# Evaluating perception system robustness for AV applications

*Shikai Geng*


4th Year Project Report
Artificial Intelligence and Computer Science
School of Informatics
University of Edinburgh

2024

# Abstract

This thesis presents an investigation into the robustness of perception systems for autonomous vehicles (AVs), focusing on the impact of environmental corruptions on model performance. Leveraging synthetic data generation and advanced object detection frameworks, this study examines how different types of corruptions—such as fog, rain, snow, and motion blur—affect the accuracy of 2D object detection models. The YOLOv8 model serves as the baseline for evaluating performance across corrupted and clean datasets, with a particular emphasis on identifying the conditions under which model performance significantly declines.

The research methodology encompasses the generation of synthetic data to minimises the influence of real-world corruptions associated with the training and testing of the YOLOv8 model under these conditions. The findings reveal that environmental corruptions adversely affect model accuracy, with varying degrees of impact across different corruption types. The study contributes to the field by demonstrating the importance of incorporating synthetic data in training procedure to enhance model robustness and by providing insights into the limitations of current perception systems in AV applications. This work not only benchmarks the resilience of AV perception models against environmental corruptions but also sets the stage for future research aimed at developing more adaptive and reliable AV systems.

# Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Shikai Geng*)

# Acknowledgements

I sincerely appreciate my supervisor Professor Subramanian Ramamoorthy for his outstanding mentorship and great encouragement. In addition, I would like to thank my friends and parents, who supported me in maintaining both physical and mental health throughout this project.

# Table of Contents


**1 Introduction** **1**
- 1.1 Motivation . . . 1
  - 1.1.1 Synthetic Data Generation . . . 1
  - 1.1.2 Model Evaluation on Corrupted Datasets . . . 2
- 1.2 Research Hypothesis and Objectives . . . 3

**2 Background** **4**
- 2.1 Related Works . . . 4
  - 2.1.1 Perception Error Model . . . 4
  - 2.1.2 Corrupted Data reviewed . . . 5
  - 2.1.3 Datasets for Autonomous Driving reviewed . . . 7
- 2.2 YOLOv8 - The Baseline Model reviewed . . . 8
- 2.3 Machine Learning Basis . . . 9
  - 2.3.1 Mathemathical Basis . . . 9
  - 2.3.2 Convolutional Neural Network . . . 10
  - 2.3.3 Feature Pyramid Network (FPN) . . . 10
  - 2.3.4 Path Aggregation Network (PAN) . . . 11

**3 Data Preparation** **12**
- 3.1 3DCC . . . 12
  - 3.1.1 Depth Completion Methods reviewed . . . 14
- 3.2 Imgaug . . . 16

**4 Evaluations** **18**
- 4.1 Experiment Setup . . . 18
- 4.2 Experiment Results . . . 18
  - 4.2.1 Accurcy . . . 18
  - 4.2.2 Confusion Matrix . . . 19
  - 4.2.3 F1 score . . . 20
  - 4.2.4 Perception-Recall curve . . . 21
- 4.3 Measure the model confidence . . . 22
  - 4.3.1 Confidence Definition . . . 22
  - 4.3.2 Results . . . 25

**5 Discussions** **26**
- 5.1 Limitations and Future Works . . . 26

5.2 Conclusions . . . . . 27

**Bibliography** **28**

# Chapter 1

# Introduction

## 1.1 Motivation

The development of autonomous vehicles (AVs) stands as a representative example of how perception systems can revolutionise human-machine interaction. AVs, engineered to navigate through complex environments without human intervention rely heavily on a integration of sensors and algorithms to interpret and respond to their surrounding environments[31]. Thus, the rigorous evaluation of such systems before their deployment becomes imperative. Traditional testing methods, such as extensive real-world trials, present considerable challenges, including impracticality and substantial risks. Furthermore, the unpredictability of real-world conditions, such as extreme weather, can corrupt sensor data, adding to the challenges of evaluating artificial intelligence (AI) and machine learning (ML)-based AV systems. These systems' complex nature, combined with the diverse scenarios they encounter, makes providing analytical assurances complicated. In this context, metrics that assess the reliability of AV perception systems, such as Perception Error Models (PEMs) [25] emerge as a vital tool.

Building on the foundation laid by perception model evaluation metrics such as PEMs, this thesis delves into two critical areas of investigation to further our understanding and assessment of AV perception systems. The first area focuses on the ***generation of synthetic data according to corruption types*** - a process to explore different methods for creating synthetic data according to corruption types; and ***evaluate 2D object detection models using corrupted data*** - a critical analysis on existing 2D object detectors to evaluate their performance in extreme conditions.

