Are we Missing Confidence in Pseudo-LiDAR Methods for Monocular 3D Object Detection?

Andrea Simonelli[1] Samuel Rota Bulò[2] Lorenzo Porzi[2] Peter Kontschieder[2] Elisa Ricci[1]
[1]University of Trento, Fondazione Bruno Kessler [2]Facebook Reality Labs

Abstract

Pseudo-LiDAR-based methods for monocular 3D object detection have received considerable attention in the community due to the performance gains exhibited on the KITTI3D benchmark, in particular on the commonly reported validation split. This generated a distorted impression about the superiority of Pseudo-LiDAR-based (PL-based) approaches over methods working with RGB images only. Our first contribution consists in rectifying this view by pointing out and showing experimentally that the validation results published by PL-based methods are substantially biased. The source of the bias resides in an overlap between the KITTI3D object detection validation set and the training/validation sets used to train depth predictors feeding PL-based methods. Surprisingly, the bias remains also after geographically removing the overlap. This leaves the test set as the only reliable set for comparison, where published PL-based methods do not excel. Our second contribution brings PL-based methods back up in the ranking with the design of a novel deep architecture which introduces a 3D confidence prediction module. We show that 3D confidence estimation techniques derived from RGB-only 3D detection approaches can be successfully integrated into our framework and, more importantly, that improved performance can be obtained with a newly designed 3D confidence measure, leading to state-of-the-art performance on the KITTI3D benchmark.

1. Introduction

By providing information about pose, location and category of objects in the 3D space, 3D object detection constitutes an enabling technology for applications like autonomous driving or augmented reality. To obtain accurate localisation performance, existing solutions rely on depth information inferred from stereo cameras or derived from Light Detection and Ranging (LiDAR) sensors. The downsides of both variants are an increase of costs, the necessity of involved recalibration routines and the inhibition of the product design form factors due to fabrication constraints.

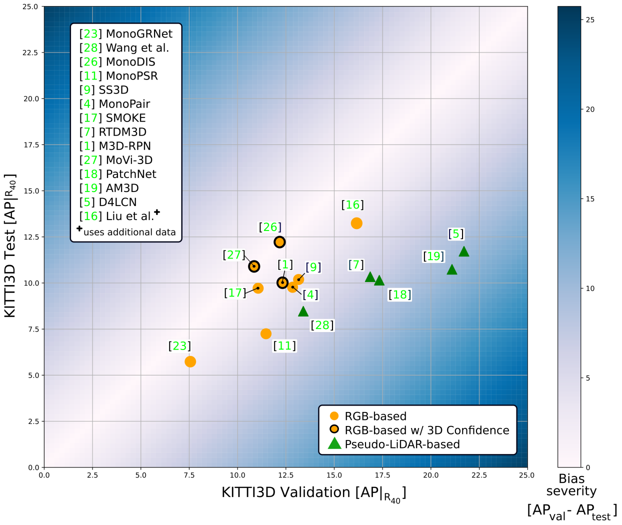

Figure 1: Performance of state-of-the-art 3D detection methods on the KITTI3D validation and test sets[1]. RGB-based methods (orange circles) exhibit a low performance discrepancy between the two sets, whereas Pseudo-LiDAR-based methods (green triangles) perform much better (up to 10 AP) on validation than on test. This indicates a bias, which we display by means of a blue-toned colormap. These results also show that the best performing RGB-based methods generally benefit from exploiting a *3D Confidence* (circled orange circles), a component which has not yet been introduced in any PL-based methods.

To overcome these issues, an emerging branch of 3D object detection methods is entirely based on monocular cameras [1, 9, 19, 20, 25, 27, 29]. Monocular cameras are a cheap alternative to the expensive LiDAR or stereo setups, but, at the same time, incur a substantially increased algorithmic complexity due to the absence of depth observations. Indeed, accurate estimation of an objects' distance to the camera is the most difficult task in monocular, image-

[1] We took as reference the performance on class *Car* in the *Moderate* difficulty, computed with the $AP|_{R_{40}}$ metric *i.e.* the one used as reference on the official KITTI3D benchmark.

based 3D object detection, making it an ill-posed problem. Despite the development of methods which focus on increasing the generalization with respect to distance [1, 27], monocular image-based methods still lag far behind their LiDAR or stereo-based counterparts.

A recent line of works [20, 31] has leveraged Convolutional Neural Networks (CNNs) for image-based depth predictions as depth substitute in monocular 3D object detection algorithms. Pseudo-LiDAR (PL) [29, 32] was promoted as a particularly effective depth representation, reporting impressive results on the challenging KITTI3D benchmark [8]. It essentially mimics a LiDAR signal for a RGB image by projecting each 2D pixel from its corresponding, estimated depth map into 3D space. With the resulting 3D point cloud, the 3D detection task is usually approached by applying state-of-the-art LiDAR-based (and thus 3D point-based) detection algorithms. PatchNet [18] has recently refuted 3D points as the source of PL's effectiveness by providing an equivalently performing implementation based on stacking 3D world coordinates as 2D maps. While this eliminates the claims of PL being advantageous due to its 3D point-based representation, their ablations confirmed the importance of operating on transformed 2D image coordinates incorporating camera intrinsics (focal length and principal point).

In this paper we argue that PL-based approaches, and more in general approaches that take depth as input, have introduced a distorted perception in the research community about their performance in the monocular setting with respect to other state-of-the-art methods that use RGB-images only. We identified two main reasons behind the issue, which constitute the two main contributions of this paper.

