Validation of Safety Metrics for Object Detectors in Autonomous Driving

Andreas Rønnestad
a.roennestad@gmail.com
Norwegian University of Science and Technology
Trondheim, Norway

Andrea Ceccarelli
andrea.ceccarelli@unifi.it
University of Florence
Florence, Italy

Leonardo Montecchi
leonardo.montecchi@ntnu.no
Norwegian University of Science and Technology
Trondheim, Norway

ABSTRACT

Object detection consists in perceiving and locating instances of objects in multi-dimensional data, such as images or lidar scans. While object detection is a fundamental step in autonomous vehicles applications, it is typically evaluated with generic metrics like precision and recall. Recently, metrics that take into account safety have been proposed in the literature. In this paper we compare two recently proposed safety metrics models for object detectors, "Planning KL divergence" and "Object Criticality Model", validating to what extent they actually measure the safety of an object detector when employed in an autonomous driving application. We base our experiments on the nuScenes dataset, and we compare the two metrics in different scenarios, both nominal ones and with the deliberate injection of detection faults. We conclude that both metrics serve as an indicator of the safety of an object detector, but they also provide different perspectives, and should therefore be used complementarily. As a by-product of this work, we also release a library for the injection of faults in experiments based on the nuScenes object detection task.

CCS CONCEPTS

• **Software and its engineering → Software safety**; • **Computing methodologies → Object detection**.

KEYWORDS

Object detectors, safety, reliability, PKL, OCM, criticality model

ACM Reference Format:
Andreas Rønnestad, Andrea Ceccarelli, and Leonardo Montecchi. 2024. Validation of Safety Metrics for Object Detectors in Autonomous Driving. In *The 39th ACM/SIGAPP Symposium on Applied Computing (SAC'24), April 8–12, 2024, Avila, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3605098.3635907

1 INTRODUCTION

The task of object detection has seen much innovation within the last decade. Much of the progress within the field can be attributed to the introduction of increasingly sophisticated deep learning approaches, made possible by the increasing processing power of modern GPUs [12]. Today, emerging safety-critical technologies rely on object detection as a fundamental part of their perceptual interface to their environments, the most prominent being automated driving assistance systems and autonomous vehicles [24].

When evaluating the performance of object detection models, the most accepted metrics rely on the concepts of *precision* and *recall* [22]. Precision is defined as the number of correct predictions out of all the predictions made, and recall as the portion of *ground truth* (GT) objects that are correctly predicted, respectively. Utilizing precision and recall for evaluating the performance of object detectors (ODs) is helpful for the basic task of *generic object detection*, associated with locating and classifying instances of objects from a number of predefined categories [17]. However, when ODs are applied in specialized and safety-critical systems, traditional metrics fail to consider the situation in which detections are made. In tasks such as autonomous driving, failing to detect specific objects poses a higher risk to the safety of the agent, its environment, and its passengers. Furthermore, the incorrect detection of non-existent objects may cause downstream components to unnecessarily interrupt the driving task. The differences in the relevance, or *criticality*, of objects, are not reflected in traditional precision- and recall-based metrics, which are instead context-agnostic.

In the last three years, context-aware (and safety-aware) metrics have been proposed, with the objective to evaluate ODs with respect to the safety and reliability of the system in which they will operate [3, 23, 30]. However, the very few proposed metrics convey different information, and no work has attempted to compare them so far.

In this paper, we validate and compare two recently proposed safety-aware metrics models for object detectors, namely PKL [23] and the OCM [3]; both rank the importance of specific detections in a scene, either explicitly or implicitly. The motivation of this paper and the corresponding experimental work is to provide a basis for comparing safety-oriented metrics for object detectors, so to motivate further work towards solid metrics for the evaluation of perception models with regard to safety. Our work aims to support system and software engineers in architecting ODs and assessing their deployment in safety-critical and reliable systems.

The rest of the paper is organized as follows. In Section 2 we introduce the background and related work, while the two selected metrics models are reviewed in more detail in Section 3. Section 4 presents the technical environment for our analysis, while Section 5 details the adopted methodology. Experiments and results are presented in Section 6 and final conclusions are drawn in Section 7.

SAC '24, April 8–12, 2024, Avila, Spain

ACM ISBN 979-8-4007-0243-3/24/04.
https://doi.org/10.1145/3605098.3635907

2 BACKGROUND AND RELATED WORK

2.1 Object Detection

The task of object detection consists in locating and classifying semantic objects of certain object classes within an input *frame*, with the frame being either a 2D visual image or a 3D point cloud.

The output of an object detection model is a list of *bounding boxes* (BBs), their *labels* and their *confidence scores* [12]. The label is the predicted semantic class of the object, and the confidence score reflects the confidence of the detection model in that prediction. BBs are tightly bound boxes encompassing objects in the scene, represented in 2D and 3D as rectangles and cuboids, respectively.

Most modern object detection approaches apply a Convolutional Neural Network (CNN) as a *backbone* for performing feature extraction. The purpose of the backbone is to extract meaningful features from the original input [12], reducing the dimensionality of the data. Typically, features extracted by the backbone serve as input to a *prediction head*, an architecture that receives high-level features and is responsible for performing the final classification task. The specific architecture of an object detector may vary, combining different backbones and different prediction heads.

2.2 Evaluation Metrics for Object Detectors

To evaluate the predictions produced by an object detector, a comparison between the predicted BBs and the BBs in the ground truth is performed. Object detectors compute BBs with an assigned confidence score. Then, a *detection threshold* is applied as a configuration parameter: all BBs with a confidence score above the selected threshold are considered as predictions, while the others are discarded.

The classification of true positives (TPs), false positives (FPs), and false negatives (FNs), is based on some definition of distance between the predicted BBs and the ground truth BBs. In this paper, we use the distance between their center points [2]. While there are several measures that can evaluate the performance of object detectors, the conventional approach is based on measures that are derived from the count of TPs, FPs, and FNs. Note that TNs are not relevant for object detectors, because in any image there are infinite BBs that should not be detected.

Precision, $P = TP/(TP + FP)$, indicates how many of the selected items are relevant; precision is 1 if all the detected objects actually exist in the GT, and 0 if none of them exist. Conversely, *Recall*, $R = TP/(TP + FN)$, indicates how many items from the GT are correctly selected. A detector with recall equal to 1 detects all the objects in the GT. Various derived metrics are grounded in precision and recall, the most commonly used being the F1-score [22] and the Average Precision (AP) [6]. The F1-score is essentially the harmonic mean of precision and recall, while the AP summarizes the precision-recall curve as the weighted mean of precision scores achieved at different detection thresholds.

While the metrics presented so far indicate the ability of ODs to accurately predict instances of objects in a scene, they do not consider the importance of such objects within the specific scenario. Research on applying ODs in safety-critical environments has raised the problem of defining safety-aware evaluation metrics. We review the most relevant ones for our work in the next section.

2.3 Related Work

To date, relatively few attempts have been made to define safety-aware metrics for object detectors. Furthermore, they all appeared within the last three years.

