Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Andreas Rønnestad

Evaluation of Safety-Oriented Metrics For Object Detectors

Validating and comparing task-specific evaluation metrics in autonomous driving scenarios

Master's thesis in Computer Science
Supervisor: Leonardo Montecchi
June 2023

NTNU
Norwegian University of
Science and Technology

Andreas Rønnestad

Evaluation of Safety-Oriented Metrics For Object Detectors

Validating and comparing task-specific evaluation metrics in autonomous driving scenarios

Master's thesis in Computer Science
Supervisor: Leonardo Montecchi
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ABSTRACT

Deep learning-based object detection has become a major pillar in safety-critical systems. With this development, solid metrics with which to evaluate the performance of detection models are crucial. For this purpose, widely acknowledged evaluation metrics such as Average Precision (AP) are commonly applied. However, when applied in autonomous vehicles, generic metrics fail to consider context and environmental factors in their evaluations. Specifically, such metrics do not effectively differentiate the relevance of objects perceived in a driving scene, thus failing to accurately reflect the significance of particularly important detections. In recent years, a number of safety-oriented evaluation metrics have been proposed to address these issues. Building on a systematic literature review identifying such approaches, this thesis focuses on the analysis and comparison of two particular metrics for the safety-oriented evaluation of object detectors. These metrics are based on the Object Criticality Model (OCM) and on Planning-KL Divergence (PKL). A rigorous experimental approach is proposed and employed for analysing the characteristics of the two metrics and the relationship between them. First, a qualitative analysis is performed on metric evaluations for detector predictions in a specific driving scenario, with and without injecting synthetic faults into predictions. Subsequently, an analysis of the relationship between OCM-related metrics and PKL is performed, investigating correlation in quantitative metric data. The sensitivity of the metrics to faults is then analysed. Lastly, a comparative analysis is performed, investigating the distinctions between safety-oriented and generic metrics for object detection. The results reveal important characteristics of the two metrics, and of their relationship. While PKL imposes a more severe penalization on false positive (FP) faults compared to false positive (FN) faults, the opposite is true for OCM-related metrics. The qualitative analysis further highlights the sensitivity of PKL to FP predictions. Furthermore, findings demonstrate that OCM-related metrics exhibit a higher sensitivity to reflecting prediction faults as the number of objects in the evaluated scenarios decreases. Again, the opposite is true for PKL. Additionally, examining the correlation between quantitative metric data, a significant increase in correlation between the metrics is reported for a decreasing number of objects in the scenarios evaluated. Finally, significant distinctions between safety-oriented metrics and generic metrics are identified, emphasizing the importance of contextual evaluation when assessing the performance of object detectors in autonomous vehicles. Through the research performed in this work, an extension of the nuScenes-devkit is developed, implementing useful functionality such as methods for the injection of faults into object detector prediction.

SAMMENDRAG

Dyp læring-basert objektdeteksjon har blitt en viktig søyle i sikkerhetskritiske systemer. Med denne utviklingen er pålitelige metrikker for å evaluere ytelsen til deteksjonsmodeller avgjørende. For dette formålet anvendes typisk velkjente, generiske metrikker som Average Precision (AP). Når slike metrikker anvendes i autonome kjøretøy, tar de imidlertid ikke hensyn til miljøfaktorer og kontekst. Spesifikt klarer ikke slike metrikker å skille mellom relevansen til ulike objekter som oppfattes i en kjøresituasjon, og reflekterer dermed ikke viktigheten av enkelte deteksjoner. I de senere årene har det blitt foreslått flere sikkerhetsorienterte evalueringsmetrikker for å takle disse problemene. Basert på en systematisk litteraturgjennomgang som identifiserte slike tilnærminger, fokuserer denne oppgaven på analyse og sammenligning av to spesifikke metrikker for sikkerhetsorientert evaluering av objektdetektorer. Disse metrikkene er basert på Object Criticality Model (OCM) og Planning-KL Divergence (PKL). I denne oppgaven blir en eksperimentell tilnærming foreslått og benyttet til å analysere egenskapene ved de to metrikkene og forholdet mellom dem. Først utføres en kvalitativ analyse av metrikk-evalueringer for prediksjoner fra en objektdetektor for en spesifikk kjøresituasjon, med og uten injiserte, syntetiske feil i prediksjonene. Deretter utføres en analyse av forholdet mellom OCM-relaterte metrikker og PKL, og korrelasjonen mellom deres kvantitative metrikk-data undersøkes. Videre analyseres sensitiviteten til metrikkene for syntetisk injiserte feil. Til slutt gjennomføres en analyse av forskjellene mellom sikkerhetsorienterte og generiske metrikker for objektdeteksjon. Resultatene avslører viktige egenskaper ved de to metrikkene og forholdet mellom dem. Mens PKL straffer falske positive (FP) feil hardere enn falske negative (FN) deteksjoner, er det motsatte tilfelle for OCM-relaterte metrikker. Kvalitativ analyse understreker denne sensitiviteten til PKL for FP-prediksjoner. Videre viser resultater at OCM-relaterte metrikker har en høyere sensitivitet til feil i prediksjoner når antall objekter i de evaluerte situasjonene reduseres. Det motsatte er tilfelle for PKL. I tillegg viser resultatene en betydelig høyere korrelasjon mellom metrikkene når antallet objekter i de evaluerte situasjonene reduseres. Til slutt identifiseres betydelige forskjeller mellom sikkerhetsorienterte og generiske metrikker, som vektlegger viktigheten av kontekstuell evaluering ved vurdering av ytelsen til objektdetektorer som anvendes i autonome kjøretøy. Gjennom arbeidet utført i denne oppgaven utvikles en utvidelse av nuScenes-devkit, som implementerer nyttig funksjonalitet som metoder for injisering av syntetiske feil i objektdetektor-prediksjoner.

PREFACE

This master thesis presents the results of research conducted through the spring of 2023 and builds on a preceding literature review conducted in the fall of 2022 [1]. It is written for the Department of Computer Science (IDI) at the Norwegian University of Science and Technology.

This work focuses on two methods for the safety-oriented evaluation metrics for object detectors, identified through the aforementioned literature review. As readers are not expected to be familiar with these works, relevant background theory is provided in this thesis. Through the research presented in this thesis, a methodology for the analysis and comparison of safety-oriented evaluation metrics for object detectors is proposed. Applying this methodology, experimental results are gathered and scrutinized. All code implemented as a consequence of this research is provided in an open repository linked in Appendix A, with the hope that it may be valuable for future research on the subject.

I would like to thank Leonardo Montecchi for supervising this work, and for useful feedback during our meetings. Additionally, I would like to thank him for providing all relevant material concerning the safety-oriented evaluation metrics proposed in the work [2], a collaborative work in which he participated as a co-author, and for scrutinizing my work and helping compile it into a research paper [3]. At last, I would like to express gratitude to my wonderful peers for my fantastic years at NTNU.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

List of all abbreviations in alphabetic order:

- **API** Application Programming Interface
- **AUC** Area Under the (ROC) Curve
- **BCE** Binary Cross-Entropy
- **CNN** Convolutional Neural Network
- **DL** Deep Learning
- **DNN** Deep Neural Network
- **FPN** Feature Pyramid Network
- **FCNN** Fully Connected Neural Network
- **GPS** Global Positioning System
- **GPU** Graphics Processing Unit
- **GT** Ground Truth
- **IMU** Inertial Measurement Unit
- **LIDAR** Light Detection And Ranging
- **MSE** Mean Squared Error
- **NDS** NuScenes Detection Score
- **NLP** Natural Language Processing
- **NN** Neural Network
- **NTNU** Norwegian University of Science and Technology
- **OCM** Object Criticality Model
- **OD** Object Detection
- **PKL** Planning KL Divergence

- **RADAR** Radio Detection And Ranging
- **ROI** Region Of Interest
- **RPN** Region Proposal Network
- **SAE** Society of Autonomy Engineers
- **SDK** Software Development Kit
- **SSN** Shape Signature Networks
- **VM** Virtual Machine

CHAPTER
ONE

INTRODUCTION

1.1 Motivation

The task of object detection, referring to the prediction of the semantic class and the location of objects represented in data, has seen much innovation within the last decade. Much of the progress within the field can be attributed to the introduction of new and increasingly sophisticated approaches within deep learning-based approaches, made possible by the increasing processing power of modern-day GPUs [4]. Today, many emerging safety-critical technologies rely on object detection as a fundamental part of their perceptual interface to the environment, the most prominent being autonomous vehicles.

When evaluating the performance of object detection models, most accepted metrics rely on the concepts of *precision* and *recall*. The concept of precision is defined as the number of correct predictions made out of all predictions made, and recall as the total number of *ground truth* (GT) objects that are correctly predicted, respectively. The most commonly used metrics, grounding in precision and recall, are the various variations of Average Precision [5]. Utilizing precision and recall as a basis for evaluating the performance of object detectors is often helpful when considering the basic task of *generic object detection*, associated with locating and classifying instances of objects from a number of predefined categories [6]. However, when such models are applied in specialized and safety-critical systems, metrics based on the classical concepts of precision and recall fail to consider the situation in which detections are made. In tasks such as autonomous driving, it is obvious that the failure to detect certain objects in a driving scenario will pose more risk to the safety of the agent, its environment, and its passengers. Furthermore, the incorrect detection of non-existent objects may cause the *planning algorithm* of the system to unnecessarily interrupt the driving task. These differences in the relevance of objects are not reflected in traditional precision- and recall-based evaluation metrics. Thus, the widely accepted and acknowledged evaluation metrics utilized for evaluating object detection models are largely agnostic to the context of the detection scenario.

In recent years, a number of situation-aware metrics have been proposed to evaluate the performance of object detectors with regard to the safety and reliability of the overall system when such models are applied in safety-critical systems. Preceding the work presented in this thesis, a literature review was conducted with the objective of mapping such approaches [1]. In the aforementioned work, five such metrics ([7], [8], [9], [2], [10]) were presented, and their characteristics were mapped and discussed in the context of *safety* and *reliability*. Each of these works introduced novel, task-specific metrics for object detectors applied in autonomous driving.

Moving forward, it is necessary to assess proposed safety-oriented metrics by means of experimental work in order to investigate their effectiveness in evaluating perception models from a safety perspective. The motivation of this thesis and the corresponding experimental work is thus to provide a basis for the comparison and assessment of safety-oriented metrics for object detectors and to motivate further work on establishing solid metrics on which perception models can be evaluated with regard to safety.

1.2 Research Objective

As discussed in the preceding section, the ever-expanding field of autonomous driving emphasizes the requirement for metrics that assess perception in self-driving systems, particularly in terms of the effect of the model's performance on the safety and reliability of the system. In the context of the evaluation of object detectors, the importance of defining the relevance, or *criticality*, of objects in different scenarios of the driving task was introduced.

The experimental work of this thesis analyzes and compares two approaches for establishing safety-oriented metrics for object detectors, namely Planning KL-divergence [10] and the Object Criticality Model [2]. Building on the work from a systematic literature review [1], the research objective of the work culminating in this thesis is to experimentally compare two approaches for the task-specific evaluation of object detection models with regard to safety and reliability. Furthermore, by examining two different approaches for evaluating detection models from a safety and reliability perspective, the objective is to contribute to the research on safety-oriented, task-specific methods for the evaluation of object detectors when applied in autonomous vehicles. To guide the efforts of the experimental work of assessing the two task-specific approaches to evaluating object detectors, three research questions were formulated:

Research Question 1. In what manner do the suggested metrics deviate from conventional, generalized metrics utilized in object detection, as well as from one another?

Research Question 2. Examining quantitative data, is there any measurable correlation between the metric evaluation results for the two approaches examined?

Research Question 3. How do the proposed metrics penalize detections that represent scenarios where the safety or reliability of the system is compromised? Does quantitative metric data exhibit indications that the metrics penalize detections that cause potential safety- or reliability issues differently?

Building on the experimental work elaborated in Chapter 4 and Chapter 6, answers to the stated research questions are discussed in Chapter 7.

1.3 Contributions

Preceding the work culminating in this thesis, a systematic literature review was performed [1], mapping proposed safety-oriented and task-specific evaluation metrics for object detectors when applied in autonomous systems. Considering the recency of the identified publications on this topic, there is limited experimental work performed validating the corresponding, proposed methods for evaluating the safety of detections in such systems. Furthermore, the experimental results identified analysing the characteristics of these metrics, are largely limited to the assessment and validation of single metrics.

To establish a basis for validating approaches for the safety-oriented evaluation of object detectors, it is necessary to map proposed approaches and to analyse their characteristics in relation to one another. This thesis focuses on the analysis and comparison of two specific metrics, aiming to comprehend the implications of their unique characteristics when evaluating detection models employed in safety-critical systems. The experimental work of this thesis thus proposes a methodology for investigating the properties of safety-oriented metrics, both independently and in relation to others. This can motivate and guide future research into the consequences of applying different methods for evaluating object detectors applied in safety-critical systems. Furthermore, the analysis performed in the experimental work of this thesis provides meaningful insights into the attributes of the specific metrics examined.

In addition to the contributions attributed to the methodology and results presented in this thesis, the software implemented to facilitate these can be should be seen as a standalone contribution. Specifically, the extension of the functionality provided in the *nuScenes devkit*[1] provides the means of performing similar evaluations with, and analyses of, the metrics examined. This includes the injection of faults in the predictions of object detection models applied. The codebase for this extended nuScenes-devkit is provided in the open repository linked in Appendix A.

Lastly, the research conducted in this thesis produced a scientific paper [3] that is currently submitted at an international conference within the field of Software Reliability. Further details about this conference are omitted at this stage due to the double-blind peer review process of the conference.

[1]The nuScenes devkit is introduced in Section 3.1, and the extended version developed for the purpose of the experimental work of this thesis is discussed in Section 5.3.

CHAPTER
TWO

BACKGROUND THEORY

2.1 Autonomous vehicles

When discussing autonomous driving systems, it is important to consider and specify the level of automation of the system in question. The Society of Autonomy Engineers (SAE) introduces a classification system based on the level of human intervention and attentiveness required in an autonomous vehicle [11]. The aforementioned system consolidates six levels of driving automation ranging from no automation to full automation. Full automation can be defined as having the vehicle perform all tasks under all conditions, with no need for human attention or interaction. Other levels of automation include driver assistance, partial automation, and conditional automation, all of which require some degree of human supervision and interaction.

For the remained of this thesis, fully autonomous systems will in this context be denoted simply as autonomous vehicles. However, the concepts presented and discussed in this work are applicable to all levels of automation that require a pipeline that employs systems and algorithms for *perception* and *path planning*. In the context of autonomous systems, the autonomous system in focus is often denoted as *ego*, and will be so for the remained of this thesis.

The autonomous system of self-driving vehicles can be divided into the *perception* system and the decision-making system [12]. The perception system is responsible for estimating the position of the vehicle and for creating an internal representation of the environment of the system utilizing data captured from various on-board sensors. Sensors applied often include visual cameras, LIDARs, RADARs, and GPS. In modern approaches to self-driving, perception often relies on DNN-based object detection models. Section 2.4 presents the background theory of such models. An example of a typical architecture for the autonomous system of a self-driving vehicle is depicted in Figure 2.1.1.

In turn, the decision-making system is responsible for navigating the vehicle to its intended location, utilizing the internal representation acquired through perception. The decision-making system typically consists of multiple subsystems. Most important for the scope of this thesis is the path-planning subsystem. The path planning system of an autonomous vehicle is responsible for computing a set of paths (i.e. sequences of poses) given the vehicles' internal representation of the environment, its location, and the rules of traffic [12]. Hence, the path planning

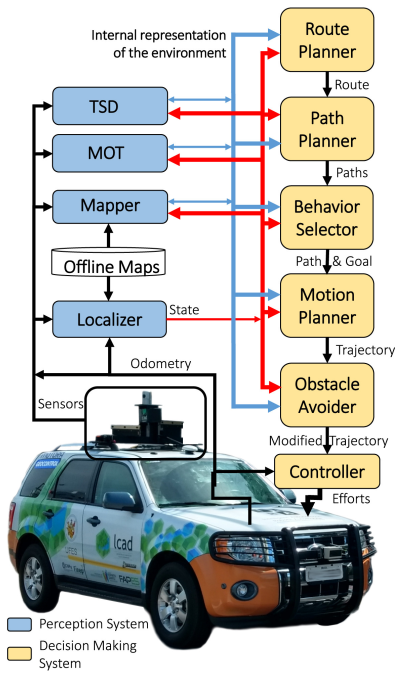


Figure 2.1.1: Typical architecture of the autonomous system of a self-driving vehicle.

Source: [12]

system relies on the environment of the vehicle as perceived by the perception system. This highlights the importance of object detection models (or other perception models) in the autonomous driving pipeline. Many approaches for path planning have been proposed, including techniques based on graph search and curve-interpolation [12]. More recently, approaches utilizing implicitly parameterized neural planners have been proposed [10].

2.2 System safety and reliability

When addressing the implications of applying technology in safety-critical systems, some important concepts and definitions need to be introduced. This allows for characterizing and analysing scenarios in which the system does not meet requirements and consequently does not provide its intended service. Especially important regarding perception in autonomous systems are the concepts of safety and reliability.

The measures of safety and reliability are typically associated with a *system*. A system, as defined by Avižienis et al. [13], refers to an entity that can interact with its environment and other systems. Moreover, a system is designed to implement a function, namely what the system is intended to do to provide a service to a user. Furthermore, we can describe the *behaviour* of the system as the sequence of actions the system executes to implement its function. In this context, the *service* provided can be defined as the system behaviour as perceived by the user.

Considering a system providing a service to a user, a *service failure* (or simply failure) occurs when the service delivered deviates from the intended service. Hence, a failure occurs when the system fails to implement its intended function

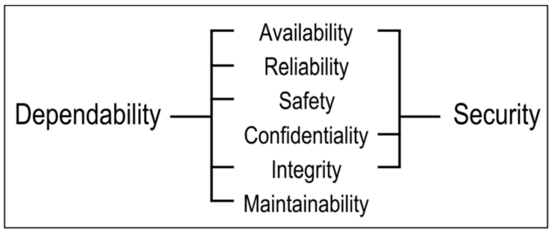


Figure 2.2.1: Attributes of dependability.

Source: [13]

[13]. The adjudged or hypothesized cause of service failure is referred to as a *fault*. Although a fault can cause system failure, this is not always the case. If a fault is not perceived in the *service interface* of the system, namely the part of the frontier between the system and its environment where service delivery takes place, it is denoted as dormant. Conversely, if the fault can be perceived by the user, it is referred to as active.

The ability of a system to avoid service failures that are more frequent and severe than is acceptable is termed the *dependability* of a system [13]. Dependability is a broad definition that encompasses multiple attributes. These attributes are summarized in Figure 2.2.1. For this thesis, the most important concepts within dependability are the concepts of *safety* and *reliability*. Definitions for these concepts are provided below.

- **Reliability**: measure of the *continuity of correct service* in a system.
- **Safety**: the absence of *catastrophic consequences* on users and the environment due to *catastrophic failures*.

Considering object detection in autonomous vehicles, the concepts of safety and reliability can serve as indicators of what can be considered good service. It is evident that the absence of catastrophic consequences and the continuity of correct service are important factors in what determines good driving. Thus, applying these concepts for the evaluation of object detection models applied in such systems can be beneficial in determining how well detectors enable the downstream task of navigation.

2.3 Deep Learning

Deep learning is a subset of the broader category of machine learning and has seen many breakthroughs over the last decade. Much of the recent advances in deep learning can be attributed to the increasing power of modern-day GPUs, allowing for neural networks to have more *trainable parameters* that can be trained efficiently. Furthermore, increasing dataset sizes allow for models to learn more meaningful and accurate representations of the underlying data distributions. Deep learning can be described as the application of deep (multi-layered) neural networks (DNNs) to approximate a function by analyzing a set of *feature vectors* drawn from the underlying distribution. DNNs have demonstrated great capabilities in extracting deep representations of data from feature vectors. These capabilities are especially due to the introduction of convolutional neural networks (CNNs) to tasks where the spatial position of the input data structures is of importance in the approximation of the function represented by the underlying data distribution. Examples of these are image- and video recognition, image classification, natural language processing and, most relevant to this thesis, object detection. Deep learning has allowed object detection to become sophisticated enough to support perception in autonomous vehicles, as well as enabling many other emerging technologies. This section will cover the background theory of deep learning-based methods. First, the theory behind NN architectures will be presented, covering NNs and CNNs in Subsection 2.3.1 and Subsection 2.3.2, respectively. Lastly, the role of data in DNNs will be discussed in Subsection 2.3.3.

2.3.1 Neural networks and Deep Learning

The purpose of a neural network (NN) is, similarly to other learning algorithms, to approximate a function $f(x)$ based on a set of feature vectors $\mathbf{x}$ provided to the network. Neural networks are organized as a set of layers of connected units of computation called *perceptrons*. In an NN with multiple layers, the output of a perceptron serves as the input of similar perceptrons in the subsequent layer of the network, depending on its network architecture [14].

A perception, given a set of inputs forming a vector $\mathbf{X}$, applies a linear transformation to its input and subsequently applies an activation function g to the result of this transformation. The linear transformation applied to $\mathbf{X}$ can be described by Equation 2.1.

$$f(\mathbf{X}) = g(\mathbf{wX} + b) \tag{2.1}$$

In the above equation, $\mathbf{w}$ represents a vector of weights, where weights are associated with the raw inputs to the NN in the case that the perceptron is located in the first layer of the network or with perceptrons in the preceding layer of the NN otherwise. Furthermore, b (the bias) is a term added to the sum, providing an additional degree of freedom to y. Both the weights and the biases of the NN are real numbers. The activation function g is applied to the result of this operation, and its output is called the *activation* of the perceptron. The choice of activation function is made by the data analyst and is usually a fixed, non-linear function. The reason for choosing a non-linear function as the activation of the perceptron is to be able to model non-linear relationships in the underlying data distribution, allowing the NN to approximate non-linear functions [14]. Apart from the non-linearity of activation functions, it is desirable for the function to be differentiable for reasons introduced later in this section. Furthermore, for the stability of training, it is desirable for the activation function to be bounded. Typically utilized are the *sigmoid* function and the *ReLU*, defined in Equation 2.3 and Equation 2.2, respectively.

$$ReLU(x) = \begin{cases} \text{x}, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

When perceptrons are organized into more than two layers excluding output layers, they are referred to as deep neural networks (DNNs). We can define *deep learning* as the class of machine learning algorithms that utilize such architectures. In any NN, the weights and biases corresponding to the nodes of computation represent the *trainable parameters*, and their optimal values are approximated in the process of *training* the network. The process of training the NN is enabled by the *backpropagation algorithm*, first introduced for learning NN-like architectures by Rumelhart, Hinton and Williams in [15]. The backpropagation algorithm is typically the algorithm utilized in updating the trainable parameters (weights and biases) of DNNs. To demonstrate how the backpropagation algorithm enables learning in DNNs by backpropagating derivatives, the critical concept of an *objective function* is first introduced. The objective function is the basis for the optimization criterion of a learning algorithm, and the choice of an objective function is a crucial part of training the parameters of a learning algorithm. An objective function is a mathematical function that quantifies the difference between the predicted output of a machine learning model and the true output, which is often referred to as the *target.* The purpose of training an NN is to minimize or maximize the objective function. Typically used are *loss functions*, namely objective functions that are minimized for the algorithm to gain a closer approximation of the function whose targets are represented by the underlying data distribution. A loss function $\mathcal{L}$ is a differentiable, mathematical function representing a penalty for the misclassification of input feature vectors for a learning algorithm making predictions. It typically takes the predicted output of the network and the target output as input and computes a numeric value that represents the difference between the two. The choice of a loss function depends on the specific problem and the nature of the data, with some commonly used loss functions for the tasks of binary classification and regression, respectively, being cross-entropy loss (BCE) and Mean Squared Error (MSE) (defined in Equation 2.4 and Equation 2.5, respectively) [14].

$$\mathcal{L}(y, \hat{y}) = -(y \ln \hat{y} + (1 - y) \ln(1 - \hat{y})) \tag{2.4}$$

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2.5}$$

In our context, y represents the target value and $\hat{y}$ represents the prediction in the above equations. For BCE, y and $\hat{y}$ are binary labels (0 or 1), and for MSE, y and $\hat{y}$ are vectors of the same length.

Backpropagation involves computing the gradient of $\mathcal{L}$ with respect to each learnable parameter in the neural network. This is done by applying the chain rule of calculus to backpropagate partial derivatives through the layers of the NN until finally differentiating the loss with respect to the input vector. If the loss function accurately represents the penalty associated with misclassification, it can then be minimized by means of a gradient optimization algorithm such as *gradient descent.* Gradient descent is an optimization problem utilized in the backpropagation algorithm to update the values of the learnable parameters of the NN once the gradient has been computed by minimizing the loss function. The weights and biases are updated in accordance with the update rule defined in Equation 2.6. The idea is to update each weight such as to take a "step" in the direction of the descending gradient of the loss function with regards to that parameter. Furthermore, another parameter γ is fixed prior to training. This parameter is called the *learning rate* and reflects how large steps the algorithm will take in the direction of the descending gradient when updating the weights.

$$w_{ij} = w_{ij} - \gamma \nabla_{w_{ij}} \mathcal{L} \tag{2.6}$$

In the above equation, γ is the learning rate, $\mathcal{L}$ is the loss function and w is the single, learnable parameter to be updated.

There are many factors that may affect the efficiency of training in an NN architecture. One such factor is the value chosen for γ. If the learning rate is too low, learning will be inefficient. On the contrary, if the learning rate is too high, loss minimization may be unstable. Other factors such as the choice of the loss function, the type of activation function utilized and the dataset utilized will have significance for the stability of the training process.

2.3.2 Convolutional neural networks

There are many possible ways to model the architecture of a DNN. The most basic architecture is one where each perceptron in every layer shares a weight with each other perceptron in the preceding and the subsequent layer. Such architectures are named fully connected neural networks (FCNNs). Although FCNNs are useful in many applications, they do not provide sufficient efficiency when the number of layers in the network grows. More specifically, the number of learnable parameters grows very quickly as we add new layers to an FCNN, making it very computationally expensive to train the network [14]. Furthermore, in many tasks, the spatial position of the input data structures is of importance for approximating the function represented by the underlying data distribution. If we consider the task of object detection in visual images, it is obvious that the spatial location of a specific pixel is of importance for the detection task. For the task of object detection, it should also be obvious that the most important information for a specific pixel to be processed in the context of its local neighbourhood, rather than in the context of the full image. An NN architecture that is particularly well suited for tasks that involve processing 2D-input data structures, such as the aforementioned, is the convolutional neural network (CNN).

Convolutional neural networks are a class of NN architectures that have *convolutional* layers. CNNs are the most extensively used class of algorithms within deep learning, being applied in several different fields such as computer vision, speech processing, and natural language processing [16]. There are several benefits to applying CNNs over fully connected architectures. The three main benefits of employing a CNN as opposed to an FCNN are, according to Goodfellow et al. [17], parameter-sharing, sparse interactions and equivariant representations [16]. The two most important of these benefits are presented below.

- Parameter-sharing: Employing convolutional layers allows for weight-sharing between perceptrons, reducing the number of learnable parameters. In a CNN, there is a single collection of weights for the whole input, known as the *receptive field*. This is in contrast to a traditional fully-connected layer, in which each element of the weight matrix is used exactly once when computing the output of a layer [17]. This property reduces the computational complexity of the model significantly.

- Sparse-interactions: Sparse connectivity between the input and output features of a convolutional layer refers to the fact that each input unit of the layer does not necessarily interact with each output unit. This is accomplished by having a smaller amount of learnable parameters for a layer than the number of inputs to the layer. This property allows for fewer parameters to be stored in memory, and thus for less computational complexity. Furthermore, it allows for fewer operations to be performed when computing the output of a layer.

A convolutional layer of a CNN typically consists of a collection of *kernels*, or *filters* (terms are used interchangeably), that represent the learnable parameters of the layer. Kernels can be defined as a fixed grid of discrete numbers or values, *kernel weights*, representing the single collection of learnable parameters for the layer [16]. The input data structure of the layer is *convolved* with the kernel, generating feature maps that are passed as inputs to the next layer of the CNN. In the process of this convolution, certain features are extracted from the input data and reflected in the parameters of the kernel. The size of the output data structures and how features are extracted are decided by the manner in which the filter interacts with the input data. More specifically, the size of the output feature vector depends on the size of the filter, its *stride* over the input vector, and an optional *padding* of the input vector. Stride refers to the step size with which the filter "slides" over the input data during convolution. Padding refers to redundant information added to the borders of the input vector, so as to preserve features near the edges of the vectors and obtain a higher dimensional output. Figure 2.3.1 demonstrates the convolution operations of a 2x2 filter on a simple, 4x4 input vector (image).

Another important type of layer in a CNN architecture is the *pooling* layer. A pooling layer performs down-sampling of the input feature vector to a lower dimension, with the purpose of reducing its dimensionality for further processing. This is done to reduce the number of learnable parameters needed in the subsequent layer of the NN, and to reduce the computational complexity of the algorithm. Furthermore, introducing pooling layers to a CNN architecture can increase the statistical efficiency of the network by hinting that the function approximated by the NN is invariant to small changes in the input [17]. Typically, pooling is performed by replacing the input of the layer at a certain location with a summary statistic of the nearby inputs [17]. *Max-pooling* and *average pooling* are common operations to apply to the feature vectors in a pooling layer. In max-pooling, a simple max operation is applied as a summary statistic to a region of the input data. In average pooling, the average of the neighbourhood is the summary statistic utilized.

For the final classification and/or regression task of a CNN, one or more fully connected neural layers are often utilized, with the feature maps representing the extracted features of the initial input in a CNN as inputs to the layers. The purpose of the other layers is thus to extract sufficiently meaningful features for the fully connected layers to appropriately classify their input.

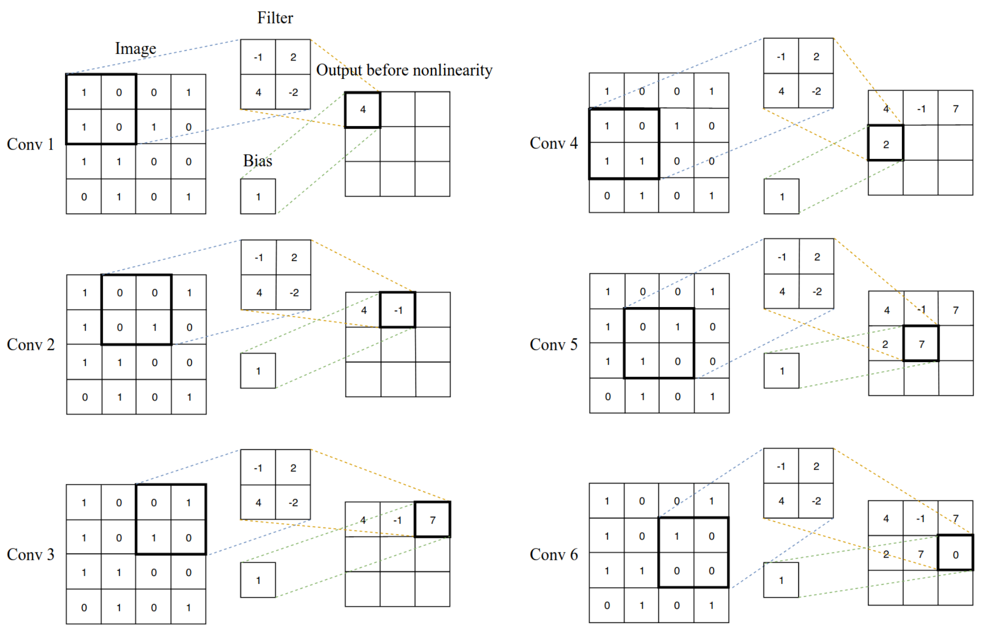


Figure 2.3.1: The convolution of a 2x2 filter with a 4x4 input image.

Source: [14]

2.3.3 The role of data

A DNN is a powerful model for approximating complex functions and relationships by learning from data. However, the accuracy of a DNN is highly dependent on the quality and quantity of the data on which it is trained to be representative of the distribution that they are intended to model. The ability of a learning model to perform well on previously unobserved input is called *generalization*. The more data that is available, and the more representative it is of the underlying function, the better the model generalizes. While the introduction of algorithms such as CNN has enabled data to be utilized more effectively and for better representations to be learned, much progress in DL in the later years can be attributed to gathering and annotating ever larger amounts of data. George E.P. Box stated that "All models are wrong, but some are useful" [18] for statistical methods. In the context of DL models, this statement holds special relevance and highlights the importance of sizeable and representative datasets for better generalization.

To achieve good performance in its predictions, DNNs (especially CNNs) require an extensively large amount of data [16]. If the available data for a specific model is not sufficient for the model to achieve good performance by training a DL model directly, this can be overcome by different methods. An important method, especially when CNNs are involved, is *transfer learning* with *fine-tuning*. Transfer learning allows for data gathered from similar tasks to be utilized when training a model. Especially in the feature extraction stage, the *pre-trained* weights of another DL model can provide meaningful extracted features for the new model, or provide a better starting point for the learnable parameters of the model than randomly initialized weights [16].

The last few years have seen the release of many major models based on DNNs, with models such as DALL-E [19]and GPT-3 [20] containing 12B and 175B learnable parameters, respectively. Although the utilization of multi-GPU high-performance computers allows for such models to be trained, annotating datasets for such sizeable models remains a major bottleneck in building such models. To overcome this problem, *data augmentation* techniques can be applied. Performing augmentations on samples can increase the size of the dataset without changing the labels of the samples. Furthermore, data augmentation can reduce the risk of *overfitting* the model [16]. Data augmentation is extensively applied in visual image- and NLP tasks. Over the last few years, *self-supervised learning* [21] techniques have also been developed to relieve the load of large-scale data annotation.

2.4 Object Detection

The task of object detection can be defined as the task of classifying and locating semantic objects of certain object classes within an input *frame*. The problem of object detection is thus two-fold, consisting of both object classification and object localization in a frame, with the frame being either a 2D visual image or a 3D point cloud.

The output of an object detection model running inference on a frame is a list of *bounding boxes* (BBs), their *labels* and their *confidence levels* [4]. The label output of the object detection algorithm is the predicted semantic class of the object, with possible labels being predefined in the *class set*. The confidence score reflects the confidence of the detection model that the detected object belongs to the predicted semantic class. Bounding boxes can be defined as 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 CNN as a *backbone* for performing feature extraction on the often large input vectors. The purpose of the backbone of the network is to extract meaningful features from the original input [4], reducing the dimensionality of the data as it propagates through the layers of the CNN. Typically, extracted features from the backbone serve as input to a *prediction head*, an architecture in which the last layers are fully connected, outputting the final predicted location and label of detected objects. More specifically, the prediction head receives high-level features from feature extraction and is responsible for performing the final regression and classification tasks. Thus, the prediction head of object detectors can be further divided into the *regressor head* and the *classifier head*, performing these respective tasks. While the specific architectures of object detection models vary, they are typically grouped into two methods, namely *two-stage detectors* and *one-stage detectors*. Two-stage detectors generally provide higher accuracy for localization and classification, whereas one-stage detectors generally achieve higher inference speeds [4]. While high accuracy is desirable, this trade-off between differing architectures is especially important to consider when there are time constraints on the application in which object detection is applied.