### 1.1.1 Synthetic Data Generation

In the rapidly advancing field of AI/ML technology, the development of high-precision object detection and segmentation models is of great importance. These models serve as the foundation for AV's perception systems, enabling them to interpret and navigate complex real-world environments. However, the journey toward achieving exceptional model accuracy entails not only enhancing the precision of these models but also ensuring their robustness and reliability under diverse conditions. This necessity becomes

particularly evident when considering the challenges posed by real-world scenarios, which often include adverse weather conditions such as heavy fog, rain, and snow, that can severely impact sensor performance.

Traditional training datasets, such as KITTI [9] and nuScenes [3], have played a crucial role in the development of perception models for autonomous driving. However, these datasets typically present sanitized and idealized conditions that do not fully encapsulate the complexities and imperfections encountered in real-world environments. This gap raises a critical question: *Can models trained exclusively on such clean data reliably perform in the real world, where conditions are far from ideal?* This question is not simply academic but has direct implications for the safety and reliability of autonomous vehicles.

Addressing this challenge requires a paradigm shift in how we prepare our models for the unpredictability of real-world operations. One promising approach is the generation of synthetic data[8] that introduces corrupted or noisy elements into training datasets. This technique simulates the data imperfections and extreme conditions, such as heavy fog and snow, that autonomous vehicles must navigate. By integrating this synthetic data into the training process, we aim to develop perception models that are not only accurate but also resilient and adaptable to the diverse scenarios they will encounter in real life.

Synthetic data generation serves two purposes. Firstly, it enhances the model's ability to cope with data imperfections, thereby improving its performance in real-world conditions where sensor accuracy is crucial. Adverse weather conditions, for example, can obscure visual cues and affect the readings of LiDAR intensity and depth values, leading to decreased prediction accuracy. Incorporating synthetic data that minimises these conditions during the training phase allows models to better anticipate and adjust to such challenges[34]. Secondly, synthetic data generation offers a practical and cost-effective alternative to extensive real-world testing. While physical world testing provides high-fidelity results, it is often unachievable due to its cost, time consumption, and associated risks. Virtual environment testing[24], on the other hand, although computationally demanding, enables the simulation of complex scenarios with a level of precision and scale that is unattainable in physical tests.

In summary, the introduction of synthetic data generation into the training of AV's perception models represents a critical step towards bridging the gap between idealised datasets and the intricate realities of the real world. By preparing models to effectively handle data imperfections and extreme conditions, we can significantly enhance their robustness, reliability, and, ultimately, their safety in autonomous driving applications.

### 1.1.2 Model Evaluation on Corrupted Datasets

The integration of synthetic data into AV perception model training is a crucial step towards making these systems more robust and reliable, even in challenging environmental conditions. To effectively measure how well these models perform, especially for 2D object detection, it's essential to test them against datasets that mimic real-world imperfections, like blurring, noise, or bad weather. This process not only tests the mod-

els' resilience but also helps pinpoint areas needing improvement. 2D object detection is vital for AVs since it helps them identify and place objects correctly, a key factor in making safe navigational decisions. By evaluating these detection systems under a range of distorted conditions, we aim to ensure they can handle the kind of visual challenges sensors face in the real world, thereby offering a thorough check on their performance.

Evaluating the performance of 2D object detectors on corrupted datasets utilises key metrics such as accuracy, recall, and precision. These metrics offer a quantitative measure of how well models can identify and localise objects, detect all relevant objects within a scene, maintain correct predictions, and consistently perform across different levels of data corruption. Initial evaluations indicate a wide range of outcomes, with some models showing a significant decline in detecting certain objects (sitting human e.g.) under extreme conditions like motion blur or extreme weather, which raises concerns about their applicability in real-world adverse conditions.

This critical analysis of 2D object detectors on corrupted datasets offers invaluable insights for future model development. It underscores the importance of including a diverse range of synthetic data conditions during training to enhance model robustness and highlights the benefits of integrating adaptive algorithms and advanced sensory processing techniques to reduce the impact of data corruption. As we move forward, developing 2D object detection models must focus on refining accuracy and precision in ideal conditions while also enhancing resilience against the unpredictable variances of real-world scenarios. This approach will be crucial in advancing the safety, reliability, and overall effectiveness of AV perception systems as they navigate the complexities of the real world.