We investigated related research works based on the following criteria: i) the work proposes a new situation-aware metric for benchmarking object detectors; ii) the metric introduced is safety-oriented (in some way); iii) experimental results are presented; and iv) factors considered in the evaluation of the detectors are documented well enough to allow for an evaluation of experimental results with respect to safety and reliability. Details of our review are not reported here for the sake of brevity; a more extended report can be found in [27]. We identified five relevant works introducing safety-related metrics, which are briefly described in the following.

The work in [18] defines task-oriented metrics to detect pedestrians. The idea is to apply the IoU (Intersection over Union, i.e., the overlapping area between the predicted BBs and the ground-truth BBs, divided by the area of union between them [22]) to the predicted BBs of pedestrians and their GTs, and measure up to which distance a specified threshold holds. The resulting metric is the maximum distance at which no ground truth BB is missed, for a specific IoU threshold. The proposal is evaluated using the CARLA simulator [5].

The metric proposed in [31] is designed to evaluate models that implement *people detection* in off-road environments. The idea is to weight BBs, both in the GTs and in the detected ones, according to a weighting factor based on the position and estimated time-to-collision with the object. The work was evaluated on the "BOSCH internal off-road dataset", which is not available to the public.

The work of [30] proposes a metric to evaluate predictions of an object detection and tracking algorithm. The authors argue that for an algorithm to be safe, multiple dimensions need to be evaluated, namely quality (of the predictions), relevance (of predicted objects with respect to the ego vehicle), and time (to produce the predictions). Each of these factors is scored with a combination of existing metrics and new heuristics, and then aggregated to produce a final safety score. The proposal was evaluated on the KITTI dataset [7].

Finally, the PKL [23] and OCM [3] metrics are the focus of this paper, and they are described in detail in Section 3.1 and Section 3.2, respectively. We chose those two for our work, because they have been both evaluated on the nuScenes dataset by their authors, but at the same time they are based on two very different underlying ideas. Both Guo et al. [10] and Ceccarelli et al. [3] compared, respectively, the PKL and the OCM against traditional metrics such as AP and mAP. However, those comparisons aimed to expose the difference of their metrics to traditional metrics. In this paper we instead aim at investigating which metric is most suited to describe the safety guarantees of ODs. We are not aware of evaluations comparing safety-aware metrics for ODs, such as the PKL or the OCM.

3 OVERVIEW OF THE SELECTED METRICS

3.1 Planning KL-divergence

In [23], Philion et al. argue that the evaluation of the performance of perception systems in autonomous vehicles should be aligned with the downstream task of trajectory planning. Planning is a crucial part of the autonomous pipeline, so the "best" detection

models should be those that make the planner compute a trajectory as close as possible to the one computed on GT information only. Based on this observation, they propose the *Planning KL-divergence* (PKL) metric, as a measure of the difference between the trajectory planned based on GT objects, and the trajectory planned based on objects detected by an object detector.

In more details, PKL is a measure of the KL-divergence [13] between the probability distribution of future positions of the vehicle, at different time steps, given the semantic observations (detections) of the detector and the ideal observations represented by GT objects [23]. As a measure of divergence, a "perfect" detection would receive a PKL score of 0, corresponding to no divergence between the distributions. Further details can be found in [23].

Evaluating a detector based on the impact of its predictions on the trajectory planned by the vehicle implicitly ranks the "importance", or criticality, of detecting specific objects in the scene. With the important assumption that the adopted planner is sufficiently robust to plan a reasonable trajectory with regard to safety, the PKL score can function as an indicator of the impact on the safety of a specific detection model.

3.2 Object Criticality Model

The authors of [3] argue that an evaluation metric applied in autonomous systems should discriminate based on the context surrounding the vehicle. The authors propose the Object Criticality Model (OCM), which shares some ideas with the work of Wolf et al. [31] discussed before. In the OCM, a criticality value is assigned to each object in a specific scene, based on safety-relevant factors relating the object and the ego vehicle. Such value is computed for both the GT objects and the predicted objects.

Three factors are considered to compute the criticality of an object B: distance, colliding trajectory, and time-to-collision, which result in three individual criticality scores, $\kappa_d(B)$, $\kappa_r(B)$, and $\kappa_t(B)$. Each of these scores ranges in the interval $[0, 1]$, with 1 meaning maximum criticality. The model depends on three parameters, $D_{\max}$, $R_{\max}$, and $T_{\max}$, which define a threshold after which the corresponding criticalities assume value 0. For example, $D_{\max} = 30$ means that for objects farther than 30 meters $\kappa_d(B) = 0$. Further details can be found in [3]. The overall criticality weight of an object, $\kappa(B)$, is defined as a linear combination of the three above scores.

The performance of a detector is then measured in terms of "how much criticality" it is able to detect. More specifically, the authors of [3] define two metrics called *reliability-weighted precision* ($P_{\mathcal{R}}$), and *safety-weighted recall* ($R_{\mathcal{S}}$), as variants of the Precision and Recall metrics in which objects are weighted based on their criticality score. The authors also define *Average Critical Precision* ($\mathrm{AP}_{\mathrm{crit}}$), as a variant of AP, computed using $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$.

4 TECHNICAL ENVIRONMENT

We review the technical environment we used for our analysis. In particular, we discuss the adopted dataset, libraries, and detection models.

4.1 The nuScenes Dataset

Recent years have seen the release of several sophisticated datasets which have played important roles in the advancement of 3D object detectors in autonomous driving. Some important examples are the KITTI [8], Waymo [29], and nuScenes [2] datasets. For our work, we used nuScenes, as it is the one that was used by the original works on PKL [23] and OCM [3].

nuScenes [2] was released in 2019 as a multimodal dataset for the task of training and evaluating perception systems for autonomous driving. The dataset has received particular acclaim for the range of different sensor types supported in the dataset. The main contributions of the nuScenes dataset are its large volumes and complexities of data, with 360-degrees sensor coverage on a real driving scenario.

nuScenes comprises 1000 driving scenes of 20 seconds each, acquired in Boston and Singapore, under a wide array of situations and environmental conditions. The 1000 scenes have been manually selected to contain relevant traffic scenarios (i.e., high-density traffic, rare object classes, potentially dangerous scenarios, etc.). Highly accurate annotations are provided with every keyframe, sampled at 2Hz. The 23 object classes are each annotated by their semantic category, bounding boxes, and attributes that include visibility, activity, and pose. In addition, the dataset includes all intermediate, non-annotated frames [2]. The frequency of objects of the *car* class is on average 20 per keyframe.

For the experimental work presented in this paper, the *nuScenes devkit* [21] is utilized extensively. The devkit implements an API for parsing and loading data from nuScenes, and functionality related to object detection. Among those, it provides functionality to perform metric evaluations of predictions produced by object detector inference over the nuScenes dataset [2]. When evaluating detector predictions, the devkit utilizes the 2D center distance on the ground plane as a match criterion.

4.2 The MMDetection3D Toolbox

MMDetection3D [19] is a part of the open-source object detection toolbox MMDetection [4], implementing a large set of detection methods and components related to 3D object detection. MMDetection provides weights for over 200 pre-trained object detection models ready to be applied. Furthermore, the Python module provides compatibility with multiple datasets, including nuScenes.

MMDetection3D provides specific integration with the nuScenes dataset and devkit, which makes it particularly suited for our work. Furthermore, its components are known to provide good results in terms of performance, speed, and memory usage when compared with other frameworks [4].

