[1] Christian Fruhwirth-Reisinger[1,2] Horst Possegger[1] Horst Bischof[1,2]
{david.schinagl,georg.krispel,reisinger,possegger,bischof}@icg.tugraz.at
[1] Graz University of Technology [2] Christian Doppler Laboratory for Embedded Machine Learning

Abstract

Widely-used LiDAR-based 3D object detectors often neglect fundamental geometric information readily available from the object proposals in their confidence estimation. This is mostly due to architectural design choices, which were often adopted from the 2D image domain, where geometric context is rarely available. In 3D, however, considering the object properties and its surroundings in a holistic way is important to distinguish between true and false positive detections, e.g. *occluded pedestrians in a group. To address this, we present GACE, an intuitive and highly efficient method to improve the confidence estimation of a given black-box 3D object detector. We aggregate geometric cues of detections and their spatial relationships, which enables us to properly assess their plausibility and consequently, improve the confidence estimation. This leads to consistent performance gains over a variety of state-of-the-art detectors. Across all evaluated detectors, GACE proves to be especially beneficial for the vulnerable road user classes,* i.e. *pedestrians and cyclists.*

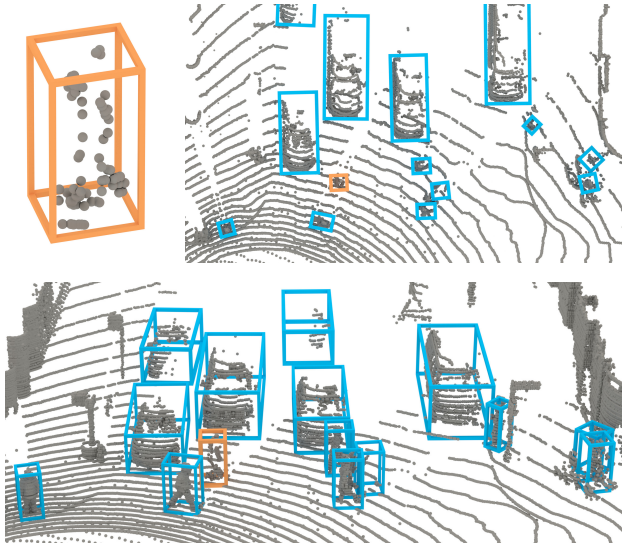

Figure 1. Context matters: a baseline detector struggles at detecting true positive objects confidently if the sampling pattern is atypical, *e.g.* the occluded pedestrian in the orange bounding box (close-up top-left). GACE exploits the geometric properties of the detection and its surrounding objects to significantly increase the score for this detection, which is intuitively correct for a human observer considering this scene.

1. Introduction

Three-dimensional perception of surrounding objects is a critical component for autonomous vehicles and robots. Many modern perception systems use point cloud data from LiDAR (Light Detection and Ranging) sensors for this task, since they can provide accurate 3D information even over long distances. The popularity of these sensors can be seen in the increased research interest in LiDAR-based 3D object detection approaches, *e.g.* [10,29,38,40,45,47] and the large number of recently published autonomous driving datasets that include LiDAR data, *e.g.* [1, 6, 20, 33].

However, the characteristics of LiDAR data impose significant challenges for object detection. Unlike pixels in an image, which are aligned in a regular grid, point clouds represent 3D data as a collection of individual points in space, each with its own set of coordinates. In addition to the unstructured nature of the data, the highly variable point density poses a major challenge. Due to the angular offset of the LiDAR beams, the density is highly dependent on the distance to the object and can be altered by occlusions in the foreground. This often requires detecting objects based on very few data points, which is especially true for classes with smaller spatial dimensions, such as pedestrians and cyclists. For example, in the Waymo Open Dataset [33], one of the most widely used and challenging datasets to date, about 30 percent of all annotated pedestrians consist of less than merely 20 points. Detecting these sparsely sensed objects naturally leads to a large number of false positive detections at test time. For this reason, determining a meaningful confidence value for the detections is critical to find a trade-off between precision and recall that adequately dis-

tinguishes true positives from false positives. The potential that could be exploited by improving the confidence score is considerable: Suppose we have an oracle that could correctly classify the detections of a SECOND [38] model on the Waymo dataset into true and false positives. This would increase the LEVEL_1 average precision for vehicles by $+3.96$AP and, more importantly, for pedestrians and cyclists by as much as $+10.71$AP and $+13.74$AP, respectively.

Existing 3D object recognition pipelines, including confidence estimation approaches, were largely inspired by 2D image-based object recognition models and then gradually adapted to point cloud processing. However, the conventional *backbone - neck - head* architecture of the 2D detection model was largely retained, *e.g.* in [8, 10, 15, 27, 38, 45]. After extracting features (point-based, voxel-based, or region-based) over multiple levels in the backbone, they are fused in the neck module and then passed to the detection head, where bounding box regression and confidence estimation are performed on a dense feature representation. Typically, separate branches are used within the head for bounding box regression and confidence estimation, each consisting of one or more fully connected layers on top of the common feature representation.

Unlike image-based object detection, there are several highly relevant geometric properties inherent to objects in 3D point clouds that have been largely unexploited in assessing the confidence of a detection. In the image case, it is usually not possible to easily derive geometric properties for the objects, such as height or orientation, unless a static and fully calibrated camera and a known or constant scene are given. In contrast, in 3D object detection, many geometric features are directly available in the object properties and associated 3D data points to better assess the presence of a real object. On the one hand, these are **instance-specific properties**, such as the dimension of the object, the heading direction, the position, or the point distribution within the bounding box. For example, as shown in Figure 2, the precision of a SECOND [38] model for detecting a vehicle is highly dependent on the size of the object and from which side it is detected by the LiDAR. It can be seen that vehicle categories between passenger cars and heavy duty vehicles (*i.e.* vehicles with a length of 6 to 13m) are harder to detect, potentially because they are underrepresented in the dataset, and that vehicles are easier to detect from behind (*i.e.* viewing angle between ± 45 degrees), presumably because of the highly reflective license plates, as shown in [25]. On the other hand, **contextual properties**, *i.e.* geometric relationships to neighboring objects, can contribute to a more reliable estimation of the confidence. For example, as shown in Figure 1, a pedestrian that appears atypical due to occlusions can be assessed more reliably by additionally taking neighboring vehicles and pedestrians into consideration.