## 1.2 Research Hypothesis and Objectives

The idea is based on the hypothesis that real-world corruptions can have an impact on the performance of perception models trained on clean datasets. This is because models trained on clean datasets usually do not generalise well to corruptions.

In this paper, we first explore different methods for creating synthetic data according to corruption types and then design 4 types of common corruptions in 2D object detection for camera sensors to comprehensively and rigorously evaluate the corruption robustness of current 2D object detectors.

# Chapter 2

# Background

## 2.1 Related Works

### 2.1.1 Perception Error Model

Sensors facilitate the interaction between devices and their environment, converting physical phenomena into quantifiable data. In automation, sensors provide real-time data essential for decision-making. Therefore, it is important to know how the sensor and the corresponding perception model perform in different situations. Wrong perception results may lead to serious accidents.

A typical challenge for sensor accuracy and sensing models is adverse weather conditions. Heavy fog, rain, and snow can obscure visual elements, affecting the light detection and ranging (lidar) intensity and, consequently, the depth value readings. These issues will affect the prediction accuracy of models trained on clean data sets and are frequently encountered in real-world scenarios. To evaluate sensor performance under such conditions, physical world testing is one approach. Although these tests offer high fidelity results, they can be costly, time consuming, and risky. An alternative is virtual environment testing, which is automatic but also demands significant computational resources for high-precision simulation.

To tackle these challenges in error analysis and perception system performance, Andrea Piazzoni et al. introduced the Perception Error Model (PEM)[9] – a virtual simulation tool for assessing the impact of perception errors on autonomous vehicle safety.

PEM works by taking a ground truth representation of the world and generating a perceived version of it, mimicking the combined function of sensing and perception subsystems in a virtual space. It is sensor-agnostic, meaning it is not tied to specific sensor models, allowing for flexible integration into simulation pipelines. PEM analyzes a collection of surrounding objects, both obstacles and road users, as perceived by the sensing and perception system. It can model the entire system, including sensors and AI processing of sensor data, without being restricted to specific sensors or AI algorithms. PEMs are useful for evaluating the performance of perception software independently. Perception errors are categorized into detections, misclassification, object parameters,

and dynamics. By contrasting ground truth with the perception algorithm's output, the magnitude of these errors can be quantified into a single metric.

### 2.1.2 Corrupted Data reviewed

In the realm of computer vision, data corruption includes a variety of distortions or perturbations that can significantly impair the effectiveness of machine learning models. These corruptions arise from a range of factors, including weather conditions, sensor inaccuracies, motion blur, occlusions, and scene rotations. The generation of corrupted data is a critical area of study, aimed at revealing the vulnerabilities of image recognition models to common distortions.

One prominent method for generating corrupted data is through algorithmically applied perturbations on real images, as demonstrated by the Two Dimensional Common Corruptions (2DCC)[11] initiative. This approach led to the creation of the IMAGENET-P dataset, designed to assess classifiers' resilience against typical perturbations. In contrast, other efforts, such as ObjectNet[1], focus on capturing corruptions within real-world settings. While this method offers realistic scenarios, it is limited by its substantial manual labor requirements, lack of scalability, and limited control over rotation, background, and viewpoint parameters.

An alternative strategy involves utilizing computer graphics-based 3D simulators, as proposed by 3DB[18] , to generate corrupted data. Additionally, open-source libraries like imgaug enable the transformation of a collection of images into a significantly larger set of slightly modified versions. The Three Dimensional Common Corruptions (3DCC) [17]approach generates corruptions with adjustable parameters, considering scene geometry and requiring RGB and depth images, and occasionally 3D meshes. This method ensures a more realistic depth of field in the corrupted images.

Corrupted data serves a pivotal role in evaluating the robustness of machine learning models, particularly through benchmarking exercises on corrupted datasets. Yinpeng Dong and colleagues present a thorough examination of the resilience of 3D object detection models used in autonomous driving against data corruption[6]. By categorizing corruptions into five levels and identifying 27 distinct types, the authors establish benchmarks on the KITTI[9], nuScenes[3], and Waymo[33] datasets. Their large-scale experiments reveal varying performances of state-of-the-art 3D detectors under different corruption types, highlighting the benchmarks' effectiveness in assessing detector robustness.

Furthermore, incorporating corrupted data into the training process can enhance model performance. By adding a corresponding set of corrupted training data to the original dataset, models can improve their accuracy and robustness, an essential consideration for applications like autonomous driving where reliability under diverse conditions is paramount.