In the experimental work presented in Section 6, two pre-trained models with weights from the MMDetection3D *model zoo* [4] are utilized, described in the following.

Pointpillars with FPN backbone. PointPillars [15] was proposed by Lang et al., as a novel encoder that learns features from vertical columns (pillars) of point clouds resulting from LIDAR scans. The PointPillars architecture consists of three stages: a feature encoder network, to transform a 3D point cloud into a pseudo-image; a 2D convolutional backbone, for extracting high-level features from the pseudo-image; and a detection head for BB classification and regression. Compared to previous point-cloud models and fusion-based models for 3D object detection, the PointPillars encoder enables faster inference [9].

In our experimental work we used the pre-trained PointPillars model with an FPN [16] backbone. This specific model will be denoted as PPT in the remainder of the paper.

SSN with RegNet Backbone. Zhu et al. [33] proposed Shape-Signature Networks (SSNs), a framework that incorporate shape information in the model, to discriminate between object classes. The proposed shape signature is designed to be compact and effective, for the purpose of high inference speed. Furthermore, the shape signature is designed to be robust to sparsity and noise in the point clouds, and thus to guide the learning of discriminative features during training. Zhu et al. [33] demonstrate that SSN-based models achieve state-of-the-art results and that the proposed shape signature achieves good scalability across various backbone networks.

The specific, pre-trained SSN model applied in this paper employs a RegNetX-400MF-SECFPN backbone and neck, based on the architectures RegNetX [26], SECOND [32], and FPN [16]. This model will be simply denoted as *SSN* in the remainder of this paper.

5 METHODOLOGY

We now discuss the methodology we adopted for analyzing the metrics introduced in Section 3.1 and Section 3.2. Experiments and results are presented in Section 6 later.

5.1 Alignment of Metrics

The summarizing metric defined in the OCM [3], AP_{crit}, is computed in a similar fashion to AP, which means that detectors are evaluated over multiple confidence thresholds. In contrast, PKL evaluates detector predictions on a single confidence threshold (τ). This difference would make the comparison sensitive to the specific threshold applied to PKL.

In [10], Guo et al. propose computing PKL (and mAP) using an "optimal confidence threshold", namely the threshold that achieves the highest F1-score, to perform a fair comparison between submissions scored in the nuScenes detection [2] task. Still, this solution is not appropriate for our objectives, because the comparison of metrics would be influenced by the behavior of the detector itself, thus making it difficult to interpret the results.

To enable a comparison of PKL with the OCM on fixed sets of BB predictions, we focus on the more detailed metrics proposed in the OCM, namely reliability-weighted precision ($P_{\mathcal{R}}$) and safety-weighted recall ($R_{\mathcal{S}}$). This allows for evaluating the OCM at a set of BB predictions at a pre-defined threshold τ.

To summarize the OCM "variants" of precision and recall into a single indicator, we decided to utilize a variation of F1-score, with reliability-weighted precision and safety-weighted recall as components. The resulting metric will be denoted $F1_{crit}$, defined as follows:

$$F1_{crit} = 2\frac{P_{\mathcal{R}} \cdot R_{\mathcal{S}}}{P_{\mathcal{R}} + R_{\mathcal{S}}}. \quad (1)$$

Another important factor to consider when comparing results with the aforementioned metrics is their relation with object classes. The $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ metrics proposed in [3] are based on precision and recall, and they are typically evaluated for each object class present in the dataset separately. In contrast, PKL evaluates to a single value that considers all the objects present in the scene, for a single scene (nuScenes sample). To compare the two metrics we only take into consideration a single semantic class, namely cars; this is achieved by filtering on the "car" class of the nuScenes dataset, for each sample evaluated. As a consequence, nuScenes samples in which no cars are present are excluded.

Table 1: Custom datasets of metric data over randomly selected nuScenes samples.

Name	Injected Errors	τ	Samples
RAW_40	None	0.40	894
FP_40	False Positive	0.40	923
FN_40	False Negative	0.40	739

5.2 Datasets of Metrics Values

The experiments introduced later in Section 6 involve quantitative analysis of metric evaluations over single nuScenes samples. To have a common baseline for the comparison of the two evaluation models (PKL and OCM), we created three custom datasets based on nuScenes. The datasets include the values of the target metrics (PKL, $P_{\mathcal{R}}$, and $R_{\mathcal{S}}$), calculated on the BBs predicted by the SSN detector (see Section 4.2), on randomly selected samples from nuScenes.

The first dataset, named RAW_40, was created by randomly selecting 1000 samples from the nuScenes dataset, including samples from both the training and the validation sets. We first ran the SSN object detector on each sample (i.e., scene), obtaining a set of predicted BBs. As discussed before, a threshold τ needs to be set to discriminate between TP and FP predictions. We used $\tau = 0.40$ for our dataset (hence the suffix "_40" in its name), because it was shown to produce the best values for PKL [10]. Evaluation was then performed, producing metric data for the task-oriented metrics PKL, $P_{\mathcal{R}}$, and $R_{\mathcal{S}}$. Note that computing the PKL metric involves executing a path planner model on each sample, which is particularly intensive in terms of time and resources.

Besides the base dataset with nominal values, we also created two datasets in which synthetic detection errors have been *injected* in the set of predicted BBs. The idea behind those additional datasets is to evaluate the impact of detection errors on the target metrics. The two datasets, FP_40 and FN_40, were created following a procedure similar to the one described above, but with an intermediate step in which false positive or false negative detections have been injected, respectively. The procedure for performing such injection is detailed in the next section.

The three datasets are summarized in Table 1, along with the number of nuScenes samples contained in each of them. For each dataset, we also report the kind of injected misdetection and the value of the threshold that was used.

Note that, even though we sampled 1000 scenes from the nuScenes dataset, some samples have been excluded because they did not match our criteria, hence the varying sizes in Table 1. As previously discussed, one main condition for exclusion is the absence of objects labeled as "car" in the GTs of the sample. Other conditions are more related to the implementation; for example, samples that receive exactly 0.0 as PKL value are excluded, because comparison is performed using the logarithm of PKL, which is undefined for zero. Furthermore, for injecting FNs we remove objects in a specific

region of the scene (see next section); therefore we excluded from FN_40 those samples that did not have any objects in that region that could be removed.

5.3 Fault Injection Procedure

The creation of FP_40 and FN_40 involved a *fault injection* [11] step, in which the predictions produced by the SSN detector were modified with synthetic detection mistakes. Different strategies exist for fault injection [20]; our approach modifies the output of the object detector (injection of interface errors [20]), thus emulating its faulty behavior. Further details on the procedure are explained in the following. The code we used to inject faults in nuScenes prediction is available in the repository of this project [28].

Injection of FPs. Injecting a FP means *adding* a bounding box in the list of BBs produced by the detector. By controlling where to place the erroneously detected object, it is possible to emulate specific contextually-relevant situations.

Our injection procedure supports a series of parameters describing the properties of the object to be injected. The main properties include: *position*, *size*, *velocity*, and *orientation* (in three dimensions) of the object to be injected, and its *confidence score*. Other properties can also be set, including its *category* (e.g., 'car') and *attributes* (e.g., 'vehicle.moving'). Furthermore, a parameter decides the *number of objects* that will be injected in a sample.