First contribution. State of the art PL-based methods report excellent performance on the KITTI3D validation set but do not show the same gains on the test set. In this work we perform an in-depth experimental study to analyze the reasons behind such inconsistency and demonstrate that top performing PL-based methods adopt a training protocol which artificially leads to high average precision on the validation set. The issue is evident in Fig. 1, where the discrepancy between the KITTI3D validation and test set performance of PL-based methods (green triangles) is much more pronounced than RGB-based methods (orange circles). Indeed, the depth estimation algorithms on which PL-based methods heavily rely are usually trained by including $\approx 30\%$ of the validation set data used for 3D object detection. Despite this issue was mentioned briefly in [29, 28], this biased training protocol was later used in many subsequent PL-based methods. This clearly indicates the necessity and the relevance for the community of a more detailed analysis, which we provide in this paper.

Second contribution. The outcome of a fair comparison on test set of PL-based methods against RGB-only based approaches on the KITTI3D benchmark is currently favouring more the latter ones. On the flip side, we found that published PL-based methods are penalized by the complete lack of a proper 3D confidence score which, as shown in Fig. 1 (circled orange circles), is becoming a fundamental component of state-of-the-art RGB-only methods. In this paper we propose, for the first time, to endow PL-based methods with a mechanism for predicting a 3D confidence, demonstrating remarkable performance gains. In particular, we show that, following previous RGB-only based methods [25], also in the case of PL, 3D confidence can be trained by directly regressing the expected loss. While this works well in practice, it is sensitive to the scale of the loss and, hence, requires some hyperparameter tuning. Moreover, it suffers from the issue of becoming overconfident as the training progresses towards an overfitting regime. In the spirit of addressing those two issues, we open a novel direction and successfully explore the possibility of having 3D confidences expressed in relative terms. Our novel finding leads to improved performance and sets the new state-of-the-art on the KITTI3D benchmark.

2. Related Works

Current approaches for monocular 3D object detection can be roughly divided in two categories: RGB-only methods, which directly address the ill-posed problem of the object's distance estimation, and PL-based methods, which leverage from automatically estimated depth maps or point clouds to recover the distance information.

Monocular RGB-only 3D detectors. Earlier approaches for monocular RGB-only 3D detection such as SSD-6D [10] and Deep3DBox [21] build on top of state of the art deep architectures for 2D detection, and exploit information from projective geometry to estimate the 3D pose and position of the objects in the scene. Mono3D [2] develops from the idea of generating 3D proposals and scoring them according to several cues, such as semantic segmentation features, object contour, and location priors. OFT-Net [24] operates by considering an orthographic feature transform to map a 2D feature map to bird-eye view. MonoGRNet [23] simultaneously estimates 2D bounding boxes, instance depth, 3D location of objects and local corners. GS3D [13] exploits an off-the-shelf 2D object detector and efficiently computes a coarse cuboid for each predicted 2D box, which is then refined to estimate the 3D bounding box. MonoPSR [11] jointly leverages 3D proposals and scale and shape estimation to accurately predict 3D bounding boxes from 2D ones. Recently, few works have proposed single-stage deep architectures [1, 27]. M3D-RPN [1] generates 2D and 3D object proposals simultaneously and exploits a post-processing optimisation and a depth-aware network to improve localization accuracy. MoVi-3D [27] is a lightweight architecture which exploits automatically generated virtual views where the object appearance is normalized with respect to distance to facilitate the detection task. Liu *et al*. [17] propose SMOKE, a deep architectures which predicts 3D bounding boxes by relying on key-point estimation as an intermediate task. MonoDIS [25] shows that training convergence and detection accuracy of 3D detection networks can be improved by considering loss disentanglement. In [25] 3D confidence for detection is also introduced for increasing performance. In this paper we show how this notion can be extended to PL and further improved introducing a relative measure of confidence.

Pseudo-LiDAR based 3D detectors. A second category of works exploit external data and network models to generate depth maps from the RGB input as an intermediate step for 3D detection. For instance, ROI-10D [20] introduces a loss to minimize the misalignment of 3D bounding boxes and exploits depth maps inferred with SuperDepth [22]. A disparity prediction module is considered in [31] and integrated into a network composed of two parts: one that generates 2D region proposals, and another that predicts 3D object location, size and orientation. Pseudo-Lidar [29] represents the first PL method, introducing the idea of interpreting depth maps as 3D point clouds which are then fed to state-of-the-art LiDAR-based 3D object detectors. In [29] the presence of a possible performance bias is also suggested but an in-depth experimental study is lacking. Pseudo-Lidar++ [32] improves the accuracy in the localisation of faraway objects by adapting a stereo network architecture and deriving a loss function for direct depth estimation. AM3D [19] proposes to integrate complementary RGB features into the PL pipeline and introduces a a specific module to map the 2D image data to the 3D point cloud. PatchNet [18] analyses the effect of depth data representation on performances and improves over previous PL models by integrating the 3D coordinates as additional channels of input data. However, all these works lack a fundamental component of state-of-the-art RGB-only based detectors [25] *i.e*. the estimation of a 3D confidence.

3. Preliminaries

We first review the monocular 3D object detection task and introduce the KITTI dataset [8] – the most influential benchmark to assess the performance of 3D detection methods. We also report the results of an experimental analysis on KITTI, highlighting the crucial role of depth estimation on the performance of state-of-the-art PL-based methods.