In the realm of autonomous driving, the reliability and accuracy of sensor data are paramount. As vehicles navigate through ever-changing environments, they encounter various conditions that can corrupt the data collected by onboard sensors, such as cameras and LiDAR[29]. Understanding these corruptions is crucial for developing

robust autonomous driving systems capable of performing safely under diverse circumstances. This article categorizes these corruptions into four primary types, considering their source and impact on sensor data: weather-level, sensor-level, motion-level, and object-level corruptions.

**Weather-level Corruptions.** Weather-level corruptions significantly affect the performance of sensor systems, especially in autonomous vehicles that must navigate through diverse atmospheric conditions. In addition to snow, rain, fog, and strong sunlight[38], other elements like hail, dust storms, and varying degrees of cloud cover also have critical impacts. For example, hail can cause physical damage to sensors, affecting their accuracy, while dust storms can severely limit visibility and sensor range. Cloud cover variations can lead to inconsistent lighting conditions, making it challenging for cameras to maintain consistent visibility and for LiDAR to accurately gauge distances. Adaptive algorithms that can adjust to these varying conditions are essential for ensuring consistent performance, necessitating advanced machine learning techniques that can predict and compensate for environmental impacts.

**Sensor-level Corruptions.** Sensor-level corruptions include issues beyond Gaussian, uniform, and impulse noise, such as sensor aging, calibration drift, and temperature-induced malfunctions[28]. Over time, sensors can degrade, leading to a gradual decrease in data quality, a phenomenon that is challenging to detect and correct. Calibration drift refers to the gradual misalignment of sensor parameters, requiring regular recalibration to ensure accuracy. Temperature fluctuations can also affect sensor performance, with extreme cold or heat impacting sensor sensitivity and signal processing capabilities. Sophisticated diagnostics and self-healing mechanisms that can detect anomalies and initiate corrective actions without human intervention are required to address these issues.

**Motion-level Corruptions.** Motion-level corruptions also include vibration-induced distortions and the effects of sudden accelerations or decelerations, in addition to motion compensation, moving objects, and motion blur. High-frequency vibrations, common in rough terrain or due to engine operation, can blur images and scatter LiDAR points[37], obscuring critical details. Sudden changes in vehicle speed can lead to discrepancies between sensor readings, complicating the task of object tracking and prediction. Advanced stabilization techniques and algorithms that can dynamically adjust to these conditions are vital for maintaining data integrity and ensuring accurate environmental modeling.

**Object-level Corruptions.** Object-level corruptions involve challenges related to shape, material, and orientation and also include issues like occlusion, reflectivity variations, and object clustering[7]. Occlusions, where objects are partially hidden by others, can significantly hinder accurate detection and identification. Reflectivity variations, especially in materials that absorb or reflect light differently, can confuse sensors and lead to inaccurate readings. Object clustering, where multiple objects are close together, poses a challenge for systems trying to distinguish between them, requiring sophisticated segmentation algorithms.

Table 2.1: Comparison existing autonomous driving datasets

| Dataset | (*)Frames | rain | fog | snow | bler |
|---|---|---|---|---|---|
| A2D2[10] | 41K | ✗ | ✗ | ✗ | ✗ |
| ApolloScape [36] | 150K | ✓ | ✗ | ✗ | ✗ |
| CityScapes[4] | 25K | ✗ | ✗ | ✗ | ✗ |
| KITTI[9] | 41K | ✗ | ✗ | ✗ | ✗ |
| nuScenes[3] | **1.4M** | ✓ | ✗ | ✗ | ✗ |
| Waymo Open[33] | 37K | ✓ | ✗ | ✗ | ✗ |
| Foggy Cityscapes[30] | 20k | ✗ | ✓ | ✗ | ✗ |

### 2.1.3 Datasets for Autonomous Driving reviewed

Autonomous driving datasets such as A2D2, CityScapes, KITTI, nuScenes, Waymo Open, and Foggy Cityscapes are foundational for developing vehicle perception systems. However, these datasets often lack in capturing comprehensive real-world corruptions like adverse weather and blurriness showen in Table 2.1, crucial for training robust detection models. This deficiency, mainly due to the high cost of collecting real-world corruption data, presents a significant challenge. The A2D2 dataset, with its extensive repository of 41,000 frames, and similarly CityScapes and KITTI, fail to include critical conditions such as rain, fog, snow, and blurriness, essential for ensuring vehicular performance reliability across diverse environmental landscapes. Conversely, ApolloScape steps forward by incorporating rainy conditions into its 150,000 frames but does not cover fog, snow, and blurriness. nuScenes and Waymo Open acknowledge rain in their respective datasets but still miss fog and snow data, while Foggy Cityscapes fills a part of this gap by featuring fog in its 20,000 frames but lacks rain, snow, and blurriness coverage.