2.4.1 Backbone networks

As mentioned, the purpose of the backbone of an object detection model is to function as the feature extractor of the architecture, producing high-level feature maps

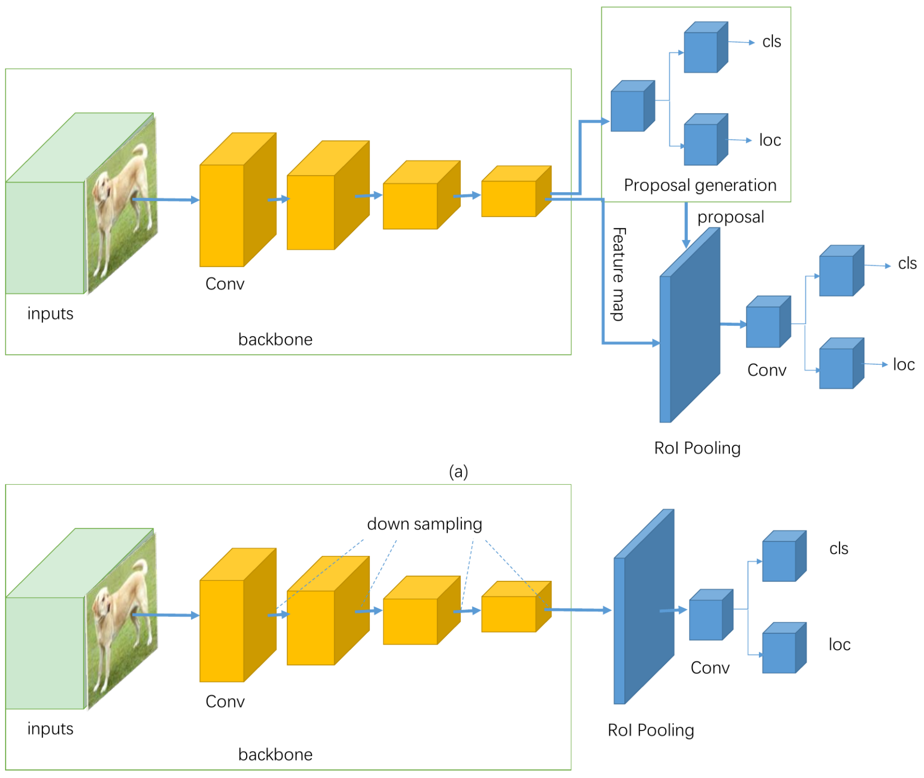


Figure 2.4.1: The basic architectures of one-stage and two-stage detectors. The architecture of two-stage detectors is depicted in the uppermost figure and the architecture of one-stage detectors in the lowermost figure. In the illustration, “cls” and “loc” refer to classification and regression, respectively.

Source: [4]

through a series of convolutional- and pooling layers, on which the classification- and regression tasks can be performed. Towards different applications for object detectors with different requirements, different backbones can be utilized with regard to the trade-off between accuracy and efficiency [4]. Larger backbones with more learnable parameters (more densely connected) are typically applied when the application requires higher degrees of accuracy. Examples of such architectures are ResNet [22], ResNeXt [23] and RegNet-based variations [24], which are all extensively used as backbone architectures. When the application requires more efficiency, more lightweight backbones can be utilized [4].

2.4.2 Two-stage detectors

Two-stage detectors can be distinguished by the two stages in which object detection is performed. The first stage is referred to as *region proposal*, in which candidate BBs, or regions that might contain objects, are proposed. In Faster-RCNN [25], arguably the most representative of two-stage detectors, this task is performed by the *region-proposal network (RPN)* [4], as opposed to the less efficient selective search performed in Fast-RCNN [26]. An RPN takes as input an image of arbitrary size and outputs a series of rectangular region proposals (BBs) with corresponding *objectness scores* (a measure of probability for membership to classes in the class set). In Faster-R-CNN, the RPN shares convolutional layers of the backbone with the Fast R-CNN architecture [26], and region proposals are generated by sliding a network over the feature maps produced by these layers. More specifically, the input of the RPN is a spatial window of the input convolutional feature maps generated by the convolutional layers of the backbone architecture [25]. Subsequently, to generate proposals, regions are mapped to lower dimensional features that are again fed into two sibling FC layers for box-regression and box-classification, respectively. In the second stage of Faster-RCNN, a *RoI-pooling* layer accepts the convolutional features and the candidate BBs from the RPN as input, and performs regression and classification on these ROIs to produce the final predictions [25]. The purpose of the RoI-pooling layer is to perform max-pooling on input ROIs of non-uniform sizes to obtain fixed-size feature maps on which the following regression and classification tasks can be performed. A simplified illustration of the modules of two-stage detectors is depicted in the top half of Figure 2.4.1 and an overview of the Faster-RCNN network in Figure 2.4.2.

2.4.3 One-stage detectors

In contrast to two-stage detectors, one-stage detectors propose predicted BBs from input frames directly without performing a region-proposal step. This property allows one-stage detectors to be more time efficient in their inference, and one-stage detectors are thus often preferred in real-time applications. A simplified illustration of the modules of generic one-stage detectors is shown in the bottom half of Figure 2.4.1.

The YOLO [27] object detection model and its variants (e.g. YOLOv3, YOLOv5) are widely used one-stage detectors. The purpose of YOLO was to propose an OD model that is time-efficient and simple to optimize by not being overly complex (and thus simpler to optimize). This was achieved in YOLO by framing detection

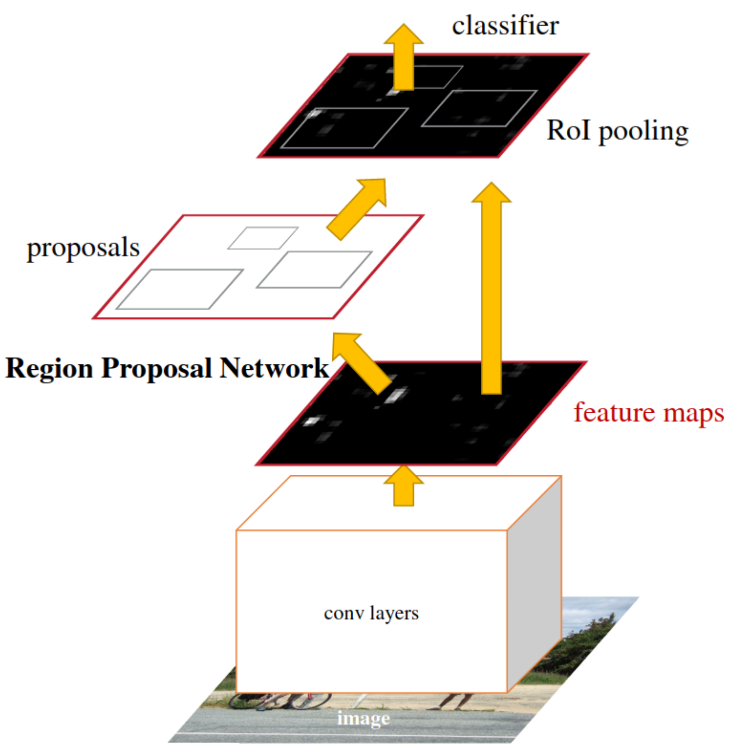


Figure 2.4.2: Overview of the Faster-RCNN two-stage detection network.

Source: [25]

as a regression pipeline, utilizing a unified architecture for extracting features from input images and performing BB regression directly on these feature maps [4]. For the YOLO model proposed by Redmond et al. [27], the feature extraction stage is composed of 24 convolutional layers, pre-trained on the ImageNet 1000-class competition dataset [28], followed by two FC layers. Despite the fast inference time of the YOLO model, omitting the Region Proposal stage performed in two-stage detectors by performing regression directly on feature maps resulting from a feature extraction performed on input frames introduces limitations with one-stage. For instance, performing regression on features extracted directly from the input image, in contrast to refining BBs resulting from a proposal stage, limits the accuracy of detections and the number of candidate BBs evaluated. For instance, the YOLO model introduced in Redmond et al. predicts less than 100 BBs per image as opposed to 2000 region proposals evaluated by the two-stage Fast-RCNN model [4].

2.5 Evaluation Metrics for Generic Object Detection

To assess the correctness of bounding boxes predicted by an object detector, a comparison between predicted BBs and ground truth boxes (GTs) is performed. For this purpose, widely acknowledged evaluation metrics such as Average Precision (AP), first presented in [29], are typically applied. With the object detection

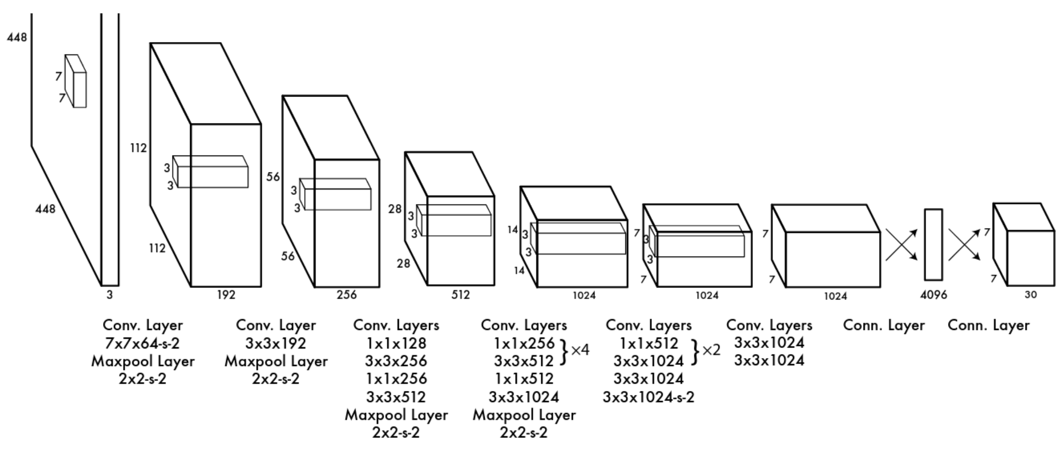

448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 2.4.3: Architecture of the feature extraction stage of YOLO.

Source: [27]

scene containing a variety of challenges, competitions and datasets, a variety of measures for validating the results of object detector inference are applied [5], with little consensus on an optimal evaluation metric for generic object detection. This section presents the background theory of some of the widely acknowledged, traditional performance metrics for ODs.

2.5.1 Precision and recall

As a basis on which to evaluate a set of BBs predicted by an object detection model, BBs are classified as either correct or incorrect following a comparison of predicted BBs with GT boxes. The criteria for this classification are based on the concept of Intersection Over Union (IOU). IOU is based on the Jaccard Index [30], and is within the scope of object detection a measure of the degree of similarity between a predicted BB and a GT. More specifically, IOU is the area of intersection between the boxes divided by the area of their union, as illustrated in Figure 2.5.1. IOU can be described in mathematical terms by the following equation:

$$J(B_p, B_{gt}) = IOU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}, \tag{2.7}$$

where J is the Jaccard similarity coefficient, B_p is the predicted BB, and B_{gt} is the GT box. An IOU of 0 indicates no overlaps between the boxes, whereas a perfectly predicted BB (corresponding to the GT location precisely) would exhibit an IOU of 1.

To classify a predicted BB as correct or incorrect, an IOU threshold is used, representing the minimum IOU for a BB to be considered a correct detection of the object represented by the GT. The choice of an IOU threshold allows for different variations of higher-level metrics such as AP in terms of how restrictive the metric is on considering detections as correct or incorrect.

In the context of evaluating predicted BBs by measuring their degrees of similarity to GT boxes, some important concepts can be defined. First, the detector's output can be labelled as a "positive" prediction to signify the presence of a predicted

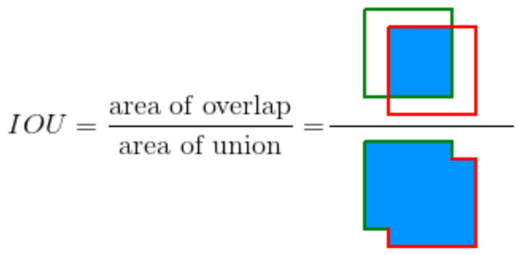


Figure 2.5.1: Intersection Over Union (IOU) illustrated with two-dimensional bounding boxes.

Source: [5]

object, and as a "negative" prediction to indicate its absence. From this, the following definitions can be identified:

- True Positive (TP): The correct detection of a GT box.
- False Positive (FP): An incorrectly predicted BB.
- False Negative (FN): An undetected GT box.

Within the scope of object detection, the concept of a True Negative (TN) does not apply as there is an infinite amount of BB locations and classifications that could apply for the prediction of non-defined objects. Building on these underlying concepts, *precision* is defined as the amount of correctly predicted BBs divided by the total amount of predictions for a set of detections. Furthermore, we define *recall* as the amount of correctly predicted BBs divided by the total amount of GTs. If we consider a dataset with G ground truths and an object detection model that outputs N predicted BBs, of which S are correct (with $S \leq G$), then we can express precision and recall formally as Equation 2.8 and Equation 2.9 [31].

$$P = \frac{\sum_{n=1}^{S} \mathrm{TP}_n}{\sum_{n=1}^{S} \mathrm{TP}_n + \sum_{n=1}^{N-S} \mathrm{FP}_n} \tag{2.8}$$

$$R = \frac{\sum_{n=1}^{S} \mathrm{TP}_n}{\sum_{n=1}^{S} \mathrm{TP}_n + \sum_{n=1}^{G-S} \mathrm{FN}_n} \tag{2.9}$$

In the above equations, P denotes precision and R denotes recall. It is desirable for a model to exhibit high values for both precision and recall, indicating that the detector successfully predicts a reasonable amount of GTs while limiting FP predictions. However, there is a trade-off between the two measures of precision and recall. A model that achieves a significantly high recall score is likely to yield a lower score for precision, and vice versa. A model that produces a low amount of predictions, but with high confidence levels, is likely to receive high precision, but poor recall scores. On the contrary, a model that produces a high amount of predictions may have high recall, but poor performance with regards to precision.

2.5.2 Average precision

As discussed in Section 2.4, the output of an object detection model consists of a predicted bounding box, a predicted class label, and a confidence score. When computing precision and recall for a set of predictions, a confidence threshold τ can be applied to include only detections that have confidence levels higher than τ. As discussed in the preceding section, there is a trade-off between precision and recall for a set of predictions made by an object detector. This trade-off can be demonstrated by applying a confidence threshold to a set of predictions. For instance, setting a low threshold for confidence would increase recall by including more low-confidence predictions, as opposed to setting a high value for τ. More specifically, recall can be defined as a decreasing function of τ, namely $R(\tau)$ [31]. However, although one would expect the opposite to be true for precision, this is not necessarily the case. The reason for this is that if we view both the numerator and the denominator of Equation 2.8 as functions of τ, which they are in the case that such a threshold is applied, they are both decreasing functions τ. Conversely, the numerator of Equation 2.9 is a decreasing function of τ, but the denominator is constant.

To visualize the trade-off between precision and recall for a specific model, the precision×recall curve is utilized. The $P \times R$ curve plots precision against recall for a set of confidence intervals. As discussed in the preceding section, a good detector should exhibit good performance in terms of both precision and recall, with precision remaining high as recall increases [5]. This corresponds to a high area under the $P \times R$ curve (AUC)[1]. Due to the non-monotonic behaviour of precision as a function of τ, the $P \times R$ curve is, itself, non-monotonic and non-continuous due to a discrete sampling of the function values at different confidence thresholds. This poses a problem with for accurate measurement of AUC, which is normally overcome by pre-processing the curve with N-point or all-point interpolation [31]. This results in an interpolated curve defined by a continuous function $P_{interp}(R)$, where R defines the recall level, a real value in $[0, 1]$.

In 11-point interpolation, AUC of the $P \times R$ curve is estimated by averaging the maximum precision values at a set of 11 equally spaced recall levels. Considering a $P \times R$ curve and a set containing 11 equally spaced recall levels, we can compute AP by Equation 2.10.

$$AP_{11} = \frac{1}{11} \sum_{R \in \{0, 0.1, \ldots, 0.9, 1\}} P_{interp}(R) \tag{2.10}$$

where

$$P_{interp}(R) = \max_{\tilde{R}: \tilde{R} \geq R} P(\tilde{R}) \tag{2.11}$$

With 11-point interpolation, AP is thus obtained by considering the maximum value for precision value given by P_{interp} whose recall value is greater than R.

[1]AUC is originally defined as *area under the ROC (receiver operating characteristic) curve*, the term stemming from the field of radar engineering [14].

2.5.3 Variations of Average Precision

As discussed in the preceding sections, different possible thresholds for IOU, as well as different methods for interpolation of the $P \times R$ curve, results in multiple possible variations of AP. The typical way of describing the traits of a specific AP indicator is to suffix the abbreviation with the IOU threshold and state the interpolation method in the subscript. Hence, AP estimated by applying 11-point interpolation under an IOU threshold of 0.50 can be expressed as $AP50_{11}$. Different variations of AP can be applied to indicate the performance of detectors in detecting objects of different sizes. For example, the MS COCO Benchmark applies three variations of the AP metric computed separately to measure the performance of a detector in detecting objects of small, medium and large sizes, respectively [4].

2.5.4 Mean Average Precision

When applying AP to evaluate object detectors, this indicator is estimated separately for each object class. To indicate the performance of the model across object classes, AP can be averaged over all object classes to compute a score more descriptive of the overall performance of the model. Considering a total number of object classes C, with AP-scores estimated for each class, represented by AP_i for the i-th class, we can define mAP as

$$mAP = \frac{1}{C}\sum_{i=1}^{C} AP_i. \tag{2.12}$$

2.5.5 F1-Score

Another common metric for the evaluation of object detector predictions is the F1 score (or F-measure). The F1-score is defined as the *harmonic mean* of the precision and recall of a model for a given confidence threshold [31], as shown in Equation 2.13.

$$F_1 = 2\frac{precision \cdot recall}{precision + recall} \tag{2.13}$$

While AP evaluates the predictions of a detector at multiple confidence thresholds (Equation 2.10), the F1-score simply assesses the correctness of a specific set of predictions.

2.5.6 NuScenes Detection Score

When object detection models are applied in specific domains, such as autonomous driving, datasets may contain metadata that can be utilized in metrics that assess predictions. The NuScenes Detection Score (NDS) [32] is an example of such a metric, and is applied as an alternative evaluation measure in the *nuScenes detection task*. In the nuScenes dataset, metadata is provided for relevant objects in the scene, such as velocity, visibility and pose. NDS is designed to take into account not only detection performance but also the quality of detections in terms of predictions of such metadata. For extensive information about the nuScenes dataset, refer to Section 3.1. For detector predictions on nuScenes data, the different error types are consolidated into the scalar score NDS as

$$NDS = \frac{1}{10}[5\text{mAP} + \sum_{\text{mTP} \in \mathbb{TP}} (1 - \min(1, \text{mTP}))], \tag{2.14}$$

where mAP is mean Average Precision, and $\mathbb{TP}$ is a set of five mean True Positive metrics quantifying the quality of detections in terms of box location, size, orientation, attributes, and velocity. These measures are Average Translation Error (ATE), Average Scale Error (ASE), Average Orientation Error (AOE), Average Velocity Error (AVE), and Average Attribute Error (AAE), respectively, and are further detailed in [32]. For each of the aforementioned metrics, mTP is computed as the average of the metric scores over all object classes ($\mathbb{C}$), namely:

$$\text{mTP} = \frac{1}{|\mathbb{C}|} \sum_{c \in \mathbb{C}} \text{TP}_c. \tag{2.15}$$

NDS is an effort to rank the performance of object detectors with regards to the application in which the models are applied and is an example of a *task-specific metric.*

2.6 Safety-Oriented Evaluation Metrics

While the evaluation measures presented thus far indicate the performance of ODs in terms of their ability to accurately and reliably predict instances of objects, such metrics do not discriminate between the objects in a specific frame based on the scenario. In safety-critical applications such as autonomous driving, this goes against intuition as it is obvious to most that some objects are more critical than others in a driving scenario. Ensuing advances in the application of object detectors in autonomous systems, efforts have been made in proposing task-specific evaluation metrics for assessing the performance of models applied in this domain. In [9], Volk et al. argue that new methods to verify the safety of a perception system are needed as objects detected by such systems can be of different importance, or *criticality*, depending on parameters such as their velocity, distance and potential damage caused by a collision. This need for evaluation metrics that consider the relevance of objects is further expressed by Ceccarelli and Montecchi [2]. In [10], Philion et al. argue that a measure for the performance of ODs applied as part of a perception system in an AV should reflect the impact that predictions made would induce on the downstream task of driving . In the project work preceding this thesis, proposed task-specific and safety-oriented metrics for object detectors were mapped in a systematic literature review [1]. For the experimental work of this thesis, two of these metrics were selected for comparison and analysis. In this section, these two metrics are introduced, and their methods of assigning relevance to objects are investigated.

2.6.1 Planning KL-divergence

In [10], Philion et al. argue that the evaluation and analysis 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, and the best-ranked detection models in terms of performance should thus be the ones that best enable the planner to compute a trajectory as close as possible to the trajectory that would be planned with perfect detections (ground truth detections). The proposed metric, Planning KL-divergence (PKL), is a measure of the KL-divergence [33] between the probability distribution of future positions at different time steps for an autonomous vehicle given the semantic observations (detections) of the detector and the future positions corresponding to ideal observations (represented by GT objects) [10]. As a measure of divergence, a "perfect" detection result will receive a PKL score of 0, corresponding to no divergence between the distributions.

Consider a system of multiple autonomous agents making perceptions (through detection) at a series of time steps, and let x_t^i denote the position of agent $i \in 1...N$ at time t. Furthermore, let o_t^i denote the observation (predictions from detection) for agent i at time t. The perfect observation, corresponding to detecting all GTs correctly, will be denoted o_t^{*i}. Starting from $t = 1$, the system state corresponding to perfect observation by all agents over a set of T time steps can be described by the probability distribution

$$P = p(x_1^1...x_T^N | o_1^{*1}...o_T^{*N}). \tag{2.16}$$

Consider another distribution Q in which the first agent, x_t^1, is considered to be an agent that does not achieve perfect observation. In this context, the KL divergence between these probability distributions can be attained to measure the difference between the system distribution resulting from perfect observation from our agent and the system in which it its observations of the first agent have a degree of noise. That is,

$$D_{KL}(P||Q) \tag{2.17}$$

where P is as defined in Equation 2.16 and

$$Q = p(x_1^1...x_T^N | o_1^1...o_T^1, o_1^{*2}...o_T^{*N}). \tag{2.18}$$

In the above equations, it is assumed that the noise present in the system is only due to noisy perception. With this assumption, Equation 2.17 represents the change in the P distribution given the noisy perception in the first agent. It is assumed that all agents make predictions for the future given their observation only at $t = 1$ (agents do not make any new observations in a time horizon of $Tsteps$). This assumption is reasonable for a sufficiently short time frame. Furthermore, it is assumed that the system moves independently at each time step, given its observations. Under these assumptions, P and Q become products of the marginal distributions over the future of all agents. The joint probabilities can then be factorized, becoming

$$P = \prod_{t=1}^{T} \prod_{i=1}^{N} p(x_t^i \mid o_1^{*i}), \quad Q = \prod_{t=1}^{T} p(x_t^1 \mid o_1^1) \prod_{i=2}^{N} p(x_t^i \mid o_1^{*i}) \tag{2.19}$$

Substituting this into (2.16) and (2.18), and taking the KL-divergence, we get

$$D_{KL}(P \mid\mid Q) = \mathbb{E}_P \left[log \frac{\prod_{t=1}^{T} \prod_{i=1}^{N} p(x_t^i \mid o_1^{*i})}{\prod_{t=1}^{T} p(x_t^1 \mid o_1^1) \prod_{i=2}^{N} p(x_t^i \mid o_1^{*i})} \right] \tag{2.20}$$

$$= \mathbb{E}_P \left[log \frac{\prod_{t=1}^{T} p(x_t^1 \mid o_1^{*1})}{\prod_{t=1}^{T} p(x_t^1 \mid o_1^1)} \right] = D_{KL}(P^1 \mid\mid Q^1) \tag{2.21}$$

Consider a sequence of predicted, semantic objects resulting from the inference of an object detector model. This sequence of predictions (observations) is denoted $s_1, ..., s_t \in S$ and the corresponding sequence of ground truth objects is denoted $o_1^*, ..., o_t^* \in O$. Furthermore, let $A : S \to O$ be an object detection model predicting o_t conditioned on s_t, and $x_1, ..., x_t$ be the sequence of positions corresponding to the sequences of sensory input and detections made. In this context, *Planning KL-divergence* can be defined as

$$PKL(A) = \sum_{0<\Delta\leq T} D_{KL}(p_\theta(x_{t+\Delta} \mid o^*_{\leq t_0}) \mid\mid p_\theta(x_{t+\Delta} \mid A(s_{\leq t_0}))) \tag{2.22}$$

where $p_\theta(x_{t+\Delta} \mid o^*_{\leq t_0})$ represents the distribution of ground truth trajectories in the dataset D,

$$\theta = \arg\min_{\theta'} \sum_{x_t \in D} -\text{log} p_{\theta'}(x_t \mid o^*_{\leq t}). \tag{2.23}$$

Evaluating detectors based on the impact of their predictions on the planned trajectory of the vehicle implicitly ranks the "importance", or criticality, of detecting specific objects in the scene. Thus, with the assumption that the planner applied is sufficiently robust to plan an optimal trajectory with regard to safety, the PKL evaluation can function as an indicator for the safety and reliability of applying a specific model in such a system.

2.6.2 Object Criticality Model

In Ceccarelli and Montecchi [2], the authors argue that in evaluating object detection models applied in autonomous vehicles, generic evaluation metrics do not address the demands of the task to which they are applied. Furthermore, the work argues that an evaluation metric applied in autonomous systems should discriminate based on the circumstance, or context, of the environment of the vehicle. To address these limitations, Ceccarelli and Montecchi propose the *Object Criticality Model (OCM)*. In the OCM, a value representing the importance, or criticality, is assigned to each object in a specific scene by considering important factors in the context. Such a criticality is computed for both the GT objects and the predicted objects. By considering object criticalities, a measure of the performance of an object detector can reflect the importance of each detection. In the OCM, such an evaluation metric is proposed by integrating object criticality values with the generic measures of precision, recall, and average precision.

As discussed above, environmental factors are considered when computing the criticality values for objects in a scene. In the OCM, three such factors are considered, consolidating the final object criticality. The three factors, and thus criticality weights considered are distance, colliding trajectory, and time-to-collision, namely $\kappa_d(B)$, $\kappa_r(B)$, and $\kappa_t(B)$. Here B represents the object for which criticality is computed. It is intuitive that distance from ego is an important factor when evaluating the criticality of an object in a traffic scenario. This is considered in the distance criticality, computed as a function of this distance between ego and B, denoted d_{egoB}. The latter two criticality weights represent the importance of the correct detection of an object regarding its likelihood of collision with the ego vehicle. The process of computing these criticalities thus involves determining the location of a potential point of collision C between the two vehicles. Here, the colliding trajectory criticality reflects the relevance of an object considering the distance between ego and the potential collision point between ego and the object, denoted d_{egoC}. The time-to-collision criticality weight represents the time it would take for B to reach this point of collision, and is computed as a function of the distance d_{BC} between B and C. The two latter criticality weights are computed with regard to the current location, speed and direction of movement for ego and the object.

$$\kappa_x(B) = max(0, -\frac{1}{Z^2}x^2 + 1) \tag{2.24}$$

As mentioned, the location of a potential collision point $C = (C_x, C_y)$ is determined in the process of computing the three criticality weights. Here, C represents the closest point that can be reached between ego and B. As discussed above, the location of a potential collision point between ego and the object considered is computed with regard to the location, speed and direction of movement for the two. To determine the coordinates for C, the velocity vectors of ego and B are utilized. From C, the criticality weights $\kappa_d(B)$, $\kappa_r(B)$, and $\kappa_t(B)$ can be computed from the second degree equation in Equation 2.24. This function is utilized so as to assign a criticality of 1.0 if the input measure, denoted x in the equation, is 0. Furthermore Equation 2.24 will decrease towards the value of 0 as x increases. In Equation 2.24, a constant value $Z > 0$ is the value for which larger values of x causes the criticality computed to evaluate to 0.

Considering the distance criticality, the value it takes on is computed by applying Equation 2.24 with $x = d_{egoB}$. Furthermore, the constant $D_{max} > 0$ is substituted for Z. D_{max} is thus a predetermined constant describing the maximum distance from ego at which objects evaluate to positive values for $\kappa_d(B)$. Thus, the distance criticality will evaluate to 1.0 if the $d_{egoB} = 0$, and decrease with increasing distances.

Similarly, for the collision distance criticality weight, $\kappa_r(B)$ is computed by Equation 2.24, with $x = d_{egoC}$ and $Z = R_{max}$. Here, $R_{max} > 0$ and is defined as the maximum distance from which collision trajectories are considered relevant for ego.

By estimating the time it takes for B to reach the potential collision point C, namely Δt, the collision time criticality weight can be computed. From the velocity vector of the object B, this can be computed. A similar approach as for the preceding criticality weights is then applied to compute $\kappa_t(B)$. Again Equation 2.24 is applied, with $x = \Delta t$ and $Z = T_{max}$, where the latter is a constant value that represents the maximum time-to-collision to consider. This constant is restricted by having $T_{max} > 0$.

$$\kappa = 1 - (1 - \kappa_d) \cdot (1 - \kappa_r) \cdot (1 - \kappa_t). \tag{2.25}$$

The final criticality value of an object B can be calculated using Equation Equation 2.25, which takes into account the three aforementioned criticality weights. Since all three criticality weights can only take on values in the range [0,1], the final criticality value (denoted as κ) is also constrained to this interval. It is worth noting that when any of the criticality weights reach the maximum value of 1.0, indicating high relevance for the object, the final criticality value will also evaluate to 1.0. Moreover, the final criticality value increases as any of the criticality weights used in its computation increases.

In an object detection scene, the aforementioned criticalities of objects can be utilized in the evaluation of object detector predictions. To establish criticality-related metrics for the evaluation of such predictions, the definition of object criticalities is integrated with the existing metrics of precision and recall. Hence, in [2], the definitions of reliability-weighted precision (denoted $P_{\mathcal{R}}$) and safety-weighted recall (denoted $R_{\mathcal{S}}$) are established. The aforementioned measures are established considering the importance of *safely* and *reliably* detecting objects through perception in an autonomous system. The definitions of safety and reliability in a system are introduced in Section 2.2.

$$P_{\mathcal{R}} = \frac{\sum\limits_{B \in TP^*} \kappa(B)}{\sum\limits_{B \in TP^*} \kappa'(B) + \sum\limits_{B \in FP^*} \kappa'(B)} \tag{2.26}$$

$$R_{\mathcal{S}} = \frac{\sum\limits_{B \in TP^*} \kappa'(B)}{\sum\limits_{B \in TP^*} \kappa(B) + \sum\limits_{B \in FN^*} \kappa(B)} \tag{2.27}$$

In the aforementioned equations, the variable $\kappa'(B)$ represents the criticality of a predicted object, while $\kappa(B)$ represents the criticality of a ground truth object. As a result, the reliability-weighted precision (Equation 2.26) takes into account the criticalities of predictions in the denominator and the ground truth criticalities in the numerator. This approach serves to penalize excessively large predicted criticality values, which could potentially compromise the reliability of the driving task. In contrast, the safety-weighted recall (Equation 2.27) places the ground truth criticalities in the denominator and the predicted criticalities in the numerator. Consequently, $R_{\mathcal{S}}$ provides an assessment of the safety of detections by penalizing low predicted values for criticality.

Similarly to the definitions of generic precision and recall, Equation 2.26 and Equation 2.27 define $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$, respectively. However, in contrast to the generic measures of precision and recall, the criticalities of objects are utilized in the above equations. Thus, the relevance of objects as evaluated by the OCM is reflected in the evaluation of a set of detector predictions.

To comprise the measures of reliability-weighted precision and safety-weighted recall into a single metric for evaluation, Ceccarelli and Montecchi [2] utilize the definition of AP to propose a new criticality-related metric, denoted AP_{crit}. This metric evaluates predictions of an object detector at multiple thresholds for confidence (τ), integrating the measures of $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$.

CHAPTER
THREE

EXPERIMENTAL TOOLS AND RESOURCES

In this section, the tools and resources applied in the experimental work of this thesis are presented. Here, Section 3.1 introduces the nuScenes dataset [32] and the corresponding nuScenes detection task and devkit. Subsequently, Section 3.2 introduces the MMDetection3D framework [34], applied for performing object detector inference upon which metric evaluations are made. Furthermore, this section introduces the detection models utilized in this thesis.

3.1 nuScenes dataset

The choice of a dataset is paramount when evaluating and training object detection models. The significance of this choice is further accentuated when these models are applied in safety-critical domains, such as autonomous driving. 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 of such datasets are the *KITTI* [35], *Waymo* [36], and *nuScenes* [32] datasets. In this section, we will cover the nuScenes dataset, which is the detection dataset utilized in the experimental work presented in this thesis.

NuScenes [32] 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 driver of this focus on data acquisition from multiple sensor types stems from the acknowledgement that different sensors are complementary in autonomous driving task. For instance, cross-modality methods for 3D object detection can utilize the advantages of certain sensors to produce better semantic- and depth representations of the environment [37]. Furthermore, Caesar et al. [32] point out the lack of multimodal datasets exhibiting the full range of challenges associated with building autonomous perception systems as one of the points nuScenes seeks to address. Later expansions of the nuScenes dataset have seen the addition of spatial map data and further semantic annotation for a more detailed view of the scenes depicted.
The main contributions of the nuScenes dataset are its large volumes and complexities of data, with 360° sensor coverage across all relevant vision and range sensors including 1x LIDAR, 6x visual cameras (360° view), RADAR, GPS and

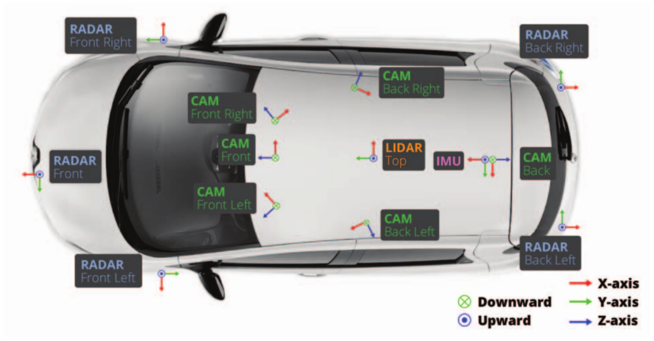


Figure 3.1.1: Sensor setup on vehicle utilized for acquiring *nuScenes* data.