To create the FP_40 dataset mentioned above, we ran the injection procedure on each of the 1000 randomly selected nuScenes samples. All the injected objects are assigned to the 'car' *category*, and their *orientation* and *vertical position* (z-coordinate) are set to the same as ego. The *confidence score* is set to 0.99, to make sure that the object is not filtered out due to a low confidence.

The size and position of the object are selected randomly. Considering a 3D coordinate system in which ego is placed at the origin, $(0, 0, z)$, the injected object has *coordinates* (I_x, I_y, z), with $I_x \in [-5, 5]$ meters from ego, and $I_y \in [-10, 30]$ meters from ego. The *size* of the object is (I_w, I_l, I_h), where $I_w \in [1.5, 3.5]$, $I_l \in [2.0, 6.0]$, and $I_h \in [1.5, 3.0]$ meters. The parameters I_x, I_y, I_w, I_l, and I_h are all drawn from their respective uniform probability distributions, in the intervals specified above. The *velocity* of the injected FP is set either as 0, or as equal to the ego velocity, each selected with 50% probability. This choice also influences whether the object would get the 'vehicle.moving' or 'vehicle.stopped' attribute. Finally, the *number of objects* injected per sample is also selected randomly, in the interval $[0, 3]$.

Injection of FNs. Injecting a FN means *removing* a bounding box from those correctly predicted by the object detector. While removing an object is conceptually simpler, there is however limited flexibility, because what can be done depends on what objects are present in the scenario.

The procedure for injecting a FN is based on two parameters: a distance d, and a probability p. First, all the objects with Euclidean distance from ego smaller than d are collected (i.e., those within a circle of radius d from ego). Then, each of them is tested for removal against probability p; if an object is removed, the procedure stops.

To create the FN_40 dataset mentioned above, the distance d was drawn randomly from a uniform distribution such that $d \in [10, 40]$, while the removal probability is set to $p = 1/4$. Similarly as for FPs, the *number of objects* to be removed per sample is also selected randomly, in the interval $[0, 3]$.

6 EXPERIMENTS AND RESULTS

In this section we validate and compare PKL and the OCM metrics. We first discuss how individual samples have been evaluated qualitatively (Section 6.1), and then we analyze the distribution of metrics on our datasets (Section 6.2). In Section 6.3 we discuss the impact of scenario size, and in Section 6.4 we analyze their statistical correlation. We then go into more details on their sensitivity to hazardous detection errors (Section 6.5), and finally we discuss their relation with traditional metrics (Section 6.6). The source code we used in the experiments, and the obtained results, are available in the repository of this project [28].

6.1 Qualitative Analysis of Individual Samples

Before proceeding to a deeper comparison between the metrics, we quantitatively analyzed their behavior on selected nuScenes samples. We first analyzed the metrics values in the nominal setup, and then after the injection of detection errors. Such analysis allowed us to have a first understanding of the two metrics, and to craft the subsequent experiments.

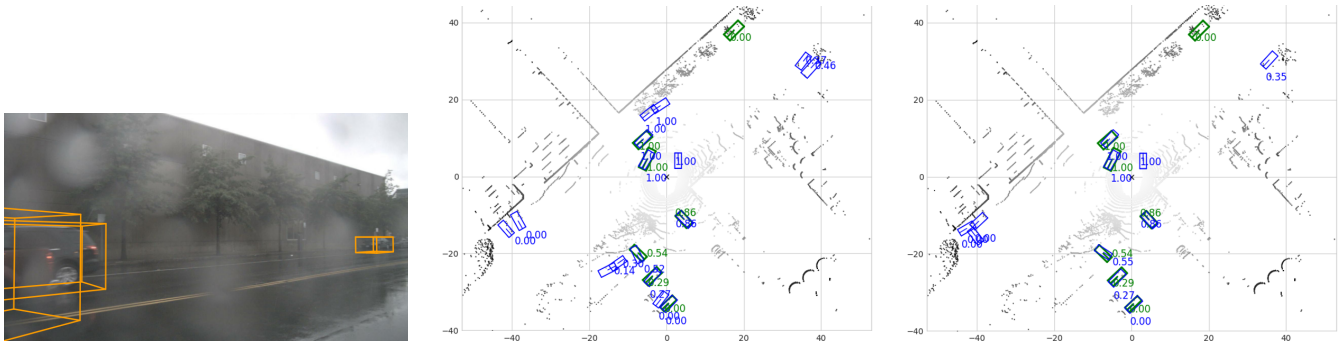
While it is not possible to report all the details here, we discuss the analysis of a representative sample, to show how the evaluation was performed. Figure 1 provides an overview of nuScenes sample with *sample token* `6c8d4379e83646d08436f6ec92b35fe5`. In the given scenario, the ego vehicle executes a right turn at an intersection while two cars approach from the right and pass in front of it. One of the vehicles makes a left turn into the same street from which the ego vehicle came, while the other vehicle proceeds straight ahead. The two aforementioned vehicles clearly have a distinctive role in the presented scenario.

Figure 1b and Figure 1c show the predictions of the PPT and SSN detectors, respectively, along with the GT objects. In the figures, the axes are measured in meters, and the coordinate system is aligned with the reference frame of the ego vehicle. Predicted objects are visualized in blue and GT objects in green; the number annotated next to the BB depicts the criticality according to the OCM. Furthermore, both figures include an injected FP, at position $(3, 5)$, which has criticality 1.0.

From the figures, we observe that the two vehicles in proximity of ego are detected by both detectors, and that they receive criticality of 1.0 according to the OCM. Note that the two detectors have different behaviors, which results in different criticality values for some objects, especially distant ones. Also, PPT (Figure 1b) detects two FPs with very high criticality, just in front of the ego vehicle.

We computed the target metrics for the analyzed sample, both in the base scenario and in two scenarios with one injected fault, one FP and one FN, respectively. Results are reported in Table 2.

We first observe that PKL and the metrics from the OCM rank the two detectors differently: SSN is ranked best by both $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ (and consequently by $F1_{crit}$), while PPT is ranked best by PKL (recall that for PKL a lower score is preferred). In fact, PKL indicates an almost perfect detection for PPT, and some more consistent



(a) Front camera frame

(b) Predictions made by the PPT detector, with an injected FP at position $(3, 5)$.

(c) Predictions made by the SSN detector, with an injected FP at position $(3, 5)$.

Figure 1: Overview of sample `6c8...fe5` of the nuScenes dataset, and example of the qualitative analysis we performed.

Table 2: Metric values computed on sample `6c8...fe5`, in different scenarios. For each scenario, the best values of each metric is highlighted in bold.