Efforts to address these gaps include the assembly of specialized datasets like Seeing Through Fog (STF)[2], Ithaca365[5], and Canadian Adverse Driving Conditions (CADC)[26], which focus on adverse weather conditions, alongside collections that highlight road anomalies from 2D images. These efforts are aimed at enhancing model resilience by providing more comprehensive environmental representations. Despite the value of these datasets, they are often limited by the costs and logistics of gathering rare, real-world data and are primarily used for model evaluation rather than training, exhibiting a notable domain gap from larger-scale training datasets. This gap is exacerbated by the varied conditions under which these datasets are collected, such as different cities, vehicles, and sensor setups, complicating the isolation of specific factors influencing model robustness.

In response to these challenges, there is a growing trend towards synthetic augmentation of clean datasets with real-world corruption simulations to benchmark model resilience. This approach, pioneered by ImageNet-C[11]in the image classification domain with 15 types of corruptions, including noise, blur, and a range of weather to digital disruptions, has been extended to other areas such as 3D object detection and point cloud recognition. Such synthetic corruption proves valuable in robustness evaluation across different facets of autonomous driving technologies, demonstrating the potential to bridge the

domain gap and improve the reliability of perception systems under diverse operating conditions.

## 2.2 YOLOv8 - The Baseline Model reviewed

In the landscape of contemporary research within the realm of object detection, this project has designated the YOLO (You Only Look Once) [15] algorithm as the baseline model, a decision underscored by its real-time performance and suitability for object detection tasks. The YOLO algorithm, celebrated for bridging the gap between speed and accuracy, emerges as the cornerstone for our exploration due to its unparalleled real-time processing abilities. This attribute is particularly pivotal in applications where the immediacy of object detection is critical for prompt decision-making and action-taking.

In detail, this work uses the YOLOv8 [14]released in January 2023 by Ultralytics; since, at the time of this writing, contributors still working on the YOLOv8 paper and aiming to share it with the community as soon as it's ready. Following the current trend, YOLOv8 is anchor-free, reducing the number of box predictions and speeding up the Non-maximum impression (NMS). YOLOv8 provided five scaled versions: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large) and YOLOv8x (extra large). The small, medium, and large models have (11151080, 25879480, 43660680) parameters and (225,295, 365) layers respectively

YOLOv8 is the latest version of the YOLO object detection model[35]. This latest version has the same architecture as its predecessors but it introduces numerous improvements compared to the earlier versions of YOLO such as a new neural network architecture that utilizes both Feature Pyramid Network (FPN)[19] and Path Aggregation Network (PAN)[20] as shown in Figure 2.1.The FPN works by gradually reducing the spatial resolution of the input image while increasing the number of feature channels. This results in the creation of feature maps that are capable of detecting objects at different scales and resolutions. The PAN architecture, on the other hand, aggregates features from different levels of the network through skip connections. By doing so, the network can better capture features at multiple scales and resolutions, which is crucial for accurately detecting objects of different sizes and shapes.The backbone of YOLOv8, CSPDarknet53, is a refined version of the Darknet architecture[27], enriched with Cross Stage Partial networks to boost feature extraction efficiency. This backbone is instrumental in processing the complex visual information presented in input images. The novel neck architecture that employs FPN and PAN facilitates effective feature fusion, bridging the backbone and the YOLO head. The YOLO head, a consistent feature across the series, utilizes these processed features to predict bounding box coordinates, objectness scores, and class probabilities, enabling the model to efficiently identify various objects.