3.1. Monocular 3D Object Detection

The monocular 3D object detection task consists in detecting and localizing all the visible objects of interest (*e.g*. cars) by means of 3D bounding boxes given a single RGB image as input. Localization must be done in 3D space, properly estimating the 3D coordinates (in meters) of the center of the object $O_i = (X_i, Y_i, Z_i)$, where X_i, Y_i are related to the horizontal and vertical translations, respectively, and Z_i is the distance of the object's center from the camera. The localization also includes the estimation of the object's metric shape $S_i = (H_i, W_i, L_i)$ representing the object's height, width and length, as well as the object's rotation R_i w.r.t. the camera reference system. The detection requires also to estimate a confidence value C_i which generally reflects the quality and determines how confident the detector is about the particular 3D detection. In this monocular setting, it is common to assume to have a calibrated camera and to know the corresponding intrinsic camera parameters.

3.2. The KITTI Dataset

The KITTI Dataset comprises a broad collection of data from street-level sequences, captured with a multi-sensor setup in the city of Karlsruhe (Germany) in 2011. The remarkable diversity of the sensors enabled many benchmarks, including *3D object detection* and *depth estimation*, which are most relevant for this work.

KITTI 3D object detection benchmark (KITTI3D). To our knowledge, all 3D object detection methods, and in particular monocular image-based ones, adopted KITTI3D as their predominant, and usually exclusive, testing field. The KITTI3D benchmark is composed of an official *training* and *testing* split, comprising 7481 and 7518 images, respectively. Following *Chen* et al. [3], it is common to split the training set into *unofficial* training and validation splits, with 3712 and 3769 images, respectively. KITTI provides 2D and 3D bounding box annotations for *Cars*, *Pedestrians* and *Cyclists*, and each box is assigned to one of the *difficulty* levels *Easy*, *Moderate* or *Hard*, depending on the object's 2D height (≈ object's distance), degree of occlusion, and truncation. KITTI3D adopts two main evaluation metrics, *i.e*., *3D Average Precision (3D AP)* and *Bird's Eye View Average Precision (BEV AP)*. As reported in [26], $\text{AP}|_{R_{40}}$ is the only legitimate 3D detection AP score, deprecating the previously used $\text{AP}|_{R_{11}}$ score.

Depth prediction benchmark. The KITTI depth prediction benchmark offers *official* training and testing splits, but it is common to split [6] the training data into *unofficial* training and validation sets of 23488 and 697 images, respectively. Depth prediction methods are inferring pixel-specific distance estimates w.r.t. the camera and are evaluated with several metrics like *Absolute Relative Error (AbsRel)*, *Squared Relative Error (SqRel)*, *etc*.

3.3. The Crucial Role of Depth

We also provide the results of an *oracle* analysis demonstrating that depth is the most influential factor for performance in monocular 3D object detection. Following the definitions in Sec. 3.1, we used KITTI3D predictions

Category	Oracle sub-task	M3D-RPN [1] Easy	 Mod.	 Hard	MonoDIS [26] Easy	 Mod.	 Hard
RGB-based	–	12.78	10.36	8.07	16.71	12.32	10.58
	$\hat{R}$	14.71	11.78	9.26	17.27	12.76	11.45
	$H\hat{W}L$	13.47	10.52	8.26	16.75	12.56	11.29
	$\hat{XY}$	22.63	17.47	13.48	29.59	22.17	19.31
	$\hat{Z}$	**34.53**	**28.35**	**22.51**	**45.99**	**38.02**	**33.48**

Category	Oracle sub-task	*Wang et al.* [29] Easy	 Mod.	 Hard	PatchNet [18] Easy	 Mod.	 Hard
Pseudo-LiDAR-based	–	23.71	12.40	10.61	31.15	16.23	13.49
	$\hat{R}$	24.04	13.39	11.13	31.60	17.43	14.58
	$H\hat{W}L$	25.73	14.50	11.64	34.19	19.01	15.58
	$\hat{XY}$	33.76	20.37	17.22	44.23	25.62	21.76
	$\hat{Z}$	**53.71**	**35.15**	**29.38**	**59.81**	**41.93**	**35.94**

Table 1: **Oracle analyses**. We computed the object detection results (*Car* 3D $AP|_{R_{40}}$) of state-of-the-art methods by substituting selected predicted components (*Oracle*) with their corresponding ground-truth value (*e.g.* $\hat{Z}$).

3D Object Detector	Depth Estimator	*Validation set* 3D $AP\uparrow$ Easy	 Mod.	 Hard	*Test set* 3D $AP\uparrow$ Easy	 Mod.	 Hard
Wang et al. [29]	BTS Eigen	24.47	13.40	10.92	9.87	6.40	5.46
PatchNet [18]	BTS Eigen	31.60	18.22	15.10	14.00	8.70	7.39
Wang et al. [29]	BTS GeoSep	17.20	9.35	7.57	10.76	6.86	5.93
PatchNet [18]	BTS GeoSep	20.79	10.55	8.90	10.88	7.42	6.51

Table 2: Pseudo-LiDAR results on KITTI3D validation and official benchmark, class *Car*, official AP$|_{R_{40}}$ metric.