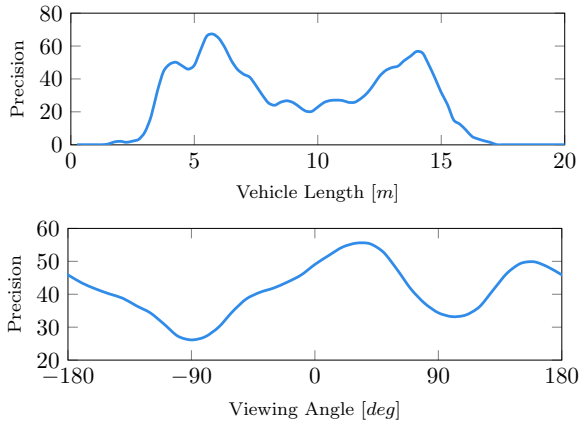

Figure 2. Precision of a SECOND [38] model for the Waymo [33] vehicle class as a function of the object length (top) and of the viewing angle (bottom), indicating from which side the vehicle is seen by the LiDAR, where 0 degrees corresponds to the rear view. These examples illustrate the strong dependence of the precision on simple geometric object properties.

Nevertheless, these simple but highly informative metric properties are neglected when estimating the confidence score in current detector architectures for the following reasons: First, in grid-based models, information such as the exact point distribution within the bounding box or the number of points is already partially discarded by the discretization (voxelization) in the preprocessing phase. Second, the bounding box properties (object dimensions & rotation) are determined in the parallel and separate box regression head and are therefore not accessible to the confidence estimation head. Finally, confidence estimation is usually performed using only features within a small area around the object (depending on the receptive field and detector), and no explicit information about neighboring objects and their geometric properties or confidence values is used, preventing a holistic estimation.

Inspired by these observations, we present *GACE*, an intuitive and highly effective method to improve the confidence estimation of any black-box detector using geometric information. Given a set of detections from the base detector, we explicitly use these neglected features to enhance the expressiveness of the confidence values with the help of these additional cues. Our model-agnostic approach is intentionally applied after the actual detector training process to perform an auxiliary geometric assessment independent of the initial features. In a detailed evaluation on the Waymo dataset, we show that GACE consistently improves the performance of several state-of-the-art detection pipelines. Furthermore, we demonstrate the generalizability and transferability of our method by applying it to other datasets and even other detectors. Without retraining them, we achieve highly compelling performance gains.

2. Related Work

LiDAR-based 3D Object Detection: Depending on how existing methods for 3D object detection in single frame LiDAR data deal with the unstructured nature of point clouds, they can be broadly categorized into point-based, grid-based, and hybrid approaches.

Point-based methods extract information directly from the individual raw 3D points [23,24,30,32,40,44]. The pioneering works, PointNet [23] and PointNet++ [24], used shared multilayer perceptrons in combination with global pooling functions to directly extract features from the irregular point cloud data. In Point-RCNN [30] features extracted in this manner are used to segment foreground points and generate proposals based on them. The advantages of point-based 3D object detectors are that there is no loss of point information due to discretization and the large receptive field, but at the cost of high computational demands.

Instead of processing the points directly, **grid-based methods** [4, 15, 21, 27, 34, 38, 39, 43, 46] discretize the non-uniform 3D points into regular grids that can then be processed with 2D/3D convolutions. Voxelnet [46], a pioneering method, divides the point cloud into uniformly spaced 3D voxels, aggregates information from the points within them and generates predictions using 3D convolutions. To better handle the large number of empty voxels, SECOND [38] introduced sparse 3D convolutions. To reduce complexity, PointPillars [15] and PillarNet [27] use a 2D grid on the ground plane to create a column representation, a single pillar-shaped voxel per location, that can be processed using 2D convolutions. As an alternative to such anchor-based methods, Centerpoint [43] predicts a bird's-eye view heat map and detects the object center using a keypoint detector. Recently, transformer-based backbones have also been used to enable long-range relationships between voxels [5,21,34,47]. Object relations within a frame and across multiple frames are mapped by Ret3D [37] using a graph and a transformer. The advantage of grid-based methods is that they can process data faster due to the regular format, but are limited by the loss of point information during the initial discretization phase.

In order to obtain both, multi-scale features and fine-grained information, **hybrid methods** process voxel and point information jointly [22, 26, 28, 29, 31, 41, 42]. PV-RCNN [28], uses a set abstraction module that combines surrounding point and voxel features at keypoints to improve the detections. Part-A^2 [31] predicts the position of parts within an object based on point features to improve the accuracy, while LiDAR R-CNN [17] uses features of a PointNet [23] model that processes points within and around box proposals. Pyramid R-CNN [19] creates point features using a pyramid grid structure to acquire fine-grained and long-range contextual information.

Confidence Estimation: In the usual *backbone - neck - head* detector architecture, after the feature extraction and aggregation, the box regression and confidence estimation are performed. This is usually done in two separate branches based on a common dense feature representation [15, 38, 40, 46]. This has the disadvantage that the accuracy of the localization is hardly included in the confidence score. Inspired by 2D object detection methods [11,12,36], IoU guided supervision is frequently used to obtain a better correlation between the classification result and the localization accuracy [9, 10, 16, 27, 28, 31, 41, 45]. Thereby, the IoU between the predicted box and the ground truth is learned in a third branch during training and then incorporated into the final confidence score at test time. Hu *et al*. [9] leverage the inherent relationship between object distance and point density to better assess a detection. Inspired by the 2D approach [3], a spatial transformation on the feature maps is done by He *et al*. [8] to better align the confidence prediction and bounding box regression. Related to confidence estimation are also calibration methods [7, 14] where the score should represent a true probability, *i.e*. how likely a detection is. Detection pipelines, however, aim at the best separation of true and false positives as optimization goal for the confidence prediction.

In our confidence estimation method we also pursue this optimization objective, but in contrast to existing approaches, we present a method for refining the confidence values for a given set of detections by exploiting the rich geometric information contained directly in the detections as well as in the underlying 3D points. While these useful cues for assessing the plausibility of a detection have been largely untapped due to the architecture of common detection pipelines, they allow us to increase the expressiveness of the confidence values.

3. Geometry-Aware Confidence Enhancement

Our goal is to optimize the confidence scores for a given set of detections from a base detector in order to better separate true positives from false positives, thus increasing the overall detection performance. We use the detector in a pure black box manner, *i.e*. we do not assume any knowledge about the architecture of the base model, nor access to its internals like parameters, features or gradients. This black-box optimization, taking only the point cloud and the set of objects detected in it as input, enables universal applicability and easy transferability of our enhancement module to any base 3D object detection pipeline. The basic idea is to revalidate the detections by exploiting as much as possible the geometric information they inherently contain.