**Intersection Over Union.** In YOLO based object detection algorithms, the term *Intersection Over Union (IoU)* stands for the metric used to quantify the accuracy of an object detector on a particular dataset. It measures the overlap between the predicted bounding box and the ground truth bounding box for each detected object. Specifically, IoU is calculated by dividing the area of overlap between the predicted bounding box

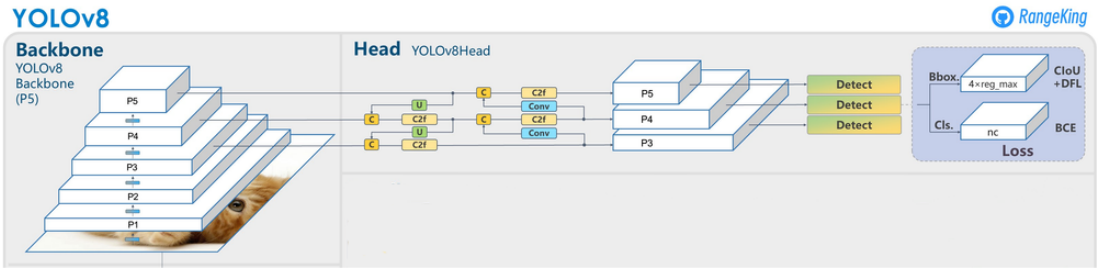


Figure 2.1: YOLOv8 Architecture [32]

and the ground truth bounding box by the area of union of these two boxes. The IoU score ranges from 0 to 1, where 1 indicates a perfect match between the predicted and the ground truth bounding boxes, and 0 indicates no overlap. In the training process of YOLO based models, a higher IOU indicates that higher

## 2.3 Machine Learning Basis

### 2.3.1 Mathemathical Basis

**Accuracy.** Accuracy is one of the most common metrics used, and it represents the proportion of true results (both true positives and true negatives) among the total number of cases examined.

$$\text{ACC} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.1}$$

However, accuracy may be misleading in datasets with imbalanced classes. In our case, since the dataset is curated to be relatively balanced (33:30), accuracy can serve as a meaningful evaluation on model performance.

**Precision.** Precision, also known as positive predictive value, measures the proportion of correctly predicted positive observations to the total predicted positives. It monotonically decreases when the FP rate increases, and thus can inform us with the FP rate.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.2}$$

**Recall.** Recall, also known as sensitivity or true positive rate, measures the proportion of actual positives that are correctly identified by the model. It is a crucial metric in situations where the cost of missing a positive case (a false negative) is high. In essence, it focuses on the model's ability to capture all relevant cases by minimizing false negatives.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.3}$$

**F1-score.** The F-score or F1 Score is the harmonic mean of precision and recall (sensitivity). It provides a balance between precision and recall in one number, capturing

the trade-off between the two. The F1 Score is useful as it reflects a balance between Recall and Precision; it is high only when both recall and precision are high.

$$\text{F-score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.4}$$

### 2.3.2 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are essential in visual data analysis, distinguished by their capacity to learn spatial hierarchies of features from images. This learning process is facilitated through convolutional layers that apply filters to input data, capturing critical attributes like edges and textures. The architecture of CNNs, which includes convolutional, pooling, and fully connected layers, enables them to efficiently extract and process meaningful patterns from the visual input.

The application of CNNs is notably exemplified in object detection tasks, such as those performed by the You Only Look Once (YOLO) series. The latest iteration, YOLOv8 [14], represents a significant advancement in this domain. It employs an anchor-free approach and a sophisticated neural network architecture that integrates Feature Pyramid Network (FPN) and Path Aggregation Network (PAN) for superior object detection across various scales.

YOLOv8's architecture is anchored by the CSPDarknet53 [27] backbone, optimized with Cross Stage Partial networks for enhanced feature extraction. This backbone processes complex visual information efficiently, while the novel neck architecture leveraging FPN and PAN ensures effective feature fusion. The YOLO head then uses these features to predict bounding boxes, objectness scores, and class probabilities with high accuracy. This streamlined approach underscores the power of CNNs in advancing object detection, highlighting their role in the ongoing evolution of computer vision technologies.

### 2.3.3 Feature Pyramid Network (FPN)

The Feature Pyramid Network (FPN) [19] is a pivotal architecture designed to enhance the capacity of convolutional neural networks (CNNs) for detecting objects across different scales. FPN effectively addresses the challenge of scale variability among objects by constructing a pyramid of features where each level represents features at a different scale. This multi-scale representation enables the model to detect objects ranging from very small to large within the same framework.

At the core of FPN's design is its top-down architecture with lateral connections. The network takes high-level semantic features from deeper, coarser layers of the network and enriches them with finer details from earlier layers through lateral connections. This process results in feature maps of varying resolutions, each carrying strong semantic information, making it possible to detect objects at different scales with high accuracy. FPN has been widely adopted in various object detection frameworks, including YOLOv8 [14], where it significantly improves detection performance by providing a robust method for handling scale variations within images.