Train set	Validation Set	$d_1\uparrow$	$AbsRel\downarrow$	$RMSE\downarrow$	$SILog\downarrow$
Eigen	Eigen validation	0.908	0.084	4.003	16.577
Eigen	Detection training	0.926	0.067	3.806	15.250
Eigen	Detection validation	0.920	0.072	3.838	16.063
GeoSep	GeoSep validation	0.904	0.093	3.627	14.019
GeoSep	Detection training	0.858	0.111	4.830	15.960
GeoSep	Detection validation	0.872	0.105	4.429	15.872

Table 3: Depth estimation results with BTS on KITTI, computed w.r.t. ground-truth depth obtained from LiDAR scans.

of state-of-the-art monocular 3D object detection methods [1, 18, 25, 29] and compared their 3D object detection performances by *substituting* sub-task predictions (*e.g. depth*) with their corresponding ground-truth values. In Tab. 1 we show that certain sub-tasks like *rotation* (R) and *shape* (W, H, L) prediction, despite the substitution with ground-truth values, do not significantly improve performance. In contrast, substituting the predicted *depth* estimation (Z) with ground truth improves substantially, meaning that *depth* is by-far the most crucial component for 3D object detection. Notably, this observation is consistent for all the different tested methods.

4. The Bias in Pseudo-LiDAR Experiments

With depth identified as most critical component in monocular 3D detection works, it becomes obvious that PL-based methods are particularly sensitive to inputs from depth estimators trained in a biased way.

4.1. The Source of the Bias

To our knowledge, all PL-based methods published so far were exclusively evaluated on the KITTI3D [8] dataset which, as described in Sec. 3.2, shares data among several benchmarks like 3D object detection and depth prediction. With the advent of PL, it is however paramount to identify potential sources of cross-pollination in task-specific dataset splits. Our investigations showed that previous, PL-based works [29, 32, 18] were built on top of DORN [7], *i.e.* a state-of-the-art depth estimator, that in turn however included a majority of images from the detection *validation* set during its training. Specifically, we found **1226/3769** (**32.5%**) images to be shared between the widely adopted Eigen *et al.* training split [6] for depth estimation and the commonly used Chen *et al.* [3] validation split for 3D object detection. When adding also the images belonging to the same capturing sequence, the numbers slightly increase to 1258/3769 (33.4%).

We illustrate the full extent of the contamination in Fig. 2, plotting GPS positions and hence the overlap of the different splits (Eigen *et al.* depth training split in black; Chen *et al.* validation split for 3D object detection in red).

In Tab. 2, we show the effect of the contamination on the validation and test scores for two state of the art PL methods [29, 18]. The rows corresponding to *Eigen* are based on biased depth as input, which was generated using BTS [12] trained on the Eigen *et al.* split. We rely on BTS [12] because it represents a novel state-of-the-art depth estimator[1]. The huge performance drops (up to 17.6 AP) between obtained validation and test scores clearly indicate the relevance of the bias issue discussed here.

4.2. Can the Bias be Removed?

As stated above, a contamination exists between the depth training and detection validation sets used by PL-based methods. To further support our hypothesis that this contamination causes a bias in the KITTI3D validation scores, we introduce a geographical separation between the two data sets. We create a novel depth training set, *GeoSep*, enforcing significant spatial separation between the datasets used for depth estimation and detection. Exploiting the GPS information included in the available KITTI benchmark data, we create two novel train/val depth splits by selecting images captured farther than 200m from any detection image (additional details are given in the supplementary material). Our new *GeoSep* data split is visualised in Fig. 2 (green markers), showing a clear safety margin between depth training (Eigen *et al.* [6], black markers) and object detection validation (red markers) splits.

[1] Differently from the depth estimator usually used by PL methods, *i.e.* DORN [7], for which official training code is not publicly available, BTS provides a complete open-source code (`https://github.com/cogaplex-bts/bts`)

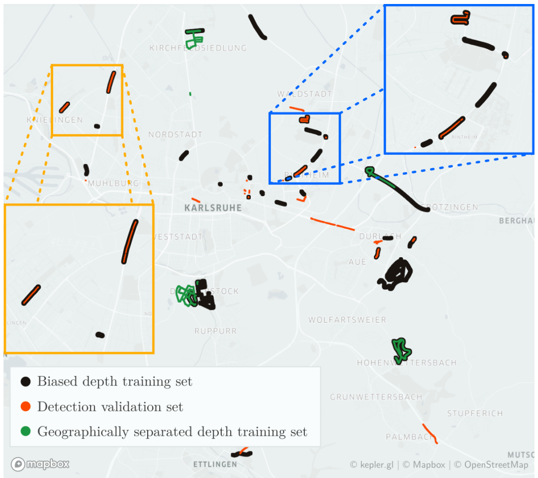

Figure 2: Geographical distribution of the biased training (black), detection validation (red) and geographically separated (green) depth training splits. Square boxes highlight parts where the overlap between the biased depth training and detection validation sets are particularly evident.

To verify whether our novel split solves the bias issue, we use it to train a depth estimation model. We use BTS [12] as depth prediction network and, again, consider the state-of-the-art PL methods in [29] and [18]. The results of our analysis, shown in Tab. 3 and Tab. 2, still indicate the presence of a bias in both depth estimation and 3D detection results. To our great surprise, despite the lack of geographical intersection between the training splits, the gap between validation and test results is still substantially higher (up to 10 AP) compared to the gap that methods using RGB-only inputs typically incur (≈ 3-5 AP). This suggests a more structured form of contamination that goes beyond the simple geographical distribution of the data, perhaps related to intrinsic factors such as the visual appearance and semantic similarity of the scenes (*e.g*. presence of similar streets).

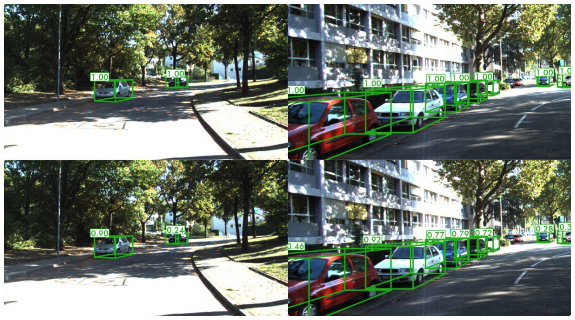

Figure 3: Qualitative results of our method with confidence scores of each detection. Top: We report the 2D confidence score that PL-based methods typically use. Bottom: We report the learned 3D confidence predicted by our method.