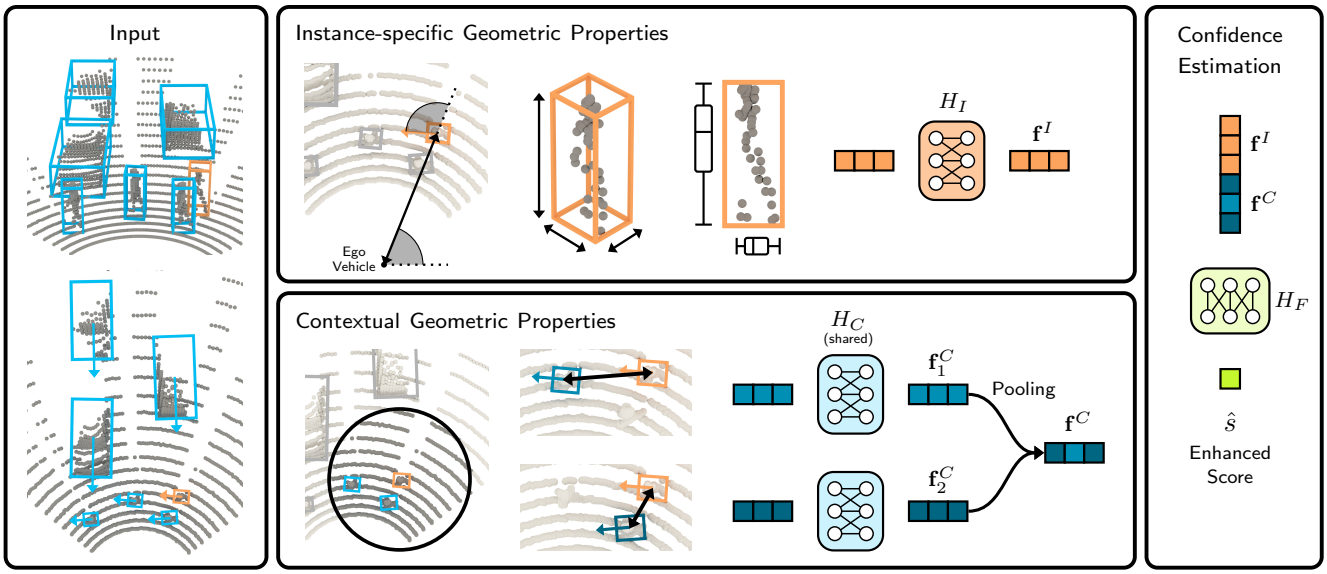

Figure 3. Schematic of GACE: To re-evaluate the confidence score of a detection (orange), we aggregate geometric properties of the detection itself and the points it contains into a feature vector (top). To capture the context of the detection, geometric relationships to neighboring detections are aggregated using a shared MLP and subsequent pooling function (bottom). By merging both features (right), we obtain a new confidence score that takes into account the underlying geometric properties of the detection.

In our proposed approach called *GACE* (**G**eometry **A**ware **C**onfidence **E**nhancement) we exploit two types of geometric information, as shown in Figure 3:

- **Instance-specific Geometric Properties (Section 3.1):** Attributes of the bounding box itself, combined with the point data inside. For example, how well is the size of the object or the heading angle supported by the point distribution?
- **Contextual Geometric Properties (Section 3.2):** Relationships to surrounding objects can provide useful information to better validate an uncertain detection, *e.g.*, a partially occluded vehicle moving in the same lane and same direction as the surrounding vehicles.

These useful cues to estimate the plausibility of an object are usually not used for the confidence estimation in common detection pipelines. The reasons are that information is often already discarded during preprocessing (discretization), essential properties such as the estimated size of the object are not available (separate box regression in a parallel branch), or the objects are only evaluated individually and not more holistically. Therefore, we explicitly use these easily accessible and rich sources of information as input to our enhancement module. After merging the instance-specific and contextual features, we determine the new confidence value of each proposed object via an auxiliary task (Section 3.3).

To generate the training data for our enhancement module, we use the black-box base model in a single inference run on the training set. The resulting set of all detections from the base model represents the training data to learn our improved confidence estimator. Formally, this can be described as follows.

Definitions & Notations: Let $\widetilde{\mathcal{X}} = \{\widetilde{\mathbf{x}}_k\}_{k=1\dots K}$ be a LiDAR point cloud, where each of the K unordered points $\widetilde{\mathbf{x}}_k \in \mathbb{R}^5$ consists of the 3D coordinates, intensity/reflectance and the elongation value. Furthermore, let $\widetilde{\mathcal{Y}} = \{\widetilde{\mathbf{y}}_i\}_{j=1\dots M}$ be the set of corresponding ground truth objects. Each object annotation $\widetilde{\mathbf{y}} = [\widetilde{\mathbf{b}}, \widetilde{y}]$ includes the bounding box parameters $\widetilde{\mathbf{b}} = [\widetilde{c}_x, \widetilde{c}_y, \widetilde{c}_z, \widetilde{d}_x, \widetilde{d}_y, \widetilde{d}_z, \widetilde{\Theta}]$ and the corresponding class label $\widetilde{y}$. For a given black-box 3D object detector F, let $F(\widetilde{\mathcal{X}}) = \mathcal{Y} = \{\mathbf{y}_{i=1\dots N}\}$ be the set of proposed detections for this input point cloud, where $\mathbf{y} = [\mathbf{b}, y, s]$. In addition to the box properties $\mathbf{b}$ and the corresponding class label y, the detector predicts a confidence value s that should ideally indicate the prevalence of a true positive example.

Confidence Optimization: Based on the known ground truth objects, a category label $\{u_i \in \{0,1\}\}_{i=1\dots N}$ can be assigned to each detection, indicating whether it is a true positive or false positive detection. Moreover, we know the IoU with a possible ground truth bounding box for each object which we define as $\{v_i\}_{i=1\dots N}$. We aim to improve the confidence estimate of the original detector by focusing exclusively on the binary classification of detections into true positive and false positive examples. We determine the revised confidence score $\{\widehat{s}_i\} = H(\mathcal{Y}, \mathcal{X})$ using our module H, where only the set of detections $\mathcal{Y}$ and the points they include $\mathcal{X} \subset \widetilde{\mathcal{X}}$ are used as input.

3.1. Instance-specific Geometric Properties

In common 2D and 3D object detection architectures, the bounding box regression and the confidence estimation are performed completely separated in different branches. This is well suited for 2D where it is desired to detect objects in the image regardless of their scale. For 3D, however, important cues for plausibility estimation remain largely unused. Let us consider the available context directly provided by an object proposal: The position of the possible object in combination with the dimensions indicates, for example, which point density is to be expected. Furthermore, the direction angle provides an indication of the expected point distribution within the box. This information can even be further refined by knowing the class of the object. This directly available geometrical knowledge about an object allows for a more profound estimation of the confidence score.