Source: [32]

IMU [32]. The sensor setup on the vehicle applied for data acquisition can be seen in Figure 3.1.1. The dataset comprises 1000 driving scenes of 20s each, acquired in Boston and Singapore, and contains a wide array of scenes, situations and environmental conditions across a diversity of locations. The 1000 scenes were manually selected to contain relevant traffic scenarios (i.e. high density traffic, rare object classes, potentially dangerous scenarios, etc.). For cameras, RADAR and LIDAR, capture frequencies are at 12Hz, 13Hz and 20Hz, respectively. Highly accurate annotations are provided with every *keyframe*, sampled at 2Hz. The 23 object classes represented in the dataset are each annotated by their semantic category, bounding box cuboids (modeled as x, y, z, width, length and yaw angles) and attributes that include visibility, activity and pose. In addition, the dataset includes all intermediate, non-annotated frames [32]. The frequency of objects of the *pedestrian* and *car* classes are 7 and 20 per *keyframe*, on average.

The full nuScenes dataset is split into three parts, namely training, validation and test sets. These subsets consist of 700, 150 and 150 scenes, respectively. Annotations are only provided for the training and validation sets, as the test set is utilized for scoring submissions to the *nuScenes detection task* [32]. Most relevant for this thesis is the nuScenes “trainval” dataset, comprising the training and validation set. Along with the publication of the nuScenes dataset, Caesar et al. [32] provide the nuScenes-devkit, taxonomy, annotator instructions, and database schema, allowing developers to experiment with the dataset with ease.

For the experimental work presented in this thesis, the nuScenes devkit is utilized extensively. The devkit implements an API for parsing and loading data from nuScenes, and development functionality related to object detection. Included in this functionality is the ability to perform metric evaluations of predictions produced by object detector inference over the nuScenes dataset [32]. For this purpose, the nuScenes devkit implements the metrics mAP, NDS, and the various TP metrics applied in Equation 2.14. When evaluating detector predictions, these metrics utilize the 2D center distance on the ground plane as a match criterion (not IOU). For the center distance, different thresholds can be applied to evaluate predictions at different match criteria.

The nuScenes devkit is implemented as a Python 3 [38] module (`nuscenes-devkit`). In addition, the devkit implements functionality for parsing and loading data from nuScenes, and development functionality related to object detection such as visualization, reference frame transformations, and more.

3.2 MMDetection3D Toolbox

MMDetection3D [34] is a part of the open-source object detection toolbox MMDetection [39], implementing a large set of detection methods, modules and components related to 3D object detection. MMDetection provides weights for over 200 pre-trained object detection models that can be applied “out of the box”. Furthermore, the corresponding Python3 [38] module provides benchmarks for, and compatibility with, multiple datasets, including nuScenes.

In the experimental work presented in Chapter 4 and Chapter 6, pre-trained detection models from the MMDetection3D toolbox are utilized. Hence, the weights provided in the MMDetection3D *model zoo* [39] are utilized to perform inference with the selected models. Furthermore, in the process of performing inference with the aforementioned models over samples from the nuScenes dataset [32], the `mmdetection3d` Python3 module is applied. The choice of utilizing the MMDetecion3D framework for the experimental work of this thesis was based on three factors. Firstly, the toolbox provides a large amount of pre-trained models that can be applied with ease and benchmarks for these models on multiple datasets, including nuScenes. Secondly, the components of MMDetection3D provide good results with regard to performance, speed and memory usage when compared with other frameworks [39], which is beneficial when performing inference with multiple models for multiple configurations of nuScenes samples and detection scenarios. Thirdly, MMDetection3D provides specific integration with the nuScenes dataset and devkit, the dataset and codebase utilized in the experimental work presented in this thesis, respectively. In the sections, the specific object detection models applied in the experimental work of this thesis are introduced.

3.2.1 Pointpillars with FPN backbone

For the purpose of performing object detection on point clouds resulting from LIDAR scans, Lang et al. [40] propose PointPillars, a novel encoder that learns features from vertical columns (pillars) of point clouds to predict 3D bounding boxes corresponding to perceived objects. This approach is an attempt at end-to-end learning by encoding a point cloud into a format appropriate for a downstream detection pipeline [40]. The PointPillars architecture consists of three stages: a feature encoder network to transform a 3D point cloud into a sparse pseudo-image, a 2D convolutional backbone for extracting high-level features from the pseudo-image, and a detection head for bounding box classification and regression. Compared to previous point-cloud models and fusion-based models for 3D object detection, applying the PointPillars encoder to transform point-clouds into pseudo-image representations enables faster (real-time) inference [41]. In addition, the Pointpillars approach performs well in terms of accuracy.

For the experimental work performed in this thesis, the pre-trained PointPillars model applied utilizes an FPN [42] backbone. This specific model applied in the subsequent experimental work will be denoted *PPT* for the remainder of this thesis.

3.2.2 SSN with RegNet Backbone

For the task of multiclass, 3D object detection, Zhu et al. [43] propose a framework that utilizes shape information to discriminate between object classes, thus benefiting the distinguishing of specific classes in such a task. This framework utilizes two distinct ingredients for this purpose, namely a shape signature objective and a number of shape-aware grouping heads introduced into the architecture. 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 evaluated, and thus to guide the learning of discriminative features during training. Shape-aware grouping heads are applied to adapt DNNs to the concept of learning through shape information. These are represented as multiple prediction heads where objects of similar shape and scale share weights. This is based on the idea that objects of larger scale should have designated deeper networks for their prediction. The networks that utilize these two key ingredients to achieve effective, multi-class, 3D object detection are denoted Shape-Signature Networks (SSNs). The work of Zhu et al. [43] 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 thesis employs a RegNetX-400MF-SECFPN backbone and neck, based on the architectures RegNetX [24], SECOND [44], and FPN [42]. This model will simply be denoted *SSN* for the remainder of this thesis.

CHAPTER
FOUR

METHOD

In this chapter, the methodology applied in the experimental work to analyze two safety-oriented evaluation metrics for object detectors is described. Applying this methodology, PKL [32] and OCM-related metrics [2] were analysed on object detector predictions over the nuScenes dataset [32].

Firstly, this chapter establishes the foundation for comparing and analyzing these metrics, outlining various methods employed in generating the datasets utilized for quantitative analysis and in aligning the metrics for comparison. Subsequently, specific experimental methodologies are provided. To begin, an overview of the methodology utilized for the qualitative analysis of metric evaluations on predictions made by two detectors for a single sample from the nuScenes dataset is provided. The details of this methodology can be found in Section 4.4. Furthermore, the metric data distributions and their correlation across different traffic scenarios and in the presence of erroneous detections were investigated by applying the methodology introduced in Section 4.5. The experimental approach applied to analyze the impact of faults on metric evaluations is presented in Section 4.6. This methodology encompasses an analysis of the effects of injecting faults into detector predictions, covering a diverse range of threats to system reliability and safety. Finally, the methodology applied in the comparative analysis of PKL and OCM-related metrics with their generic counterparts is described in Section 4.7. This methodology serves to identify and explore the differences between these metrics in the context of object detection.

4.1 Aligning metrics

To enable a comparison of PKL with OCM-related metrics on specific sets of BB-predictions for the samples evaluated (in contrast to evaluating them threshold-independently), the lower-level metrics proposed in the OCM, namely reliability-weighted precision ($P_{\mathcal{R}}$) and safety-weighted recall ($R_{\mathcal{S}}$) were in focus. This allowed for evaluating the OCM on a set of BB predictions filtered on a pre-defined confidence threshold τ. Hence, throughout the experimental methodologies introduced in this chapter, predicted BBs were filtered at a value τ prior to metric evaluation.

To comprise the OCM variations of precision and recall into a single performance indicator for detection at a specific τ, the F1-score was applied together with reliability-weighted precision and safety-weighted recall. The resulting metric is denoted $F1_{crit}$.

$$F1_{crit} = 2\frac{P_{\mathcal{R}} \cdot R_{\mathcal{S}}}{P_{\mathcal{R}} + R_{\mathcal{S}}} \tag{4.1}$$

As the high-level metric of the OCM [2], AP_{crit}, is computed in a similar fashion to AP, detectors are evaluated at multiple confidence thresholds for an evaluation of the AUC when AP_{crit} is computed. In contrast, PKL evaluates detector predictions at a specific confidence threshold (τ). These qualities make a comparison between PKL and AP_{crit} highly sensitive to the specific threshold applied to filter detections before evaluating with PKL. In [45], Guo et al. propose the use of an "optimal confidence threshold" for which to evaluate PKL for the purpose of performing a fair comparison between submissions scored in the nuScenes detection [32] task with regards to both mAP and PKL. This optimal confidence threshold is the threshold at which detector predictions exhibit the highest F1-score. While a similar method would allow for a more fair comparison between PKL and AP_{crit}, the examination of the relationship between a threshold-dependent and a threshold-independent metric is susceptible to a level of ambiguity due to the sensitivity of single-threshold metrics to confidence thresholds. This ambiguity can hinder the interpretability of the findings. Furthermore, the computation of an optimal confidence threshold would require evaluation with PKL to be performed at a sequence of pre-defined thresholds, and would thus require a significant amount of computational power. This was not considered feasible considering the limited time frame of this work.

Another important factor to consider when comparing metric results with the aforementioned metrics is their methods of evaluation across object classes. The $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ metrics proposed by Ceccarelli and Montecchi [2] are based on precision and recall, and are thus evaluated for each object class present in the GT sets of samples separately. In contrast, PKL evaluates to a single value which takes into consideration all objects present for a sample in the respective GT- and prediction sets. For the purpose of the comparative analyses proposed in this thesis, it was thus beneficial to consider only one object class in the subsequently elaborated experimental work. To evaluate both metrics, only taking into consideration a single semantic class of objects, the only objects considered in this experimental work were cars. This was achieved by filtering on the "car" class of the nuScenes dataset for each sample evaluated. In the experimental results, nuScenes samples in which no cars are present (i.e. no basis for evaluation) are discarded.

Alias	**Injected Fault Class**	τ	**Size**
RAW_40	None	0.40	894
FP_40	False Positive	0.40	923
FN_40	False Negative	0.40	739

Table 4.2.1: Datasets of metric data over randomly selected samples.

4.2 The datasets

In the experimental work introduced in Section 4.5 and Section 4.7, quantitative analyses of metric evaluations over single nuScenes samples were performed. These experiments required the construction of sufficiently large datasets of metric results for meaningful insights into the relationship of the metrics to be attained. For a dataset to be sufficiently sizeable to be considered representative of the underlying data distribution, it should comprise as large a number of samples as possible. However, given the limited time frame of this work, a viable number of samples were selected as part of this experimental methodology. Hence, for the main quantitative experiments introduced in Section 4.5 and Section 4.7, 1000 samples were selected at random from the nuScenes "trainval" dataset [32], comprising samples from both the training and validation sets[1]. From predictions of SSN over the selected samples, three datasets of metric evaluations were created at confidence threshold $\tau = 0.40$. These datasets comprised metric evaluations of raw detector predictions over the samples selected, as well as evaluations of the same predictions modified by FN and FP injection, respectively. Recall that the injection of an FP entails inserting a bounding box into the prediction set, whereas the injection of an FN entails removing a correctly predicted box from the prediction set. For the selected nuScenes samples, evaluation was first performed, producing metric data for the task-oriented metrics PKL, $P_{\mathcal{R}}$, $R_{\mathcal{S}}$. Subsequently, $F1_{crit}$ scores were computed from the latter two. Furthermore, evaluation was performed with the generic metrics of precision, recall, and F1-score.

To generate the metric data of modified predictions containing faults (i.e. FPs or FNs), an approach utilizing the implemented functionality introduced in Subsection 5.3.3 was employed, generating datasets of evaluations over predictions in which FPs or FNs had been injected. The specific methodology applied in generating the fault-injected predictions utilized in this approach is elaborated in Section 4.3.

Recall that in creating the datasets, only cars were considered when evaluating predictions. Hence, prior to the metric evaluation, all predictions and GT objects were filtered on the object class "car". The datasets generated are summarized in Table 4.2.1. In the table, the individual datasets are given an alias describing the type of faults injected in predictions prior to evaluation, and the confidence threshold at which predictions were filtered. In Table 4.2.1, τ thus refers to this confidence threshold. This choice of confidence threshold is elaborated upon and discussed in Subsection 7.1.1. Furthermore, "Size" refers to the size of the dataset

[1]A limitation of this experiment is the processing time requirements for performing an evaluation of samples and storing the results, this is elaborated upon in Subsection 7.1.4.

in terms of the number of samples evaluated (after excluding predictions without a sufficient basis for evaluation for any of the metrics). As discussed, predictions without sufficient basis for performing an evaluation were excluded in the datasets of metric data, hence the varying dataset sizes in Table 4.2.1. An example of this is samples that do not contain instances of cars. For data analysis purposes, samples that receive a PKL evaluation of 0.0 were excluded, as the logarithm of 0 is undefined[2].

[2]The negative logarithm of PKL values was utilized as a means of comparison with OCM metrics in the subsequent analysis.

4.3 Injection of faults

By the methodologies introduced in Section 4.5 and Section 4.7, an analysis of quantitative metric results derived from the *fault-injected* predictions of SSN was performed. By injecting faults into detector predictions, the sensitivity and response of the metrics to faults in detection could be mapped.

To inject faults, i.e. either false negative or false positive predictions, into the predictions of detection models, a method of generating such misdetections was required. For this purpose, methodologies for the automatic injection of FP and FN detections into detector predictions were applied. These methodologies are detailed in this section. To represent a variety of fault scenarios in the datasets resulting from the application of these methodologies, a degree of randomness was introduced when applying the methods for fault injection described in Subsection 5.3.3.

4.3.1 False positives

When injecting FPs into a set of predictions, i.e. inserting an incorrect prediction into the prediction set of a detector prior to evaluation by metrics, a series of parameters describing the properties of the injected "object" must be provided upon instantiation. An overview of these parameters and their values upon instantiation is provided in Subsection 5.3.3.

For the purpose of examining metric evaluations across a range of different scenarios in which an object detector predicts FPs, a degree of randomness was introduced in the injection of such faults. Specifically, the number of FPs inserted, as well as the position, size and velocity of individual FP objects inserted into the prediction set were all determined with different degrees of randomness. The values these parameters could take on for an arbitrary FP injection into a prediction set were each drawn from their respective, uniform probability distributions. These are presented in the following:

Number of FPs: The number n of FPs injected into a given sample was a random integer $n \in \{0, 1, 2, 3\}$. Hence, a minimum of 0 and a maximum of 3 FPs could be inserted into the predictions of a given sample.

Position: For the ego-centric, top-down coordinate frame in which the ego vehicle is positioned at $(x_{ego}, y_{ego}) = (0, 0)$, the coordinates (x_{fp}, y_{fp}) of an injected FP had $x_{fp} \in [-5, 5]$ and $y_{fp} \in [-10, 30]$.

Size: The size of an injected FP was determined by the variables $(height, length, width)$, where $height \in [1.5, 3.0]$, $width \in [2.0, 6.0]$, and $height \in [1.5, 3.5]$.

Velocity: The velocity of the injected FP is set as either 0 or as equal to the ego velocity, each selected with 50% probability.

4.3.2 False negatives

For the injection of FNs into the prediction sets of samples, i.e. removing a correctly predicted bounding box from the prediction set of a detector prior to evaluation by metrics, the number of (potential) FNs to be injected was determined in an analogous fashion to the injection of FPs. Furthermore, utilizing the method for injection described in Subsection 5.3.3, the distance from ego, $dist$, up to which objects were considered for removal, could take on values $dist \in [10, 40]$, drawn from a uniform distribution of real numbers. Moreover, for each sample, the procedure removed all BBs with a Euclidean distance up to $dist$ away from ego with a 25% probability.

4.4 Examining single sample metric evaluations

To initiate an analysis of the OCM-related metrics and PKL, a qualitative analysis was performed on a single sample from the nuScenes dataset [32] and its corresponding SSN predictions and metric evaluations for PKL and OCM-related metrics. The experimental results gathered as a consequence of this analysis are presented in Section 6.1. In this experiment, one sample was selected for analysis. The process in which this sample was selected for analysis consisted of examining single, randomly selected samples from the nuScenes dataset. The selection of this sample was performed by reviewing a selection of 20 randomly drawn samples from the nuScenes "trainval" dataset and selecting a sample that represented a dynamic traffic scenario in which the correct prediction of a number of objects is imperative. This corresponded to selecting a sample where GT objects of high criticality in terms of the OCM were present. For the sample selected, the predictions resulting from inference by the two object detection models introduced in Section 3.2 (PPT and SSN) were evaluated by OCM-related metrics and PKL, and analysis of metric results was performed. Prior to this analysis, $F1_{crit}$ scores were computed, and predictions by both detectors were filtered at a confidence threshold of $\tau = 0.15$. This choice of threshold was made to exclude an unnecessary number of FP predictions present in the prediction set prior to filtering. For the same sample and corresponding predictions selected for analysis, an analysis of the metric evaluations resulting from injecting an FP and an FN into these prediction sets was performed. These injections were performed manually by utilizing the methods described in Subsection 5.3.3. The objective of this experiment was to understand qualitatively how the respective metrics evaluate different traffic scenarios represented in the perception of the system and to assess how misdetections can impact such evaluations. Hence, the aforementioned experiment addresses Research Question 1.

4.5 Metric correlation and analysis

Addressing Research Question 2, a quantitative analysis of metric data was performed to assess the relationship between OCM-related metrics and PKL. Specifically, metric values of SSN predictions over a significant number of samples were subject to analysis. This analysis consisted of metric data visualization and analysis of statistical correlation in the metric data. This was first performed on metric data over a large, randomly selected set of nuScenes samples. Subsequently, the impact of erroneous detections on metric evaluations was investigated by inserting FP and FN detections into SSN predictions over the same, randomly drawn set of nuScenes samples. Moreover, an analysis of the impact the number of objects represented in a sample has on metric evaluations was performed.

In this experiment, metric data distributions from the datasets summarized in Table 4.2.1 were utilized. As mentioned in Table 4.2.1, metric data generated at a single confidence threshold $\tau = 0.40$ was in focus. The choice of analysing data at a single confidence threshold was made to reduce the scope of the analysis performed. In Subsection 7.1.1, the choice of, and reasoning behind the confidence threshold for this experimental work is elaborated upon.

The following describes the general methodology followed for quantitatively analysing the datasets utilized.

1. For datasets RAW_40, FP_40 and FN_40, $F1_{crit}$ was computed from $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$, and the frequency of occurrence for values of PKL and $F1_{crit}$ was visualized. The negative logarithm of PKL results was utilized for the remainder of this analysis to reconcile them with metric scores based on the OCM.

2. The relationships between the variables represented by the metrics were analysed by visualising the joint distributions between single sample evaluations for PKL and OCM-related metrics, respectively. Scatter plots were utilized to plot $F1_{crit}$, $P_{\mathcal{R}}$, and $R_{\mathcal{S}}$ against PKL with the aim of analyzing the behaviour of the joint distributions under different circumstances as represented by the unmodified and fault-injected datasets.

3. By visualizing and examining the distributions between PKL and the OCM-related metrics under different levels of constraint on the number of GT objects in samples considered, the impact of limiting the number of GT objects considered on the metric distributions was examined. This involved excluding samples that contained more than a specific number of GT objects in the datasets examined. In this step, metric distributions were visualized for arbitrary levels of restriction on the number of objects considered.

4. For a decremental number of GT objects considered in samples evaluated, an analysis of statistical correlation coefficients indicating the relationship between the metrics $F1_{crit}$ and PKL was performed. This analysis quantified the observed change in correlation between the metrics for a decreasing total number of objects considered in evaluations. Furthermore, indicators of the statistical significance of the observed correlation coefficient values were computed and assessed.

The following sections will elaborate upon the stepwise methodology presented above by introducing important concepts regarding the overall approach presented.

4.5.1 Step 1 and 2: Analysis of metric distributions

The experimental results corresponding to the application of the subsequent methods are presented Section 6.2 and Subsection 6.3.1.

For the datasets generated by applying the approach described in Section 4.2, histograms were employed as a visualization tool to illustrate the frequencies of occurrence for the metric values of PKL and $F1_{crit}$ across the selected samples. Specifically, the distributions of metric values were visualized for both unmodified prediction sets (RAW_40) and prediction sets that had been subjected to injections (FN_40 and FP_40). This involved grouping the metric values into 15 bins of equal width. The distributions represented by the respective datasets were subsequently subject to analysis.

For the purpose of examining the relationship between OCM-related metrics and PKL, *scatter plots* were employed, visualizing the joint distributions between PKL and $F1_{crit}$, $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$. For the three datasets in focus, data points were plotted with OCM-related metrics on the x-axis and the negative natural logarithm of PKL on the y-axis. The joint distributions between PKL and the measures introduced in the OCM [2] ($P_{\mathcal{R}}$ and $R_{\mathcal{S}}$) were visualised to assess the two components of $F1_{crit}$ individually. This enabled the subsequent analysis of the specific reliability- and safety-related components of the OCM for different scenarios.

While PKL by design provides a value indicating its evaluation of detection performance for single nuScenes samples, the metrics that comprise the OCM were designed to culminate in computing an average over multiple samples. Although the indicators $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ are applicable to any set of predictions paired with GT values, the results of their application to smaller sets of predictions (e.g. the sets corresponding to single nuScenes samples) are to a higher degree sensitive to the number of objects present in the sets. For instance, consider two samples where one contains a single car object and the other sample contains two car objects, all with GT criticalities of 1.0. If a detector successfully predicts the object (i.e. predicts a TP) for the first sample but mischaracterizes its criticality as 0.9, the safety-weighted recall would evaluate to $R_{\mathcal{S}} = 0.9$. Suppose for the sample with two cars, the model predicts two TPs for the cars but correctly predicts the criticality for one while mischaracterizing the other criticality as 0.9, similar to the first sample. In this case, the safety-weighted recall would evaluate to $R_{\mathcal{S}} = 0.95$. Hence, the same degree of error present in two prediction sets can result in different values for recall, depending on the number of TPs in the sample. This dependence on the number of TP predictions can also be inferred from Equation 2.27, and a similar example can be made for $P_{\mathcal{R}}$. This indicates that the evaluations of $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ depend on the total number of TPs in the set. As the PKL metric is implicitly parametrized, the factors considered in the context of detection upon evaluating detector predictions can only be determined through experimental work. Hence, PKL is less interpretable compared to the metrics proposed in the OCM.

Considering the discussion above, analysing the change in metric data distributions when limiting the number of factors considered in the evaluation of object detector predictions is beneficial for understanding the relationship between the metrics. This restricts the number of factors that may be considered in PKL evaluation and promotes the interpretability of its metric evaluations. Furthermore, this enables an analysis of the sensitivity of OCM-related metrics to the number of predictions and GT objects (and thus TP predictions) considered in the evaluation. To examine these changes, constrained versions of the datasets from Section 4.2 were generated by imposing restrictions on the total number of GT objects in the samples. This allowed for analyzing visual changes in the distributions and changes in the degree of statistical correlation between the metrics.

If the number of GT objects in a sample i is o_i, that number can be computed from metric results by

$$o_i = \sum_{n \in \mathbb{S}} TP_n + \sum_{n \in \mathbb{S}} FN_n. \tag{4.2}$$

where $\mathbb{S}$ is the set of metric evaluations over detector predictions on the sample (with a binary definition of TP and FP based on the match criterion specified).

Hence, to constrain a given dataset to the set of samples in which the total number of GTs is less than or equal to an integer number N, the restriction $o_i \leq N$ was imposed on each sample in the datasets summarized in Table 4.2.1. Samples which did not meet this criterion were excluded from the metric results analysed. The result of applying such a constraint on a dataset is a dataset of a smaller than or equal size to the original (where samples that do not meet the criterion $o_i \leq N$ are excluded). With the constrained datasets generated, the aforementioned visualizations were examined for the datasets corresponding to $N = 8$ and $N = 4$, respectively. These choices for N-values on which to visualize constrained datasets were arbitrary reference points for numbers of objects considered, for which trends of the change in the distributions can be comprehended and for which it can be investigated whether there is a change in the level of agreement between the metrics for different restrictions.

4.5.2 Step 3 and 4: Statistical Analysis

As both PKL and $F1_{crit}$ represent a measure of the performance of an object detector by considering context, they either explicitly or implicitly rank the importance of specific detections in a scene. Furthermore, both metrics are influenced to a degree by the likelihood of objects interfering with the trajectory of ego. Thus, a degree of correlation between the respective metric scores is expected under conditions where the evaluation metrics can be aligned. By statistically analyzing the relationship of the metrics under different conditions, the objective was to investigate under which conditions a relationship between the variables exists. The experimental results collected as a result of applying this methodology for statistical analysis of metric correlation are presented in Subsection 6.3.2.

To establish statistical analysis of the change in the joint distributions between PKL and $F1_{crit}$ for decremental values for N, the correlation coefficients of Spearman [46] and Pearson [47], respectively *the Pearson product-moment correlation coefficient (r_P)* and *Spearman's rank order correlation coefficient (r_S)*, were applied. As mentioned in step four of the general methodology presented in this section, these indicators of correlation were examined for different values of N in the experiment. Furthermore, the associated p-values and confidence intervals were analyzed to provide insights into the statistical significance of the findings.

The Pearson r_P gives an indication of the strength of the linear relationship between two random variables drawn from a population that follows a bivariate normal distribution. In contrast, the Spearman r_S indicates the strength of a monotonic relationship between the variables (i.e. does not assume linearity). Both of the coefficients range between -1 and 1, with the strength of a positive association between the variables being indicated by higher values, and the strength of a negative relationship being indicated by lower values. A value of 0 indicates no association between the variables (i.e. no correlation). As the Pearson r_P is best applied when the variables are assumed to follow a bivariate normal distribution, which cannot be assumed for the datasets of metric data presented in Table 4.2.1, both coefficients were reported and analyzed to provide depth to the statistical insights they report. Puth et al. [46] argue that the Pearson coefficient can offer an effective measure of the linear relationship between variables even when the assumption of a bivariate normal distribution is violated. However, to explore both linear and non-linear trends in the data, both measures of correlation are reported in the experimental results. Furthermore, since r_S does not assume any particular distribution for the variables, this allowed for possible pitfalls due to misguided statistical analysis to be avoided.

r_P and r_S can be defined according to Equation 4.3 and Equation 4.4, respectively.

$$r_P = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}} \tag{4.3}$$

The above equation defines the Pearson coefficient when applied to a sample[3], where x_i and y_i are datapoints from evaluation over a dataset, and $\bar{x}$ and $\bar{y}$ are means computed over the respective statistical samples. The Spearman's rank order correlation coefficient is equivalent to the Pearson coefficient performed on the *ranks* of the data rather than on the raw data points [46]. Hence, it can be defined as

$$r_S = \frac{\sum_{i=1}^{N} x_{i,r} y_{i,r}}{\sqrt{\sum_{i=1}^{N} x_{i,r}^2 \sum_{i=1}^{N} y_{i,r}^2}}, \tag{4.4}$$

where r_S represents the strength of association between the two variables.

For the experimental results presented in Subsection 6.3.2, the *p-value* for statistical significance is reported for both correlation coefficients. P-values indicate the probability of observing test values as least as extreme as the observed values under the *null hypothesis*[4]. In other words, the p-value is an indication of the likelihood that the results observed are occurring by chance. For relatively small sample sizes (<500 data points), the use of the p-value is generally not recommended as the main reported measure for statistical significance. However, [48] proves that a strong monotonic relationship exists between p-values and their corresponding correlation coefficients, even for a small sample size (<100 data

[3]In this context, a sample refers to a statistical sample drawn from the underlying distributions from which metric results are drawn.

[4]The null hypothesis is the hypothesis that no relationship exists between the metrics.

points). Nevertheless, confidence intervals are additionally reported for r_P and r_S in the experimental results, computed utilizing the Fisher transformation [49].

4.6 Analysing the impact of misdetections

In this part of the experimental work, an analysis of the OCM-related metrics and PKL was performed with regard to their respective evaluations of scenarios where misdetections (FN or FP predictions) could potentially lead to safety or reliability threats for the system. The objective of this experiment was to determine whether the metrics proposed in the OCM [2] and PKL [10] exhibit similar traits with regard to penalizing faults, and to investigate to which degree the metrics discriminate between two categories of misdetections, namely false positives and false negatives. Furthermore, the experiment subsequently introduced aimed to assess to which degree the aforementioned metrics evaluate the safety and reliability of perception in an autonomous system. Hence, this experiment addresses Research Question 3 of this thesis. Experimental results corresponding to the subsequent methodology are presented in Section 6.4.

To analyze metric results with regard to faults, an analysis of the degree to which the analysed metrics penalize misdetections was performed by examining mean metric scores for an increasing number of faults injected into detector predictions over a set of nuScenes samples. Hence, datasets of evaluations at different increments for the number of faults injected for a specific set of nuScenes samples were required.

To construct the aforementioned datasets, 100 nuScenes samples were randomly drawn from the nuScenes “trainval” dataset. From predictions resulting from inference by SSN over the selected samples, prediction sets were injected with an incremental number of FP and FN predictions. For each increment of the two types of fault, datasets were compiled from metric evaluations over the fault-injected prediction sets, representing evaluations of increasingly modified predictions. This was performed for an incremental number of faults injected, ranging from 0-5 inserted/removed BBs. Prior to evaluating predictions, BBs from the prediction set corresponding to each sample were filtered at different confidence thresholds for FPs and FNs, $\tau_{FP} = 0.15$ and $\tau_{FN} = 0.4$, respectively. The reason for these choices of confidence thresholds is expanded upon at the end of this section.

To inject faults into prediction sets, a similar approach to what is described in Section 4.3 was applied with regard to the randomized properties of the FPs injected. For the injection of FN predictions, the distance up to which TP predictions were considered for removal ($dist$) in the method described in Subsection 5.3.3 is set to 40 meters. This distance was selected to ensure that enough predictions were considered for removal to produce an average number of FN injections similar to the average number of FP injections for the corresponding dataset. It should be noted that while FP injections will always be executed, the number of possible FN injections for a set of predictions is limited to the number of objects correctly predicted by the detector. Therefore, when analyzing quantitative data, the true number of false negatives injected, on average, can deviate from the intended number of injections. This deviation is investigated in the experimental results presented in Section 6.4, and its implications on experimental results are discussed in Subsection 7.2.3.

Excluding the aforementioned peculiarities for this specific experimental approach, the injection of faults was performed analogously to the methodology presented Section 4.3. The specific methodology applied in the process of compiling datasets representing evaluations of incremental numbers of faults is detailed in the steps below.

1. 100 samples were drawn at random from the nuScenes “trainval” dataset.
2. Bounding boxes from the prediction set of each sample were filtered at a different confidence threshold for FP and FN-related injections, namely $\tau_{FP} = 0.15$ and $\tau_{FN} = 0.4$, respectively.
3. FPs or FNs were separately injected into the prediction sets of SSN for each sample, in five rounds. For each round, the number of injections into the prediction sets was incremented. This was performed for 1-5 faults of both classes, producing an equal number of different modified prediction sets for both categories of misdetections.
4. For unmodified predictions of SSN and for all modified prediction sets, metric evaluations were performed with OCM-related metrics and PKL. The resulting metric data corresponded to five different, fault-injected prediction sets for both classes of fault.
5. The $F1_{crit}$ score was computed for predictions in all datasets, and the mean metric values for $F1_{crit}$, $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, and PKL were computed over the predictions for the 100 samples.
6. Resulting mean values were visualized and analyzed with respect to their relative differences and changes in mean results.

In the experimental methodology described above, the choice of confidence thresholds on which to filter predicted BB predictions differ for the datasets containing injected FPs and FNs. When analysing the sensitivity of PKL to specific safety and reliability-related faults (namely FNs and FPs, respectively), evaluating predictions that contain both types of faults can lead to ambiguity with regard to the PKL score. As PKL comprises to a single measure of both reliability and safety-related factors of detector predictions, it is beneficial to limit the number of the other class of misdetection when examining the metrics' penalization of a specific class of faults. This limits the number of faults of the opposing class than the one injected, and consequently to more interpretable results with regards to PKL. Using a higher confidence threshold allows for minimizing the number of FPs present in the dataset upon evaluation of the penalization of FNs. Conversely, a low threshold allows for minimising the number of FNs present in the prediction sets prior to analysing the impact of FPs. Hence, a confidence threshold of $\tau = 0.15$ was enforced for predictions when analyzing the penalization of FPs, and a threshold of $\tau = 0.40$ was enforced when analyzing the penalization of FNs in the aforementioned experiment.

4.7 Comparative analysis of generic metrics with their safety-oriented adaptations

The OCM-related metrics examined in the experimental work of this thesis are closely related to their generic counterparts, namely precision, recall, and F1-score. To understand the relationship between these generic metrics and their safety and reliability-oriented adaptations, a comparative analysis was performed. As a comparison to the analyses performed in the preceding experiments, this analysis comprised methodologies utilized in preceding experiments. Corresponding metric results are presented in Section 6.5.

Applying the experimental methodology presented in this section, an analysis of the generic performance measures of precision, recall and F1-score was performed. Contrasting the experimental results presented in the preceding sections of this chapter, datasets of metric results created over the same samples and predictions selected in these experiments were visualized and assessed. Specifically, analogous experiments to those introduced in Subsection 4.5.1 and Section 4.6 were performed utilizing similar datasets of metric results for precision, recall, and F1-score. Hence, a comparison of the generic metrics with PKL and OCM-related metrics was performed. In the stepwise methodology presented below, the approach applied in this process is elaborated upon and described.

1. Utilizing the datasets introduced in Section 4.2, precision, recall, and F1-score were computed. Furthermore, histograms representing the frequencies of occurrence for the aforementioned metric values over the datasets were visualized together with joint distributions between PKL and the metrics. The results were analyzed in the context of results for safety-oriented metric adaptions. This approach follows the methodology of Subsection 4.5.1.

2. Similar to the approach described in Subsection 4.5.1, metric distributions over samples constrained by limiting the total number of GTs present were analysed, and put in context with results from the aforementioned experiment (by comparison). In this step, the distributions were visualized for different levels of restriction on the number of GT objects considered. This was performed for the same levels of restriction as the N-values selected in Subsection 4.5.1.

3. Following an analogous methodology to the experiment introduced in Section 4.6, the impact of injecting an incremental number of faults into SSN predictions over 100 nuScenes samples was analysed. An equivalent dataset of samples to the randomly drawn samples analysed in Section 4.6 was utilized.

The objective of this analysis was to understand the relationship between the generic metrics and their safety-oriented counterparts, namely OCM-related metrics and PKL. As the OCM-related metrics examined in the experimental work of this thesis are closely related to their generic counterparts, namely precision, recall, and F1-score, a better understanding of this relationship provides context to preceding experimental results by means of comparative analysis with widely acknowledged, commonly applied metrics for object detectors applied in autonomous vehicles. Furthermore, an analysis of the difference between evaluations by the aforementioned classes of metrics when compared with PKL evaluations provides insights into the degree to which PKL and OCM-related metrics assess and reflect similar factors in the context of detection. Hence, the experimental work introduced in this section addresses Research Question 1 of this thesis.

CHAPTER
FIVE

TECHNICAL SETUP

5.1 Object Criticality Model

Presented in Ceccarelli and Montecchi [2], the OCM, with which a safety metric is constructed, is implemented on top of the nuScenes devkit. More specifically, the implementation extends the devkit and provides ease of use by integrating the computation of low-level and high-level OCM-related metrics with the framework for the evaluation of generic metrics provided by the nuScenes devkit. Furthermore, the extension of the nuScenes devkit module provided by Ceccarelli and Montecchi implements functionality for visualizing criticalities of GT- and predicted objects in nuScenes samples and scripts for evaluating detector predictions with OCM-related metrics and for visualizing metric results.