The persistence of the bias using both depth training splits (Eigen or *GeoSep*) make us draw the conclusion that fair comparisons should not, at least in these settings, be performed on the KITTI3D validation set. On the other hand, the fact that published PL-based methods are not able to surpass state-of-the-art RGB-only based methods (see Tab. 5 first block) is an indication that the test set itself does not suffer from the same type of bias, thus preserving its validity for the sake of fair comparisons. Following these conclusions, all the comparisons related to our second contribution will be made on the official test set while the validation set will be used for ablation studies. Despite our study only partially identifies the source of the bias, this work provides the first analysis of the issue revealing potentially unfair comparisons and we encourage the community to take it into account for future works.

5. 3D Confidence for PL-based Methods

As described in the previous section, the performance of PL-based methods is deeply influenced by the upstream depth estimation task. We will now demonstrate that the estimation of the 3D confidence has an equally relevant role.

The 3D confidence can be thought as an estimate of the quality of the 3D detection which, as described in Sec. 3.1, has to be associated to each 3D bounding box. In datasets such as KITTI3D, this confidence takes an active role in the computation of the metrics (*e.g*. Average Precision). In light of this fact we observed that existing Pseudo-LiDAR methods do not perform the 3D confidence estimation in any way but rely on the class probability coming along with the 2D detections. By doing so, the confidence adopted by current PL-based methods is actually agnostic to the quality of the 3D predictions and therefore not effective for the role it should take. On top of this, as shown in Fig. 3, we observed that 2D detectors are often too confident and therefore the need for a 3D confidence seems essential. For this reason, we propose to endow PL-based methods with the ability of estimating the 3D confidence.

5.1. Proposed Architecture

In order to describe how the confidence is estimated we first provide an overview of the general architecture we adopt, similarly to other PL-based methods [29, 18], and subsequently detail our contribution.

PL-based 3D detection architecture. The architecture commonly adopted by state-of-the-art PL-based methods, which we also use in this work, is depicted in Fig. 4 (excluding the red block, *i.e*. our contribution). It can be divided into three main branches, namely *2D Detection*, *Pseudo-LiDAR* and *3D Detection*. The *2D Detection* and *Pseudo-LiDAR* components usually exploit pre-trained architectures and have the purpose of understanding where the object of

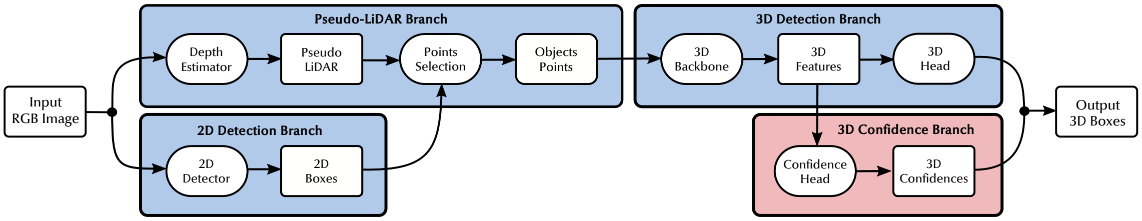

Figure 4: Architecture of a generic Pseudo-LiDAR-based method integrating the proposed 3D confidence component.

interest are in the image, as well as of estimating the per-pixel depth, respectively. The per-pixel depth-map is then converted to Pseudo-LiDAR 3D point-cloud and, finally, the points belonging to each object are selected and filtered to discard elements corresponding to *e.g.* road, occlusions. The 3D detection block is responsible for the estimation of the output 3D bounding boxes, taking as input the selected PL points to perform a point-based 3D detection by means of an initial *3D Backbone* followed by a *3D Head*.

3D confidence head. In the following we describe our main contribution, *i.e.* an approach which endows the PL-based methods under consideration with the ability to predict a self-supervised 3D confidence. In order to reliably and accurately estimate the 3D confidence of bounding boxes, appropriate 3D-related feature representation need to be computed. For this reason, in this work we introduce an additional branch in the architecture, namely the *3D Confidence Branch*, which, as shown in Fig. 4 (red block), takes as input the set of *3D Features* computed by the *3D Backbone* and outputs a single value C_i, *i.e.* the 3D confidence, for each object. In the presence of $K > 1$ classes the output is a set of K confidences C_i^k, one for each class k. Note that our proposed *3D Confidence Branch* is not tied to any particular architecture and requires minimal modifications to existing PL approaches. An example of a simple implementation is by mirroring the architecture of the *3D Head*, thus leading to limited computational complexity and minimal overhead in term of inference time.

5.2. Learning the 3D Confidence

In this paper we propose two different approaches for 3D confidence prediction. The first approach, which we denote as *Absolute 3D confidence* estimation is inspired by previous RGB-only based methods [26], while the second strategy, called the *Relative 3D confidence* estimation method, is introduced with this paper. In both cases, given a 3D bounding box B_i and the corresponding ground-truth $\hat{B}_i$, the loss for the 3D confidence prediction C_i^{3D} takes the following cross-entropy form:

$$L_{\text{conf}}(C_i^{\text{3D}}|B_i,\hat{B}_i) = -T_i \log C_i^{\text{3D}} - (1-T_i)\log(1-C_i^{\text{3D}}),$$

where T_i is the target confidence value that takes a different value for the absolute and relative confidences, as described below. In case of multiple object categories, we assume to have independent 3D confidence predictions per class.