We extract these basic properties and transform them into a compact representation using a multilayer perceptron (MLP) H_I. As input parameters, we first use the object parameters, *i.e.* the position (c_x, c_y, c_z), the size (d_x, d_y, d_z), and the heading angle Θ of the bounding box, as well as the initial confidence value s_i of the detection estimated by the base detector. Additionally, we use the distance $\|\mathbf{c}\|$ from the LiDAR sensor to the object center, and the angle α between the line of sight to the object and the heading angle of the object:

$$\alpha = \Theta - \text{atan2}(c_y, c_x). \tag{1}$$

This angle describes from which side an object is seen from the LiDAR center, independent of the position of the object relative to the LiDAR, *e.g.* a vehicle driving directly towards the LiDAR always has the same angle α, no matter from which direction the vehicle approaches. We complement these cues with information about the points $\mathcal{X}_\mathbf{b} \subset \mathcal{X}$ inside the bounding box $\mathbf{b}$. Besides the overall number of points $|\mathcal{X}_\mathbf{b}|$, we extract elementary low-level statistics. Therefore, we scale the box and the associated points to a uniform size and then align them w.r.t. their center and yaw angle. In this canonical representation we compute the mean, standard deviation, minimum and maximum of $\mathcal{X}_\mathbf{b}$ for all axes, denoted as $\mathcal{X}_\mathbf{b}^{\text{mean}}, \mathcal{X}_\mathbf{b}^{\text{std}}, \mathcal{X}_\mathbf{b}^{\text{min}}, \mathcal{X}_\mathbf{b}^{\text{max}}$. We then aggregate these attributes into one feature vector representing the instance-specific plausibility per object as

$$\mathbf{f}^I = H_I\left(\left[\mathbf{b}, \alpha, \|\mathbf{c}\|, |\mathcal{X}_\mathbf{b}|, \mathcal{X}_\mathbf{b}^{\text{mean}}, \mathcal{X}_\mathbf{b}^{\text{std}}, \mathcal{X}_\mathbf{b}^{\text{min}}, \mathcal{X}_\mathbf{b}^{\text{max}}\right]^\top\right), \tag{2}$$

where we pass all angles as direction vectors $(\cos(\cdot), \sin(\cdot))$ and normalize all metric properties to unit length using the corresponding maximum value range.

3.2. Contextual Geometric Properties

Especially in the case of uncertain detections, *e.g.* detections that are far away and therefore consist of only a few 3D points, or detections that are partially occluded and therefore appear atypical, geometric contextual information can be very useful in assessing a confidence score. Examples include a vehicle that is heavily occluded but in a convoy with other vehicles, or cyclists moving in a group. However, this information is usually not available for the confidence estimation, since the object proposals are evaluated individually within the receptive field but without explicit knowledge of the objects detected in its vicinity and their properties. In order to assess the plausibility of a detection $\mathbf{y}$ in a more holistic way, we use the geometric relations to surrounding objects in the scene. We thereby consider all neighbors that are within a certain radius r around the object to be evaluated.

We therefore create a representation per object, which captures the relationships to its neighboring objects. In order to be independent of the number of neighboring objects, we first create a feature vector for each neighbor, which we then combine into a unified representation using a symmetric pooling function. As input parameters per neighbor we use the distance to the object to be evaluated $\|\mathbf{c} - \mathbf{c}_n\|$, the direction vector $\mathbf{c} - \mathbf{c}_n$, the difference between the two heading angles $\Theta - \Theta_n$, and the neighbor's class label y_n. To incorporate the validity of the neighbor, we leverage $\mathbf{f}_n^I$, which represents the instance-specific properties of the neighbor. This information is encoded via the shared weight MLP H_C to form the feature vector $\mathbf{f}_n^C$ for each individual neighbor,

$$\mathbf{f}_n^C = H_C\left(\left[\|\mathbf{c} - \mathbf{c}_n\|, \mathbf{c} - \mathbf{c}_n, \Theta - \Theta_n, y_n, \mathbf{f}_n^I\right]^\top\right), \tag{3}$$

where the angle $\Theta - \Theta_n$ is provided as direction vector and the metric features are normalized to unit length.

Finally, we aggregate the information from the individual neighbors into a unified representation for later processing, as shown in Figure 3. This requires accumulating features of an unknown (and varying) number of neighbors. To impose no constraint on the number of features, we take inspiration from the pooling step of point-based detectors, *e.g.* [23], and employ a symmetric max pooling function to form a plausibility signature $\mathbf{f}^C$ over the feature vectors of all neighbors. Thus, in this representation the geometric relations to the surrounding objects are accumulated for the subsequent confidence estimation, taking into account the respective local properties of the neighbors.

3.3. Data Fusion & Confidence Prediction

To estimate the new confidence value for a detection, we merge the instance-specific and contextual geometric features. The instance-specific information encoded in $\mathbf{f}^I$, as well as the contextual information in $\mathbf{f}^C$ are concatenated and processed using H_F, a MLP with sigmoid output function, to estimate the new confidence score

$$\widehat{s} = H_F\left(\left[\mathbf{f}^I, \mathbf{f}^C\right]^\top\right). \tag{4}$$

We train the whole module including the two feature encoding networks H_I and H_C using an end-to-end training strategy. As loss function $\mathcal{L}_{\text{conf}}(\widehat{s}, u)$ for this task we use the focal loss [18] to focus learning on hard examples. The goal is to divide the set of detections as good as possible into true or false positive examples using the binary category label $u \in \{0, 1\}$ as target. As auxiliary task during training we estimate the IoU with the ground truth bounding box in a further output $\widehat{v}$ of H_F. Therefore, we add an IoU-guidance [16, 41, 45] L1-loss term $\mathcal{L}_{\text{IoU}}(\widehat{v}, v)$. This increases the importance of the point distribution statistics within the features, as they provide evidence of the bounding box accuracy from which the confidence estimate also benefits. The overall loss function is therefore

$$\mathcal{L} = \mathcal{L}_{\text{conf}}(\widehat{s}, u) \; + \; \lambda_{\text{IoU}} \; \mathcal{L}_{\text{IoU}}(\widehat{v}, v), \tag{5}$$

where λ_{IoU} is a hyper-parameter to adjust the influence of the auxiliary task.

4. Experiments

To demonstrate the benefits of GACE, we evaluate our black-box confidence optimization method on several well-known state-of-the-art 3D object detection pipelines. In particular, we apply it to the pillar-based PointPillars [15], the voxel-based SECOND [38] and Centerpoint [43], as well as the hybrid methods Part-A^2 [31], PV-RCNN [28] and PV-RCNN++ [29].