	Base Scenario		*Injected FP*		*Injected FN*	
Metric	PPT	SSN	PPT	SSN	PPT	SSN
$P_{\mathcal{R}}$	0.533	**0.915**	0.466	**0.733**	0.455	**0.880**
$R_{\mathcal{S}}$	0.989	**0.998**	0.989	**0.998**	0.717	**0.720**
$F1_{crit}$	0.693	**0.955**	0.633	**0.845**	0.556	**0.799**
PKL	**0.677**	3.673	20.74	**5.169**	4.50	**3.31**

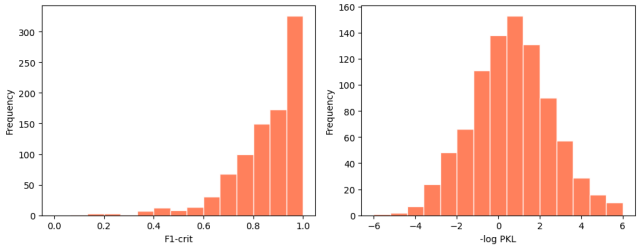


Figure 2: Distribution of $F1_{crit}$ (left) and $-\log(\text{PKL})$ (right) on the RAW_40 dataset.

deviation for SSN. Note also that $P_{\mathcal{R}}$ is very low for PPT, due to the two FPs being detected almost in front of ego (see Figure 1b).

When injecting the FP, there is a reduction in the values of $F1_{crit}$ and PKL for both detectors, as expected. For $F1_{crit}$, the decrease is caused by $P_{\mathcal{R}}$ (reliability score), which decreases sharply as expected; $R_{\mathcal{S}}$ instead remains unchanged. Interestingly, the PKL score for PPT sharply worsens to 20.74. This is probably due to the combined effect of the injected FP with the two pre-existing ones, which causes a big deviation of the planned trajectory. Instead, the PKL value for SSN experiences only a minor increase, to 5.17, and thus, after injecting the FP, the SSN turns out as the best detector also according to PKL.

For the injection of a FN, the correct prediction of the car closest to ego was removed. This is the vehicle just at the front left of ego, turning left at the intersection. We observe that OCM-related metrics are impacted similarly on the two detectors by the introduction of a FN. Also, both detectors are evaluated approximately equally in terms of $R_{\mathcal{S}}$. $F1_{crit}$ is substantially lower for PPT than for SSN; this is due to the strong difference in $P_{\mathcal{R}}$, which is in turn caused by the two FPs close to ego. Note that while $R_{\mathcal{S}}$ is not influenced by injecting FPs, the definition of $P_{\mathcal{R}}$ also depends on FNs. This explains the change observed in $P_{\mathcal{R}}$ for both detectors, compared to the base scenario.

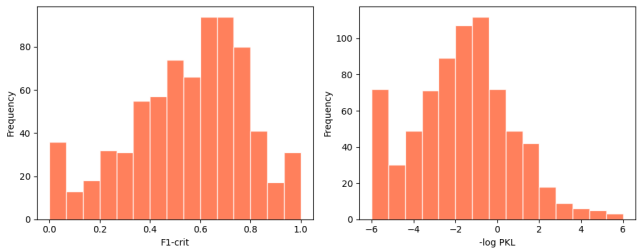
For PKL, a small increase is seen for PPT after the injection of the FN. Interestingly, for SSN, the PKL value is instead slightly *lower* (i.e., better) in the scenario where a FN has been injected. This means that the path computed from the set of objects in which the car closest to ego is removed is actually closer to the GT than the one computed based on the full set of objects predicted by SSN.

6.2 Metrics Distribution

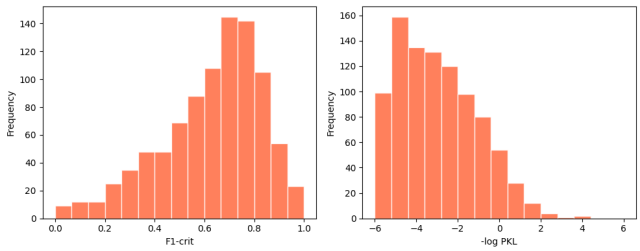
Figure 2 shows the distribution of $F1_{crit}$ (left) and $-\log(\text{PKL})$ (right) on the RAW_40 dataset. Note that for PKL the plot shows the *negative logarithm* of the score, because PKL $\in [0, \infty)$. Also, note that for PKL < 1.0 the negative logarithm evaluates to positive values. Thus, for the right-hand histogram in Figure 2, values larger than 0 (in the negative logarithm) actually represent base PKL scores smaller than 1.0, i.e., good scores. This particular characteristic of PKL should be considered when interpreting the results.

Even taking that into account, a dissimilarity between the distributions of metric results can be observed. $F1_{crit}$ rates more than one third of the samples with the highest score (1.0), but then the number of samples rapidly decreases for lower $F1_{crit}$ values. This means that it is easy to distinguish "safe" scenarios, from scenarios where context-relevant misdetections occurred. Note that to have $F1_{crit} = 1.0$ it is necessary that both $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ are equal to 1.0.

Figure 3 shows the distribution of the same metrics on the datasets FN_40, FP_40 in which synthetic detection errors have been injected. The first observation is that both metrics are heavily affected by the injected faults, which, we recall, are injected close to the ego vehicle. They are therefore measuring, to some extent, how "safe" the predictions of the underlying detector are. We also



(a) Distributions on the FN_40 dataset



(b) Distributions on the FP_40 dataset

Figure 3: Distribution of $F1_{crit}$ (left) and $-\log(\text{PKL})$ (right) on the FN_40, FP_40 dataset, which include the injected detection errors.

note that PKL seems to be more heavily affected by FP injections, while $F1_{crit}$ is more heavily affected by FN injections.

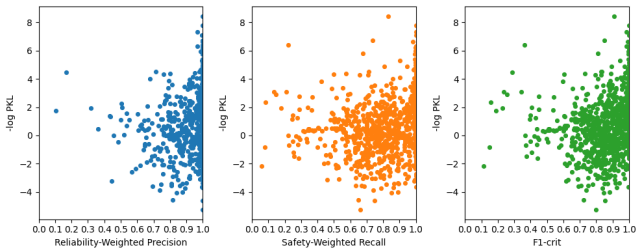
To further analyze the relations between the OCM metrics and PKL, in Figure 4 we plot their joint distributions on the samples from our datasets. Each row of plots refers to a different dataset, while each column refers to a different OCM metric. For all the plots, the y axis reports the $-\log(\text{PKL})$ value associated with a sample, while the x axis reports $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, or $F1_{crit}$, respectively.

When observing the distributions on the base dataset (RAW_40, Figure 4a) we note very little relation between the OCM metrics and PKL. In particular, samples that achieve optimal values on one metric (i.e., 0.0 for PKL and 1.0 for $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, and $F1_{crit}$), may assume very disparate values on the metric(s) from the other model. We visually observe that the $R_{\mathcal{S}}$ penalizes missed detections (low values of safety-weighted recall in Figure 4b, central figure), while the $P_{\mathcal{R}}$ penalizes false positives (low values of reliability-weighted precision in Figure 4b, left figure). This difference is not evidently captured, instead, by using PKL only.

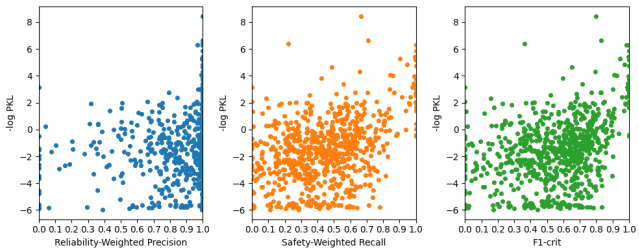
It is possible to notice a better alignment between the metrics on the FN_40 and FP_40 datasets. In particular, the most visible agreement, albeit still barely visible, is on the FN_40 dataset, between $R_{\mathcal{S}}$ and $-\log(\text{PKL})$ (middle plot in Figure 4b). In fact, it can at least be noticed that samples with high $R_{\mathcal{S}}$ values also exhibit a good PKL score.