The criticality values computed for objects with the OCM are dependent on three parameters, namely D_{max}, R_{max}, and T_{max} [2]. These three parameters were introduced in Subsection 2.6.2, and correspond to the maximum distances, collision distances, and collision times considered, respectively. For the purpose of the analysis presented in this thesis, these parameters will be set as constant values as an analysis of different configurations of these parameters has already been performed in [2], and as it is outside the scope of this project to evaluate the metric with regards to different configurations. However, it should be noted that different choices for these parameters will induce different experimental results. In [2], Ceccarelli and Montecchi present metric results for object detectors across a range of different configurations for $(D_{max}, R_{max}, T_{max})$. Thus, the choice of these values should be considered in any application or experimental work that utilizes the OCM. The constant values selected for these parameters were $D_{max} = 30.0$, $R_{max} = 20.0$, and $T_{max} = 10.0$. The choice of these values was based on having sufficiently high values for the three parameters for considering as many possible scenarios in which the perceived objects may interfere with ego.

5.2 Planning KL-divergence module

To evaluate detector predictions over nuScenes samples with the PKL metric, this work utilizes functionality implemented in the `planning-centric-metrics` module [10] for Python3. This module provides the means for evaluating detections with the PKL metric, as well as useful functionality for visualization and for training neural planner models for application in evaluation with PKL. In the experimental work performed and described in this thesis, the model of a neural planner provided by Philion et al. [10] was utilized for evaluating predictions with PKL, initialized with default weights provided in the corresponding repository.

5.3 The modified nuScenes devkit

Extending the modified nuScenes devkit module presented in Section 5.1 (implementing the OCM), additional functionality was implemented to accommodate the requirements of the experimental work performed and presented in this thesis. In this section, some of the essential functionality implemented as part of this research and utilized to collect the subsequent experimental results is mapped. In the experimental work performed, the `nuscenes-devkit` Python SDK [32] was utilized. All implemented functionality extending the `nuscenes-devkit` has been tested for versions 3.7 and 3.8 of the nuScenes devkit and for version 3.8.3 of Python [38]. The extended nuScenes devkit module and all implemented software are linked in Appendix A. As discussed in Section 3.1, the match criteria applied to discriminate between TP and FP predictions in nuScenes is the 2D center distance on the ground plane (not IOU). For the experimental work performed, the match limit for this parameter is set at a constant value, namely $d = 2.0$. This implies that predicted BBs with center distances less than 2.0 from a GT are considered true positive detections. In the following subsections, some of the changes made to the nuScenes devkit, and some of the functionality implemented, will be introduced.

5.3.1 Single sample evaluation

For evaluating object detection models, the nuScenes devkit module [32] implements generic evaluation metrics, including AP and mAP. Additionally, the devkit provides the means of evaluating the predictions of an object detector with NDS, presented in Equation 2.14. The aforementioned evaluation metrics enable the evaluation of detections over quantitative data by accumulating low-level performance measures such as the number of TPs, FPs, and FNs. However, as the experiments of this thesis, introduced in Chapter 4, required computing metric evaluations for predictions on single samples of the nuScenes dataset, this functionality was implemented in the extended version of `nuscenes-devkit`. This entailed implementing an alternative to the main evaluation method provided in the devkit.

5.3.2 PKL and OCM metrics

The experiments introduced in Chapter 4 apply the nuScenes devkit extensively for evaluating detector predictions on nuScenes samples with the task-oriented metrics of OCM and PKL. Thus, it was necessary to incorporate metric evaluation with these metrics into the nuScenes devkit module. In the extended nuScenes devkit provided by Ceccarelli and Montecchi [2], detector predictions can be evaluated with OCM-related metrics by accumulating metric results over multiple nuScenes samples (e.g. over the whole nuScenes “trainval” dataset). Building on this extended version of the devkit, the library provided in Appendix A enables the evaluation of detector predictions over single nuScenes samples with OCM-related metrics and with PKL. Hence, for a set of nuScenes samples, single sample metric evaluations, as well as multiple sample evaluations on accumulated metric data, can be performed utilizing the implemented functionality.

5.3.3 Injection of faults

A fundamental requirement for the experimental work introduced in the preceding chapter is the capacity of injecting faults into the prediction sets of object detection models. This entails modifying these prediction sets by either removing (FN) or inserting (FP) bounding boxes in detections. In the extended `nuscenes-devkit` module, methods are implemented for the removal and insertion of BB in the prediction set of a detector. In this section, the implementational details of the methods for injecting faults are described. Furthermore, the Python3 [38] code implementing these methods are provided in Appendix B.

5.3.3.1 False positives

To inject a false positive detection into the prediction set of an object detection model, a method is implemented in the `nuscenes-devkit` module. The code implementing this functionality is provided in Algorithm B.1. This method allows for the injection of a false positive prediction into the prediction set of an object detector, subsequent to model inference over single nuScenes samples. As parameters, the method takes the (x_{fp}, y_{fp}) position of the object to be injected relative to the ego reference frame in which ego is positioned at $(x_{ego}, y_{ego}) = (0, 0)$. Furthermore, the method takes a parameter describing the size $(height, length, width)$ of the FP to be injected. This allows the FP to be placed in specific locations and to have specific sizes based on the experiment performed. Finally, a boolean value is taken as a parameter describing whether or not to set the velocity of the injected FP equal to ego velocity or equal to 0 in all directions.

When instantiating a prediction in the nuScenes devkit, a set of parameters describing the properties of the object is required. As mentioned, some of these properties are taken as parameters in the method implemented for injection. For ease of implementation, other properties of the injected object take on predetermined values. These properties, and the values they assume for injected BBs, are presented below.

Detection score: 0.99

Attribute name: ’vehicle.moving’ if velocity is matching ego velocity, or ’vehicle.stopped’ otherwise.

Detection name: ’car’.

Orientation: set as equal to ego orientation.

Z-coordinate of position: set as equal to ego z-coordinate.

In the above, the detection score is the confidence level of the detector for the specific detection. This is initialized to 0.99 to avoid the bounding box being filtered out due to confidence threshold filtering in the experiments. The attribute name describes attributes for objects in nuScenes, providing additional descriptions of the activities of objects. As the object initialized is a car, this is set to represent a moving vehicle if the injected object has a non-zero velocity, and to represent a stopped vehicle otherwise. Furthermore, the detection name corresponds to the object class of the predicted object, which is set to represent a car for the experiments performed. Most significant in terms of consequences for the subsequent experimental work, the orientation and z-coordinate correspond to the global orientation and z-position of the injected BB, respectively. These values are always set as equivalent to the corresponding properties for the ego vehicle in the global reference frame, for the specific sample considered. The implications of the aforementioned implementational choices are discussed in Subsection 7.1.2

5.3.3.2 False negatives

To inject a false negative detection into the prediction set of an object detector, i.e. removing a correctly predicted BB from the prediction set, a method is implemented in the `nuscenes-devkit` module. The code implementing this method is provided in Algorithm B.2. Given a set of bounding boxes predicted by an object detector over a single nuScenes sample, this method allows removing a single, or multiple, correctly predicted bounding boxes, i.e. TP predictions. The method implementing the functionality for injecting FN predictions takes as a parameter a distance measure, specifying the Euclidean distance from the ego vehicle up to which correctly predicted BBs are considered for removal. This parameter thus enables not considering objects that are further away from ego than a specified distance. Furthermore, another parameter is taken in the method, namely a boolean value indicating whether to remove one or all TP predictions within the aforementioned distance from ego. This allows for predictions to be modified to simulate significant safety hazards in detection. If multiple possible TP predictions are present within the distance specified upon calling the method described above, predictions are selected for removal based on their Euclidean distance from ego on the ground plane. More specifically, the most proximate TP in relation to ego is removed first. The method described above enables the injection of FNs into the predictions of an object detector over nuScenes samples.

5.4 IDUN High-Performance Compute Cluster

IDUN [50] is NTNU's high-performance compute cluster, providing an administrated and highly available compute platform for students and faculty members. The IDUN project is a joint effort between a number of faculties and shareholders at NTNU. At the time of writing, IDUN incorporates 92 designated GPUs, 80 of which are financed and leveraged by the Department of Computer Science, and 1932 CPU cores.

For the experimental work of this thesis, a VM hosted on IDUN was utilized. This allowed for utilizing a designated amount of storage, CPU, and GPU resources. These resources included two Tesla P100 PCIe 16GB GPU compute nodes, built based on the NVIDIA Pascal GPU architecture, and 950GB of storage. These resources were applied in the experimental work of this thesis by performing inference with object detection models over the nuScenes dataset and performing metric evaluations of the resulting detector predictions.

CHAPTER

SIX

EXPERIMENTS AND RESULTS

Following the methodologies outlined in Chapter 4, experiments were performed and results were collected. In this chapter, these results are presented and analysed.

6.1 Examining single sample metric evaluations

This section examines detector predictions and metric evaluations for a selected nuScenes sample, following the experimental methodology introduced in Section 4.4. This is performed for both raw detector predictions on the sample, and for predictions that are modified by injecting synthetic faults. For the two object detection models SSN and PPT, predictions on single samples from the nuScenes dataset were filtered at confidence threshold $\tau = 0.15$ and visualized. Furthermore, both GTs and predictions are filtered to only include the "car"-class before evaluation. The samples examined were selected qualitatively.

Firstly, single nuScenes sample and its corresponding GTs and predictions from SSN and PPT are examined and analyzed. For these scenarios, the corresponding evaluations of detector predictions according to $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$, and PKL are subsequently assessed. For the repeatability of the experiments performed in this section, the *sample token* (the unique identifier for nuScenes samples) of the sample analyzed is presented below.

Sample token: 6c8d4379e83646d08436f6ec92b35fe5

In Figure 6.1.1, the point cloud resulting from the LIDAR scan is visualized together with the environment and annotated GTs for the sample. In Figure 6.1.2, the camera frames corresponding to the sample analysed are depicted. 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. In this scenario, there is some uncertainty as to the importance of detecting the two aforementioned vehicles. It can be argued that the vehicles are important to detect due to their proximity to ego. However, one could also claim that there is no danger of the vehicles interfering with ego and that the correct or incorrect detection of the vehicles does not affect ego's ability

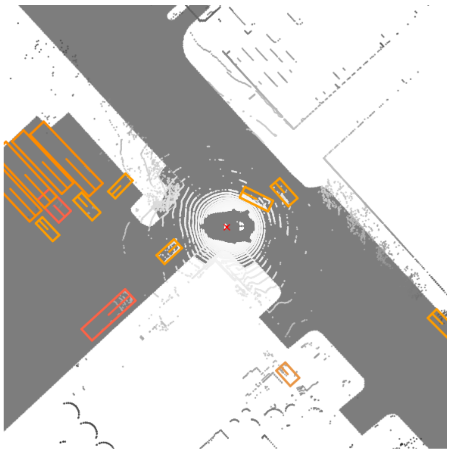
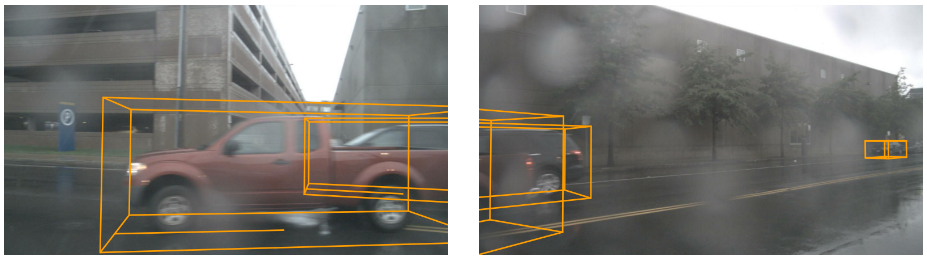


Figure 6.1.1: LIDAR-scan and environment of the sample.



(a) Left-front camera.

(b) Front camera

Figure 6.1.2: Camera frames corresponding to the sample.

to make a right turn. This ambiguity can highlight the difference between the two task-specific metrics for which results are subsequently examined.

6.1.1 Raw predictions

In Figure 6.1.3, the predictions of SSN and PPT for the sample, as well as GT objects, are visualized. In the figure, the axes are measured in meters, and the coordinate system is aligned with the reference frame of the ego vehicle. Furthermore, predicted BBs are visualized in blue and GT objects in green with the corresponding criticalities for the prediction and the GT annotated next to their bounding box rectangles. In Figure 6.1.3, it is observed that the two aforementioned vehicles in proximity to ego both receive criticalities of 1.0. To understand how the OCM evaluates this criticality, the criticality weights κ_d, κ_r, and κ_t, introduced in Subsection 2.6.2, are considered. It is evident that the closest vehicle (turning left) is the most important and fruitful object to analyze in this particular sample. Thus, the criticality weights are discussed for this object. The predicted BB receives the same criticality as the GT object when rounded to two decimal points in Figure 6.1.3. Hence, values are only presented for predicted boxes. For both PPT and SSN, the aforementioned criticality weights evaluate to $\kappa_d = 0.94$, $\kappa_r = 0.91$, and $\kappa_t = 1.0$. Out of these three values, the *collision time criticality* is the highest, evaluating to the maximum criticality of 1.0. This implies that as evaluated by the OCM, the ego is on a potential collision path with the vehicle. As discussed in Subsection 2.6.2, the reason for this is that k_t is computed only with

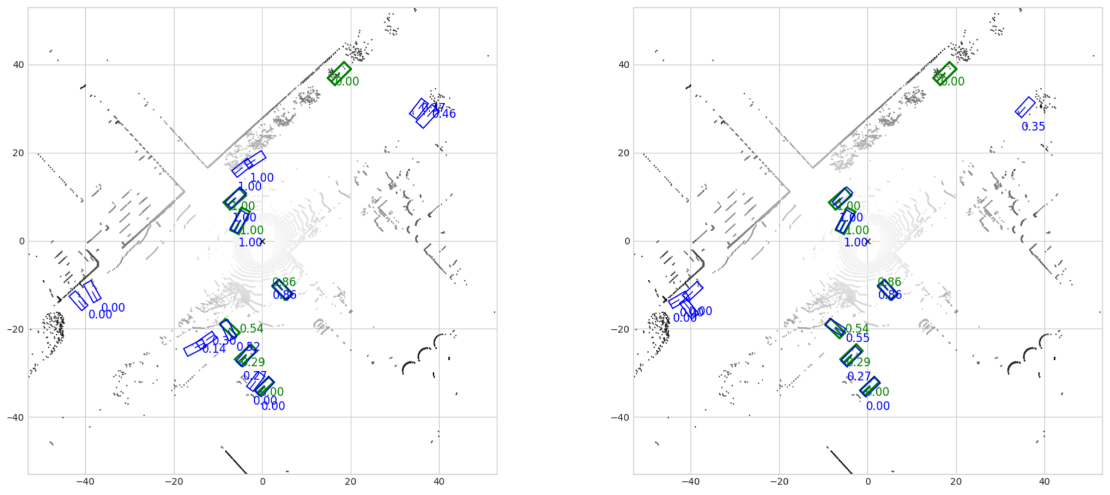


Figure 6.1.3: Predicted objects (blue), GTs (green) and criticalities assigned by the OCM for PPT (left) and SSN (right) on a single sample.

regards to the current speed and direction of movement of the vehicle evaluated, and thus exhibits a high criticality due to the velocity vector of ego pointing in the direction of the incoming vehicle. The same is true for the *collision distance criticality*, which evaluates to a value proximate to the maximum.

Metric	**PPT**	**SSN**
$\mathbf{P}_{\mathcal{R}}$	0.533	0.915
$\mathbf{R}_{\mathcal{S}}$	0.989	0.998
$\mathbf{F1_{crit}}$	0.693	0.955
PKL	0.677	3.673

Table 6.1.1: Metric results on single sample.

Table 6.1.1 presents the metric results corresponding to the BBs predicted by the PPT and SSN detectors. In the table, metric values are rounded to three decimal points. It is observed that the main metrics $F1_{crit}$ and PKL rank the predictions of the two detectors differently. For SSN, $F1_{crit}$ indicates near-perfect detection of objects with positive criticality values, exhibiting a score near 1.0. PKL, however, suggests a slight divergence from the GT planned path (path planned with GT predictions), indicated by a value of $PKL = 3.67$. It is noted, however, that a PKL value of 3.67 is a good score and that the difference between PKL evaluations for the two detectors is likely due to a less significant error in the prediction of the orientation, size or location of an object in the scene. For PPT, the $F1_{crit}$-measure evaluates to a lower value. This is due to $P_{\mathcal{R}}$ assuming a value of 0.533, and it is observed in Figure 6.1.3 (left figure) that this is caused by the detector predicting a number of false positive detections with high criticalities. Most notably, PPT predicts the presence of two non-existing objects in front of the vehicle, whose criticalities both evaluate to 1.0. The $R_{\mathcal{S}}$ score for the PPT predictions indicates that all critical objects were correctly predicted. Note that

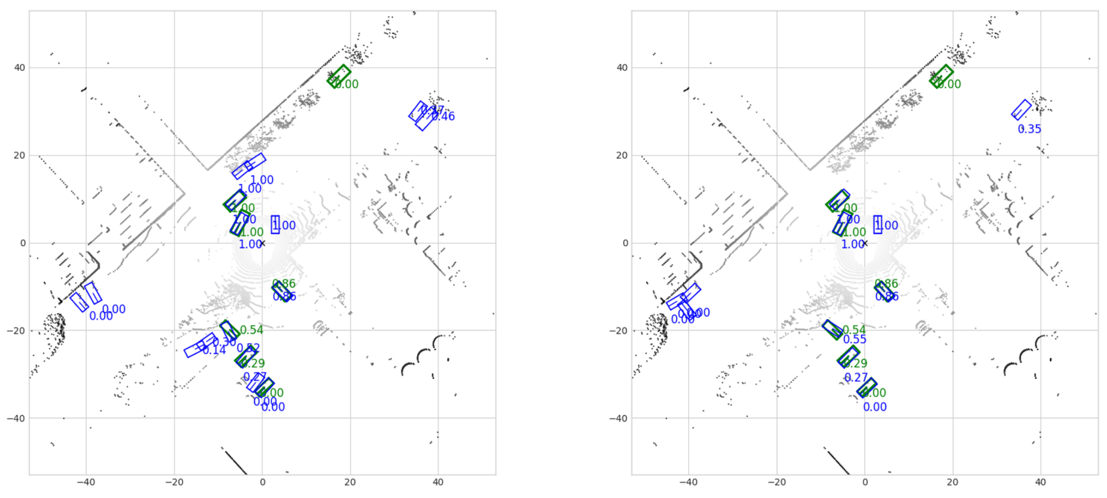


Figure 6.1.4: Predicted objects, GTs, and criticalities with injected FP for PPT (left) and SSN (right).

the confidence threshold at which the detections were filtered at $\tau = 0.15$, giving high values for safety-weighted recall. For PKL, the predictions by PPT are evaluated to 0.677, indicating very little deviance from the planned trajectory as a result of these detections.

6.1.2 Fault injected predictions

In this section, the detector predictions for the sample examined in the previous section are subject to injection of misdetections. The implications of injecting faults into the prediction sets of the detectors in the context of evaluation are investigated. Utilizing the methods for fault injection described in Subsection 5.3.3, one FN and one FP are injected into the predictions of PPT and SSN. Hence, through the injection of faults, detections that have the potential to cause safety hazards or reliability issues in the downstream task of planning are investigated. This enables a qualitative evaluation of metric results for such scenarios, and thus an analysis of these results when compared to metric evaluations of predictions where such faults have not been introduced.

6.1.2.1 Injected false positive

For the injection of a false positive, a bounding box representing a vehicle of object class "car" was inserted into the prediction sets of both detectors at the position $(x, y) = (3, 5)$ relative to ego. The size of the injected object was $(h, l, w) = (1.7, 4.0, 2.0)$, and its velocity was set to 0 in all directions. The FP was injected at the selected location to directly interfere with the ego trajectory, and thus to represent a detection that interrupts the service of the autonomous system and in consequence affects its reliability.

In Figure 6.1.4, the predictions and GT objects, along with the injected FP are visualized. In the figure, the injected object is located close to the front-right of ego. Expectedly, a high criticality value of 1.0 is derived for the injected FP for both detection models. For both PPT and SSN, the criticality weights evaluate to $\kappa_d = 0.92$, $\kappa_r = 0.98$, and $\kappa_t = 0.88$. Due to the injected object being in such close proximity to ego and having a velocity of 0, the perceived object has a short collision distance and collision time, explaining the high values for the criticality weights.

Metric	**PPT**	**SSN**
$\mathbf{P}_{\mathcal{R}}$	0.466	0.733
$\mathbf{R}_{\mathcal{S}}$	0.989	0.998
$\mathbf{F1}_{\mathbf{crit}}$	0.633	0.845
PKL	20.74	5.169

Table 6.1.2: Metric results for predictions with one injected FP.

The metric results for the modified prediction sets of PPT and SSN are presented in Table 6.1.2. Seen in contrast to Table 6.1.1, there is a decreased in the values for the evaluations of $F1_{crit}$ and PKL for both detectors as a result of injecting an FP. Most notably, the PKL score for the predictions of PPT evaluates to PKL = 20.74, whereas the PKL score for the SSN predictions is 5.17. For the detections of PPT, this indicates a divergence from the GT trajectory of ego caused by the introduction of false positive detection. However, to reason about the change in PKL-score as a result of the injection of an FP, the other predictions of the detector must be considered. Examining Figure 6.1.4, this divergence can be explained by considering the other misdetections present in the prediction set of PPT. As discussed in the preceding section, the presence of two cars is incorrectly predicted by PPT in front of ego. Although these FPs do not sufficiently impact the planned trajectory of ego to cause a considerable increase in the PKL evaluation on their own, as seen in Table 6.1.1, they provide important context to the evaluation performed on the FP-injected predictions. Taking into account the trajectory planned by the planning algorithm subsequent to the introduction of the FP, it is likely that the joint presence of the injected FP and the preexisting false positives would induce a more considerable divergence from the planned path. Considering the OCM measures, the introduction of an FP leads to a decrease in $P_{\mathcal{R}}$, and thus to a decrease in $F1_{crit}$ for both PPT predictions and SSN predictions compared to results presented in the previous section. This decrease is, however, larger for SSN. This can be explained by considering the number of FPs with considerable criticalities already present in the predictions for PPT, and the equation for computing $P_{\mathcal{R}}$, introduced in Subsection 2.6.2. For a larger number of misdetections present when computing the reliability-weighted precision, the addition of another FP has a less significant impact on the final score.

6.1.2.2 Injected false negative

For the injection of a false negative detection, a single, correct prediction was removed from the prediction sets of PPT and SSN. The specific object selected for removal was the vehicle closest to ego, i.e. the vehicle turning left at the intersection. Evaluated by a center distance threshold of 2.0, both object detectors correctly predicted BBs for this object. The vehicle representing this object was

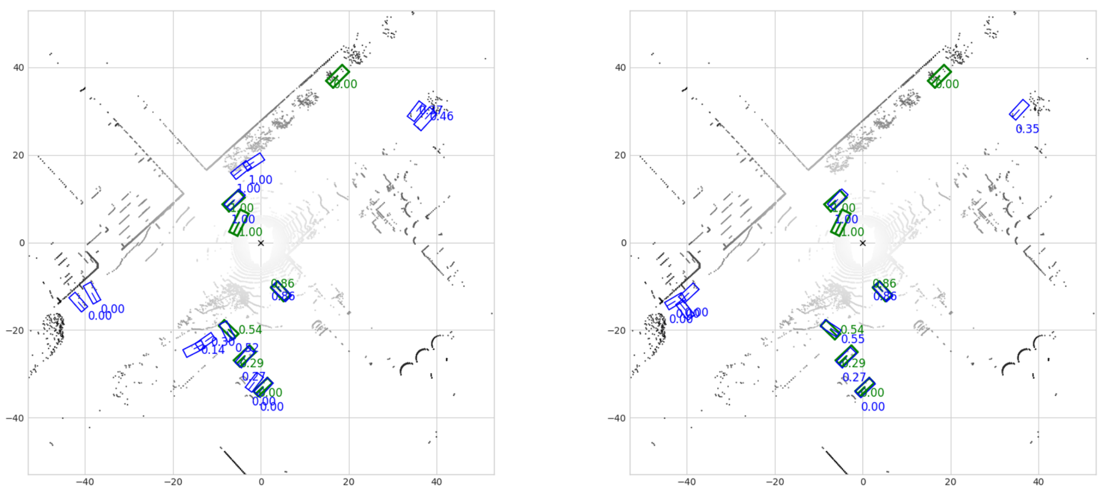


Figure 6.1.5: Predicted objects, GTs, and criticalities with injected FN for PPT (left) and SSN (right).

selected due to the high perceived criticality for the vehicle following observations in the visualization of the sample, i.e. Figure 6.1.1. Note that this high perceived criticality is not related to criticality as defined by the OCM, but rather on the perceived importance of the detection viewed in the context of the sample. Thus, the prediction was chosen for removal with the intent of analyzing the impact of removing high-criticality objects on metric evaluations. This enables a qualitative assessment of how the metrics evaluate potentially hazardous detection scenarios where the safety of the system could be compromised. To inject the FN detection, the methods introduced in Section 4.3 were utilized.

In Figure 6.1.5, it is observed that the predictions do not contain a correctly predicted bounding box (plotted in blue) for the selected object. This corresponds to PPT and SSN failing to detect the GT object the vehicle represents and simulates a scenario in which the safety of the system is potentially compromised. Thus, following the injection, an FN detection is reflected in the measures for performance.

Metric	**PPT**	**SSN**
$\mathbf{P}_{\mathcal{R}}$	0.455	0.880
$\mathbf{R}_{\mathcal{S}}$	0.717	0.720
$\mathbf{F1}_{\mathbf{crit}}$	0.556	0.799
PKL	4.50	3.31

Table 6.1.3: Metric results for predictions with one injected FN.

The metric evaluations of the detectors' predictions injected with a false negative are presented in Table 6.1.3. It is observed that OCM-related metrics are impacted similarly by the introduction of an FN for the two detectors when compared with metric evaluations of the unmodified predictions (i.e. raw predictions) presented in Table 6.1.1. In particular, $R_{\mathcal{S}}$ sees a decrease of ≈ 0.27 for both detectors.

In addition, both sets of detections are evaluated approximately equally in terms of safety-weighted recall. As presented in the previous section, the criticality of the predicted BB for the removed object evaluates to $\kappa = 1.0$ for both detectors. Furthermore, as discussed in Subsection 6.1.1, the raw predictions for the two detectors are evaluated approximately equally in terms of $R_{\mathcal{S}}$. Thus, analogous results for safety-weighted can be expected for the two detectors when this particular detection is removed from their predictions. For $F1_{crit}$ a similar comparison can be made, with approximately equivalent decreases in metric values for the two detectors relative to the metric data for the raw predictions. However, as observed in the previous section, $F1_{crit}$ is substantially lower, at $F1_{crit} = 0.556$ for PPT than for SSN, with $F1_{crit} = 0.799$. This difference is induced by their disparate scores for reliability-weighted precision, which is in turn caused by a prevalence of FPs present in the predictions of PPT. Note that while safety-weighted recall is not influenced by introducing FP detections into the prediction set, the definition of reliability-weighted precision allows the value it takes on to be influenced by removing correct predictions. This explains the change observed in $P_{\mathcal{R}}$ for both detectors, compared to metric data over raw predictions.

Considering PKL, a small increase is seen for PPT following the injection of the FN detection relative to PKL evaluation over the raw predictions. This indicates that the removal of the predicted BB causes a slightly larger divergence from the path planned by considering the GT objects. Interestingly, in the case of SSN, PKL exhibits a marginally lower value for the fault-injected predictions compared to the raw predictions. This suggests that the path planned based on FN-injected predictions assumes a smaller divergence from the path planned by considering GT objects than to the path planned with raw predictions. More specifically, this entails that predicting the BB causes a greater divergence from the original path than not predicting it. Although the difference in PKL values is negligible for raw predictions and FN-injected predictions, investigating how such a result can occur is important when analyzing the characteristics of PKL. An explanation of this result is that the removed BB exhibits some degree of error that causes a specific divergence, as reflected by PKL. If the properties of the object detected (i.e. location, speed, direction of movement) are such that the object is not likely to interfere with ego, then the prediction of it (with some degree of error) may cause a greater value for Planning KL-divergence than simply not making a prediction for it.

6.2 Analysis of metric distributions

In this section, part of the results collected by applying the methodology introduced in Subsection 4.5.1 are presented. Specifically, visualisations forming the basis for the analysis of metric distributions are presented and analysed. Recall that metric data utilized in this experiment stem from a single object detector, namely SSN. Furthermore, scores for PKL are presented on the negative logarithmic form ($-log$ PKL) as the PKL value is not bounded for positive values.

6.2.1 Nominal metric data

Following the approach of Subsection 4.5.1, the histograms representing the frequencies of occurrence in the dataset RAW_40 for $F1_{crit}$ and PKL, respectively, are visualized in 6.2.1.

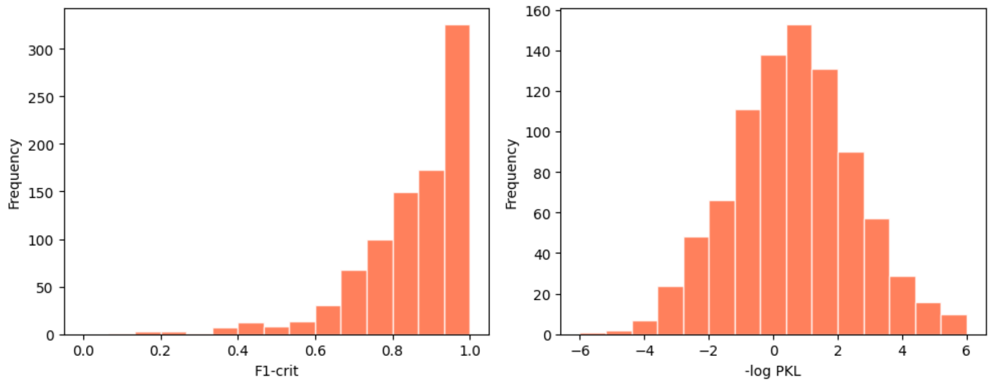


Figure 6.2.1: Histogram for the metric distributions of RAW_40.

In Figure 6.2.1, metric values are divided into 15 bins of uniform width, where the frequencies of occurrence for metric values falling into specific bins are visualized by the height of the bars representing each bin. In the plot, a visual dissimilarity between the distributions of metric results can be observed, with PKL scores seemingly being more evenly distributed for the samples evaluated. It should be noted that for instances with $PKL < 1.0$, the negative logarithm evaluates to positive values. Thus, for the right-hand histogram of Figure 6.2.1, the bins larger than 0 represent PKL scores less than 1 and approaching 0 (i.e. very good scores for PKL). This suggests that the distributions plotted Figure 6.2.1 represent more similarity between the metrics than is perceived. This quality of the PKL values and the fundamental differences of the metrics should thus be considered when interpreting results.

Figure 6.2.2 shows the joint distributions for metric results between OCM-related metrics and PKL. As is observed in the histograms depicted in Figure 6.2.1, a significant number of predictions evaluate to high scores for both OCM-related metrics and PKL. Considering the individual components of the OCM, higher values are achieved for $P_{\mathcal{R}}$ than for $R_{\mathcal{S}}$ on average. This is likely a consequence of the confidence threshold selected ($\tau = 0.40$) permitting more FN detections than FP predictions to be present in the prediction sets evaluated.

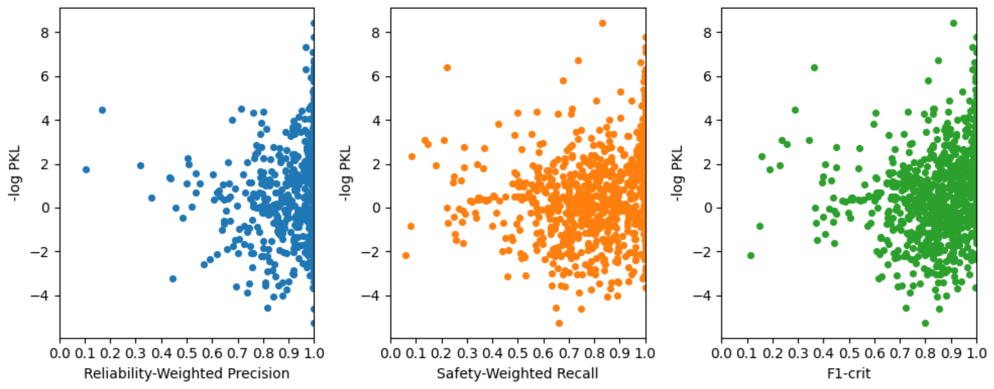


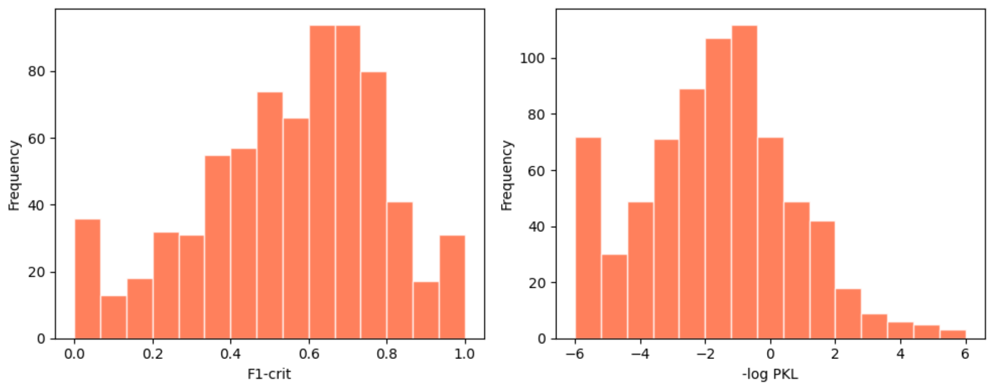
Figure 6.2.2: Scatter plot showing the joint distributions between PKL and the task-oriented metric evaluations represented in RAW_40.

6.2.2 Fault injected metric data

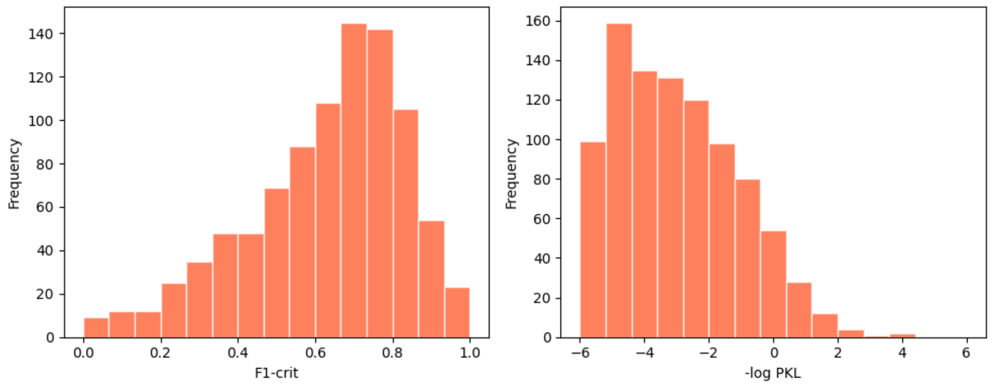
In this section, the distributions of metric data in the fault-injected datasets presented in Table 4.2.1, namely FN_40 and FP_40, are analyzed. As performed for the non-injected dataset in the preceding section, the distributions of metric data for both metrics and datasets are visualized in Figure 6.2.3.