Datasets: For the evaluation of our approach we use the Waymo Open Dataset [33] as well as the KITTI Dataset [6]. The Waymo dataset is one of the largest and most challenging public datasets for autonomous driving research. It provides 798 training scenes and 202 validation scenes, where each scene consists of about 200 LiDAR samples covering the full 360° field-of-view. In total, the dataset consists of over 8.9M annotated objects classified into vehicles, pedestrians and cyclists. We follow the common evaluation protocol using the standard metrics average precision (AP), as well as average precision with heading (APH), where true positives are weighted by their heading accuracy. In addition, we use the KITTI dataset, one of the most popular datasets for 3D object detection. We thereby use the standard split [2] into a training set (3712 samples) and validation set (3769 samples). We also follow the common evaluation practice using average precision with 40 recall points. For both datasets, the 3D IoU threshold for a true positive sample is 0.7 for vehicles and 0.5 for pedestrians and cyclists.

Implementation Details: Our experimental setup[1] is based on the open-source toolbox OpenPCDet [35]. To ensure the reproducibility of our results, all base models used in the experiments have been trained on the training set using the default configuration and default training policies of OpenPCDet. This includes augmenting the data by randomly rotating, scaling and flipping the point cloud, as well as ground truth sampling [38], *i.e.* the placement of objects from other training examples in the current frame. To create the training data for our module, we use the base model as a black-box detection pipeline. In a single inference run on the training set, we collect the output of the base model, *i.e.* the set of all detections, which represents our training data. The actual training process of our module is therefore entirely decoupled from the base model.

Our sub-networks for feature transformation, *i.e.* H_I and the shared network H_C, as well as the confidence prediction network H_F are two-layer MLP's with 256 feature dimensions. The feature vector $\mathbf{f}^I$ in which the instance-specific information is encoded is 128-dimensional and $\mathbf{f}^C$ for the contextual information is 64-dimensional. When determining the contextual geometric properties, we accumulate the neighboring objects that are within a radius of $r = 40m$.

We train our model end to end with the Adam [13] optimizer and a learning rate of 0.001. The weighting of the auxiliary loss term during training is set to $\lambda_{\text{IoU}} = 0.5$. In all experiments we use the same architecture as well as hyperparameters, regardless of the underlying black-box base model. We train our module over 5 epochs on a single NVIDIA® GeForce® RTX 3090 GPU, which takes less than 10 minutes due to our favorable lightweight model size and low feature dimensions.

4.1. Main Results

Table 1 summarizes the results of our confidence optimization on the Waymo dataset for different baseline detection pipelines. For each detector architecture, we report the performance of the base detector as well as the results after applying our GACE module. Please note that for better traceability, the results of the base detectors correspond to the respective OpenPCDet implementations (see OpenPCDet modelzoo). Some reported baseline scores are even higher than in their original papers due to augmentation techniques used in OpenPCDet.

[1] https://github.com/dschinagl/gace

Method	Overall				Vehicles (IoU=0.7)				Pedestrians (IoU=0.5)				Cyclists (IoU=0.5)			
	LEVEL_1		LEVEL_2		LEVEL_1		LEVEL_2		LEVEL_1		LEVEL_2		LEVEL_1		LEVEL_2	
	mAP	mAPH	mAP	mAPH	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH
PointPillars [15]	64.72	57.07	58.57	51.73	70.99	70.35	62.79	62.20	66.36	47.15	58.27	41.32	56.81	53.71	54.66	51.67
+ GACE (Ours)	69.25	61.24	62.98	55.73	71.92	71.28	63.63	63.04	72.18	51.97	64.06	45.96	63.64	60.47	61.25	58.20
Improvement	**+4.53**	**+4.17**	**+4.41**	**+4.00**	**+0.93**	**+0.93**	**+0.84**	**+0.84**	**+5.82**	**+4.82**	**+5.79**	**+4.64**	**+6.83**	**+6.76**	**+6.59**	**+6.53**
SECOND [38]	65.13	60.81	59.01	55.12	70.93	70.30	62.65	62.07	65.67	54.96	57.78	48.25	58.78	57.18	56.59	55.05
+ GACE (Ours)	70.17	66.13	63.74	60.06	71.56	70.92	63.22	62.63	71.71	61.87	63.27	54.37	67.22	65.59	64.73	63.16
Improvement	**+5.04**	**+5.32**	**+4.73**	**+4.94**	**+0.63**	**+0.62**	**+0.57**	**+0.56**	**+6.04**	**+6.91**	**+5.49**	**+6.12**	**+8.44**	**+8.41**	**+8.14**	**+8.11**
Part-A^2 [31]	70.30	66.66	63.53	60.27	73.35	72.81	64.73	64.24	70.02	61.01	60.83	52.85	67.53	66.18	65.03	63.73
+ GACE (Ours)	73.07	69.21	66.24	62.77	73.99	73.43	65.38	64.87	72.36	62.93	63.21	54.81	72.84	71.28	70.13	68.63
Improvement	**+2.77**	**+2.55**	**+2.71**	**+2.50**	**+0.64**	**+0.62**	**+0.65**	**+0.63**	**+2.34**	**+1.92**	**+2.38**	**+1.96**	**+5.31**	**+5.10**	**+5.10**	**+4.90**
Centerpoint [43]	73.01	70.35	66.79	64.33	72.87	72.33	64.76	64.27	74.48	68.22	66.55	60.81	71.69	70.50	69.06	67.92
+ GACE (Ours)	75.58	72.98	69.19	66.76	74.49	73.99	66.19	65.73	78.62	72.48	70.44	64.73	73.64	72.48	70.94	69.83
Improvement	**+2.57**	**+2.63**	**+2.40**	**+2.43**	**+1.62**	**+1.66**	**+1.43**	**+1.46**	**+4.14**	**+4.26**	**+3.89**	**+3.92**	**+1.95**	**+1.98**	**+1.88**	**+1.91**
PV-RCNN [28]	71.11	66.78	64.40	60.50	74.79	74.17	66.17	65.61	72.06	61.46	63.00	53.56	66.48	64.72	64.03	62.34
+ GACE (Ours)	73.38	68.89	66.65	62.57	75.20	74.55	66.57	65.97	73.84	62.97	64.88	55.14	71.12	69.16	68.50	66.60
Improvement	**+2.27**	**+2.11**	**+2.25**	**+2.07**	**+0.41**	**+0.38**	**+0.40**	**+0.36**	**+1.78**	**+1.51**	**+1.88**	**+1.58**	**+4.64**	**+4.44**	**+4.47**	**+4.26**
PV-RCNN++ [29]	75.72	73.05	69.22	66.73	77.30	76.81	68.92	68.47	78.91	72.42	70.43	64.41	70.95	69.90	68.31	67.31
+ GACE (Ours)	76.76	74.02	70.31	67.75	77.42	76.93	69.05	68.59	79.59	72.97	71.34	65.18	73.26	72.16	70.54	69.48
Improvement	**+1.04**	**+0.97**	**+1.09**	**+1.02**	**+0.12**	**+0.12**	**+0.13**	**+0.12**	**+0.68**	**+0.55**	**+0.91**	**+0.77**	**+2.31**	**+2.26**	**+2.23**	**+2.17**

Table 1. Performance comparison on the Waymo Open Dataset [33] validation set for different baseline methods with and without our confidence enhancement module GACE.