Absolute 3D confidence. Inspired by [25], the absolute 3D confidence is trained by directly regressing the loss of the prediction. This boils down to setting $T_i^{\text{abs}} = e^{-\frac{1}{\beta}\ell(B_i,\hat{B}_i)}$ as the target confidence, where $\ell(B_i,\hat{B}_i)$ is the loss incurred by the bounding box prediction and $\beta > 0$ is a temperature parameter. Since this approach leads to a 3D confidence which reflects the quality of a 3D detection in absolute terms we call it *Absolute 3D Confidence*.

Relative 3D confidence. We also propose a novel approach which aims at overcoming a couple of issues that affect loss-based confidences like the one described above. The first is that they are sensitive to the scale of the loss values, which requires to tune scaling factors. The second is that they are not immune to the issue of the network becoming overconfident as the training progresses towards an overfitting regime. We solve these two issues by shifting the semantics of the score that the network provides with each prediction from an absolute confidence to a relative confidence. A typical confidence score is supposed to be representative of the absolute quality of the prediction. Instead, the score that we require the network to learn is representative of the quality of the prediction relative to other typical predictions done by the network. For this reason, this novel confidence is regarded as *Relative 3D Confidence*.

Consider a training set containing n 3D objects, where $\ell_i \triangleq \ell(B_i,\hat{B}_i)$ denotes the loss incurred by the model on the predicted 3D bounding box for the ith object. We propose to regress as a confidence C_i^{3D} for the prediction B_i, the proportion of 3D objects in the training set on which the model performs equal or worse than on object i, *i.e.* our target confidence prediction is given by

$$T_i^{\text{rel}} = \frac{1}{n-1}\sum_{\substack{j=1\\ j\neq i}}^{n} \mathbb{1}_{\ell_j \geq \ell_i}\,,$$

where $\mathbb{1}_P$ denotes the indicator vector for predicate P. The proposed confidence is inherently relative because it does not depend on the actual absolute values of the losses, but rather on their ordering. In order to train a model to regress the new confidence, we need to compute the value of T_i^{rel} for each 3D object i in the mini-batch. However, to compute

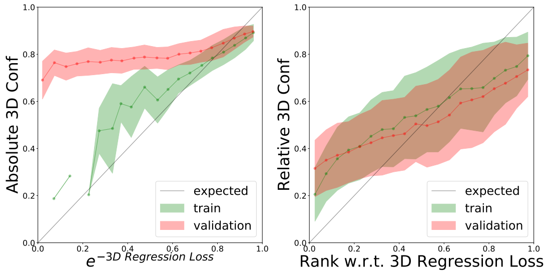

Figure 5: Binned scatter plot of the *Absolute* (left) and *Relative* (right) 3D confidences on the training (green) and validation (red) splits. Since the x-axis represents what the confidence is supposed to regress, the ideal curve lies on the diagonal. On the training set, both the absolute and relative confidences follow the expected curve. On the validation set however only the relative confidence aligns with the expected curve, whereas the absolute confidence consistently overestimates it, signaling the issue of being overconfident.

such target values, we need to access the loss incurred on each 3D object in the training set, which is computationally demanding. An alternative solution consists in tracking the past loss values for each 3D object in the training set. This is feasible, but given the frequent updates of the model the past losses would be soon outdated. Also, there exist augmentation strategies, for which it might be hard to match predictions across epochs. Interestingly, there is a very simple, stochastic procedure that allows us to train the desired confidence by only accessing loss values in a mini-batch of at least 2 elements. Specifically, we randomly pair the bounding box prediction for 3D object i in the mini-batch with the prediction obtained for another, distinct 3D object π_i in the same set. Given this assignment, we compute a binary target value $\hat{T}_i$ for training the confidence as $\hat{T}_i = \mathbb{1}_{\ell_i \leq \ell_{\pi_i}}$, and plug this into the cross-entropy loss L_{conf} that we use to train the confidence. To see why this works, fix a 3D object i in the training set. Then the target variable $\hat{T}_i$ is a random variable, where π_i is uniformly sampled among the other $n-1$ 3D objects in the training set. Accordingly, the expected value $\mathbb{E}[\hat{T}_i]$ of $\hat{T}_i$ yields exactly T_i^{rel}. This is also the value to which the predicted confidence C_i^{3D} will tend to if trained with the cross-entropy loss L_{conf}, assuming the losses will eventually converge during training. If multiple classes are present, we have independent confidence predictions per class and the random pairing procedure is applied only between 3D objects in the mini-batch having the same class. No loss is computed if a 3D object is the only one of a given class in the mini-batch.

Pros and cons of relative confidences. Our score takes the role of a *relative* confidence, which does not give information about the quality of a prediction in absolute terms, but rather assesses the quality of the prediction relative to other predictions done by the network. As an extreme example, a nearly perfect prediction can get zero confidence if that is the prediction that incurred the highest error in the training set. Similarly, a bad prediction can get high confidence score if it is the best prediction the network ever made. An interesting property of our confidence score is the invariance to order-preserving transformations of the losses. This renders the score more robust to scenarios where the losses change over time, which is the setting that is encountered at training time. Additionally, our confidence score does not suffer from the problem of the network becoming overconfident, because of the relative nature of our score (see Fig. 5). On the downside, an absolute confidence is typically helpful to filter predictions based on a threshold. Doing the same with a relative confidence might be cumbersome. This is why we actually combine our relative confidence score from the 3D detection head with the 2D *absolute* confidence score from the 2D detection. Indeed, the 2D detection confidence works well for the sake of removing bad quality predictions, but lacks resolution for discriminating the quality of the predictions left. This is where our relative confidence comes into play, since it does not suffer from the issue of becoming overconfident.