6.3 Sensitivity to Scenario Size

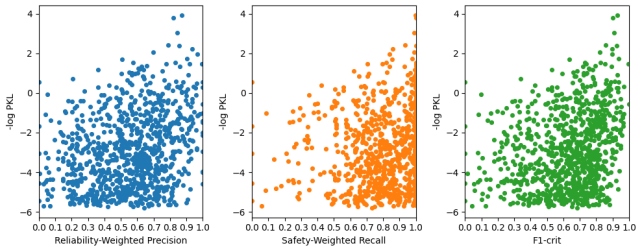
Previous work on PKL [10] found that PKL scores rapidly deteriorate when the number of cars in the scenario increases. Based on this finding, we decided to compare the OCM and PKL metrics on modified versions of our datasets, in which only samples that have



(a) Metrics joint distributions over the RAW_40 dataset



(b) Metrics joint distributions over the FN_40 dataset



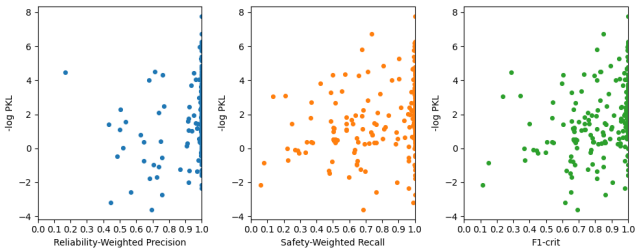
(c) Metrics joint distributions over the FP_40 dataset

Figure 4: Joint distribution between the OCM metrics and PKL on the different datasets.

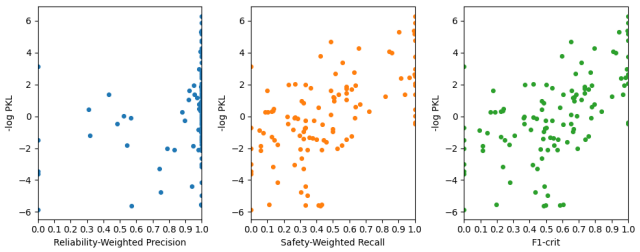
N objects or less in their GT set are retained. In the following, we refer to those datasets as *constrained* datasets.

The joint distribution of metrics on the constrained datasets is shown in Figure 5. Considering the FN_40 dataset, there indeed seems to be more agreement between the metrics as the number of objects in the evaluated scene decreases (Figure 5b). Fewer data points with low values of PKL and high values of $F1_{crit}$ can in fact be observed, when compared to Figure 4. Furthermore, there appears now to be an evident linear trend between $R_{\mathcal{S}}$ and $-\log(\text{PKL})$, when FNs are injected (middle plot in Figure 5b). Interestingly, a similar but weaker trend can be observed between $P_{\mathcal{R}}$ and $-\log(\text{PKL})$, when instead FPs are injected (left-hand plot in Figure 5c). Such symmetry can be explained by the fact that the PKL score is affected by both FNs and FPs, while $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ are mainly affected by only one of the two kinds of detection errors.

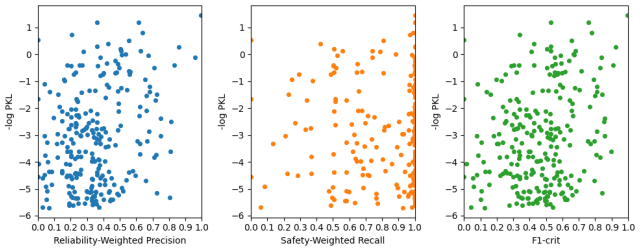
Viewed in the context of all the visualized joint distributions, safety-weighted recall ($R_{\mathcal{S}}$) appears to be more correlated with PKL than reliability-weighted precision ($P_{\mathcal{R}}$). $F1_{crit}$ also exhibits some degree of correlation with PKL on all the datasets, when they are constrained for $N = 4$. Note however that a linear trend between



(a) Metrics joint distributions over the constrained RAW_40 dataset ($N = 4$)



(b) Metrics joint distributions over the constrained FN_40 dataset ($N = 4$)



(c) Metrics joint distributions over the constrained FP_40 dataset ($N = 4$)

Figure 5: Joint distribution between the OCM metrics and PKL on the *constrained* datasets where the number of objects in the scenario is limited to 4.

R_S and $-\log(\text{PKL})$ implies a non-linear trend with respect to the real PKL value.

6.4 Statistical Correlation

To obtain statistical evidence of the correlation between PKL and OCM metrics, the Spearman [25] and Pearson [1] coefficient are computed; namely *the Pearson product-moment correlation coefficient (r_P)* and *Spearman's rank order correlation coefficient (r_S)*.

The Pearson r_P gives an indication of the strength of the linear relationship between two random variables. In contrast, the Spearman r_S indicates the strength of a monotonic relationship between the variables (i.e., it does not assume linearity). Both of the coefficients range in the interval $[-1, 1]$, with the strength of the association between the variables being indicated by the absolute value, and the direction being indicated by the sign. A value of 0 indicates no association between the variables (i.e., no correlation).

Table 3 reports the r_P and r_S coefficients computed on PKL and $F1_{crit}$, on different versions of our datasets, both constrained (with $N = 4$ and $N = 8$) and unconstrained (we indicate it with $N = \infty$).

Table 3: Correlation coefficients between PKL and $F1_{crit}$ when limiting the number of objects in the scenario (N).

Dataset	N	r_P	r_S	p_P	p_S	*Size*
RAW_40	∞	0.207	0.318	4.094e-10	1.872e-22	894
	8	0.280	0.365	1.505e-15	1.616e-9	447
	4	0.343	0.408	1.392e-7	2.258e-10	224
FN_40	∞	0.346	0.327	3.112e-22	6.774e-20	739
	8	0.503	0.503	5.382e-21	6.177e-21	305
	4	0.577	0.598	1.820e-11	2.163e-12	114
FP_40	∞	0.242	0.251	8.163e-14	1.089e-14	923
	8	0.290	0.268	1.119e-10	2.720e-9	476
	4	0.230	0.198	0.240e-3	0.169e-2	250

The *p-value* for statistical significance is also reported in the table, indicating the probability of observing test values at least as extreme as the observed values under the *null hypothesis*. In other words, the p-value is an indication of the likelihood that the observed results could be occurring by chance.

In the table, p_P and p_S refer to the p-values for Pearson and Spearman coefficients, respectively, which indicate the probability of observing values with the same properties under the *null hypothesis*; p-values below 0.001 are typically regarded as indication of statistical significance [14]. Finally, *Size* refers to the size of the constrained datasets at the corresponding value of N.

As shown in the table, the coefficients always indicate a positive correlation between PKL and $F1_{crit}$. Furthermore, the corresponding p-values always stay sufficiently low to indicate statistically significant results, except for the FP_40 dataset when constrained to $N = 4$. Note that, despite being smaller, the other two datasets constrained to $N = 4$ achieve a much lower p-value (i.e., stronger significance).