For all baseline detectors and all classes, adjusting the confidence level with our approach leads to an increase in performance without exception. The overall performance gains range from +1.02mAPH (LEVEL_2) for PV-RCNN++ to +4.94mAPH (LEVEL_2) for SECOND, demonstrating the significance of this geometric information in estimating a confidence score for a detection.

Object Classes: The performance improvements are significantly higher for the classes of pedestrians and cyclists. Intuitively, objects of these vulnerable road users contain significantly fewer 3D points due to their smaller spatial extent compared to vehicles. This effect is even aggravated by possible occlusions. Furthermore, the class of cyclists in particular is underrepresented in the training data. These properties make objects of these classes more difficult to detect, which leads to a higher number of false positives, but also makes confidence estimation more complex. Especially in these cases, our confidence enhancement method benefits from the additional geometric information when separating false positives from true positives, resulting in a higher impact of GACE.

Base Detectors: The overall performance gain is highest when applying GACE on pillar-based (PointPillars [15]) and voxel-based methods (SECOND [38] and Centerpoint [43]). These methods lose valuable point information already in their preprocessing stage, *i.e.* by voxelization. Thus, the optimization potential for these methods is higher than for the hybrid methods, which explicitly incorporate point-level information at detection locations. However, by explicitly exploiting both instance-specific and contextual geometric properties of the detections, GACE also leads to a significant performance improvement for these methods, *i.e.* Part-A^2 [31], PV-RCNN [28] and PV-RCNN++ [29], most notably for the classes of pedestrians and cyclists.

Precision/Recall Plots: To better illustrate the impact of our method, Figure 4 shows the precision/recall plots before and after applying GACE with a SECOND [38] model as the base detector (See the supplemental material for the other base detectors). Since recall remains unaffected, the significant performance gains of GACE are entirely due to an increase in precision. The better separation of true and false positives is mainly seen in the more challenging classes of pedestrians and cyclists, where the number of false detections is significantly higher and confidence estimation is more difficult. Furthermore, we see that the precision gains are higher in regions of higher recall, *i.e.* for detections with a lower confidence score. Especially these initially underrated detections of objects of the vulnerable classes represent a safety risk that can be reduced by GACE.

Range-based Evaluation: Table 2 summarizes our evaluation across the different distance ranges of Waymo. This also shows that GACE can consistently improve the performance of all baseline detectors and in all sub-ranges, especially in the far range. Detections at long distances are more challenging due to the lower point density, making it harder to distinguish true positives from false positives. Therefore, the geometric information is even more valuable in these cases, *e.g.* from +2.93mAPH (LEVEL_2) for PV-RCNN++ up to +4.97mAPH (LEVEL_2) for PointPillars. The detailed evaluation results for all subclasses can be found in the supplemental material.

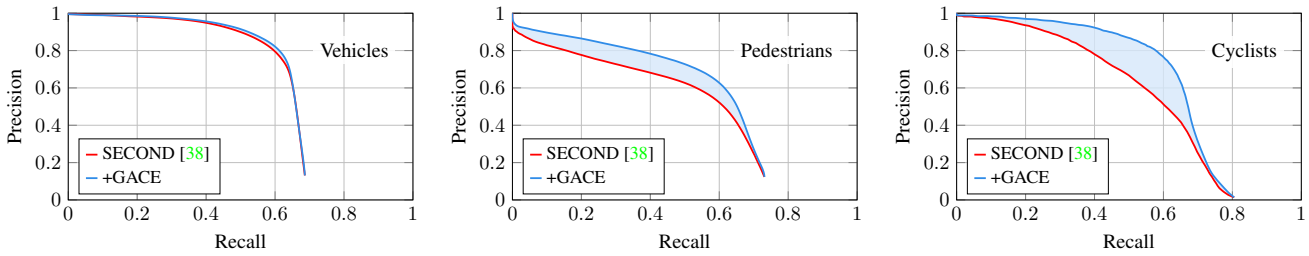

Figure 4. Precision-recall plots for a SECOND [38] model as base detector with and without our confidence enhancement module on the Waymo Open Dataset [33] validation set for LEVEL_2 APH. Especially at higher recall levels, GACE's better separation of true and false positives leads to higher precision.

Method	Overall (LEVEL_2) 0-30m mAP / mAPH	30-50m mAP / mAPH	50m-Inf mAP / mAPH
PointPillars [15]	75.10 / 67.59	55.31 / 48.53	37.18 / 31.00
+ GACE (Ours)	78.32 / 70.68	60.17 / 53.01	42.88 / 35.97
Improvement	**+3.22 / +3.09**	**+4.86 / +4.48**	**+5.70 / +4.97**
SECOND [38]	75.55 / 71.87	56.55 / 52.42	37.73 / 33.39
+ GACE (Ours)	78.64 / 75.39	60.10 / 56.13	41.69 / 37.39
Improvement	**+3.09 / +3.52**	**+3.55 / +3.71**	**+3.96 / +4.00**
Part-A^2 [31]	79.93 / 76.93	61.77 / 58.01	40.13 / 36.44
+ GACE (Ours)	81.62 / 78.52	64.63 / 60.59	45.06 / 40.94
Improvement	**+1.69 / +1.59**	**+2.86 / +2.58**	**+4.93 / +4.50**
Centerpoint [43]	81.29 / 79.12	65.45 / 62.66	46.45 / 43.27
+ GACE (Ours)	82.94 / 80.90	67.56 / 64.76	50.20 / 46.80
Improvement	**+1.65 / +1.78**	**+2.11 / +2.10**	**+3.75 / +3.53**
PV-RCNN [28]	79.77 / 76.24	62.57 / 58.10	42.51 / 37.77
+ GACE (Ours)	81.25 / 77.66	64.76 / 60.11	47.01 / 41.90
Improvement	**+1.48 / +1.42**	**+2.19 / +2.01**	**+4.50 / +4.13**
PV-RCNN++ [29]	83.08 / 80.98	67.86 / 65.04	48.69 / 45.26
+ GACE (Ours)	83.59 / 81.43	68.69 / 65.83	51.81 / 48.19
Improvement	**+0.51 / +0.45**	**+0.83 / +0.79**	**+3.12 / +2.93**

Table 2. Consistent performance gains on the Waymo Open Dataset [33] validation set across distance ranges for different baseline methods for LEVEL_2 over all classes.