Examining Figure 6.2.3, it is evident that PKL penalizes predictions in which FPs have been injected more rigorously than $F1_{crit}$. Specifically, comparing with results in the RAW_40 dataset analysed in the preceding section, a more significant left shift is observed for the distribution of PKL values in Figure 6.2.3b than in Figure 6.2.3a. This indicates lower performance in terms of PKL when considering predictions injected with FPs. This penalization is more thoroughly investigated in subsequent experimental results. In Figure 6.2.3, a significant left shift can also be observed in both distributions of $F1_{crit}$ for the fault-injected datasets when compared with Figure 6.2.1. Additionally, the distribution of values for $F1_{crit}$ are more evenly distributed across metric values for the FN_40 dataset.

In Figure 6.2.4, the scatter plots representing the joint distributions between PKL and $F1_{crit}$, $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ for the two fault-injected datasets are visualized. A noticeable disparity in terms of $F1_{crit}$ is observed for distributions represented in the two datasets. Specifically, metric values appear to be more evenly distributed in the FN_40 dataset. It is anticipated for the $P_{\mathcal{R}}$- and $R_{\mathcal{S}}$-based distributions to exhibit differences for the two types of injected errors in the prediction sets, illustrated in Figure 6.2.4a and Figure 6.2.4b. This is the consequence of the two measures being designed to reflect different types of faults in prediction. However, different distributions for their harmonic means ($F1_{crit}$) may signify a difference in the penalization of FNs and FPs as both measures comprising the metric are reflected equally.

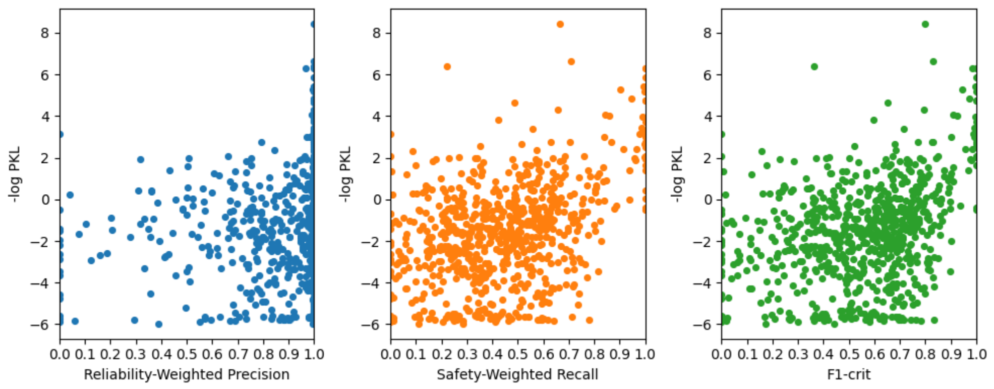


(a) Histogram for the FN_40 dataset.

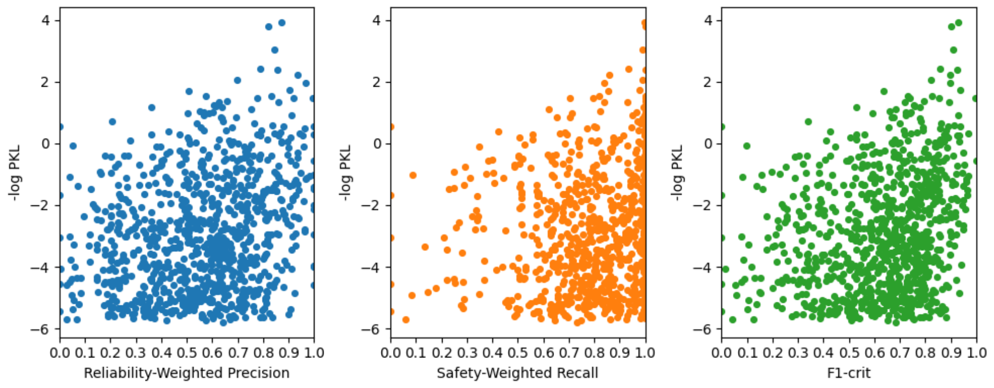


(b) Histogram for the FP_40 dataset.

Figure 6.2.3: Figures (a) and (b) show the frequency of occurrence for metric values in the FN_40 and FP_40 datasets, respectively.



(a) Scatter plot for metric data over the FN_40 dataset.



(b) Scatter plot for metric data over the FP_40 dataset.

Figure 6.2.4: Figures (a) and (b) visualize the joint distributions between the task-oriented metric evaluations represented in the FN_40 and FP_40 datasets, respectively. In comparison with the RAW_40 dataset, the changes in distributions reflect the types of fault injected.

6.3 Metric correlation for constrained detections

In this section, part of the results collected by applying the methodology presented in Section 4.5 are presented. Specifically, the methodologies introduced in Subsection 4.5.1 and Subsection 4.5.2 were applied to produce the subsequent results. These results provide insights into the correlation between OCM-related metrics and PKL by means of visual analysis of metric distributions and the analysis of statistical correlation. Furthermore, the results investigate the consequences of constraining the datasets utilized by restricting the total number of objects considered in the samples evaluated.

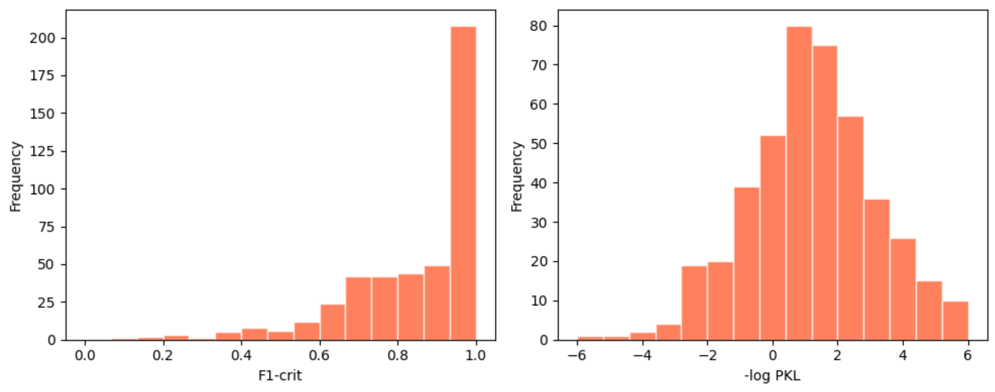
6.3.1 Analysis of metric distributions

In Figure 6.2.2, it can be observed that PKL scores tend to vary over samples with congruent evaluations for $F1_{crit}$ (i.e. samples that are evaluated similarly with regards to $F1_{crit}$). It is difficult to analyze the reason for the disparity of PKL values for detections that are similarly scored by $F1_{crit}$ due to the environmental factors considered in PKL evaluations being implicitly learned. To enable an assessment of the relationship between the two metrics, and to analyze the sensitivity of OCM-related metrics to the number of objects present in samples, the changes in the distributions examined in the previous section are reviewed for datasets in which samples are excluded based on the number of GT objects they represent, as described in Subsection 4.5.1.

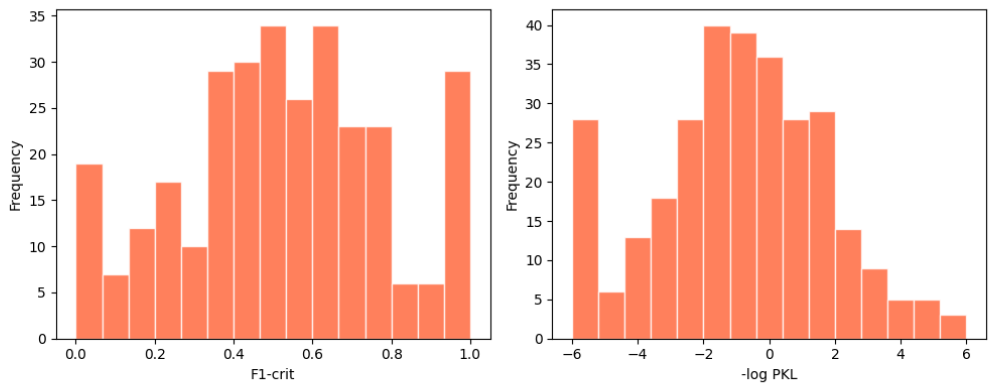
Constrained datasets for which samples that have less than or equal to N objects in their GT set are visualized analogously to the full datasets presented in the preceding section. More specifically, metric data for the datasets resulting from the constraint $N = 8$ (implying $o_i \leq 8$) and $N = 4$ are visualized. In the subsequent analysis, the datasets of reduced size resulting from these constraints on the number of objects will be referred to as *constrained datasets*.

For $N = 8$, Figure 6.3.1 depicts the histograms representing the distributions of metric data in the constrained datasets. Examining the distribution of PKL results for the RAW_40 dataset (Figure 6.3.1a), a right skew is observed in the distribution compared to similar metric results for the unconstrained RAW_40 dataset. This indicates that the predictions (and samples) disregarded as a consequence of constraining the dataset with $N = 8$ performed worse in terms of PKL than the samples where $o_i \leq 8$. Similar observations are made for PKL distributions in the datasets FP_40 and FN_40. The overall increase in PKL as a result of excluding evaluations for samples with a higher number of objects indicates that PKL generally exhibits better scores for predictions on samples with a lower number of objects. Conversely, for $F1_{crit}$, the distributions depicted in Figure 6.3.1 generally indicate lower performance for the constrained datasets, in contrast to similar distributions for the full datasets visualized in presented in Section 6.2. This is most evident for the distribution of $F1_{crit}$ in the FP_40 dataset. The aforementioned observations suggest that the means of PKL and $F1_{crit}$ scores over a given dataset are inversely proportional to the increase and decrease, respectively, of the number of objects represented in the samples evaluated.

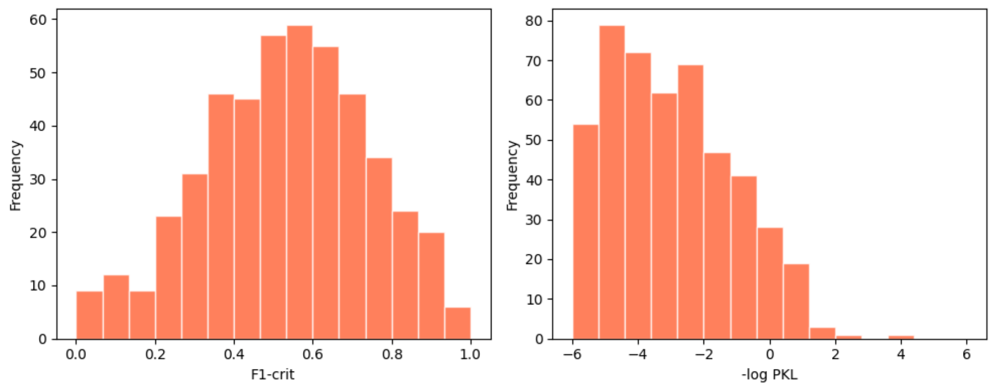
In Figure 6.3.2, the joint distributions between OCM-related metrics and PKL are visualized through scatter plots. In the histograms provided, it can be observed



(a) Histogram for the RAW_40 dataset.



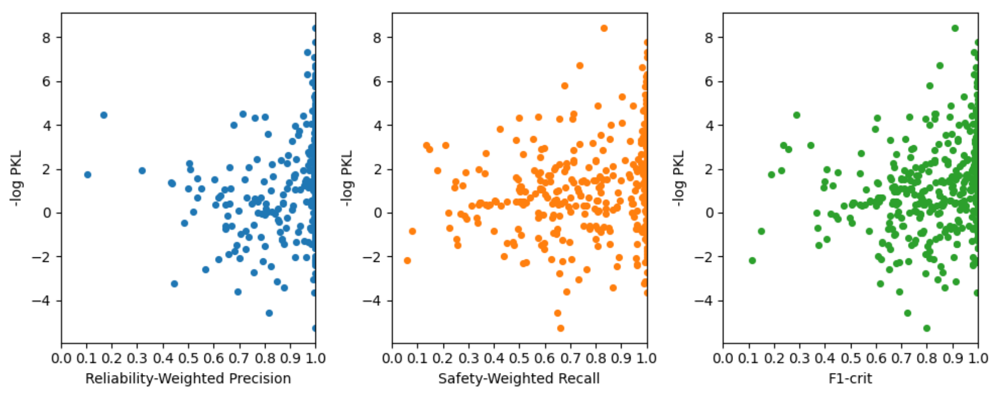
(b) Histogram for the FN_40 dataset.



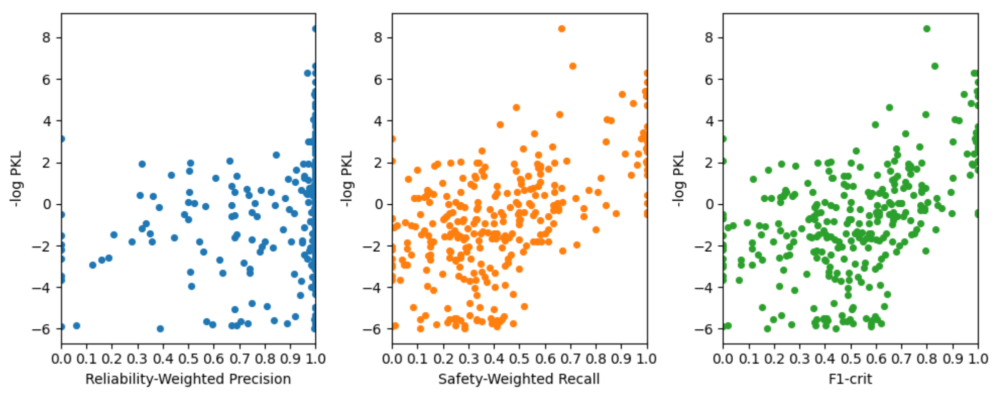
(c) Histogram for the FP_40 dataset.

Figure 6.3.1: Figures (a), (b), and (c) show the marginal metric distributions for the constrained datasets RAW_40, FN_40, and FP_40, where $N = 8$.

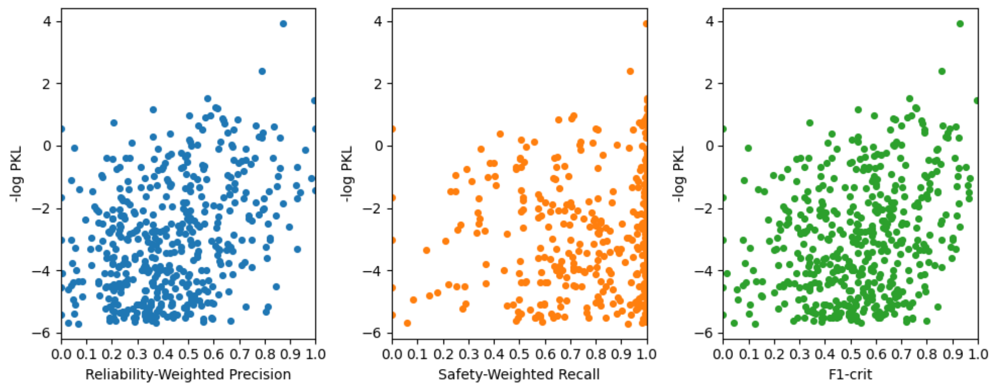
that over the fault-injected datasets, a smaller fraction of data points corresponding to poor PKL scores are present when constraining the FN_40 and FP_40 datasets with $N = 8$. Specifically, there is a smaller fraction of data points corresponding to predictions that exhibit high scores for $F1_{crit}$ and low scores for PKL (compared to the unconstrained sets depicted in Figure 6.2.2 and Figure 6.2.4). This indicates a higher degree of "agreement" between the two in terms of evaluating the predictions reflected in the constrained dataset. For $F1_{crit}$ and PKL, there appears to be a higher degree of correlation between the metrics as a consequence of the shifts in their distributions. Especially over data from FN_40, the distri-



(a) Scatter plot for metric data over the RAW_40 dataset.



(b) Scatter plot for metric data over the FN_40 dataset.

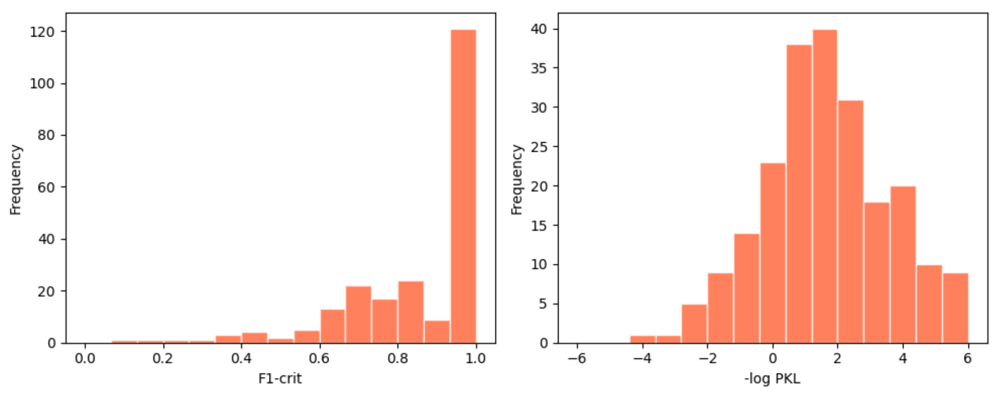


(c) Scatter plot for metric data over the FP_40 dataset.

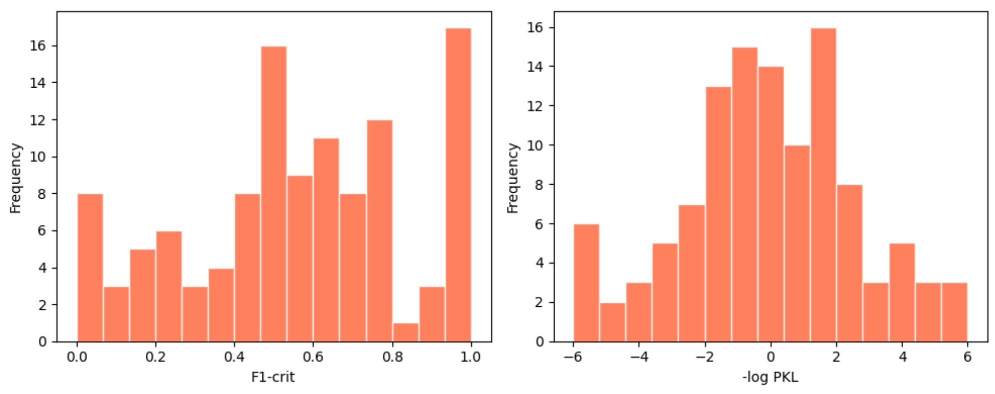
Figure 6.3.2: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets RAW_40, FN_40, and FP_40, where $N = 8$.

butions of data points appear to follow a linear trend. Considering $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ for the corresponding datasets, their joint distributions indicate that a decrease in high-scoring predictions for the measures causes the overall decrease in $F1_{crit}$ for lower values of the constraint variable N.

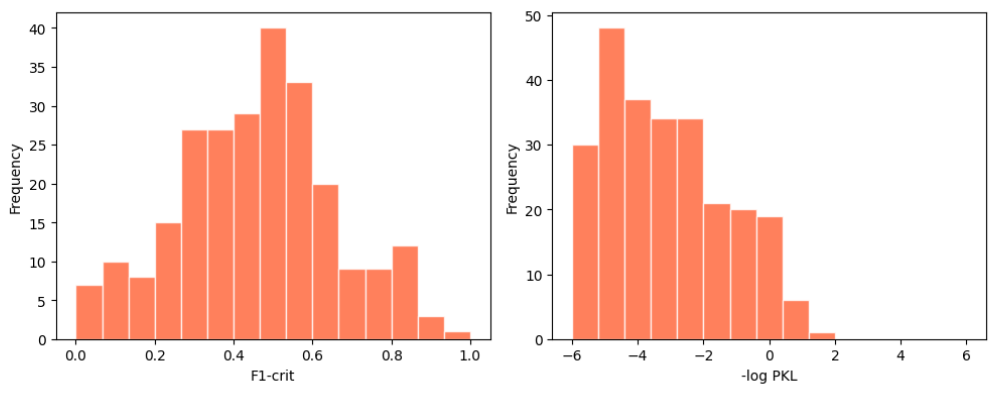
In Figure 6.3.3, histograms representing the distributions of metric results are depicted for datasets constrained with $N = 4$. Comparing these to the histograms depicted in Figure 6.3.1, it is evident that there is a decrease in the fraction of values corresponding to low-scoring predictions for PKL. Considering $F1_{crit}$,



(a) Histogram for the RAW_40 dataset.



(b) Histogram for the FN_40 dataset.

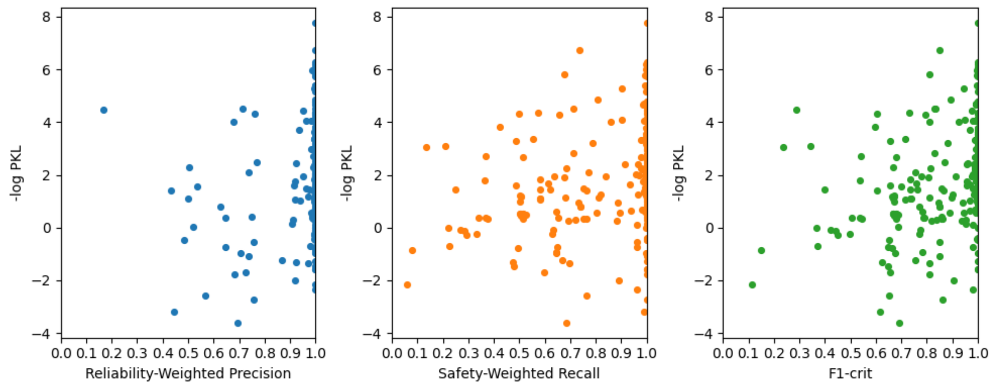


(c) Histogram for the FP_40 dataset.

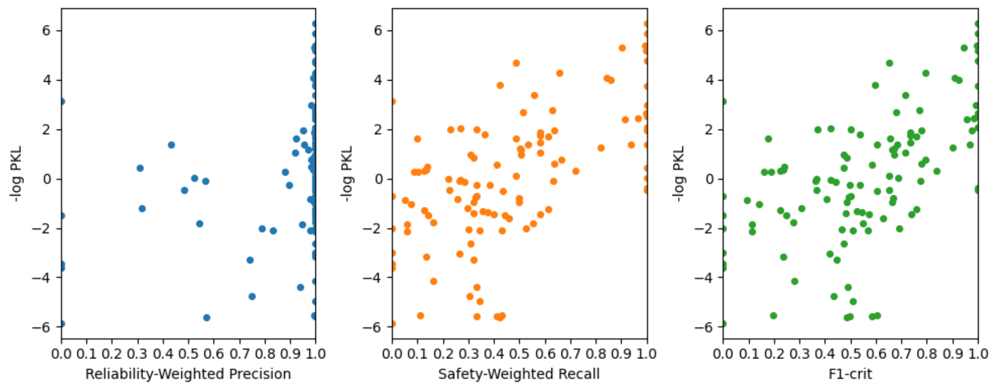
Figure 6.3.3: Figures (a), (b), and (c) show the marginal metric distributions for the constrained datasets RAW_40, FN_40, and FP_40, where $N = 4$.

there appears to be an overall increase in the fraction of high-scoring predictions for RAW_40 and FN_40, but a decrease for FP_40. This indicates a similarity in the response of the distributions of the two metrics to the constraints imposed for the RAW_40 and FN_40 datasets and a more evident difference in their evaluations for the FP_40 dataset.

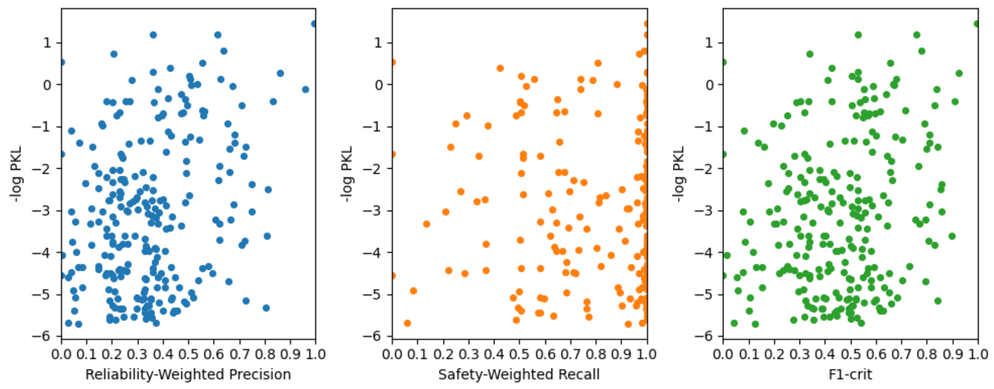
As observed in previously illustrated distributions for PKL, PKL appears to penalize false positives more severely than false negatives considering the respective, fault-injected datasets FP_40 and FN_40. Analyzing the distribution of PKL represented by FP_40 for both constrained datasets (with $N = 8$ and $N = 4$), it is observed that the distribution is skewed left, with values for the negative logarithm of PKL not exceeding 2.



(a) Scatter plot for metric data over the RAW_40 dataset.



(b) Scatter plot for metric data over the FN_40 dataset.



(c) Scatter plot for metric data over the FP_40 dataset.

Figure 6.3.4: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets RAW_40, FN_40, and FP_40, where $N = 4$.

In Figure 6.3.4, the joint distributions between PKL and OCM-related metrics are depicted for datasets with a constraint of $N = 4$. Examining data represented by FN_40 in Figure 6.3.4b, there appears to be a more pronounced linear relationship between $F1_{crit}$ and PKL as the number of objects in the evaluated scenes decreases. This is demonstrated by the reduction in data points with low PKL values for higher $F1_{crit}$ values, compared to the distributions for unconstrained datasets and datasets constrained with $N = 8$. It is important to note that the joint distributions are depicted on the logarithmic scale of the y-axis. As a linear trend is observed for the measures $F1_{crit}$ and $R_{\mathcal{S}}$ for FN_40, this implies a non-linear trend for the true metric values.

Considering all the joint distributions analyzed, it can be inferred that safety-weighted recall exhibits a higher correlation with PKL compared to reliability-weighted precision. This correlation appears to be increasingly evident for decreasing values of the constraint variable N (and thus for decreasing numbers of objects represented by the samples evaluated). One possible explanation for this observation could be the aforementioned observed difference in the penalization of false positive predictions between PKL and reliability-weighted precision.

6.3.2 Statistical Analysis

In the previous section, joint distributions between PKL and $F1_{crit}$ were analyzed for the scenarios $N = \infty$, $N = 8$, and $N = 4$. To quantify the relationship between these measures for these constraints, the Pearson correlation coefficient (r_p) and the Spearman correlation coefficient (r_S) were computed. As discussed in Subsection 4.5.2, these coefficients measure the strength and direction of linear and monotonic relationships, respectively. Additionally, the p-values associated with these coefficients were computed to assess the statistical significance of the findings. The results are summarized in Table 6.3.1-Table 6.3.3.

Table 6.3.1: Correlation on constrained detections for RAW_40.

N	$\mathbf{r_P}$	$\mathbf{r_S}$	$\mathbf{p_P}$	$\mathbf{p_S}$	Size
N=∞	0.207	0.318	4.094e-10	1.872e-22	894
N=8	0.280	0.365	1.505e-15	1.616e-9	447
N=4	0.343	0.408	1.392e-7	2.258e-10	224

Table 6.3.2: Correlation on constrained detections for FN_40.

N	$\mathbf{r_P}$	$\mathbf{r_S}$	$\mathbf{p_P}$	$\mathbf{p_S}$	Size
N=∞	0.346	0.327	3.112e-22	6.774e-20	739
N=8	0.503	0.503	5.382e-21	6.177e-21	305
N=4	0.577	0.598	1.820e-11	2.163e-12	114

Table 6.3.3: Correlation on constrained detections for FP_40.

N	$\mathbf{r_P}$	$\mathbf{r_S}$	$\mathbf{p_P}$	$\mathbf{p_S}$	Size
N=∞	0.242	0.251	8.163e-14	1.089e-14	923
N=8	0.290	0.268	1.119e-10	2.720e-9	476
N=4	0.230	0.198	0.240e-3	0.169e-2	250

In the tables, p_P and p_S refer to the p-values for Pearson and Spearman, respectively, and "Size" refers to the size of the constrained datasets at this value of N. Recall that lower values for the constraint variable imply smaller datasets generated, as a consequence of excluding samples.

For the three values of N, for which the distributions of the metrics were visualized in the preceding section, the p-values stay sufficiently low to indicate statistically significant results[1] for all but p_S for $N = 4$ on the FP_40 dataset. This

[1]As a rule of thumb, p-values below 0.001 will be regarded as indications of statistical significance [48].

is despite it being the largest dataset at this level of constraint. This indicates that the statistical significance of the results is independent of the size of the datasets. Examining the correlation coefficients, there is a consistent increase in their values for decreasing values of N for all but the FP-injected dataset at $N = 4$. The aforementioned results for the FP-injected dataset are likely a consequence of PKL penalizing such faults more than the OCM-related metrics.

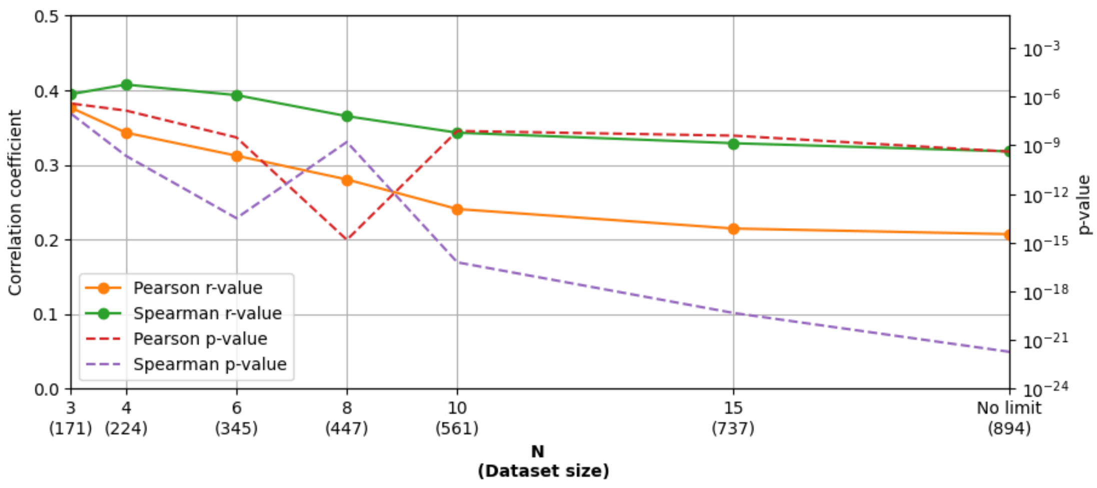
As the three values of N investigated thus far were arbitrarily selected and qualitatively reviewed, a thorough analysis of the change in correlation coefficients for a more considerable number of values for N is necessary for inferring more meaningful insights into their dependency of the metrics on N.

In the subsequent analysis, analogous metric correlation data to what was introduced for three values of N in Table 6.3.1-Table 6.3.3 is generated for a total of 7 different values of N. In this experiment, constrained datasets were generated for the 7 values of N and evaluated with regard to correlation. Subsequently, the change in correlation coefficients and p-values was visualized. An upper limit for the constraint variable on the datasets was set at $N = 15$. Furthermore, the low limit of $N = 3$ was selected as a lower number for the constraint variable N would result in the datasets containing too few samples for a reliable evaluation of statistical correlation.

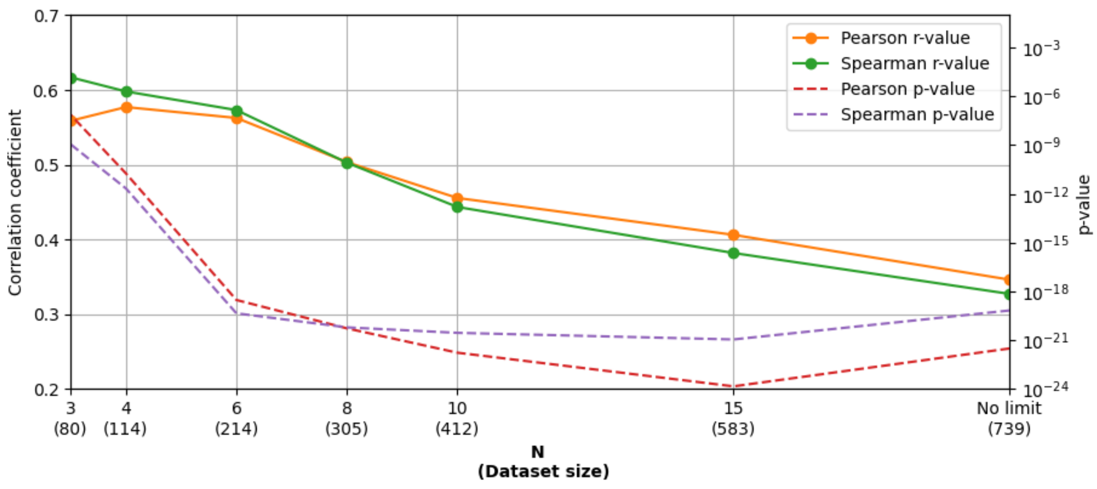
In Figure 6.3.5, the changes induced in the correlation coefficients and their corresponding p-values by different constraints on the numbers of objects are visualized. Examining Figure 6.3.5a (RAW_40) and Figure 6.3.5b (FN_40), there is a consistent increase in the values of the correlation coefficients for decreasing values of N when $N \geq 4$. This increase is the most prominent over the FN_40 dataset, with both coefficient values above 0.50 for samples constrained with $o_i \leq 8$. However, for FP_40, there is not much change in the correlation coefficients, with all values falling in the range 0.2-0.3. Moreover, although there is a slight increase in the values when imposing the restriction $N = 15$, the correlations decrease for lower values of N. The correlation for both coefficients is reported at their lowest values for $N = 4$ (although corresponding p-values at this constraint indicate uncertain statistical significance). The aforementioned observations may indicate an increased difference in penalization of FPs for the two metrics with a decreasing N. If PKL in fact penalizes FP detections more than $F1_{crit}$, as suggested by the data distributions examined in Subsection 6.3.1, this can be explained by considering that for lower numbers of objects present, an injected FP will be more likely to affect the vehicle trajectory, thus accentuating the differences in penalization.