	Validation 3D AP			*Test 3D AP*		
Method	Easy	Mod.	Hard	Easy	Mod.	Hard
Wang et al.	24.47	13.40	10.92	14.17	8.47	7.29
+ *Absolute 3D Conf*	32.44	20.84	17.26	18.56	10.99	9.31
+ *Relative 3D Conf*	**34.56**	**22.04**	**18.87**	**18.74**	**11.04**	**9.41**
PatchNet	30.53	17.33	12.80	15.70	10.15	8.79
+ *Absolute 3D Conf*	37.04	23.26	18.78	22.21	12.51	10.46
+ *Relative 3D Conf*	**38.60**	**23.68**	**19.51**	**22.40**	**12.53**	**10.64**

Table 4: KITTI3D Validation and Test set $AP|_{R_{40}}$ results.

6. Experiments

We test the validity of our proposed 3D confidence measures by considering the deep architectures implemented in two common PL methods, *i.e.* the first PL method, *Wang et al.* [29], and a current top-performing state-of-the-art method, PatchNet [18]. We modify their architectures to include our proposed *3D Confidence Head*, which in all our experiments is implemented mirroring the existing *3D Head*. In detail, it is implemented as one set of fully-connected layers for [29] and three distance-specific fully-connected modules for [18]. We follow the schedules and hyperparameters choices of [18] and [29], with the only addition of the 3D Confidence loss which is given a weight of 1.0. Additional implementation details are given on the supplementary material. We follow the experimental protocol of all PL-based works and evaluate our method on the KITTI3D [8] benchmark. All the results presented and reported in this work have been computed with the official $AP|_{R_{40}}$ metric.

Experimental results. In Tab. 4 we investigate the influence of the 3D confidence on the PL-LiDAR methods [29]

Method	*Car 3D AP* Easy	Mod.	Hard
OFTNet [24]	1.61	1.32	1.00
FQNet [15]	2.77	1.51	1.01
ROI-10D [20]	4.32	2.02	1.46
GS3D [13]	4.47	2.90	2.47
MonoGRNet [23]	9.61	5.74	4.25
Wang et al. [29]	9.87	6.40	5.46
MonoDIS [25]	10.37	7.94	6.40
MonoPSR [11]	10.76	7.25	5.85
Mono3D-PL [30]	10.76	7.50	6.10
SS3D [9]	10.78	7.68	6.51
MonoPair [4]	13.04	9.99	8.65
SMOKE [17]	14.03	9.76	7.84
RTDM3D$^{\mathbf{A}}$ [14]	14.41	10.34	8.77
M3D-RPN [1]	14.76	9.71	7.42
MoVi-3D [27]	15.19	10.90	9.26
PatchNet [18]	15.68	11.12	10.17
AM3D [19]	16.50	10.74	9.52
MonoDIS [26]	16.50	12.20	10.30
D4LCN [5]	16.65	11.72	9.51
Liu et al.$^{\mathbf{A}}$ [16]	21.65	**13.25**	9.91
Our PatchNet	**22.40**	12.53	**10.64**

Table 5: Test set SOTA $AP|_{R_{40}}$ official results on KITTI3D for class *Car*. Best scores in bold, runner-ups underlined. **A** = trained with additional data and only on class *Car*.

Method	*Cyclist 3D AP* Easy	Mod.	Hard	*Pedestrian 3D AP* Easy	Mod.	Hard
M3D-RPN [1]	0.94	0.65	0.47	4.92	3.48	2.94
MoVi-3D [27]	1.08	0.63	0.70	8.99	5.44	4.57
MonoDIS [26]	1.17	0.54	0.48	7.79	5.14	4.42
D4LCN [5]	2.45	1.67	1.36	4.55	3.42	2.83
SS3D [9]	2.80	1.45	1.35	2.31	1.78	1.48
MonoPair [4]	3.79	2.12	1.83	**10.02**	**6.68**	**5.53**
Our - PatchNet	**7.79**	**4.32**	**3.98**	3.00	1.81	1.59

Table 6: Test set SOTA $AP|_{R_{40}}$ official results on KITTI3D of published multi-class methods for *Cyclist* and *Pedestrian*. Best scores in bold, runner-ups underlined.

and [18]. In particular, we compute the 3D object detection metrics on the validation and test set of KITTI3D with the baseline methods as well as with the addition of the *3D Confidence Head* (+ *3D Confidence*) trained with both the absolute and relative learning procedure. As shown in the table, we observe a major improvement on the $3D\ AP$. This validates our hypothesis about the importance of having a 3D confidence prediction component in PL-based methods. The relative 3D confidence also consistently outperforms the absolute one, demonstrating the validity of our proposed relative formulation. In Tab. 5, 6 we compare our results with state-of-the-art methods on the KITTI3D test set. Our method based on PatchNet structure achieves state-of-the-art performance on the classes *Car* and *Cyclist*, while it does not surpasses previous approaches on the *Pedestrian*. We ascribe this behaviour to the fact that instances of the class *Pedestrian* are extremely rare in the *Eigen* depth training set, thus having a negative impact on the depth map quality.