Baseline	Instance Specific	Contextual	Aux-Loss $\mathcal{L}_{\text{IoU}}$	LEVEL_2 mAP	LEVEL_2 mAPH
✓				59.01	55.12
✓	✓			62.66	58.44
✓		✓		60.33	56.90
✓	✓	✓		63.37	59.58
✓	✓		✓	62.94	58.78
✓		✓	✓	60.19	56.77
✓	✓	✓	✓	63.74	60.06

Table 3. Ablation experiments for a SECOND [38] model as base detector over all classes on the Waymo Open Dataset [33] validation set. We show the impact of each GACE component.

Box Properties	# Points	Viewing Angle	Point Stat.
$56.98^{+1.86}$	$55.96^{+0.84}$	$55.86^{+0.74}$	$58.15^{+3.03}$

Table 4. Impact of the different instance-specific feature groups when used exclusively for a SECOND model as base detector on the Waymo Open Dataset [33] over all classes (LEVEL_2 mAPH).

4.2. Ablation Studies

Main Components: We evaluate the contribution of our proposed instance-specific and contextual properties as well as the influence of the auxilary IoU loss for a SECOND [38] model as baseline in Table 3. Incorporating the instance-specific geometric properties already leads to a significant performance increase of +3.65AP/+3.32APH. Including also the contextual information, *i.e.* incorporating the relationships to the neighboring detection hypotheses further increases the performance by another +0.71AP/+1.14APH. Compared to the contributions of these two components, there is only a minor impact of incorporating the IoU guidance, leading to additional +0.37AP/+0.48APH. Note that the degradation when adding $\mathcal{L}_{\text{IoU}}$ to only contextual features is caused by the lack of point information, which does not allow a reasonable estimation of the IoU. Overall, the instance-specific geometric information contributes stronger than the contextual geometric information.

Instance-Specific Properties: We analyze the contribution of each feature group within the instance-specific properties, namely *box properties* $(\mathbf{b}, \|\mathbf{c}\|)$, *number of points* $(|\mathcal{X}_\mathbf{b}|)$, *viewing angle* (α), and *point statistics* $(\mathcal{X}_\mathbf{b}^{\text{mean}}, \mathcal{X}_\mathbf{b}^{\text{std}}, \mathcal{X}_\mathbf{b}^{\text{min}}, \mathcal{X}_\mathbf{b}^{\text{max}})$. Table 4 shows the impact of each group on the overall performance when used exclusively, indicating a high contribution of box properties and point statistics. A complete list of the combinations can be found in the supplemental material.

Contextual Properties Radius: The dependence of the performance on the chosen context radius r is shown in Figure 5. It can be seen that the performance increases significantly up to $\sim 15m$, followed by a slight degradation starting at $\sim 40m$, which illustrates the importance of neighboring objects in the near and middle ranges.

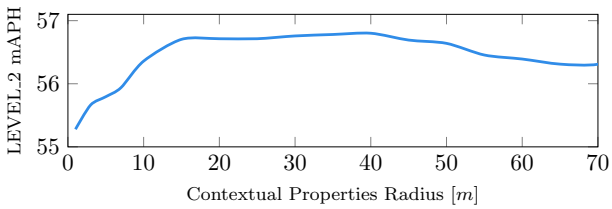

Figure 5. Performance of purely context-based GACE for different radius r (SECOND base model / Waymo overall).

Method	Car	Pedestrian	Cyclist
SECOND (KITTI)	81.61	51.14	66.74
+GACE (SECOND Waymo)	82.04 **+0.43**	57.11 **+5.97**	70.99 **+4.25**
PointPillars (KITTI)	78.39	51.41	62.94
+GACE (SECOND Waymo)	78.51 **+0.12**	55.30 **+3.89**	67.94 **+5.00**
Part-A^2 (KITTI)	82.92	59.73	70.10
+GACE (SECOND Waymo)	82.94 **+0.02**	64.21 **+4.48**	72.16 **+2.06**
PV-RCNN (KITTI)	82.86	53.64	70.42
+GACE (SECOND Waymo)	82.84 **-0.02**	61.06 **+7.42**	72.70 **+2.28**

Table 5. Model Transfer: Applying a GACE module trained on SECOND detections on Waymo (without LiDAR elongation, which is not available on KITTI) to different detectors on the KITTI dataset (moderate difficulty / @R40).

4.3. Generalization and Transferability

Since we use only the detection attributes and corresponding properties of the underlying point cloud as input, a GACE module trained on base detector A can be directly applied to another base detector B. Furthermore, since all metric features are normalized to their maximum possible value and the statistical parameters (point distribution) are computed from a normalized unit-length box, GACE can be applied not only to a different detector, but also directly to a different detector on a different dataset. Therefore, to demonstrate the general applicability of GACE, we freeze a GACE module trained on detections from a SECOND detector on Waymo, and apply it directly to detections of a SECOND detector on the KITTI dataset [6]. As shown in Table 5, first row, this also leads to considerable performance improvements despite the distribution shift to a different dataset (different LiDAR sensor, different country). Even more remarkably, significant performance gains are also obtained when the same GACE module is applied even to a different base detector, *e.g.* +7.42AP for PV-RCNN on pedestrians while achieving the same performance for cars. The results demonstrate the excellent generalization capability of GACE and the general validity of the geometric information regardless of the dataset.

4.4. Runtime Analysis

In Table 6 we show the runtime analysis of GACE for a 360° field-of-view Waymo point cloud using a single NVIDIA® GeForce® RTX 3090 GPU. The computationally most intensive part is the feature extraction, especially extracting the statistical features for all detections. However, this can be solved efficiently via PyTorch `einsum`. During inference, we first compute the instance-specific plausibility $\mathbf{f}^I$ for each detection via H_I, which is then used as input to H_C for the corresponding neighbors. Overall, GACE is capable of processing ~ 490 Waymo point clouds per second with ~ 100 detections each.

Feature Extraction	Points in Boxes Query	0.56 ms
	Geometric & Statistical Features	0.98 ms
	Neighboring Object ID's Query	0.07 ms
Network Inference	H_I (Instance-specific)	0.14 ms
	H_C (Contextual)	0.17 ms
	H_F (Confidence Estimation)	0.12 ms
Overall		**2.04 ms**

Table 6. GACE runtime analysis per 360° field-of-view Waymo point cloud on a single NVIDIA® GeForce® RTX 3090 GPU.