The Spearman coefficient reports a consistently higher correlation than Pearson over the RAW_40 dataset. As the Spearman coefficient is a measure of the strength of a monotonic relationship between variables, this may indicate a relationship that is monotonic and non-linear between metric values over the RAW_40 dataset (Pearson is a measure of a linear relationship). For the fault-injected sets, the correlation coefficients are more analogous, not diverging by more than ≈ 0.04 across all values of N. This indicates that the relationship between the variables is better described as a linear association over the fault-injected datasets (a linear relationship is a special case of a monotonic relationship). This evaluation is, however, susceptible to ambiguity. This is elaborated upon in Subsection 7.2.2.

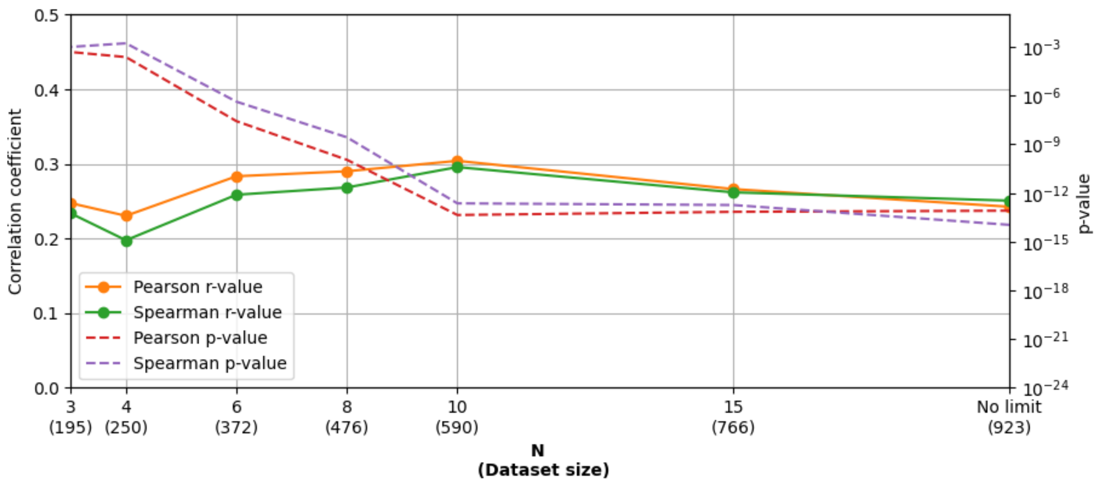
Considering the p-values observed in the plots of Figure 6.3.5, a similar observation to the preceding preliminary analysis of arbitrary values for N is made



(a) Correlation on constrained RAW_40 dataset.



(b) Correlation on constrained FN_40 dataset.



(c) Correlation on constrained FP_40 dataset.

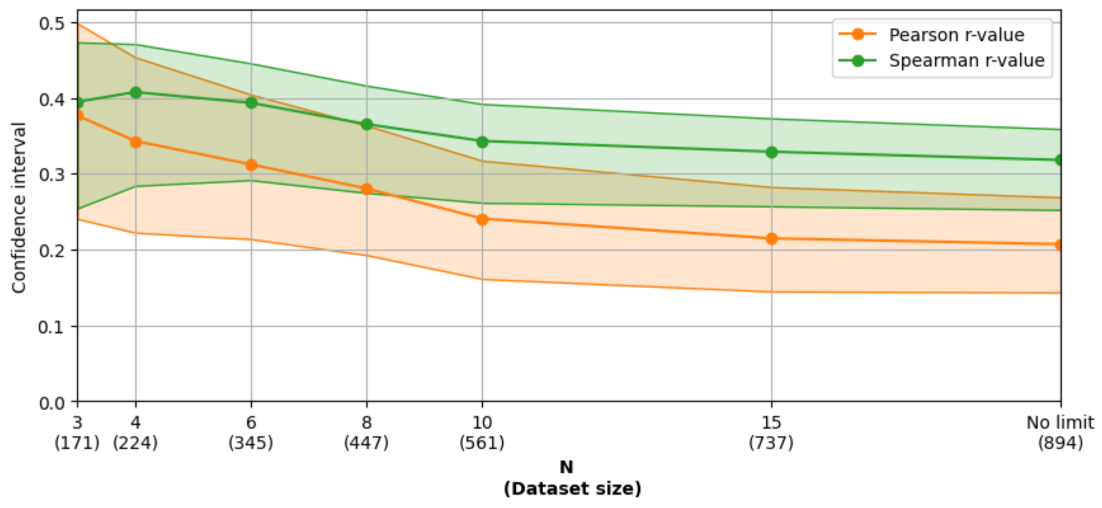
Figure 6.3.5: Change in correlation coefficients and corresponding p-values for decreasing values of N over contained datasets. In the plots, solid lines correspond to the left-hand y-axis and dashed lines correspond to the right-hand y-axis.

for the coefficients over all datasets. Specifically, all p-values stay low enough for statistical significance to be indicated by the results, excluding $N \in \{3, 4\}$ over FP_40. Furthermore, p-values for both correlation coefficients generally decrease with increasing sample size in Figure 6.3.5b and Figure 6.3.5c. This is expected behaviour for increasing sample sizes, implying more data over which the null hypothesis can be evaluated[4]. In Figure 6.3.5a, for $N = 8$, a spike can be observed for p_S and a dip for p_P. For all other values, the p-values show a consistent decline for increasing N-values, with p_S having a considerably lower value. For FN_40, the reason for the increase in p-values from $N = 15$ to $N = \infty$ is likely due to more outliers being included in the dataset as a consequence of an increased number of objects affecting the PKL evaluation (inducing high scores).

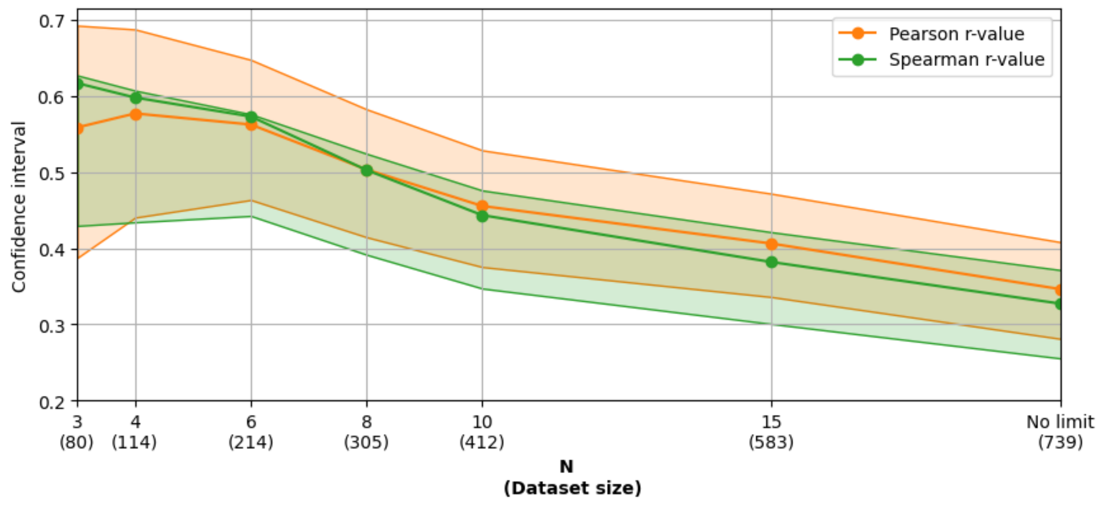
To further analyse the significance of the values observed for r_P and r_S, the confidence intervals for the coefficients are provided. The confidence intervals are computed at 95% confidence and are complementary to the p-values visualized in Figure 6.3.5. In Figure 6.3.6, these confidence intervals are plotted together with the correlation coefficients for the range of values for N examined in the preceding paragraph.

Expectedly, the confidence intervals widen for decreasing values of N and decreasing dataset sizes, accordingly, in Figure 6.3.6a and Figure 6.3.6b (for RAW_40 and FN_40, respectively). However, for both datasets, the intervals are sufficiently narrow to indicate a statistically significant trend for the increase in correlation for decreasing values of the constraint variable N. For RAW_40 and FN_40, the intervals for r_S are more narrow than that of r_P, indicating higher statistical significance in the observed results for the Spearman correlation. However, for FN_40, the interval for r_P is considerably higher than the corresponding Spearman interval with regard their high confidence limits.

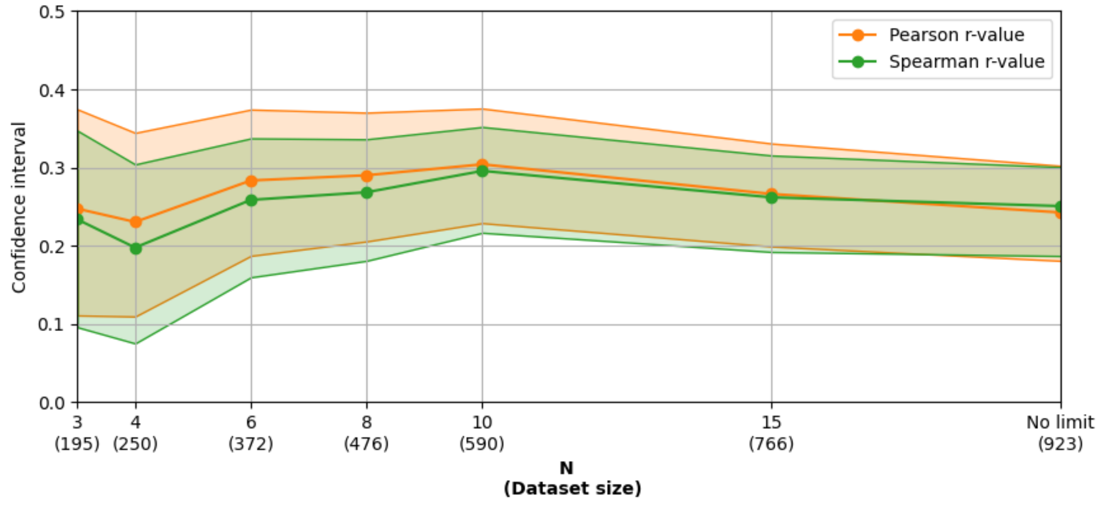
Over the FP_40 dataset, the confidence intervals visualized in Figure 6.3.6c indicate a smaller degree of statistical significance regarding observed values for constraints $N \leq 6$. This is especially prominent in the confidence intervals for r_S. This complements what was observed for the p-values visualized in Figure 6.3.5c.



(a) Correlation coefficients and corresponding p-values for RAW_40 dataset.



(b) Correlation coefficients and corresponding p-values for FN_40 dataset.



(c) Correlation coefficients and corresponding p-values for FN_40 dataset.

Figure 6.3.6: Change in the correlation coefficients and their confidence intervals at 95% confidence.

6.4 Analysing the impact of misdetections

Applying the methodology introduced in Section 4.6, an analysis of the impact of introducing an incremental number of faults in the prediction set of a detector on metric results is performed. In this experiment, predictions made by SSN over 100 nuScenes samples were modified by injecting an incremental number of faults, ranging from 0 to 5. This was performed for both types of injected faults, after which the predictions were evaluated by the metrics. For each increment of the number of faults, the mean of metric results was computed for $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$ (computed from the aforementioned), and PKL. The resulting values were plotted for an increasing number of injected faults. Recall that BBs in the prediction sets of the detector over the samples were filtered at confidence thresholds $\tau = 0.15$ and $\tau = 0.40$ for predictions injected with FPs and FNs, respectively.

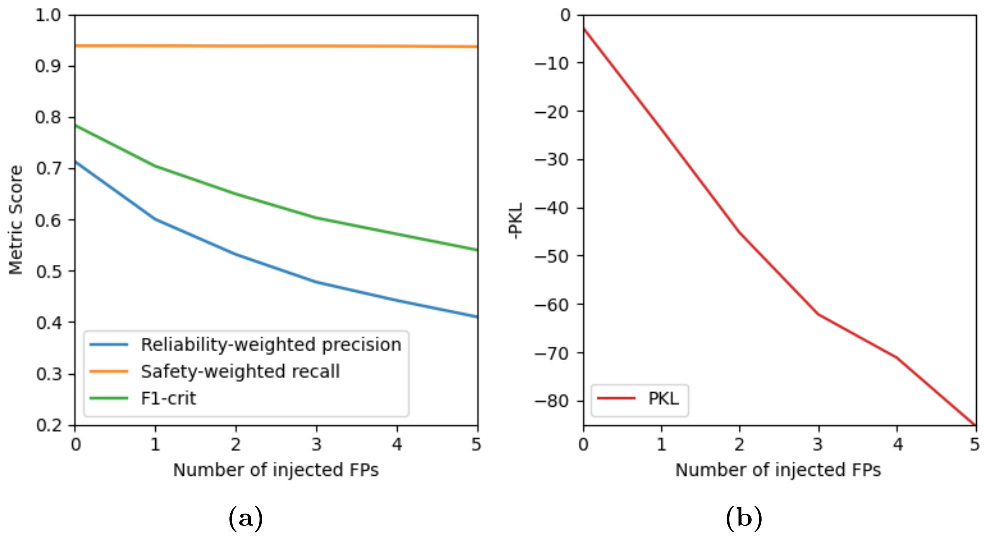
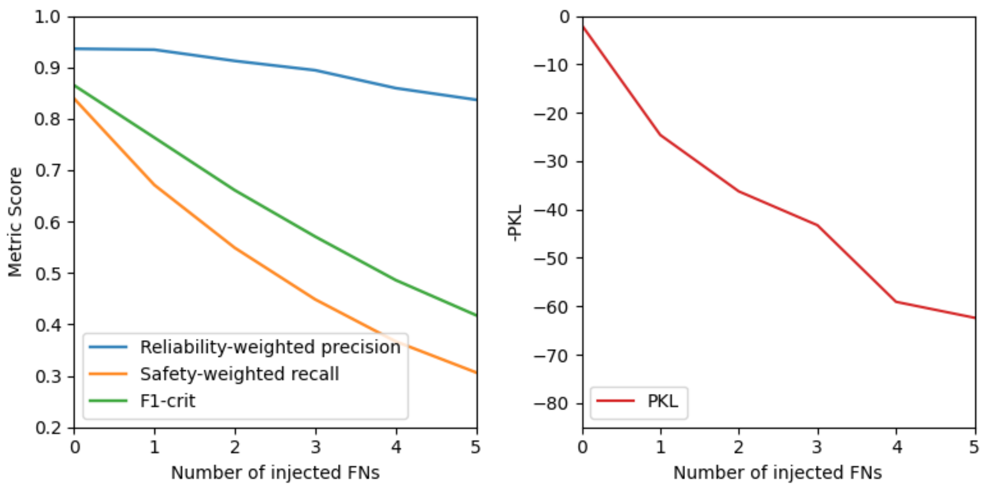


Figure 6.4.1: Mean $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$ and PKL for 0-5 injected False Positives.

In Figure 6.4.1 and Figure 6.4.2, plots visualise the change in the means of metric results for injected FPs and FNs, respectively. For both classes of faults, both $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ are plotted in addition to $F1_{crit}$ to enable an interpretation of how the individual components of $F1_{crit}$ penalize misdetections with regards to safety- and reliability-related faults. Furthermore, in the visualizations provided, PKL values are visualized. For PKL, observed values are negated solely for visualization purposes, to enable comparative visual analysis. Hence, it should be noted that the negative logarithm of PKL values is not applied in this specific experiment. Note that as the number of possible FN injections for a prediction set is the same as the number of correctly predicted BBs in the set, the "real" number of injected FNs was, on average, less than the specified number. The real numbers of FNs injected are presented in Table 6.4.2, and discussed in Subsection 7.2.3.

As can be inferred from Equation 2.27, the safety-weighted recall does not change with an increasing number of FPs. Figure 6.4.1a demonstrates this for $R_{\mathcal{S}}$, which remains constant at 0.94. In contrast, $P_{\mathcal{R}}$ starts at 0.71 and decreases with an increasing number of faults. For $P_{\mathcal{R}}$, the lowest value of 0.41 was observed for 5 faults injected.

For injected FPs, Figure 6.4.1a visualizes observed values for the harmonic mean of $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$, namely $F1_{crit}$. Furthermore, observed values for PKL evaluated over the same dataset are visualized in Figure 6.4.1b. Comparing the aforementioned plots, there appears to be a more prominent decrease[2] in PKL values than in values for $F1_{crit}$ for increasing numbers of injected FPs.



(a) **(b)**

Figure 6.4.2: Mean $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, $F1_{crit}$ and PKL values over 100 samples for 0-5 injected False Negatives.

Considering the injection of FNs, reliability-weighted precision (Equation 2.26) allows for taking on a different value if a correctly predicted BB is removed from the evaluated predictions. For this reason, a decrease in $P_{\mathcal{R}}$ can be observed in Figure 6.4.2a. In turn, this affects $F1_{crit}$ by causing it to take on lower overall values. For $R_{\mathcal{S}}$, a steep decrease of 0.53 is observed between 0 and 5 injected FNs.

Metric	**Injected FPs**	**Injected FNs**
$P_{\mathcal{R}}$	0.30	0.10
$R_{\mathcal{S}}$	0.0	0.53
$F1_{crit}$	0.24	0.45
PKL	83	60

Table 6.4.1: The decrease in means of metric values induced by 5 injected faults.

Complementing the visualizations provided, the total changes in all metric values between 0 and 5 injected faults for both classes of misdetection are presented in Table 6.4.1. While PKL on average penalizes the injection of FPs more than FNs for the injections performed over SSN predictions for the samples selected, the opposite is true for $F1_{crit}$ in the observed results on the FN-injected data. However, the magnitude of the difference in penalization for the two metrics differs.

[2]Note that a decrease for PKL values in these plots implies an increase in true PKL values.

The overall decreases in the evaluation of PKL corresponding to the injection of 5 injections of FPs and FNs are 83 and 60, respectively. For $F1_{crit}$, these decreases are 0.24 and 0.45. This suggests that PKL penalizes FNs only 40% what $F1_{crit}$ does, but FPs 259% more than $F1_{crit}$. For the data presented in Table 6.4.1, this implies that PKL penalizes FPs more severely than FNs by a relative difference of 138.3%. Instead, $F1_{crit}$ penalizes FNs more severely than FPs by a difference of 187.5% for the data evaluated. The possible causes of these results are investigated in Subsection 7.2.3.

Injecting false negative detections into the predictions of an object detector entails removing a correctly predicted object from its prediction set. Thus, the number of FNs injected in the prediction set of a detector for a specific sample depends on the predictions of the detector, and thus on the number of GT objects present in the sample. Furthermore, the number of FNs injected depends on the argument provided to the method of removal presented in Subsection 5.3.3, specifically the Euclidean distance up to which objects are considered for removal. Thus, the number of such faults present in predictions is not always equivalent to what is implied in Figure 6.4.1 and Figure 6.4.2 for all samples. This is due to the fact that for some prediction sets, the number of correctly predicted BBs is less than the number of injected faults specified. Hence, for each number of FNs intended to be injected, the true, average number of FNs present in the resulting predictions are computed and displayed in Table 6.4.2. The implications of these results, presented in Table 6.4.2, are discussed in Subsection 7.1.2.

Injected FNs	**Mean FNs**
1	0.951
2	1.90
3	2.86
4	3.75
5	4.55

Table 6.4.2: True numbers of injected false negatives.

6.5 Comparative analysis of precision, recall, and F1-score with their safety-oriented adaptations

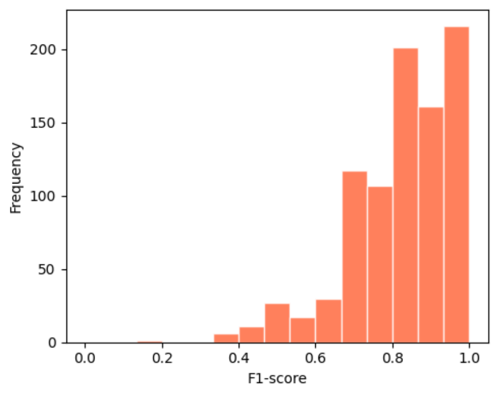
In this section, an analysis of the generic performance measures of precision, recall and F1-score is performed. By contrasting the experimental results presented in the preceding sections of this chapter, the evaluations of the metrics are visualized and assessed over similar data. More specifically, similar experiments to what is introduced in Subsection 4.5.1 and Section 4.6 are performed utilizing the same datasets described in these methodologies. Thus, a comparison of the generic metrics with PKL and OCM-related metrics is performed. The objective of this analysis is to understand the relationship between the aforementioned, generic metrics and their safety-oriented counterparts, namely OCM-related metrics and PKL. A better understanding of this relationship gives context to the preceding experimental results presented by comparative analysis with commonly applied metrics for object detectors in the field of autonomous driving.

6.5.1 Analysis of metric distributions

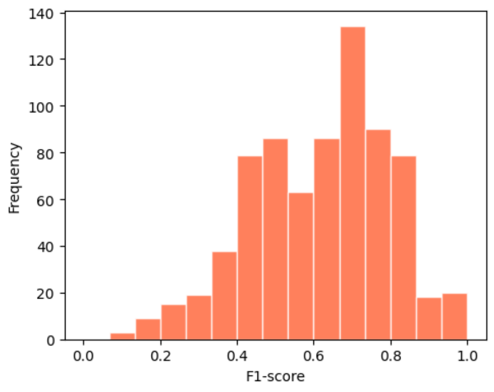
In the subsequent experimental results, data visualizations for the joint distributions between PKL and precision, recall, and F1-score are presented. By examining these results in the context of results presented in Subsection 6.3.1, a comparative analysis of the generic metrics with their OCM-related counterpart is subsequently performed. The method with which the experimental work was performed is analogous to the method introduced in Subsection 4.5.1, only differing with regard to the metrics that are analyzed.

In Figure 6.5.1, the distributions of metric results for the F1-score over the three datasets summarized in Table 4.2.1 are visualized through histograms. In each histogram, metric values are divided into 15 bins of uniform width, with the height of the bars representing each bin indicating the frequency of occurrence for each range of metric values. Seeing the distributions in the context of the same distributions visualized for $F1_{crit}$ in Subsection 6.3.1, some important distinctions can be made between the measures. Considering the distribution of the F1-score in RAW_40, it is observed that the frequency of values does not decrease monotonically for lower scores. Rather, higher frequencies of values are observed for specific bins. This indicates that for the F1-score, prediction evaluations are more likely to fall into specific intervals of values for the RAW_40 dataset, represented by the bins of the histograms. Considering evaluations over the fault-injected datasets, the histograms depicted in Figure 6.5.1b and Figure 6.5.1c also reflect the inclination of F1-score values to fall into specific intervals of values. Compared to the distributions for $F1_{crit}$ presented in Subsection 6.3.1, the highest bars in the histograms representing the F1-score distribution over fault-injected datasets are considerably higher.

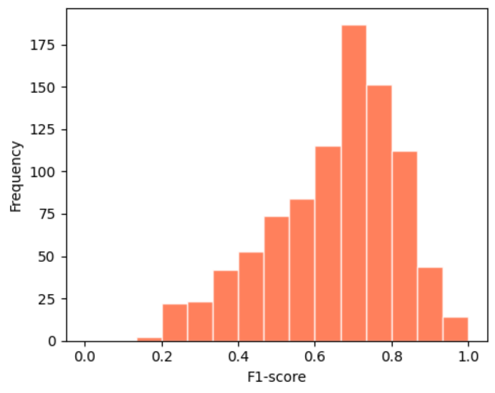
Contrasting the plots depicted Figure 6.2.2 and Figure 6.2.4, metric results of precision, recall, and F1-score are plotted against PKL in the scatter plots depicted in Figure 6.5.2 for the three datasets RAW_40, FN_40, and FP_40. In the plots depicted in Figure 6.5.2, the bands of similar values for the F1-scores confirm the aforementioned observations and explain the high frequencies of occurrence in specific bins of the histograms depicted in Figure 6.5.1. For the metric results over



(a) Histogram for the RAW_40 dataset.



(b) Histogram for the FN_40 dataset.

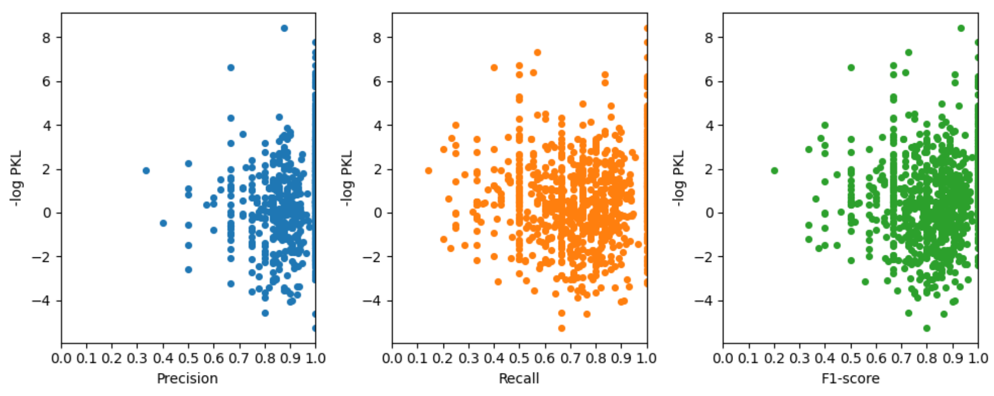


(c) Histogram for the FP_40 dataset.

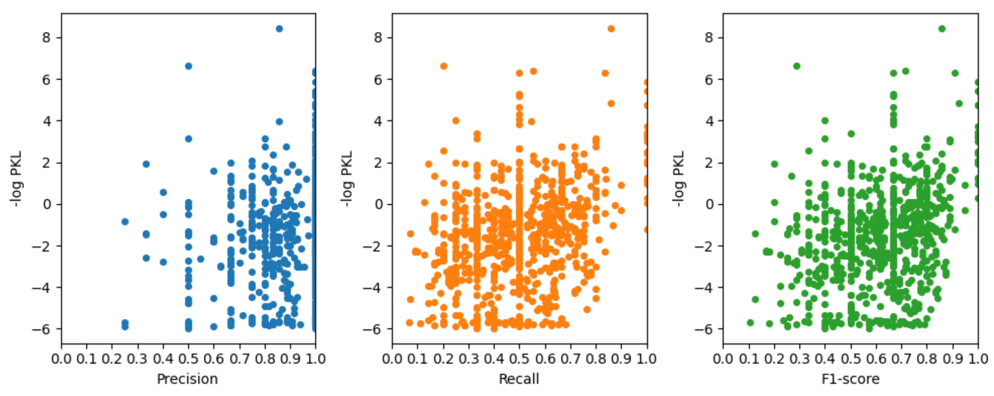
Figure 6.5.1: Figures (a) and (b) show the frequency of occurrence for metric values in the RAW_40, FN_40, and FP_40 datasets, respectively.

the fault-injected datasets, it can be observed that very few data points correspond to a maximum F1-score of 1.0. This is expected, as all generic detection metrics applied by definition equally reflect all predictions considered upon evaluation. This implies that fault-injected predictions can never reach a perfect score when evaluated with such metrics. Moreover, this indicates that the few data points corresponding to precision, recall, and F1-score values of 1.0 in turn correspond to samples where no faults were introduced due to the random nature of the number of injected faults described in Section 4.3. Conversely, for OCM-related metrics, if injected FPs or FNs correspond to objects that have a criticality 0, $F1_{crit}$ may evaluate to a perfect score even with such injections performed.

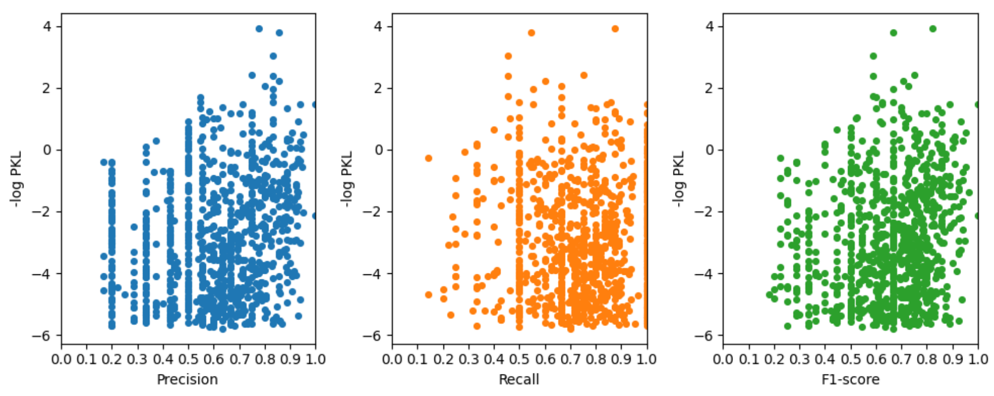
The aforementioned observations of bands of precision, recall, and F1-score values can be explained by considering the binary concepts of TP/FP predictions applied in the generic metrics and the number of objects present in a sample for which predictions are evaluated. Consider recall computed for predictions on a single sample in which three GTs are present. In this case, the possible values recall can take on when evaluating the predictions are limited. More specifically, since the denominator of Equation 2.9 will sum to the number of GTs, there are 4 possible values recall can take on, namely Recall $\in \{0, \frac{1}{3}, \frac{2}{3}, 1\}$. The definition of precision is more flexible with regard to the values it can take on, depending on the number of false positives predicted. However, in practice, for a reasonably good



(a) Scatter plot for metric data over the RAW_40 dataset.



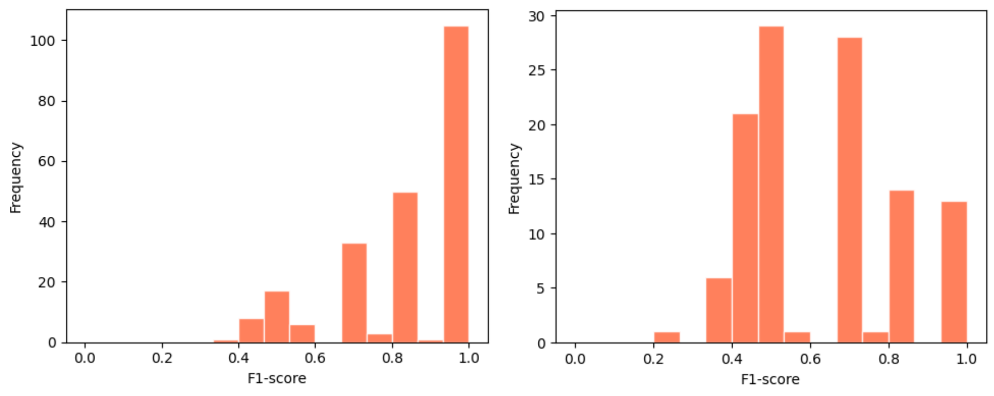
(b) Scatter plot for metric data over the FN_40 dataset.



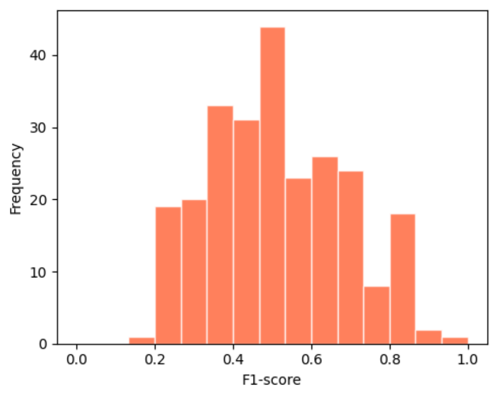
(c) Scatter plot for metric data over the FP_40 dataset.

Figure 6.5.2: Figures (a), (b), and (c) visualize the joint distributions between PKL and traditional precision, recall and F1-score computed over the RAW_40, FN_40, and FP_40 datasets, respectively. Bands of metric values for the generic metrics can be observed.

object detector and predictions filtered at a confidence threshold, the number of FPs should not be too high. This would limit the number of values precision can take on (by Equation 2.26) and induce similar behaviour for precision values to what is observed in Figure 6.5.2. Considering the aforementioned discussion, the number of values the generic metrics can take on will decrease with the number of objects present. In the subsequent analysis, the distributions of metric values are visualized for a restriction on the maximum number of objects present in the samples evaluated. A similar approach to what was performed in Subsection 6.3.1 is applied, and metric distributions are visualized for $N = 4$, limiting the maximum amount of GT objects to 4.



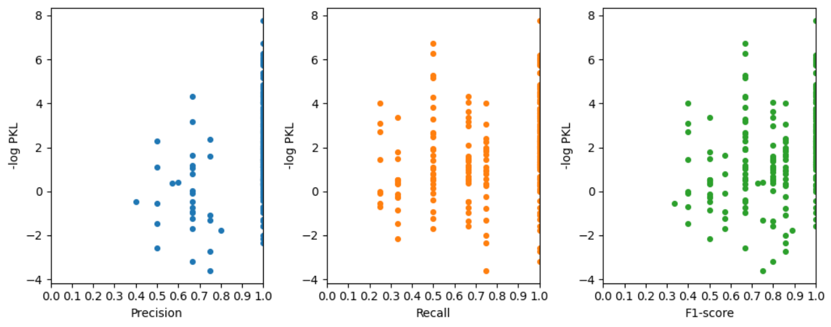
(a) Histogram for the RAW_40 dataset. **(b)** Histogram for the FN_40 dataset.



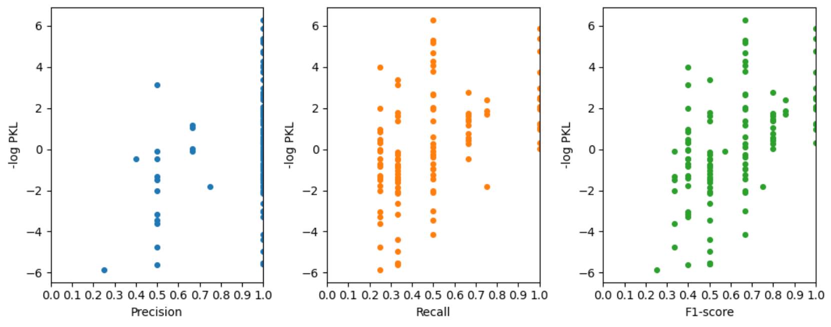
(c) Histogram for the FP_40 dataset.

Figure 6.5.3: Figures (a) and (b) show the frequency of occurrence for metric values in the constrained datasets RAW_40, FN_40, and FP_40, respectively, where $N = 4$. It is observed that most values fall into a number of specific intervals.

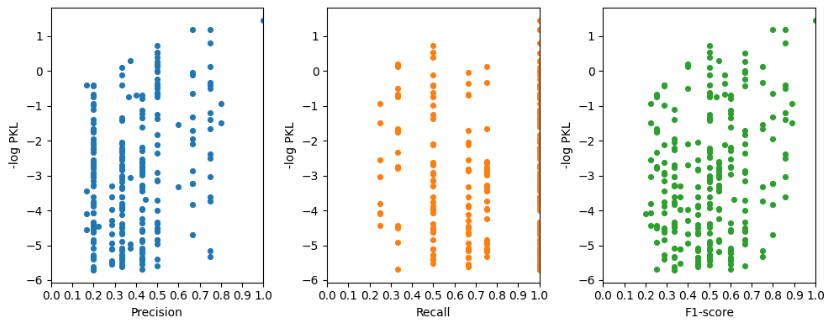
In Figure 6.5.3, the histograms representing the distributions of F1-score values evaluated over the constrained datasets are depicted. For metric results over the RAW_40 and FN_40 datasets, similar to the indications of preceding observations, it can be observed that metric values to a great extent fall into specific bins. However, in contrast to the histograms depicted in Figure 6.5.1a and Figure 6.5.1b, values for F1-score almost exclusively fall these intervals when $N = 4$ is applied. This can be explained by considering the aforementioned example, in which the reduced numbers of GTs in the samples evaluated cause restrictions on the possible outcomes of evaluating the predictions with precision, recall, and consequently F1-score. For predictions over the FP_40 dataset, F1-scores are more evenly distributed in the central bins of the histogram. As discussed in the preceding paragraph, this is likely due to the dependence of precision on the number of FP predictions. As there can be false positives present in detector predictions prior to the injection of faults, and due to the possible number of FPs injected, precision can take on more values when evaluated over the FP_40 dataset.