The coefficients in Table 3 also confirm what was observed, informally, from the plots of Figure 5: i) the strongest correlation is observed when injecting FN detections (FN_40 dataset), and ii) the correlation between the metrics progressively improves when reducing the number of objects in the considered samples. The latter does not however apply for FP injections, for which a weak correlation is observed even with fewer objects.

The highest correlation is observed on the dataset with injected FNs, when limiting to 4 or less ground truth objects. Both r_P and r_S are in fact close to 0.6, with p-values lower than 10^{-10}, which indicate a consistent positive correlation between the two metrics.

6.5 Sensitivity to Hazardous Detection Errors

In this section, we analyze the impact of detection mistakes on the two metrics models. We progressively inject detection errors on the same nuScenes samples, using the injection method described in Section 5.3. The method injects misdetections of objects that are close to the ego vehicle, which are therefore hazardous detection errors. For computational reasons, we used only 100 randomly selected nuScenes samples for this experiment. For each of them, we injected FPs and FNs, separately, an incremental number of times, from 1 to 5, and we computed the target metric for each of those modified scenarios.

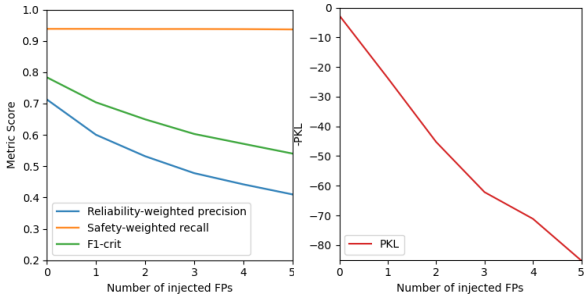


Figure 6: Mean values of $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$ (left), and PKL (right), over 100 samples for 0–5 injected FPs.

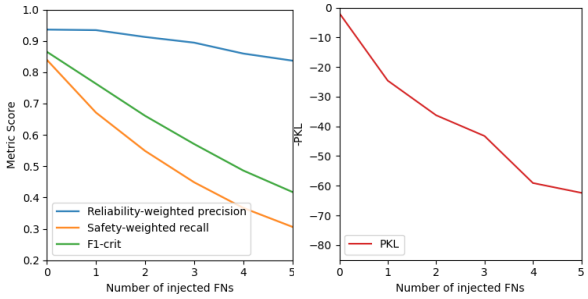


Figure 7: Mean values of $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$ (left), and PKL (right), over 100 samples for 0–5 injected FNs.

Figure 6 and Figure 7 show the mean of the target metrics on the 100 samples, at varying the number (and type) of injected faults. For visualization purposes, the negated values of PKL are displayed in the plot; recall in fact that an optimal PKL value is 0.0, and it increases otherwise.

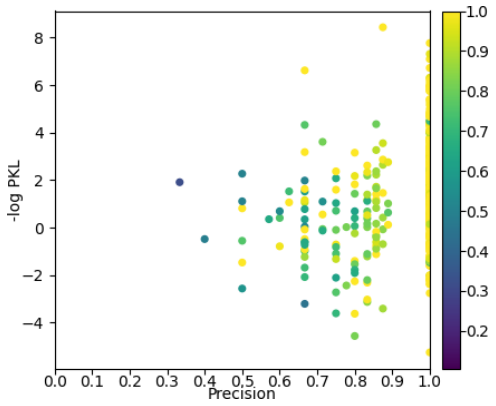
We notice that both models correctly penalize the hazardous misdetections, although with some differences. As noticed also in the earlier analyses, PKL is more strongly impacted by FP injections, while the opposite holds for $F1_{crit}$. By its construction, with the OCM it is possible to distinguish between FPs and FNs, by looking at the individual $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ components; different detection errors are instead indistinguishable with PKL.

The two OCM components have a slightly different behavior: safety-weighted recall ($R_{\mathcal{S}}$) remains constant when an increasing number of FPs are injected (Figure 6), while reliability-weighted precision ($P_{\mathcal{R}}$) is somehow affected by FNs (Figure 7).

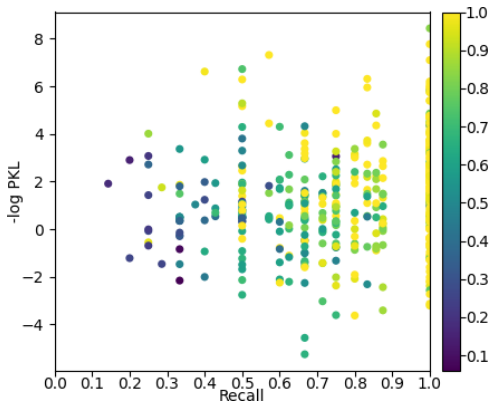
Also, PKL seems to continue penalizing multiple errors in the same way, while $F1_{crit}$ has a sharper decrease in the beginning and tends to stabilize for a higher number of injected misdetections. However, this same behavior of PKL is responsible for the increased dependence of PKL on the number of objects in the scenario, as discussed in [10].

6.6 Relation to Traditional Metrics

We conclude our analysis by discussing the relation between the PKL, the OCM, and the traditional Precision and Recall metrics. The plots in Figure 8 visualize such relation on the RAW_40 dataset, constrained to $N = 8$.



(a) x = Precision, $y = -\log(\text{PKL})$, $z = P_{\mathcal{R}}$



(b) x = Recall, $y = -\log(\text{PKL})$, $z = R_{\mathcal{S}}$

Figure 8: Relation between PKL, OCM metrics, and traditional metrics (Precision and Recall) on the constrained RAW_40 dataset, with $N = 8$.

In Figure 8a, each nuScenes sample in the dataset is positioned according to the Precision (x-axis) and $-\log(\text{PKL})$ score (y-axis) obtained by SSN, and it is colored according to $P_{\mathcal{R}}$. Figure 8a shows instead Recall and $R_{\mathcal{S}}$, following a similar organization.

While some samples get a similar score for generic metrics and safety-oriented ones, most samples are evaluated differently by these metrics. Furthermore, it is evident that each of the three metrics models provide a different perspective. In Figure 8b, samples with a good PKL score (i.e., $-\log(\text{PKL}) \geq 0$) are assigned very disparate values of Precision and Recall and of $R_{\mathcal{S}}$. A similar behavior can be observed in Figure 8a for Precision and $P_{\mathcal{R}}$.

Samples with good $R_{\mathcal{S}}$ have often also a good Recall value, which follows from the definition of the metric. The opposite is instead not true, as there are samples with low Recall but very high $R_{\mathcal{S}}$. A similar behavior can be observed between Precision and $P_{\mathcal{R}}$. This behavior is due to the focus of $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ on critical objects, meaning that low-criticality objects are less reflected in the OCM than in their generic counterparts.