5. Limitations

While our method has proven effective in improving object detection performance for vulnerable classes such as pedestrians and cyclists, it is worth noting that it may not offer such significant benefits for simpler classes such as vehicles. This is because all detectors typically have few false positives for these *easier-to-detect* objects. Therefore, there is less room for improvement for this class. However, it is important to consider the context in which our method is applied. Even if the performance improvement is not as large for simpler classes, the overall impact of our method on safety should not be underestimated. In many real-world scenarios, pedestrians and cyclists are particularly vulnerable, and any improvement in their detection can make a significant contribution to safety.

6. Conclusion

We proposed GACE, a method to better evaluate object hypotheses from black-box LiDAR-based 3D object detectors by explicitly assessing numerous geometric properties inherent in the detections. This enables a better separation between true and false positive detections and thus significantly improves the performance. In a comprehensive analysis, we were able to show the performance of our method for several state-of-the-art detectors and also demonstrate the generalisation capabilities of GACE. This underlines the importance and general validity of the geometric information inherent in 3D detections, which has been largely neglected in the past.

Acknowledgements This work was partially funded by the Austrian FFG project iLIDS4SAM (878713) and by the Christian Doppler Laboratory for Embedded Machine Learning.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proc. CVPR*, 2020. 1

[2] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. *Advances in neural information processing systems*, 28, 2015. 6

[3] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: Object Detection via Region-based Fully Convolutional Networks. In *NeurIPS*, 2016. 3

[4] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel R-CNN: Towards High Performance Voxel-based 3D Object Detection. In *Proc. AAAI*, 2021. 3

[5] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing Single Stride 3D Object Detector with Sparse Transformer. In *Proc. CVPR*, 2022. 3

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proc. CVPR*, 2012. 1, 6, 9

[7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On Calibration of Modern Neural Networks. In *Proc. ICML*, 2017. 3

[8] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure Aware Single-stage 3D Object Detection from Point Cloud. In *Proc. CVPR*, 2020. 2, 3

[9] Jordan SK Hu, Tianshu Kuai, and Steven L Waslander. Point Density-Aware Voxels for LiDAR 3D Object Detection. In *Proc. CVPR*, 2022. 3

[10] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. AFDetV2: Rethinking the Necessity of the Second Stage for Object Detection from Point Clouds. In *Proc. AAAI*, 2022. 1, 2, 3

[11] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask Scoring R-CNN. In *Proc. CVPR*, 2019. 3

[12] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of Localization Confidence for Accurate Object Detection. In *Proc. ECCV*, 2018. 3

[13] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proc. ICLR*, 2015. 6

[14] Fabian Kuppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate Confidence Calibration for Object Detection. In *Proc. CVPR*, 2020. 3

[15] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection from Point Clouds. In *Proc. CVPR*, 2019. 2, 3, 6, 7, 8

[16] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. GS3D: An Efficient 3D Object Detection Framework for Autonomous Driving. In *Proc. CVPR*, 2019. 3, 6

[17] Zhichao Li, Feng Wang, and Naiyan Wang. LiDAR R-CNN: An Efficient and Universal 3D Object Detector. In *Proc. CVPR*, 2021. 3

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *Proc. ICCV*, 2017. 6

[19] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid R-CNN: Towards Better Performance and Adaptability for 3D Object Detection. In *Proc. ICCV*, 2021. 3

[20] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Chunjing Xu, et al. One Million Scenes for Autonomous Driving: ONCE Dataset. 2021. 1

[21] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel Transformer for 3D Object Detection. In *Proc. ICCV*, 2021. 3

[22] Jongyoun Noh, Sanghoon Lee, and Bumsub Ham. HVPR: Hybrid Voxel-Point Representation for Single-stage 3D Object Detection. In *Proc. CVPR*, 2021. 3

[23] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. CVPR*, 2017. 3, 5

[24] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, 2017. 3

[25] David Schinagl, Georg Krispel, Horst Possegger, Peter M Roth, and Horst Bischof. OccAM's Laser: Occlusion-Based Attribution Maps for 3D Object Detectors on LiDAR Data. In *Proc. CVPR*, 2022. 2

[26] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3D Object Detection with Channel-wise Transformer. In *Proc. ICCV*, 2021. 3

[27] Guangsheng Shi, Ruifeng Li, and Chao Ma. PillarNet: High-Performance Pillar-based 3D Object Detection. In *Proc. ECCV*, 2022. 2, 3

[28] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In *Proc. CVPR*, 2020. 3, 6, 7, 8

[29] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN++: Point-Voxel Feature Set Abstraction With Local Vector Representation for 3D Object Detection. *IJCV*, pages 1–21, 2022. 1, 3, 6, 7, 8

[30] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. In *Proc. CVPR*, 2019. 3

[31] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network. *PAMI*, pages 2647–2664, 2021. 3, 6, 7, 8

[32] Weijing Shi and Ragunathan Rajkumar. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud. In *Proc. CVPR*, 2020. 3

[33] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In *Proc. CVPR*, 2020. 1, 2, 6, 7, 8

[34] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. SWFormer: Sparse Window Transformer for 3D Object Detection in Point Clouds. In *Proc. ECCV*, 2022. 3

[35] OpenPCDet Development Team. OpenPCDet: An open-source toolbox for 3D object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 6

[36] Lachlan Tychsen-Smith and Lars Petersson. Improving Object Localization with Fitness NMS and Bounded IoU Loss. In *Proc. CVPR*, 2018. 3

[37] Yu-Huan Wu, Da Zhang, Le Zhang, Xin Zhan, Dengxin Dai, Yun Liu, and Ming-Ming Cheng. Ret3D: Rethinking Object Relations for Efficient 3D Object Detection in Driving Scenes. *arXiv:2208.08621*, 2022. 3

[38] Yan Yan, Yuxing Mao, and Bo Li. SECOND: Sparsely Embedded Convolutional Detection. *Sensors*, 18(10):3337, 2018. 1, 2, 3, 6, 7, 8

[39] Bin Yang, Wenjie Luo, and Raquel Urtasun. PIXOR: Real-time 3D Object Detection from Point Clouds. In *Proc. CVPR*, 2018. 3

[40] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3D Single Stage Object Detector. In *Proc. CVPR*, 2020. 1, 3

[41] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-Dense 3D Object Detector for Point Cloud. In *Proc. ICCV*, 2019. 3, 6

[42] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. HVNet: Hybrid Voxel Network for LiDAR Based 3D Object Detection. In *Proc. CVPR*, 2020. 3

[43] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3D Object Detection and Tracking. In *Proc. CVPR*, 2021. 3, 6, 7, 8

[44] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proc. CVPR*, 2022. 3

[45] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud. In *Proc. AAAI*, 2021. 1, 2, 3, 6

[46] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In *Proc. CVPR*, 2018. 3

[47] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer for 3d object detection. In *Proc. ECCV*, 2022. 1, 3