### 2.3.4 Path Aggregation Network (PAN)

The Path Aggregation Network (PAN) [21] further refines the concept of feature fusion in convolutional neural networks, aiming to enhance the flow of information across different scales. PAN builds upon the foundation laid by FPN by adding an additional bottom-up pathway, which augments the flow of low-level detail to the topmost layers of the network. This enhancement ensures that even the highest-level feature maps retain fine-grained details, crucial for accurate object detection, especially for small objects.

PAN's architecture is characterized by its efficient aggregation of features at multiple levels, improving the feature hierarchy within the network. This is achieved by creating a more effective pathway for information from the initial layers of the network, which capture rich detail and texture information, to reach the output layers directly. Such an architecture not only bolsters the detection capabilities of models like YOLOv8 [14] but also contributes to the overall robustness and precision of the detection process across varying object sizes and complexities. By ensuring a seamless integration of both high-level semantic and low-level detail features, PAN plays a crucial role in the evolution of CNN architectures towards more sophisticated and nuanced object detection methodologies.

# Chapter 3

# Data Preparation

The main objective of this part of the work is to generate corrupted datasets as a virtual environmental dataset used for benchmarking model robustness by artificially adding corruptions to the existing clean dataset. The KITTI 2D object detection dataset was selected as the base dataset for generating the corrupted dataset for the experiments. The KITTI 2D object detection dataset consists of 7481 training images and 7518 test images, comprising a total of 80256 labelled objects.

Four common types of corruption are selected for the experiments: fog, rain, snow, and Motion Blur. Fog, rain, and snow are the weather that can easily affect the camera-based object detection model, and motion corruption is the corruption type that has the greatest impact on camera-based object detection models[6].

Table 3.1: Comparison existing corruption methods

| Method | depth include | AI based | weather condition include |
|---|---|---|---|
| 3DCC[17] | ✓ | ✓ | ✓ |
| 2DCC[11] | ✗ | ✓ | ✓ |
| ObjectNet[1] | ✗ | ✓ | ✗ |
| 3DB[18] | ✗ | ✗ | ✗ |
| Imageaug[16] | ✗ | ✓ | ✓ |
| Imagecorruptions[22] | ✗ | ✗ | ✓ |
| Augmix[12] | ✗ | ✓ | ✗ |

In our work, We studied and compared the methods in the Table 3.1 and selected two methods to generate corruption.

## 3.1 3DCC

First, I chose 3DCC for adding corruption to the dataset. The work on 3D Common Corruptions (3DCC) presents a novel framework for evaluating and enhancing the robustness of computer vision models by incorporating realistic, geometry-aware image

corruptions. Unlike previous benchmarks that apply uniform 2D modifications, 3DCC leverages scene depth information to simulate corruptions that are more representative of real-world scenarios, such as motion blur that varies with object distance or lighting changes affecting different parts of a scene differently.

3DCC has significant advantages for developing more robust AI systems. Firstly, it introduces a more challenging and comprehensive set of corruptions that models may encounter in real-world applications, thereby providing a more accurate measure of their performance and resilience. Secondly, by considering the 3D geometry of scenes, it helps in identifying and addressing the specific vulnerabilities of models to spatially varying perturbations, leading to improvements in model design and training methodologies.

Utilizing 3DCC can be particularly beneficial in applications where models are deployed in dynamic and visually complex environments, such as autonomous driving, medical imaging, and augmented reality. By training and testing models against 3DCC's suite of corruptions, developers can better ensure that their models are not only accurate under ideal conditions but also reliable when faced with the unpredictable nature of real-world visual data.

Moreover, the framework's extendability and efficiency in generating corruptions make it a versatile tool for the community. Researchers and practitioners can adapt 3DCC to their specific needs, applying it to various datasets and using it to explore new dimensions of robustness research. The ongoing development and use of 3DCC have the potential to significantly advance our understanding of AI robustness, pushing the field towards the creation of models that truly understand and can navigate the complexity of the 3D world.

These features of 3DCC make it by far the best method of generating corruption for this experiment.