Method	Depth Estimator	*Validation 3D AP* Easy	Mod.	Hard	*Test 3D AP* Easy	Mod.	Hard
PatchNet	BTS GeoSep	20.79	10.55	8.90	10.88	7.42	6.51
+ Abs. 3D Conf.	BTS GeoSep	23.37	15.49	12.70	17.38	10.30	8.78
+ Rel. 3D Conf.	BTS GeoSep	**24.51**	**17.03**	**13.25**	**17.69**	**10.85**	**9.37**

Table 7: Validation and Test set $AP|_{R_{40}}$ results on KITTI3D obtained with the *GeoSep* depth training split.

In Tab. 7 we also report the results of *PatchNet + 3D Confidence* obtained by relying on a depth estimator trained on the *GeoSep* depth training set. Again, this table demonstrates the merit of our contributions in term of 3D confidence estimation. Additionally, relatively to our first contribution, we notice that a gap between the validation and test set results still exists, indicating the bias issue. However, this does not invalidate our observations regarding the effectiveness of the proposed 3D confidence.

7. Conclusions

In this paper we have shown that top-performing Pseudo-LiDAR-based works suffer from a bias in the reported validation scores for the KITTI3D benchmark. The source of the issue is partially due to an overlap that exists between the training set used to train the upstream depth estimators, providing the depth in input to the PL-based methods, and the validation set used for 3D object detection. In an attempt to validate the hypothesis we constructed an geographically separated training set for the depth estimators by ensuring geographical separation to the detection validation set. However we found that this is not sufficient to remove the bias in the validation set, which indicates the existence of a more structured nature of the issue. As a consequence, future works involving PL-based methods on KITTI3D should avoid comparative analysis against other methods using the validation set, but rather rely on the test set. In the second part of our work, we provided an architectural change to PL-based methods aimed at endowing them with the ability of predicting 3D confidences. We showed that with this simple change PL-based methods get remarkable improvements on the KITTI3D benchmark establishing a new state of the art.

Acknowledgements

We would like to thank Andreas Geiger for supporting our experiments on the KITTI3D benchmark. We also thank Xinzhu Ma, Wanli Ouyang and Garrick Brazil for sharing their detections and for helpful discussions. E. Ricci acknowledges financial support from the H2020 EU project MARVEL - Multimodal Extreme-Scale Data Analytics for Smart Cities Environments.

References

[1] Garrick Brazil and Xiaoming Liu. M3D-RPN: Monocular 3d region proposal network for object detection. In *ICCV*, pages 9287–9296, 2019. 1, 2, 4, 8

[2] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, 2016. 2

[3] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G. Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NIPS*, 2015. 3, 4

[4] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *CVPR*, pages 12093–12102, 2020. 8

[5] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *CVPR*, pages 11672–11681, 2020. 8

[6] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, pages 2650–2658, 2015. 3, 4

[7] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *CVPR*, 2018. 4

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The kitti vision benchmark suite. In *CVPR*, 2012. 2, 3, 4, 7

[9] Eskil Jorgensen, Christopher Zach, and Fredrik Kahl. Monocular 3d object detection and box fitting trained end-to-end using intersection-over-union loss. In *CVPR*, 2019. 1, 8

[10] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, October 2017. 2

[11] Jason Ku, Alex D. Pon, and Steven L. Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *CVPR*, 2019. 2, 8

[12] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv:1907.10326*, 2019. 4, 5

[13] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *CVPR*, 2019. 2, 8

[14] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *ECCV*, 2020. 8

[15] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3d object detection. *CoRR*, abs/1904.12681, 2019. 8

[16] Yuxuan Liu, Yuan Yixuan, and Ming Liu. Ground-aware monocular 3d object detection for autonomous driving. *RAL*, 2021. 8

[17] Zechen Liu, Zizhang Wu, and Roland Tóth. SMOKE: Single-stage monocular 3d object detection via keypoint estimation. *CoRR*, abs/2002.10111, 2020. 3, 8

[18] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *ECCV*, 2020. 2, 3, 4, 5, 7, 8

[19] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang, and X. Fan. Accurate monocular object detection via color- embedded 3d reconstruction for autonomous driving. In *ICCV*, 2019. 1, 3, 8

[20] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. ROI-10D: Monocular lifting of 2d detection to 6d pose and metric shape. In *CVPR*, 2019. 1, 2, 3, 8

[21] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *CVPR*, July 2017. 2

[22] Sudeep Pillai, Rares Ambrus, and Adrien Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation. In *ICRA*, 2019. 3

[23] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for 3d object localization. In *AAAI*, 2019. 2, 8

[24] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *CoRR*, abs/1811.08188, 2018. 2, 8

[25] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel Lòpez-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, pages 1991–1999, 2019. 1, 2, 3, 4, 6, 8

[26] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection: From single to multi-class recognition. In *TPAMI*, 2020. 3, 4, 6, 8

[27] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Elisa Ricci, and Peter Kontschieder. Towards generalization across depth for monocular 3D object detection. In *ECCV*, pages 9287–9296, 2020. 1, 2, 8

[28] Xinlong Wang, Wei Yin, Tao Kong, Yuning Jiang, Lei Li, and Chunhua Shen. Task-aware monocular depth estimation for 3d object detection. In *(AAAI)*, 2020. 2

[29] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019. 1, 2, 3, 4, 5, 7, 8

[30] Xinshuo Weng and Kris Kitani. Monocular 3d object detection with pseudo-lidar point cloud. In *ICCVW*, 2019. 8

[31] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, June 2018. 2, 3

[32] Yurong You, Yan Wang, Wei-Liu Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *CoRR*, abs/1906.06310, 2019. 2, 3, 4