Finally, we note that Precision and Recall align on specific values, producing some definite "vertical bands" in Figure 8, which does not happen with their OCM counterparts (e.g., refer to Figure 5). This is due to the fact that Precision and Recall are, by definition, constrained to a discrete set of values, which furthermore depends

on the number of objects in the scenario. For example, in a scenario with 4 GT objects, both Precision and Recall can only assume values in the set $\{0, 0.25, 0.5, 0.75, 1\}$. The same is not true for $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$, which instead may assume continuous values in the entire interval $[0, 1]$, independently on the number of objects in the scenario.

7 CONCLUSION

In this paper, we elaborated on a recent line of thought for which object detectors in safety-critical domains should prioritize the detection of objects that are most likely to interfere with the decisions of the autonomous actor. Very recent works in the literature proposed metrics to evaluate object detectors following this concept. Especially, we focused on two metrics models for object detectors, PKL and OCM, that are intended for the autonomous driving domain and whose code has been released publicly.

We evaluated PKL and OCM to understand to what extent correct detections and misdetections may impact safety and reliability, and how this can be captured by the two metrics. We conclude that both metrics serve as an indicator of the safety of an object detector, but they also provide different perspectives. Concisely, OCM is able to distinguish between a safety issue and a reliability issue, while PKL does not make this distinction. All the software used in this paper is organized in a library, and released for reproducibility [28] and to encourage the usage of PKL and OCM.

As future work we aim at defining a training procedure of object detectors to maximize PKL or OCM. Intuitively, the object detector is intended to reward the detection of objects that are relevant (close and in colliding trajectories), and it is expected instead to be far less effective in the detection of objects that are not relevant for the tasks. Practically, this can be realized by a proper training phase, where the usual loss measurement approach is modified according to the principles and measures established by OCM and PKL. Another direction consists in involving humans in the evaluation, to understand to what extent those metrics actually reflect the perception of a human driver on the safety of specific scenarios.

REFERENCES

[1] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. *Springer Topics in Signal Processing* 2 (2009), 1–4.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. Nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 11618–11628.

[3] Andrea Ceccarelli and Leonardo Montecchi. 2023. Evaluating Object (Mis)Detection From a Safety and Reliability Perspective: Discussion and Measures. *IEEE Access* 11 (5 2023), 44952–44963.

[4] Kai Chen et al. 2019. MMDetection: Open MMLab Detection Toolbox and Benchmark. arXiv 1906.07155. arXiv:1906.07155 [cs.CV]

[5] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. 1–16.

[6] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88 (2010), 303–338.

[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. 2013. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research* 32, 11 (2013), 1231–1237.

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 3354–3361.

[9] Alireza Ghasemieh and Rasha Kashef. 2022. 3D object detection for autonomous driving: Methods, models, sensors, data, and challenges. *Transportation Engineering* 8 (2022).

[10] Yiluan Guo, Holger Caesar, Oscar Beijbom, Jonah Philion, and Sanja Fidler. 2021. The efficacy of Neural Planning Metrics: A meta-analysis of PKL on nuScenes. arXiv 2010.09350. arXiv:2010.09350 [cs.CV]

[11] Mei-Chen Hsueh, T.K. Tsai, and R.K. Iyer. 1997. Fault injection techniques and tools. *Computer* 30, 4 (1997), 75–82. https://doi.org/10.1109/2.585157

[12] Licheng Jiao, Fan Zhang, Fang Liu, Shuyuan Yang, Lingling Li, Zhixi Feng, and Rong Qu. 2019. A survey of deep learning-based object detection. *IEEE Access* 7 (2019), 128837–128868.

[13] Taehyeon Kim, Jaehoon Oh, NakYil Kim, Sangwook Cho, and Se-Young Yun. 2021. Comparing Kullback-Leibler Divergence and Mean Squared Error Loss in Knowledge Distillation. arXiv 2105.08919.

[14] Eugene Komaroff. 2020. Relationships Between p-values and Pearson Correlation Coefficients, Type 1 Errors and Effect Size Errors, Under a True Null Hypothesis. *Journal of Statistical Theory and Practice* 14, 3 (2020).

[15] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2019-June. 12689–12697.

[16] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Vol. 2017-January. 936–944.

[17] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. 2020. Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision* 128, 2 (2020), 261–318.

[18] Maria Lyssenko, Christoph Gladisch, Christian Heinzemann, Matthias Woehrle, and Rudolph Triebel. 2021. From Evaluation to Verification: Towards Task-oriented Relevance Metrics for Pedestrian Detection in Safety-critical Domains. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 38–45. https://doi.org/10.1109/CVPRW53098.2021.00013

[19] MMDetection3D Contributors. [n. d.]. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d (Accessed: December 29, 2023).

[20] Roberto Natella, Domenico Cotroneo, and Henrique S. Madeira. 2016. Assessing Dependability with Software Fault Injection: A Survey. *ACM Comput. Surv.* 48, 3, Article 44 (Feb. 2016), 55 pages. https://doi.org/10.1145/2841425

[21] nuScenes Contributors. [n. d.]. nuScenes devkit. https://github.com/nutonomy/nuscenes-devkit (Accessed: December 29, 2023).

[22] Rafael Padilla, Sergio L Netto, and Eduardo AB Da Silva. 2020. A survey on performance metrics for object-detection algorithms. In *2020 international conference on systems, signals and image processing (IWSSIP)*. IEEE, 237–242.

[23] Jonah Philion, Amlan Kar, and Sanja Fidler. 2020. Learning to Evaluate Perception Models Using Planner-Centric Metrics. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 14052–14061.

[24] Cristiano Premebida, Gledson Melotti, and Alireza Asvadi. 2019. RGB-D object classification for autonomous driving perception. *RGB-D Image Analysis and Processing* (2019), 377–395.

[25] Marie-Therese Puth, Markus Neuhäuser, and Graeme D. Ruxton. 2015. Effective use of Spearman's and Kendall's correlation coefficients for association between two measured traits. *Animal Behaviour* 102 (2015), 77–84.

[26] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Designing network design spaces. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 10425–10433.

[27] Andreas Rønnestad. 2022. Investigation of Safety-Oriented Metrics for Object Detectors. Specialization Project, Department of Computer Science, NTNU.

[28] Andreas Rønnestad, Andrea Ceccarelli, and Leonardo Montecchi. 2023. GitHub Repository with source code and results used in this paper. https://github.com/andreas-roennestad/Evaluation-Of-Safety-Oriented-Metrics-for-Object-Detectors (Accessed: December 29, 2023).

[29] Pei Sun et al. 2020. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2443–2451.

[30] Georg Volk, Jörg Gamerdinger, Alexander Von Betnuth, and Oliver Bringmann. 2020. A Comprehensive Safety Metric to Evaluate Perception in Autonomous Systems. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems, ITSC 2020*.

[31] Mirja Wolf, Luiz R. Douat, and Michael Erz. 2021. Safety-Aware Metric for People Detection. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Vol. 2021-September. 2759–2765.

[32] Yan Yan, Yuxing Mao, and Bo Li. 2018. Second: Sparsely embedded convolutional detection. *Sensors (Switzerland)* 18, 10 (2018).

[33] Xinge Zhu, Yuexin Ma, Tai Wang, Yan Xu, Jianping Shi, and Dahua Lin. 2020. SSN: Shape Signature Networks for Multi-class Object Detection from Point Clouds. In *Computer Vision – ECCV 2020*, Vol. 12370 LNCS. 581–597.