(a) Scatter plot for metric data over the RAW_40 dataset.



(b) Scatter plot for metric data over the FN_40 dataset.



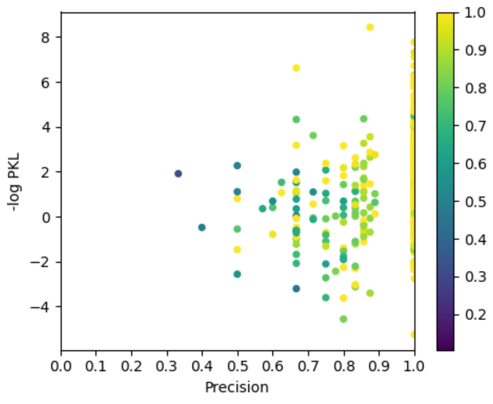
(c) Scatter plot for metric data over the FP_40 dataset.

Figure 6.5.4: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets RAW_40, FN_40, and FP_40, respectively, where $N = 4$. The bands of values for generic metrics are increasingly visible for lower values of N.

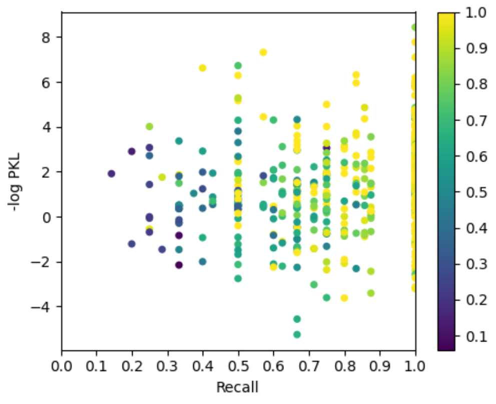
Figure 6.5.4 visualizes the joint distributions of PKL and the generic detection metrics for the three constrained datasets. Expectedly, more clearly defined bands of values for the F1-score are observed with $N = 4$. In Figure 6.5.4b, the bands corresponding to different numbers of GTs present in the samples can be observed. Recall that in the aforementioned example, considering the possible values of recall with three GT objects represented in a sample, the four values recall could take one were in $\{0, \frac{1}{3}, \frac{2}{3}, 1\}$. In Figure 6.5.4b, bands of metric data points can be observed at the latter three values, corresponding to samples with three objects present. Furthermore, for samples with four GT objects, bands of values can be observed for all possible recall values, namely values in Recall $\in \{0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1\}$, excluding 0. Similar results can be observed, corresponding to samples with two ground-truth objects present. For precision over the FP_40 dataset, similar bands of results can be observed, although the number of possible values has a greater degree of freedom as a consequence of a varying number of FPs present in the predictions of the detector. Thus, the distribution for the F1-score, being the harmonic mean of the two prior, follows a similar pattern and reflects the possible values the measures can take on.

Comparing the aforementioned observations with the distributions observed in Figure 6.3.4, it has been observed that OCM-related metrics *can* be more descriptive of detector performance with regards to the context of the detection when evaluated over small datasets (e.g. that of a single sample), compared to generic evaluation metrics. This comes as a result of the greater degree of freedom in possible values the metrics can evaluate, utilizing the concept of criticality values for objects.

It has been observed that the set of possible metric scores for precision and recall are restricted compared to their safety-oriented adaptations proposed in [2]. This consequently causes differences in the evaluations of the respective metrics. To better understand these differences between specific evaluations of predictions with generic and OCM-related metrics, the joint distributions between PKL and the generic precision and recall are visualized with data points coloured based on their corresponding values for $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$. Thus, the subsequent visualizations represent metric values for three metrics, namely PKL and corresponding precision and recall-related metrics. In the subsequent illustrations, the datasets representing the metric distributions are constrained with $N = 8$.



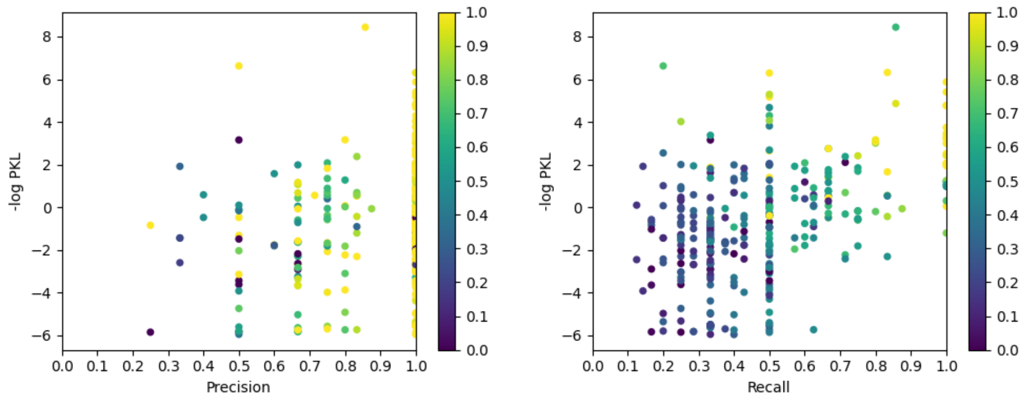
(a) Scatter plot for precision.



(b) Scatter plot for recall.

Figure 6.5.5: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets RAW_40, where $N = 8$ and dots are coloured based on corresponding OCM-related metrics.

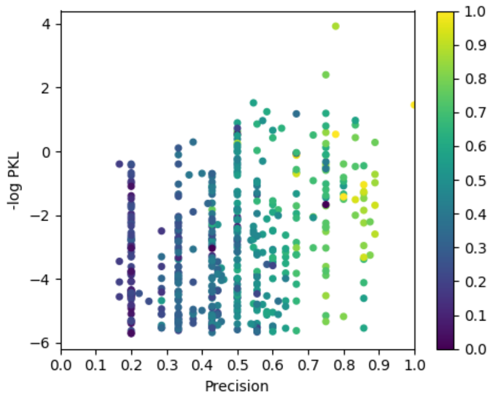
The joint distribution between PKL and generic precision and recall for the constrained dataset RAW_40 is visualized in Figure 6.5.5. Each data point in the scatter plot is coloured based on its corresponding values for reliability-weighted precision and safety-weighted recall. Examining the distribution, it is evident that while some metric values for the generic and safety-oriented metrics align, a substantial number of predictions are evaluated differently by these metrics. Specifically, both precision and recall show that many predictions, which are scored perfectly by the OCM-related metrics, perform worse under the generic variants. This is evident considering the number of yellow-coloured data points not exhibiting perfect evaluation for precision and recall in Figure 6.5.5. Although most predictions that receive perfect scores in terms of OCM-related metrics also evaluate to 1.0 under the generic metrics for precision and recall, the opposite is not true for a significant number of predictions. This is expected, as predictions that contain FNs or FPs that correspond to low-criticality objects would be less reflected in the OCM-related metrics than in their generic counterparts. However, the degree to which predictions are evaluated differently by the two approaches is significant and reflects the greater degree of values OCM-related measures can take on.



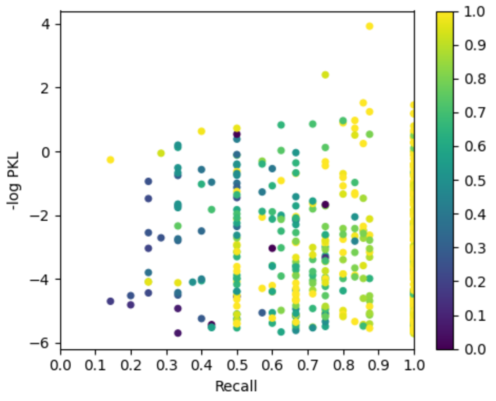
(a) Scatter plot for precision. **(b)** Scatter plot for recall.

Figure 6.5.6: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets FN_40, where $N = 8$ and dots are coloured based on corresponding OCM-related metrics.

For predictions over the FN_40 dataset, Figure 6.5.6 visualizes the joint distribution of metric results for PKL and the generic precision and recall, with the colour of dots in the scatter plots indicating the corresponding scores for OCM-related metrics. What was observed for precision over the RAW_40 dataset, is also the case in the joint distribution depicted in Figure 6.5.6a, with several predictions scoring higher for $P_{\mathcal{R}}$ than for precision. A similar observation can be made in the case of the recall measure, whose distribution is illustrated in Figure 6.5.6b. Predictions that assume high values for recall exhibit higher values with regard to safety-weighted recall. Conversely, low-scoring predictions for the recall measure achieve lower scores for the OCM-related recall metric. Comparing with the distribution illustrated in Figure 6.5.5b, there is an increase in data points that fall in the range 0.1-0.4, most of which assume lower values with regards to $R_{\mathcal{S}}$. It is indicated by the colour of these dots that predictions within the range of 0-0.1 for $R_{\mathcal{S}}$ do not assume scores below 0.1 for ordinary recall. This highlights a disparity between OCM-related metrics and generic metrics. By definition, when any number of true positive predictions exist in the set of predictions generated by an object detector, recall cannot assume a value of 0. The same is true regarding perfect scores for recall, namely that recall cannot exhibit a perfect score if any number of FNs are present in the prediction set. This implies that predictions that evaluate to perfect scores for both recall and safety-weighted recall over the fault-injected FN_40 dataset are predictions in which no FN has been injected due to the random nature of the method for injection described in Section 4.3. These reservations emphasize the difference in the potential range of evaluation scores between generic recall and safety-weighted recall.



(a) Scatter plot for precision.



(b) Scatter plot for recall.

Figure 6.5.7: Figures (a), (b), and (c) show the joint metric distributions for the constrained datasets FP_40, where $N = 8$ and dots are coloured based on corresponding OCM-related metrics.

Figure 6.5.7 shows similar distributions to the aforementioned, but for the FP_40 dataset. In the scatter plot, for precision and recall, it can be observed that the metric values in many cases correspond to predictions that are evaluated to lower values for $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$, respectively. For precision, the predictions for which this is the case likely correspond to high-criticality objects being injected, in which case they are reflected more negatively in the safety-oriented adaption of precision. However, this can be observed in particular for recall, with many low evaluations for safety-related recall corresponding to high values in the plot depicted in Figure 6.5.6b. The opposite is also observed, namely high-scoring predictions for $R_{\mathcal{S}}$ performing worse with regards to recall. This indicates that there are many scenarios in which predictions contain FN predictions, where the corresponding objects are of low criticality with regard to the OCM.

6.5.2 Analysing the impact of misdetections

Following an analogous method to what is performed in Section 4.6, this part of the experimental results examines the impact of injecting faults into the predictions of a detector on the generic metrics examined. More specifically, utilizing the predictions of SSN over a set of 100 nuScenes samples, an incremental number of false positives and false negatives are injected and the means of metric evaluations are computed. The number of FPs and FNs injected ranges from 0 to 5, and mean metric values are plotted for precision, recall and F1-score. As introduced in section Section 4.6, BBs in the prediction sets of the detector over the samples are filtered at confidence threshold $\tau = 0.15$ and $\tau = 0.40$ for predictions injected with FPs and FNs, respectively. The subsequent experimental results are seen in the context of the experimental results introduced in Section 6.4, with the objective of investigating the difference in penalization of misdetections between OCM-related metrics and their generic counterparts.

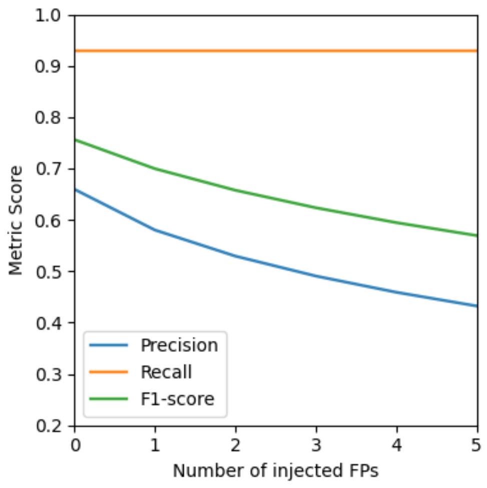


Figure 6.5.8: Mean precision, recall and F1-score over 100 samples for 0-5 injected False Positives.

In Figure 6.5.8, the decrease in mean precision, recall, and F1-score values as a consequence of injecting an incremental number of FPs into the prediction sets of SSN over the samples is visualized. As observed in Section 6.4, recall remains constant at $R = 0.94$ for an increasing number of FP injections. Expectedly, a decrease in precision, and consequently in the F1-score, is observed for an incremental number of FP injections. With precision at $P = 0.65$, the unmodified predictions over the dataset evaluate to a lower average value for precision compared to $P_{\mathcal{R}}$ over equivalent data. Furthermore, a less steep decrease in precision and F1-score as a consequence of the injection of increasing numbers of FPs is observed when compared with analogous results for reliability-weighted precision and $F1_{crit}$. The lowest value for precision is observed at $P = 0.43$, slightly above the lowest observed value for $P_{\mathcal{R}}$. The aforementioned results imply that the injected FPs are more harshly penalized by reliability-weighted precision in contrast to what is reflected in the decrease in precision, as a result of high criticalities for injected BBs. As discussed in Section 4.6, FP injections are generated at a location in close enough proximity to ego to likely be of some importance in terms of criticality. Thus, as high-criticality objects are injected as false positive detections, the aforementioned results indicate that the OCM correctly reflects the importance of correctly predicting objects in proximity to ego.

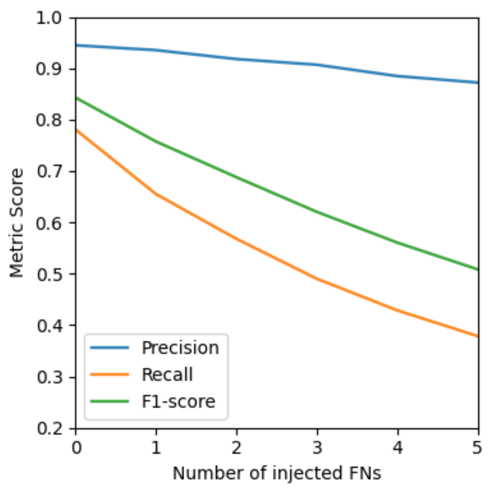


Figure 6.5.9: Mean precision, recall and F1-score over 100 samples for 0-5 injected False Negatives.

Figure 6.5.9 visualizes the decrease in mean precision, recall, and F1-score values as a consequence of injecting an incremental number of FNs into the prediction sets of SSN over the samples. Similar to results for $P_{\mathcal{R}}$ presented in Section 6.4 for injected false negative predictions, a slight decrease can be observed for precision in Figure 6.5.9. Although the decreases in the respective precision-based metrics are similar, there is a minor difference between the two values. Specifically, while the lowest observed value for reliability-weighted precision is at $P_{\mathcal{R}} = 0.85$, evaluated at 5 injected faults, the lowest observed value for precision is $P = 0.87$, for the same number of injected FNs. Considering recall, there is a steep decrease observed for incremental numbers of injected faults. With observed values ranging from $R = 0.78$ (0 FNs) to $R = 0.38$ (5 FNs), this represents an overall decrease of 0.40 in mean recall values. However, a larger decrease is observed for $R_{\mathcal{S}}$ in the results discussed in Section 6.4. Consequently, a larger decrease in $F1_{crit}$ is observed, in contrast to the F1-score. This indicates that safety-weighted recall penalizes the injected FNs more than the traditional recall measure. This is a consequence of the removed BBs assuming higher values for criticality than other objects in the scene, reflecting the removal of objects based on distance from ego as introduced in Subsection 5.3.3. The aforementioned observations show that traditional recall does not penalize the FN predictions injected as severely as its OCM-related counterpart. The same is true for precision and F1-score.

Table 6.5.1 presents the difference in mean metric scores for precision, recall and F1-score resulting from the injection of 5 faults for both FPs and FNs. Contrasting this table with results for the OCM-related metrics, presented in parentheses in Table 6.5.1, the differences in these results give an indication of the difference in penalization of faults between generic metrics and their safety-oriented adaptations as introduced in [2]. In the context of similar results for the OCM-related metrics, it can be observed that the differences in evaluation scores induced by injecting 5 faults are larger for all safety-oriented adaptions of the

Metric	**Injected FPs**	**Injected FNs**
Precision	0.23 (0.30)	0.07 (0.10)
Recall	0.0 (0.0)	0.40 (0.53)
F1-score	0.19 (0.24)	0.33 (0.45)
PKL	83	60

Table 6.5.1: The decrease in means of generic metric values induced by 5 injected faults. Corresponding decreases for OCM-related metrics $P_{\mathcal{R}}$, $R_{\mathcal{S}}$, and $F1_{crit}$ are shown in parentheses and PKL is included.

generic metrics. This validates a larger penalization of faults by the OCM-related metrics.

CHAPTER
SEVEN

DISCUSSION

In this thesis, two proposed safety-related evaluation metrics for object detectors were analysed and assessed on the nuScenes dataset [32]. The metrics were two novel approaches for proposing a metric that considers the context in which detections are made in the evaluation process. In this process, an experimental methodology was established and applied to analyse the specific attributes of the metrics, both independently and in relation to each other. Although two metrics were selected for analysis in this experimental work, the approach taken in their evaluation can be applied to the analysis of other similar metrics. In the work preceding this thesis, a literature review was conducted, identifying other approaches [1].

7.1 Considerations and limitations

7.1.1 Significance of the confidence threshold

When analyzing different datasets of detections that have been subject to fault injection, the choice of confidence threshold on which to filter predictions before performing evaluation is critical. For a low threshold, sample predictions will contain an abundance of false positives prior to the injection of faults. Conversely, for a high threshold, predictions are likely to contain false negatives prior to injection. As discussed, the PKL and OCM-related metrics have significant differences with regard to their evaluation of system safety and reliability, and with regards to their method of evaluation. Specifically, PKL provides a single-value evaluation of a specific set of predictions filtered at a predetermined confidence threshold. Furthermore, PKL consolidates the evaluation of safety and reliability-related faults into its single evaluation score, namely the Planning KL-divergence [10]. Ceccarelli and Montecchi [2], on the other hand, propose separate measures for evaluating reliability and safety.

The experimental methodology introduced in Section 4.6 was developed to explore the *impact* of injecting two types of fault in detector predictions on the metrics examined. Hence, the objective of the experiment was to quantify the change occurring in metric evaluations as a result of the aforementioned injections. When performing such an experiment, the choice of confidence threshold on which to filter predictions is important. Consider the injection of an incremental number of FP faults in a set of detector predictions where the detection model has failed to predict a number of GTs. For the predictions evaluated, the number of FNs would remain constant with an incremental number of FPs injected. As has been demonstrated, the $P_{\mathcal{R}}$ measure is dependent on the number of TP predictions present, and will thus be impacted by the FNs present prior to injection. Furthermore, it was demonstrated in Section 6.1 that the injection of analogous misdetections can induce a different change in PKL scores depending on the faults present in the prediction set prior to injection. This illustrates the necessity of limiting the number of faults represented by the predicted BBs when analysing the impact of misdetections.

In the experimental work presented in Section 6.4, PKL and OCM-related metrics are analysed in terms of their degree of penalizing FN and FP predictions. As was demonstrated, choosing a high confidence threshold when evaluating a dataset that has been subject to FP injection may lead to findings being less interpretable due to the presence of FNs in the predictions. Conversely, selecting a low threshold when analyzing the impact of injected FNs may cause a problem of interpretability when assessing safety-weighted recall. In the methodology applied in the experiment, this was addressed by selecting two different thresholds for the analysis of the impact of FPs and FNs, respectively. In addition, comprising the two OCM-related measures into the $F1_{crit}$ metric (introduced in Equation 4.1) provided a single evaluation score for OCM-related measures, with which results for PKL could be compared, promoting the interpretability of the results. In Section 7.4, other approaches to analysing the impact of misdetections on the evaluation metrics are explored.

For analysing metric correlation, the methodology introduced in Section 4.5 was applied. In the experiment proposed, metric evaluation was performed for predictions filtered at a threshold of $\tau = 0.40$. This threshold was selected to represent an arbitrary mid-range threshold at which an abundance of FPs and FNs were not present in the predictions. Furthermore, this choice was made to limit the scope of the experimental work and was justified by limiting the scope of the analysis to measuring the correlation between PKL and $F1_{crit}$. By focusing on the $F1_{crit}$ measure, the impact of the aforementioned ambiguity on results is limited by consolidating safety and reliability-related factors assessed by the OCM-related metrics into one score that could be assessed at a specific confidence threshold.

7.1.2 Methods for injection and datasets

For the analysis of quantitative metric data performed in the experimental work presented in Chapter 4 and Chapter 6, metric data for fault-injected predictions over nuScenes samples were produced. Specifically, when generating the fault-injected datasets of predictions over nuScenes samples resulting in metric data utilized in the experiments introduced in Section 4.5 and Section 4.6, the methods for injecting faults described in Subsection 5.3.3 were applied. Hence, results collected in the aforementioned experiments were influenced by the implementation of the methods for fault injection, and the arguments provided to these methods upon injecting false positive or false negative predictions. In this section, the implementational choices considering methods for injection and the generating of datasets utilized will be discussed.

As described in Subsection 5.3.3, the methods implemented for injection enable inserting or removing predicted bounding boxes from the prediction sets resulting from inference by an object detector on a single nuScenes sample. When injecting FP predictions, i.e. inserting BBs into a prediction set, a number of properties for the injected BBs are predetermined based on the specific scenario, and thus equivalent for all FP predictions injected for one sample. While some of these predetermined properties are insignificant for the results of the subsequent metric evaluation of predictions, such as the detection score (the confidence score), others have an impact on these results. Specifically, the orientation and z-coordinate of an injected FP detection are always initialized as equivalent to their corresponding values for ego. These implementational choices were made for simplicity in utilizing the aforementioned methods in constructing larger datasets of fault-injected predictions corresponding to detections on nuScenes samples.

The predetermined orientation of an injected BB should be seen in context with the possible initialized values for the velocity of the object. As seen in Subsection 5.3.3, the velocity of an injected object either corresponds to 0 or to the ego velocity, depending on the provided argument. The combination of these two restrictions on the possible scenarios simulated by the injection of an FP has the consequence of limiting the possible number of generated detection scenarios that correspond to reliability issues for autonomous systems. For example, it is impossible to inject a false positive prediction corresponding to a vehicle approaching ego. It should be noted that the aforementioned limitations arising from the discussed implementational choices are limitations on the possible scenarios simulated by injecting FPs, not on the comparative analysis of safety-oriented metrics. Despite the limitations on the possible scenarios simulated, the scenarios that were simulated in the experimental work of this thesis represented plausible misdetections upon which the differences between metric evaluations could be examined.

Considering the z-coordinate of injected FPs, initializing this value to be equivalent to the corresponding property for ego can generate scenarios in which the BB injected is initialized with a z-coordinate not corresponding to ground level. There were no identified limitations induced by this on the experimental work of analysing metric evaluations. However, the extent to which this affects the metric results (specifically for PKL) can be investigated further.

When injecting FN detections, i.e. removing a correctly predicted BB from the prediction set, a parameter is provided to the method described in Subsection 5.3.3, indicating the distance up to which predicted BBs are considered for removal. This parameter should be seen in the context of the other parameter provided to the method, namely the parameter indicating whether to remove all or one single BB within the distance specified. As seen in the description of the method provided, and in the code provided in Appendix B, the most proximate TP in relation to ego is removed first when injecting FNs. Upon generating the dataset FN_40, this may have had the consequence of introducing FNs that are more proximate to ego compared to the corresponding injections of FPs, culminating in the dataset FP_40. This should be considered when analysing results for the experimental work utilizing these datasets.

In the experimental work introduced in Section 4.5, datasets of predictions injected with faults were generated. Subsequently, analyses of metric evaluations over these datasets were performed. As discussed in Section 4.2, a degree of randomness was introduced when generating such datasets. Specifically, for selecting the number of faults injected and for the location, size and speed of injected FPs, a random element was applied in the methods applied. Similarly, a degree of randomness was involved in determining the distance up to which the injection of FNs is considered for objects. For the experiment introduced in Section 4.6, similar datasets were generated by introducing the same random variables, excluding the number of faults injected and the distance considered in the injection of FNs.

Considering both classes of faults introduced, the difference in the means for injecting them induces a disparity between the number of injections for each class. This was observed in the experimental results presented in Section 6.4, in which the number of FNs observed was lower on average than the number of FPs, for the same number of injections implied by the number of calls to the injection methods described in Subsection 5.3.3. Moreover, for the experimental work examining metric correlation, the dataset FN_40 was generated with a random value for the distance up to which injections were considered. This further restricted the number of FN injections possible for each sample in the experimental work introduced in Section 4.5. This specific methodology applied for the injection of faults into the dataset upon evaluation represented in FN_40 were made were presented in Section 4.3.

7.1.3 Choice of parameters

As discussed in Section 5.1, all experiments performed and presented in this thesis apply the same parameters considered when computing criticality weights for objects with the OCM. The values of these parameters were $D_{max} = 30.0$, $R_{max} = 20.0$, and $T_{max} = 10.0$. Furthermore, as discussed, the choice of these values was based on having sufficiently high values for the three parameters for considering as many possible scenarios in which the perceived objects may interfere with ego. However, it might be the case that an increase in correlation can be observed when considering lower limits represented by these parameters. As the OCM computes future trajectories for objects by extrapolating linear trajectories represented by the speed and direction of movement in a specific time-step, selecting a lower value T_{max} may cause more realistic future trajectories, and thus criticalities, to be generated. This should be considered when interpreting experimental results presented in this thesis and in any future work on the subject.

As reported in the work by Ceccarelli and Montecchi [2], different values for the aforementioned parameters consequently produce different results when considering the evaluations with regard to reliability-weighted precision and safety-weighted recall. As a consequence, different instantiations of these values will lead to different results for the experiments performed. Although analysing the impact of this choice of parameters on the experiments performed is outside the scope of this thesis, such an analysis would add depth to the experimental results and allow for more understanding considering the relationship between OCM-related metrics and PKL. This is elaborated upon in Section 7.4.

The 2D center distance on the ground plane is the match criteria applied when discriminating between TP and FP predictions over the nuScenes dataset [32]. In all experiments performed, the limit for 2D center distance is set to the constant value $d = 2.0$. This choice was made to restrict the scope of the analysis, and thus to perform and present a more comprehensible analysis of metric evaluations. Although the choice of center distance limit will not impact PKL evaluations, OCM-related metrics will reflect this choice of match criteria. Thus, a different limit for center distance would cause different evaluations for OCM-related metrics.

7.1.4 Dataset limitations

In the results presented in Table 6.4.2, it is observed that the true numbers of FN detections introduced in the fault-injected datasets diverge from the number of intended FNs as the latter increases. Thus, the amount of uncertainty associated with the observations also increases for an increase in the number of injected FNs. However, the values presented in Table 6.4.2 stay sufficiently close to the intended number of injections to present significant, although not conclusive, findings. Hence, for a more in-depth analysis of the impact of injecting false negative detections, the predictions evaluated could be verified to contain the number of FNs injected. One way of doing this is to draw the samples evaluated from a distribution of samples in which the number of ground truth detections is higher than the maximum number of FNs injected in the experiment. Furthermore, detector predictions over the samples selected should be examined, and the predictions that do not contain a sufficient amount of True Positive detections should be excluded. However, performing the aforementioned version of the experiment would compromise the diversity of the scenarios represented in the samples, by excluding samples with fewer than a sufficient number of GT objects. This would introduce another factor of uncertainty considering that the number of GT objects present in the samples evaluated has an impact on the evaluations of OCM-related metrics, as discussed in the preceding chapter.

Another limitation to consider, induced by the limited time frame of this work, is the size of the datasets utilized. As performing detector inference and metric evaluations over nuScenes samples is computationally expensive, datasets of 1000 and 100 nuScenes samples were utilized in the experiments introduced in Section 4.5 and Section 4.6, respectively. Given a longer time frame, analogous experiments would be performed on larger sets of nuScenes [32] samples, and preferably on the entirety of the dataset. This is restated in Section 7.4.

7.2 Discussion of results

For the purpose of guiding the experimental work performed, three research questions were formulated in the introductory chapter of this thesis. Research Question 1 and Research Question 2 addressed the relationship between the investigated metrics and their attributes compared to one another, and to generic metrics for evaluating detection models. Furthermore, Research Question 3 addressed the sensitivity of the safety-oriented metrics to faults. The experimental methodologies presented in Chapter 4 were applied to answer these questions. In this section, the limitations of the applied methodologies and the results stemming from the application of these methods will be discussed and applied to address the research questions stated.

7.2.1 Research Question 1: Metric properties

In the work of Guo et al. [45], an experiment is performed, computing PKL and mAP scores for three detectors for which results have been submitted in the nuScenes detection task [32]. The aforementioned metrics were computed for predictions made for different constraints on the number of GT objects present in the samples included. Results presented in the aforementioned work showed that the best scores for mAP were achieved for the highest numbers of GT objects present in the samples upon which predictions were made. Conversely, the best scores for PKL were achieved for the lowest number of total objects present in the samples evaluated. It should be noted that in the aforementioned results, metrics were evaluated on predictions over multiple samples and median PKL values were utilized in reported results.

In the visualizations for the joint distributions between OCM-related metrics and PKL presented in Section 6.2 and Subsection 6.3.1, the aforementioned trend for PKL was observed. In particular, it was observed in the metric distributions that for lower numbers of GTs present in samples evaluated, lower scores (indicating better evaluations) for PKL were reported, especially considering fault-injected predictions. Furthermore, a left shift in the distribution of $F1_{crit}$ was observed for lower values for the constraint variable N. This indicates that although the OCM-related metrics take into consideration the context of detections, scores for metrics based on reliability-weighted precision and safety-weighted recall follow a similar trend to what was observed in [45] for mAP. As a consequence of the difference in the metrics compared and the method with which results were gathered, this comparison does not provide conclusive evidence that this is the case for the OCM-related metrics. However, it is fruitful to consider the traits of generic detection metrics (such as mAP) in the analysis of OCM-related metrics, as they are on the same foundational measures of precision and recall.

In Section 6.5, visualizations are provided (Figure 6.5.5-Figure 6.5.7) indicating the difference in OCM-related metrics and their corresponding generic metrics for the scatter plots of joint distributions between PKL and the generic precision and recall. The subsequent analysis of these distributions with their corresponding differences in metrics visualized demonstrated a greater degree of freedom in OCM-related metrics, in contrast to their generic counterparts. Specifically, it was observed that when incorrect predictions are present in a prediction set, precision and recall cannot evaluate to perfect scores. Conversely, when correct predictions are present in a prediction set, the aforementioned generic metrics cannot exhibit scores of 0.0. Furthermore, the analysis of joint distributions between PKL and generic metrics demonstrated a general limitation of the number of possible values for precision, recall, and F1-score with a decreasing number of predictions evaluated. This limitation was increasingly evident for lower numbers of objects in the predictions evaluated, as illustrated in Figure 6.5.4. This is in contrast to OCM-related metrics, which can evaluate to a continuous set of values, another consequence of applying object criticalities. It is discussed in the Subsection 7.2.2 that the safety and reliability associated with specific detections are more reflected in OCM-related metrics as the number of objects in a sample decreases. For the generic versions of precision and recall, it can be argued that the opposite is true. With a smaller number of predictions evaluated, the limitation on values that precision and recall can take on would restrict the descriptiveness of the measures. In addition, objects with less relevance will be increasingly influential in the final evaluation of the measures.

In Figure 6.5.5-Figure 6.5.7, illustrating joint distributions between PKL and generic metrics and the corresponding values for OCM-related metrics, it was observed that OCM-related metrics generally exhibit higher values than the corresponding generic metrics for high-ranked predictions. Furthermore, the opposite was true for low-ranking metrics, with OCM-related metrics assuming lower values than their generic counterparts. This demonstrated that OCM-related metrics can evaluate to near-perfect scores even with misdetections present in detector predictions (when such predictions are not deemed relevant by the OCM), as indicated by the lower values for corresponding, generic metrics. Conversely, OCM-related metrics can achieve values near 0, even with a significant degree of correct predictions. This highlights the impact of object criticalities in OCM-related metrics and an important difference between the safety-oriented and generic adaptations of precision and recall.

Addressing *Research Question 1*, the above discussion highlighted some key differences between OCM-related and generic metrics. These differences demonstrated through the experimental work in Section 6.5, in turn, underscore some important similarities between the OCM-related metrics and PKL. OCM-related metrics and PKL both rely on the notion of attributing importance to objects, either through explicit or implicit means, respectively. In contrast to the generic metrics for object detection examined, this enables the metrics to consider the context of detection and evaluate detector predictions based on their potential consequences for the systems in which they are applied. Furthermore, it was demonstrated that when applied to small sets of predictions corresponding to single samples, the assignment of relevance to objects allows the task-oriented metrics to take on a larger number of values, enabling a more descriptive evaluation compared to the generic metrics (derived from the binary measures of TPs, FPs, and FNs).

7.2.2 Research Question 2: Metric correlation

In the experimental results presented in Subsection 6.3.1, a higher correlation between $F1_{score}$ and PKL was both perceived in the visualizations provided and reflected in the computed correlation coefficients for results over the RAW_40 and FN_40 datasets. Despite the limited sample size evaluated, with fewer than 1000 samples, and a decreasing number of samples as the evaluated value of N decreased, the statistical significance of the findings was supported by the computed p-values and confidence intervals. These statistical measures provided sufficient evidence to conclude that there is a statistically significant increase in correlation as the number of objects considered decreases. Furthermore, the findings indicated that this increase is considerably higher for predictions injected with faults corresponding to FNs. This indicated more consensus between the two metrics PKL and $F1_{crit}$ when evaluating false negative predictions for samples with a smaller number of GTs. For FP_40, however, the findings were more inconclusive, with confidence intervals indicating less statistical significance for results than for the aforementioned data. Furthermore, a decrease in correlation was observed for

$N \leq 10$. Reviewing these results in the context of other experimental results presented, this can be a consequence of a difference in the penalization of FP predictions between the metrics.

Throughout the experimental work presented in Chapter 6, analyses were conducted on metric data that involved the evaluation of detector predictions on individual samples. Hence, the evaluated sets of bounding boxes consisted of a relatively small number of predictions. Specifically, the experimental results presented in Chapter 6 examined the characteristics and distinctions among generic metrics, OCM-related metrics, and PKL. While PKL is specifically designed to evaluate sample predictions in this manner, OCM-related and generic metrics are typically intended for application on cumulative, low-level metric data. For instance, the aforementioned metrics are typically applied to the total number of observed TP, FP, and FN predictions over a set of multiple samples.

When evaluating predictions over single samples, reliability-weighted precision and safety-weighted recall have been demonstrated to be sensitive to the number of objects for which predictions are made. An example of this was introduced in Subsection 4.5.1, where two different recall values were computed for two predictions that exhibit the same degree of error. Examining Equation 2.26 and Equation 2.27, both metrics are influenced by the number of TP predictions present in the prediction set evaluated. The more true positive detections present, the less an injected fault is reflected in the final scores of the measures. For predictions made by an object detector over samples with varying numbers of objects present, the more objects present in the GT set of a sample, the more potential TP predictions can be made. This, as a consequence, diminishes the impact of injected faults on the $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ measures. This can explain the observed increasing correlation between $F1_{crit}$ and PKL with decreasing values of N over the FN_40 dataset (as illustrated in Figure 6.3.5b) if the metrics similarly assess FN predictions, as fewer objects are present on average in this dataset. Considering FP predictions, this can explain the decrease in correlation observed for the metrics if the metrics penalize FP predictions differently, as this underlying difference would be more reflected in metric results for lower values of N.

Considering correlation for increasingly constrained datasets, is important to investigate what might cause the aforementioned disparity in terms of correlation for different fault-injected datasets. As mentioned, the observed results FP_40 may be explained by a difference in penalization between the metrics. If such a difference in penalization exists, this would be increasingly reflected in metric data for OCM-related metrics as the constraint on the number of GTs present increased (i.e. when evaluating samples with fewer objects). This is a consequence of injected faults being reflected more on average in the final evaluation of OCM-related metrics if fewer objects are present, as discussed in the preceding paragraph. This should also be considered when discussing the increasing correlation over the FN_40 dataset. Assuming that the metrics penalize FNs similarly, a reduced number of objects present in the scenes evaluated would cause more agreement between the metrics as the injected FN is reflected more in the evaluations of the OCM-related metrics.

Additionally, when examining the aforementioned correlations for fault-injected datasets, the results of Guo et al. [45], discussed in the preceding section, should be considered. The results presented in that work demonstrated decreasing PKL values for fewer objects considered in evaluated samples. PKL evaluates the divergence between trajectories planned with predictions corresponding to GTs and the prediction from a detection model, respectively. Hence, it is intuitive that an increasing number of objects considered upon evaluation would cause an increase in PKL values on average, as more GT objects correspond to more potential errors considered in the evaluation that may cause the planned trajectory to diverge from the one planned with GT predictions. This would become particularly evident when faults are injected into the predictions of the detector, as more objects present in the path planned could cause a larger divergence from the GT path than would be the case otherwise. An example demonstrating this was presented in the Section 6.1, where the introduction of an equivalent FP in both prediction sets of PPT and SSN resulted in a more substantial impact on the PKL evaluation for PPT than for SSN. This was due to the higher number of FP predictions in the PPT predictions set prior to injection, compared to those of SSN.

Addressing *Research Question 2*, the results of Section 6.3 and the discussion above indicate that OCM-related metrics and PKL exhibit a considerable degree of correlation under certain circumstances. Particularly, there was an observed increase in the correlation between the metrics as the number of GT objects considered decreased, indicating a stronger reflection of object relevance in the metric results. This implies that similar factors are considered in PKL and the OCM upon assigning relevance to objects, but that the dependence of OCM-related metrics on the number of TP predictions in the prediction set yields varying degrees of perceived correlation. Furthermore, results for statistical correlation coefficients indicated that the relationship between the metrics is best described as increasing and linear for metric data over fault-injected predictions, and as monotonic for non-injected predictions.

7.2.3 Research Question 3: The impact of misdetections

In the experiment performed in Section 6.4, applying the methodology proposed in Section 4.6, an analysis of the impact of faults on metric evaluations for detector predictions was performed. For PKL, the results indicated a difference in the degree of penalization for false positive and false negative metrics. More specifically, PKL results over the samples evaluated at different numbers of faults indicated a higher degree of penalization of FPs in contrast to FNs. Additionally, in Section 6.2, a left shift in the distribution of PKL was observed for the dataset FP_40, contrasting the corresponding distribution for FN_40. With findings from multiple experiments indicating a higher degree of penalization of FPs for PKL, there is strong evidence that such a difference in penalization exists. Furthermore, considering the qualitative analysis of a sample over which the predictions of SSN and PPT were injected with a false positive prediction presented in Section 6.1, the considerable increase in PKL illustrated in Table 6.1.2 indicated that the measure is highly sensitive to predictions with multiple FPs. In this example, two FP predictions present in the detector predictions prior to injection caused a significantly higher value for PKL than would otherwise be exhibited. These results indicated that the introduction of FP predictions caused the planned path to diverge more from the path planned with GT predictions than would be the case with introduced FNs.

In the experimental results of Section 6.4, it was observed that $R_{\mathcal{S}}$ saw a larger decrease for injected FNs in contrast to $P_{\mathcal{R}}$ for injected FPs. As a consequence, similar observations were made regarding $F1_{crit}$. However, the role of the methodology applied to produce these observations for the OCM-related metrics should be considered when interpreting these results. The dependency of $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$ on the number of TPs evaluated, and thus on the number of GT objects present was discussed in the preceding chapter. Specifically, it was demonstrated that injected misdetections are reflected more in the scores of the metrics when a smaller number of TPs are present in the prediction set evaluated. Considering the analysis of the impact of misdetections performed in Section 6.4, the same set of samples and predictions were utilized in both the analysis of injected FPs and the analysis of injected FNs. While the removal of TP predictions (injection of FNs) impacts the overall sensitivity of OCM-related metrics to faults, this is not the case for the injection of FP objects. Hence, it is noted that for the two cases examined, a more severe penalization of injected FNs would be observed for both $P_{\mathcal{R}}$ and $R_{\mathcal{S}}$. This explains the observed difference in penalization observed. While it can be claimed by the preceding results and discussion that a difference in penalization for OCM-related metrics regarding FNs and FPs exists, it has been shown that this is a limited depiction of the difference in penalization of faults for OCM-related metrics and that the number of TPs should be considered. This is an important consideration when utilizing the aforementioned metrics for evaluating the performance of detectors. Furthermore, this highlights an important distinction between OCM-related metrics and PKL considering the penalization of misdetections, namely that PKL does not exhibit a dependence on the evaluation of other predictions in the prediction set, whereas OCM-related metrics do.

As discussed in Section 6.4, a decrease in reliability-weighted precision is observed when injecting an incremental number of FN detections into the dataset of predictions. This decrease is a consequence of the dependence of $P_{\mathcal{R}}$ on the number of TP predictions in the prediction set, as discussed in the preceding section. Furthermore, it was demonstrated in the aforementioned experimental work that $R_{\mathcal{S}}$ is not affected similarly by the injection of FPs. This is a result of no dependence of safety-weighted recall on the number of FP predictions in the evaluated prediction set. In contrast to FP-injections, the decrease in $P_{\mathcal{R}}$ as a consequence of FN-injection has the consequence of decreasing the overall value of $F1_{crit}$. Hence, to evaluate the impact of injected FN detections on OCM-related metrics, particularly on how OCM-related metrics evaluate the safety of the predictions, the value for $R_{\mathcal{S}}$ should be emphasized when examining the metric results presented in Section 6.4 (in place of $F1_{crit}$).

Addressing *Research Question 3*, while there is evidence that PKL in fact penalizes false positive prediction more harshly than false negatives, this is more subtle for OCM-related metrics. It has been shown that the penalization of faults for OCM-related metrics is highly dependent on the number of TP predictions in the evaluated dataset. In the experiment of Section 6.4, the same dataset of samples was utilized for both analysing FP injections and FN injections. Hence, as fewer TP-objects were present on average in the FN-injected dataset, these injections were more prominent in the metric results. It can thus be correctly stated that there is a difference in the penalization of FPs and TPs for OCM-related metrics, but it should be noted that this perceived difference is highly influenced by the size of the prediction sets evaluated (corresponding to single samples). For increasing sizes of the prediction sets evaluated, this difference in penalization will be decreasingly evident for OCM-related metrics, given that the number of TP predictions increases.

7.3 Evaluating the safety and reliability of detections

It has been discussed and observed that OCM-related metrics' abilities to reflect immediate safety hazards or reliability issues in the predictions of object detection models depend highly on the number of correct predictions in the same prediction set. This affects the ability of the metrics to reflect the true importance of specific objects in traffic scenarios and thus affects the correlation observed correlation between the aforementioned metrics and PKL, as discussed in Subsection 7.2.2. Conversely, PKL does not exhibit a similar dependency on the number of TP predictions, and can to a greater degree reflect the immediate importance of objects in a more diverse range of scenarios with regard to other predictions made by a detector. However, it has been observed that the PKL measure is also sensitive to the number of GT objects present in samples, assuming lower values for samples in which fewer objects are present. The two approaches for the task-specific evaluation of object detectors investigated in this thesis thus exhibit differences in terms of their sensitivity to the number of objects evaluated. While PKL displays a sensitivity to higher numbers of objects considered, the opposite is indicated by OCM-related measures. Hence, in the context of analysing detector predictions, the metrics can complement one another by providing unique insights into the safety and reliability of such predictions.

To determine how well a specific metric evaluates the safety and reliability of object detection models applied in autonomous systems, the aforementioned factors should be considered. Furthermore, the degree of interpretability exhibited by metrics should be considered. OCM-related metrics distinguish between safety and reliability-related misdetections and compute criticality values for objects analytically, whereas PKL comprises all factors considered in evaluating predictions into a single evaluation score. Furthermore, PKL utilizes an implicitly parametrized path planning algorithm as the basis upon which evaluation is performed. Hence, PKL exhibits interpretability issues in contrast to OCM-related metrics. It is intuitive that the applicability of metrics utilized in safety-critical systems is in part determined by the level of speculation needed in interpreting them. Hence, when applying less interpretable evaluation metrics, rigorous analyses of their evaluation of detector predictions over a multitude of scenarios are needed. For this purpose, it is fruitful to apply interpretable, safety-oriented metrics for the purpose of comparative analysis. The experimental work presented in this thesis is an example of such an approach.

7.4 Future work

7.4.1 Use ground-truth data as a basis of injection

As discussed in Subsection 7.1.1, filtering predictions at a particular confidence threshold when analysing the impact of misdetections introduces limitations with regard to analysing the behaviour of reliability-weighted precision and safety-weighted recall. To more precisely quantify the penalization of the metrics for the two classes of faults, namely FPs and FNs, different approaches can be proposed. One specific approach is to simulate perfect detection by generating a dataset of predictions that correspond to the ground truth objects of nuScenes samples. Such a dataset could be utilized as a basis for injecting faults, in which injected FNs and FPs would be isolated. More specifically, when analyzing metric evaluations over predictions injected with a specific type of fault, no FP or FN predictions would be present in the predictions prior to evaluation. This would enable a more precise analysis of fault penalization for the metrics.

7.4.2 Expand the datasets

For the experimental work presented in Section 6.4, the dataset of metric results analysed was generated over a set of 100 nuScenes samples. As such a small dataset is prone to inducing errors in the analysis of metric data generated over it, producing a larger dataset over a larger quantity of samples would enable a more thorough analysis of the impact of misdetections. Similarly, for the analysis of correlation presented in Section 6.3, utilizing larger datasets than those summarized in Table 4.2.1 would promote more statistically significant measures of metric correlation. Furthermore, taking steps in ensuring that the data applied is representative of a multitude of driving scenarios would be beneficial when performing an analysis of metric correlation and the penalization of faults. This could be performed by utilizing nuScenes [32] meta-data, comprising time, location, and weather information.

7.4.3 Explore metric parameters and detectors

Considering the implementation of the OCM, the parameters $(D_{max}, R_{max}, T_{max})$, utilized in the assignment of criticalities, were assigned constant values for all experimental work of this thesis. In the future, the impact of employing different configurations of these parameters should be investigated. This would provide depth to the results presented in this thesis, and promote a more detailed analysis of OCM-related metrics and their relationship with PKL. Furthermore, the methodology for experimental work described in Chapter 4 should be performed by employing a more diverse range of object detection models to make predictions over the samples examined. This would enable an analysis of metric results computed over a larger number of different prediction scenarios.

7.4.4 Apply the methodology to other metrics

Finally, as stated in the introductory chapter of this thesis, it is an objective of this work to motivate further research by proposing an approach for evaluating safety-oriented evaluation metrics for object detectors. Hence, applying similar methodologies for evaluating different approaches for implementing novel and safety-oriented evaluation metrics is encouraged.

CHAPTER

EIGHT

CONCLUSIONS

In [1], a comprehensive literature review was carried out to identify safety-oriented metrics proposed for object detectors applied in autonomous systems. This thesis focused on experimental investigations, comparing and analyzing two approaches employed for evaluating object detection models in autonomous vehicles. Specifically, these two approaches were based on the Object Criticality Model [2] and Planning KL-divergence [10]. In Section 1.2, the research objective of this work was identified, and three research questions were formulated to guide the experimental work of this thesis.

Firstly, Research Question 1 stated: *"In what manner do the suggested metrics deviate from conventional, generalized metrics utilized in object detection, as well as from one another?"*.

Secondly, Research Question 2 stated: *"Examining quantitative data, is there any measurable correlation between the metric evaluation results for the two approaches examined?"*.

Lastly, Research Question 3 stated: *"How do the proposed metrics penalize detections that represent scenarios where the safety or reliability of the system is compromised? Does quantitative metric data exhibit indications that the metrics penalize detections that cause potential safety- or reliability issues differently?"*.

Applying the methodology introduced in Chapter 4, experimental results were collected and presented in Chapter 6, followed by a discussion in Chapter 7. Collectively, this content addressed the research questions and provided substantial experimental evidence to corroborate the findings.

To facilitate a comparative analysis of OCM-related metrics and PKL, different methodologies were applied to investigate the metrics' relationships and properties. Additionally, the effect of prediction faults on these metrics was quantified, utilizing the fault injection methods introduced in Subsection 5.3.3. The experimental work involved several stages. Firstly, a qualitative analysis was performed on metric evaluations on detector predictions on a single sample from the nuScenes dataset, with and without introducing synthetic faults into these predictions. Subsequently, an examination of the relationship between OCM-related metrics and PKL was carried out, investigating the correlation between quantitative metric data for both metrics and the corresponding data distributions. The impact of injecting an increasing number of faults into a given prediction set on metric evaluations was then mapped, providing insights into the penalization of reliability and safety-related faults for the metrics. Lastly, a comparative analysis was performed to identify distinctions between OCM-related metrics, PKL, and generic metrics in the context of object detection.

A qualitative analysis was performed of metric evaluations of predictions made on individual samples from the nuScenes [32] dataset in Section 6.1. This analysis specifically investigated the sensitivity of PKL and OCM-related metrics to injected faults. Furthermore, in Section 6.5, a comparative analysis between OCM-related and generic metrics was performed. The findings highlighted limitations associated with general metrics, particularly the correlation between the range of metric values and the number of evaluated objects. Consequently, this exemplified the enhanced comprehensibility of metrics that assign criticalities to objects for small datasets of predictions corresponding to single samples. The aforementioned analyses highlighted important characteristics of the metrics examined and functioned to add nuance to the subsequent experimental work, and addressed *Research Question 1*, investigating the characteristics of task-specific metrics and their distinct properties when compared with generic metrics.

Subsequently, an analysis of the distributions of, and the relationship between the metrics was performed by extensive data visualization and statistical analysis in Section 4.5. Results indicated a statistically significant increase in correlation for a decreasing number of objects considered in predictions. Specifically, a higher correlation was observed with fewer objects present for datasets that were not subject to false positive injections. Hence, it can be concluded that a relationship exists for the metrics, but that this relationship is highly dependent on the number of TP predictions evaluated. In Subsection 7.2.2, it is argued that this increase in correlation for decreasing numbers of objects is a consequence of specific faults in the predictions being increasingly reflected in OCM-related metrics. This is further corroborated by subsequently discussed results indicating a significant difference in the penalization of FP predictions between PKL and OCM-related metrics. The aforementioned experimental work addressed *Research Question 2*, concerning metric correlation.

Finally, an investigation of the impact of faults on metric evaluations was performed. This investigation involved analysing metric distributions in Subsection 6.3.1 and studying the effects of incrementally injecting faults in Section 6.4. The results provided compelling evidence that PKL imposes a more severe penalty on false positive predictions compared to false negative predictions. Conversely, it was demonstrated that OCM-related metrics exhibit a higher penalization of FNs than FPs. However, the extent of fault-penalization for OCM-related metrics was demonstrated to be highly dependent on the number of TP predictions in the detector predictions evaluated, with lower numbers of TPs implying a higher penalization of faults. These results addressed *Research Question 3*, which specifically focused on the impact of misdetections on the evaluation metrics analysed.

Through extensive experimental work, a methodology for analysing safety-oriented metrics was applied. In this process, new functionality was implemented, building on the nuScenes-devkit [32], and provided in Appendix A. These should be considered contributions that can serve to drive similar research on the subject forward in the future. Furthermore, the research conducted in this thesis produced a scientific paper [3] that is currently submitted at an international conference within the field of Software Reliability. Due to the double-blind peer review process of the conference, further details of this conference are omitted.

REFERENCES

[1] A. Rønnestad. *Investigation of Safety-Oriented Metrics for Object Detectors*. Project report in TDT4501: Computer Science, Specialization Project. Faculty of Information Technology and Electrical Engineering, 2019.

[2] A. Ceccarelli and L. Montecchi. “Evaluating Object (Mis)Detection From a Safety and Reliability Perspective: Discussion and Measures”. In: *IEEE Access* 11 (2023). Cited by: 0; All Open Access, Gold Open Access, pp. 44952–44963. DOI: 10.1109/ACCESS.2023.3272979.

[3] A. Ceccarelli, L. Montecchi, and A. Rønnestad. “Validation of Safety Metrics for Object Detectors in Autonomous Driving”. In: (2023).

[4] L. Jiao et al. “A survey of deep learning-based object detection”. In: *IEEE Access* 7 (2019). Cited by: 546; All Open Access, Gold Open Access, Green Open Access, pp. 128837–128868. DOI: 10.1109/ACCESS.2019.2939201.

[5] R. Padilla, S. L. Netto, and E. A. B. Da Silva. “A Survey on Performance Metrics for Object-Detection Algorithms”. In: vol. 2020-July. Cited by: 316. 2020, pp. 237–242. DOI: 10.1109/IWSSIP48289.2020.9145130.

[6] L. Liu et al. “Deep Learning for Generic Object Detection: A Survey”. In: *International Journal of Computer Vision* 128.2 (2020). Cited by: 1076; All Open Access, Green Open Access, Hybrid Gold Open Access, pp. 261–318. DOI: 10.1007/s11263-019-01247-4.

[7] M. Lyssenko et al. “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)*. 2021, pp. 38–45. DOI: 10.1109/CVPRW53098.2021.00013.

[8] M. Wolf, L. R. Douat, and M. Erz. “Safety-Aware Metric for People Detection”. In: vol. 2021-September. Cited by: 1. 2021, pp. 2759–2765. DOI: 10.1109/ITSC48978.2021.9564734.

[9] G. Volk et al. “A Comprehensive Safety Metric to Evaluate Perception in Autonomous Systems”. In: Cited by: 7. 2020. DOI: 10.1109/ITSC45102.2020.9294708.

[10] J. Philion, A. Kar, and S. Fidler. “Learning to Evaluate Perception Models Using Planner-Centric Metrics”. In: Cited by: 17; All Open Access, Green Open Access. 2020, pp. 14052–14061. DOI: 10.1109/CVPR42600.2020.01407.

[11] J. SAE. “3016-2018, taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles”. In: *Society of Automobile Engineers, sae. org* (2018).

[12] C. Badue et al. “Self-driving cars: A survey”. In: *Expert Systems with Applications* 165 (2021). Cited by: 312. DOI: 10.1016/j.eswa.2020.113816.

[13] A. Avižienis et al. “Basic concepts and taxonomy of dependable and secure computing”. In: *IEEE Transactions on Dependable and Secure Computing* 1.1 (2004), pp. 11–33. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-12344308304&doi=10.11092fTDSC.2004.2&partnerID=40&md5=47dddbeed3c59fa4b803fbfbf8f1bfda.

[14] A. Burkov. *The Hundred-Page Machine Learning Book.* Andriy Burkov, 2019. ISBN: 9781999579517. URL: https://books.google.no/books?id=0jbxwQEACAAJ.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning representations by back-propagating errors”. In: *Nature* 323.6088 (1986). Cited by: 15539, pp. 533–536. DOI: 10.1038/323533a0.

[16] L. Alzubaidi et al. “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions”. In: *Journal of Big Data* 8.1 (2021). Cited by: 774; All Open Access, Gold Open Access, Green Open Access. DOI: 10.1186/s40537-021-00444-8.

[17] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* http://www.deeplearningbook.org. MIT Press, 2016.

[18] G. Box. “Robustness in the Strategy of Scientific Model Building”. In: *Robustness in Statistics.* Ed. by R. L. LAUNER and G. N. WILKINSON. Academic Press, 1979, pp. 201–236. ISBN: 978-0-12-438150-6. DOI: https://doi.org/10.1016/B978-0-12-438150-6.50018-2.

[19] A. Ramesh et al. *Zero-Shot Text-to-Image Generation.* 2021. arXiv: 2102.12092 [cs.CV].

[20] T. B. Brown et al. “Language models are few-shot learners”. In: vol. 2020-December. Cited by: 1884. 2020. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85107491666&partnerID=40&md5=0066bc76607662ac4ae639b93ae004f4.

[21] L. Jing and Y. Tian. “Self-Supervised Visual Feature Learning with Deep Neural Networks: A Survey”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (2021). Cited by: 339; All Open Access, Green Open Access, pp. 4037–4058. DOI: 10.1109/TPAMI.2020.2992393.

[22] K. He et al. “Deep residual learning for image recognition”. In: vol. 2016-December. Cited by: 99732; All Open Access, Green Open Access. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[23] S. Xie et al. “Aggregated residual transformations for deep neural networks”. In: vol. 2017-January. Cited by: 4875; All Open Access, Green Open Access. 2017, pp. 5987–5995. DOI: 10.1109/CVPR.2017.634.

[24] I. Radosavovic et al. “Designing network design spaces”. In: Cited by: 475; All Open Access, Green Open Access. 2020, pp. 10425–10433. DOI: 10.1109/CVPR42600.2020.01044.

[25] S. Ren et al. “Faster R-CNN: Towards real-time object detection with region proposal networks”. In: vol. 2015-January. Cited by: 21712. 2015, pp. 91–99. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84960980241&partnerID=40&md5=18aaa500235b11fb99e953f8b227f46d.

[26] R. Girshick. “Fast R-CNN”. In: vol. 2015 International Conference on Computer Vision, ICCV 2015. Cited by: 14260. 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[27] J. Redmon et al. “You only look once: Unified, real-time object detection”. In: vol. 2016-December. Cited by: 21160; All Open Access, Green Open Access. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[28] O. Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision* 115.3 (2015). Cited by: 21521; All Open Access, Green Open Access, pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[29] G. Salton and M. McGill. *Introduction to modern information retrieval.* New York, NY: McGraw-Hill, 1983.

[30] P. Jaccard. “Etude de la distribution florale dans une portion des Alpes et du Jura”. In: *Bulletin de la Societe Vaudoise des Sciences Naturelles* 37 (Jan. 1901), pp. 547–579. DOI: 10.5169/seals-266450.

[31] R. Padilla et al. “A comparative analysis of object detection metrics with a companion open-source toolkit”. In: *Electronics (Switzerland)* 10.3 (2021). Cited by: 166; All Open Access, Gold Open Access, pp. 1–28. DOI: 10.3390/electronics10030279.

[32] H. Caesar et al. “Nuscenes: A multimodal dataset for autonomous driving”. In: Cited by: 775; All Open Access, Green Open Access. 2020, pp. 11618–11628. DOI: 10.1109/CVPR42600.2020.01164.

[33] T. Kim et al. *Comparing Kullback-Leibler Divergence and Mean Squared Error Loss in Knowledge Distillation.* 2021. DOI: 10.48550/ARXIV.2105.08919.

[34] M. Contributors. *MMDetection3D: OpenMMLab next-generation platform for general 3D object detection.* https://github.com/open-mmlab/mmdetection3d. 2020.

[35] A. Geiger, P. Lenz, and R. Urtasun. “Are we ready for autonomous driving? the KITTI vision benchmark suite”. In: Cited by: 7023. 2012, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

[36] P. Sun et al. “Scalability in perception for autonomous driving: Waymo open dataset”. In: Cited by: 499; All Open Access, Green Open Access. 2020, pp. 2443–2451. DOI: 10.1109/CVPR42600.2020.00252.

[37] M. Zhu et al. “Cross-modality 3d object detection”. In: Cited by: 5; All Open Access, Green Open Access. 2021, pp. 3771–3780. DOI: 10.1109/WACV48630.2021.00382.

[38] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[39] K. Chen et al. *MMDetection: Open MMLab Detection Toolbox and Benchmark*. 2019. arXiv: 1906.07155 [cs.CV].

[40] A. H. Lang et al. "Pointpillars: Fast encoders for object detection from point clouds". In: vol. 2019-June. Cited by: 1121; All Open Access, Green Open Access. 2019, pp. 12689–12697. DOI: 10.1109/CVPR.2019.01298.

[41] A. Ghasemieh and R. Kashef. "3D object detection for autonomous driving: Methods, models, sensors, data, and challenges". In: *Transportation Engineering* 8 (2022). Cited by: 1; All Open Access, Gold Open Access. DOI: 10.1016/j.treng.2022.100115.

[42] T.-Y. Lin et al. "Feature pyramid networks for object detection". In: vol. 2017-January. Cited by: 11686; All Open Access, Green Open Access. 2017, pp. 936–944. DOI: 10.1109/CVPR.2017.106.

[43] X. Zhu et al. "SSN: Shape Signature Networks for Multi-class Object Detection from Point Clouds". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12370 LNCS (2020). Cited by: 13; All Open Access, Green Open Access, pp. 581–597. DOI: 10.1007/978-3-030-58595-2_35.

[44] Y. Yan, Y. Mao, and B. Li. "Second: Sparsely embedded convolutional detection". In: *Sensors (Switzerland)* 18.10 (2018). Cited by: 911; All Open Access, Gold Open Access, Green Open Access. DOI: 10.3390/s18103337.

[45] Y. Guo et al. *The efficacy of Neural Planning Metrics: A meta-analysis of PKL on nuScenes*. 2021. arXiv: 2010.09350 [cs.CV].

[46] M.-T. Puth, M. Neuhäuser, and G. D. Ruxton. "Effective use of Spearman's and Kendall's correlation coefficients for association between two measured traits". In: *Animal Behaviour* 102 (2015), pp. 77–84. ISSN: 0003-3472. DOI: https://doi.org/10.1016/j.anbehav.2015.01.010.

[47] J. Benesty et al. "Pearson correlation coefficient". In: *Springer Topics in Signal Processing* 2 (2009). Cited by: 1438, pp. 1–4. DOI: 10.1007/978-3-642-00296-0_5.

[48] E. Komaroff. "Relationships Between p-values and Pearson Correlation Coefficients, Type 1 Errors and Effect Size Errors, Under a True Null Hypothesis". In: *Journal of Statistical Theory and Practice* 14.3 (2020). Cited by: 6. DOI: 10.1007/s42519-020-00115-6.

[49] *Fisher's z-transformation*. DOI: 10.1093/oi/authority.20110803095820772.

[50] M. Själander et al. *EPIC: An Energy-Efficient, High-Performance GPGPU Computing Research Infrastructure*. 2019. arXiv: 1912.05848 [cs.DC].

APPENDICES

APPENDIX

A

CODEBASE

All code developed for the purposes of the research performed in this thesis is provided in the Github repository linked below. Further explanations are given in the readme-file of the repository.

Github repository

- `https://github.com/andreas-roennestad/Thesis-Evaluating-Safety-Oriented-Metrics-for-Object-Detectors`

APPENDIX

B

METHODS FOR FAULT-INJECTION

Here, the code for the injection methods presented in Subsection 5.3.3 is provided. The subsequent code adds depth to the understanding of the experimental work introduced in Chapter 4, in which fault-injected object detector predictions are evaluated by the metrics investigated. For further context to the implementation, read Subsection 5.3.3. In the subsequent Python [38] code-snippets, the `add_FP()` and `add_FN()` methods are class methods, bound to the `DetectionEval`-class implemented in the file `./eval/detection/evaluate.py` of the modified nuScenes devkit provided in the repository linked to in Appendix A. To understand the context of the functionality utilized, it is necessary to examine this file in its entirety. However, for an understanding of the general parameters utilized to inject faults, and to understand the general methodology of injection, this is not necessary. Comments are provided in the code.

B.1 False positive injection

```
def add_FP(self, sample_token: str, pos: Tuple[float, float],
             size: Tuple[float, float, float],
             match_ego_speed = False):
    """
    Add a FP prediction in coordinates in coordinates coords wrt. ego
        reference frame
    :param sample_token: Sample to operate on.
    :param pos: Tuple[x, y]. 2D coordinates of FP in relation to ego,
        where
     positive x and y point to right and front of ego, respectively.
     z coordinate is set equal to ego z.
    :param size: Tuple[h,l,w]. Size of injected BB.
    :param match_ego_speed: Boolean. Whether to match velocity of ego
        or to have null-velocity.
    """
    print("Adding␣FP␣at␣position␣{}␣relative␣to␣ego␣at␣sample␣{}".
       format(pos, sample_token))
    # Get ego reference
    sample = self.nusc.get('sample', sample_token)
    sd_record = self.nusc.get('sample_data', sample['data']['
       LIDAR_TOP'])
    cs_record = self.nusc.get('calibrated_sensor', sd_record['
       calibrated_sensor_token'])
    pose_record = self.nusc.get('ego_pose', sd_record['ego_pose_token
       '])
    ego_translation = pose_record['translation']
    ego_rotation = pose_record['rotation']
    ego_speed = self.pred_boxes[sample_token][0].ego_speed
    # Create FP box
    fp_box = DetectionBox(
        sample_token=sample_token,
        translation=ego_translation,
        rotation=ego_rotation,
        size=size,
        detection_score=0.99,
        num_pts=25,
        attribute_name='vehicle.moving' if match_ego_speed==True else
             'vehicle.stopped',
        velocity=(ego_speed[0],ego_speed[1]) if match_ego_speed==True
             else (0,0),
        nusc=self.nusc,
    )

    # Create Box instance.
    box = Box(center=fp_box.translation, size=fp_box.size,
               orientation=Quaternion(fp_box.rotation),
               velocity=(fp_box.velocity[0],fp_box.velocity[1],0),
               name=fp_box.detection_name, crit=fp_box.crit,
               crit_t=fp_box.crit_t, crit_r=fp_box.crit_r,
               crit_d=fp_box.crit_d)
    # Move box to ego vehicle coord system.
    box.translate(-np.array(pose_record['translation']))
    box.rotate(Quaternion(pose_record['rotation']).inverse)
```

```
# Place FP at pos coordinates (note: x and y coordinates are
# in opposite positions in ego frame: trans=(y,x,z))
box.center[0]+=pos[1]
box.center[1]-=pos[0] # positive x-axis to right
# Transform to global ref frame
box.rotate(Quaternion(pose_record['rotation']))
box.translate(np.array(pose_record['translation']))
# Re-initialize Detection-Box to match new translation (to get
    correct criticalities)
fp_box_m = DetectionBox(
    sample_token=sample_token,
    translation=box.center,
    rotation=ego_rotation,
    size=size,
    detection_score=0.99,
    num_pts=25,
    attribute_name='vehicle.moving' if match_ego_speed==True else
        'vehicle.stopped',
    velocity=(ego_speed[0],ego_speed[1]) if match_ego_speed==True
        else (0,0),
    nusc=self.nusc,
)

# Add box
self.pred_boxes.add_boxes(sample_token, [fp_box_m])
```

B.2 False negative injection

```
def add_FP(self, sample_token: str, pos: Tuple[float, float],
           size: Tuple[float, float, float],
           match_ego_speed = False):
    """
    Add a FP prediction in coordinates in coordinates coords wrt. ego
        reference frame
    :param sample_token: Sample to operate on.
    :param pos: Tuple[x, y]. 2D coordinates of FP in relation to ego,
        where
     positive x and y point to right and front of ego, respectively.
     z coordinate is set equal to ego z.
    :param size: Tuple[h,l,w]. Size of injected BB.
    :param match_ego_speed: Boolean. Whether to match velocity of ego
        or to have null-velocity.
    """
    print("Adding␣FP␣at␣position␣{}␣relative␣to␣ego␣at␣sample␣{}".
       format(pos, sample_token))
    # Get ego reference
    sample = self.nusc.get('sample', sample_token)
    sd_record = self.nusc.get('sample_data', sample['data']['
       LIDAR_TOP'])
    cs_record = self.nusc.get('calibrated_sensor', sd_record['
       calibrated_sensor_token'])
    pose_record = self.nusc.get('ego_pose', sd_record['ego_pose_token
       '])
    ego_translation = pose_record['translation']
    ego_rotation = pose_record['rotation']
    ego_speed = self.pred_boxes[sample_token][0].ego_speed
    # Create FP box
    fp_box = DetectionBox(
        sample_token=sample_token,
        translation=ego_translation,
        rotation=ego_rotation,
        size=size,
        detection_score=0.99,
        num_pts=25,
        attribute_name='vehicle.moving' if match_ego_speed==True else
            'vehicle.stopped',
        velocity=(ego_speed[0],ego_speed[1]) if match_ego_speed==True
            else (0,0),
        nusc=self.nusc,
    )

    # Create Box instance.
    box = Box(center=fp_box.translation, size=fp_box.size,
              orientation=Quaternion(fp_box.rotation),
              velocity=(fp_box.velocity[0],fp_box.velocity[1],0),
              name=fp_box.detection_name, crit=fp_box.crit,
              crit_t=fp_box.crit_t, crit_r=fp_box.crit_r,
              crit_d=fp_box.crit_d)
    # Move box to ego vehicle coord system.
    box.translate(-np.array(pose_record['translation']))
    box.rotate(Quaternion(pose_record['rotation']).inverse)
```

```
# Place FP at pos coordinates (note: x and y coordinates are
# in opposite positions in ego frame: trans=(y,x,z))
box.center[0]+=pos[1]
box.center[1]-=pos[0] # positive x-axis to right
# Transform to global ref frame
box.rotate(Quaternion(pose_record['rotation']))
box.translate(np.array(pose_record['translation']))
# Re-initialize Detection-Box to match new translation (to get
    correct criticalities)
fp_box_m = DetectionBox(
     sample_token=sample_token,
     translation=box.center,
     rotation=ego_rotation,
     size=size,
     detection_score=0.99,
     num_pts=25,
     attribute_name='vehicle.moving' if match_ego_speed==True else
          'vehicle.stopped',
     velocity=(ego_speed[0],ego_speed[1]) if match_ego_speed==True
          else (0,0),
     nusc=self.nusc,
)

# Add box
self.pred_boxes.add_boxes(sample_token, [fp_box_m])
```

NTNU

Norwegian University of
Science and Technology