In order to successfully generate a corrupted image using 3dcc, 3dcc requires the following inputs:

Table 3.2: Comparison of existing corruption methods

| Input | Detail |
|---|---|
| RGB Image | The base RGB image serves as the foundation for corruption generation. This image captures the visual content of the scene and serves as the canvas upon which corruptions will be applied. |
| Depth Map | The depth map provides crucial spatial information about the scene, allowing 3DCC to understand the three-dimensional layout of objects. Depth maps are typically obtained using depth-sensing technologies such as LiDAR or stereo vision. |
| Corruption Parameters | 3DCC offers a range of parameters that allow users to customize the type and severity of corruption applied to the image. These parameters may include variables such as fog density, rain intensity, motion blur strength, and lighting variations. |

Despite its innovative approach, 3DCC is not without challenges. The reliance on accurate depth information can limit its applicability, particularly in scenarios where such data is unavailable or of low quality. Furthermore, the current set of corruptions, while extensive, may not cover the full gamut of real-world anomalies, suggesting room for expansion and refinement.

However, difficulties were encountered when implementing the 3dcc method, as the 3dcc method requires a depth map to generate the corrupted type with depth of field information, and it requires a dense depth map, which creates a great challenge for applying the 3dcc method to the KITTI dataset. Since the sensor used in KITTIdataset to collect depth information is a Velodyne laser scanner, the depth data in the dataset is only a sparse point cloud with a density of about 30%, which is completely unable to be used as an input for 3dcc.

### 3.1.1 Depth Completion Methods reviewed

Depth completion methods[13] are critical for enhancing sparse depth information, a common challenge in computer vision tasks like autonomous navigation and 3D mapping. These methods aim to fill in missing or unreliable depth data to create comprehensive depth maps essential for accurate scene understanding. Two principal strategies for depth completion include:

- **Point Cloud Approach:** This technique transforms sparse depth inputs into point clouds, which are then densified through various interpolation methods. It might leverage geometric algorithms or deep learning models to infer the missing depth information based on the spatial distribution of known points. This approach is particularly useful in LiDAR-based depth acquisition, where depth data might be inherently sparse but highly accurate.
- **Monocular Camera Depth Estimation:** This method utilizes convolutional neural networks (CNNs) to predict depth from 2D images. Trained on large datasets with depth annotations, these models learn to identify visual cues associated with depth, such as object size, perspective, and shading, to estimate a dense depth map from monocular cues alone.

In an attempt to derive a dense depth map for this project, both depth-completion and monocular depth estimation techniques were explored. Unfortunately, the efforts faced setbacks when the generated images did not align with expectations, leading to unsatisfactory outcomes. Consequently, the implementation of the 3dcc method, aimed at integrating these depth completion strategies, was put on hold. This decision reflects the challenges and complexities inherent in depth completion, highlighting the need for further research and experimentation to refine these methods for practical applications.

Figure 3.1: Original image

Figure 3.2: Generated depth map

Figure 3.3: Corrupted image from 3DCC

## 3.2 Imgaug

As an alternative to 3dcc this experiment ended up using the imageaug library(cite) as the final method of adding corruptions to the dataset.

The imgaug library stands out in the field of machine learning for its comprehensive suite of image augmentation capabilities, designed to significantly enhance dataset diversity and robustness. This Python library supports a wide range of augmentation techniques for images, and it extends its functionality to keypoints/landmarks, bounding boxes, heatmaps, and segmentation maps. Its design focuses on efficiency, operating effectively across multiple CPU cores, and offers a stochastic interface that simplifies the creation of complex augmentation pipelines. This flexibility and power make imgaug an invaluable tool for preparing datasets in tasks requiring high levels of visual recognition and analysis accuracy, such as autonomous driving and medical image processing.

Fog: The project use imgaug library to implement it, and use the predefined severities level 3 from 1, 2, 3, 4, 5 to simulate fog environment. Rain: The project set the parameter of rainfall density as 0.10 from 0.01, 0.06, 0.10, 0.15, 0.20 in RainLayer in imgaug library to simulate rain environment. Besides, the project also add a 30%-opacity gray mask layer, and reduce the brightness by 30%.Snow: The project use the imgaug library[16] to implement it, and use the pre-defined severitie level 2 from 1, 2, 3, 4, 5 to simulate snow environment, also add a 30%-opacity gray mask layer, and reduce the brightness by 30%. Blur: The project use imgaug library to implement it, and use the predefined zoom factor 2 from 1, 2, 3, 4, 5 to simulate blur condition.

The KITTI 2D objection detection dataset contains 7481 training and 7518 test samples. As we do not have access to labels to the test set, Our corrupted KITTI dataset is constructed upon the training set. Therefore our dataset will contain 4 types of corruption, each containing 7481 images that have been added with the corresponding corruption type.

Figure 3.4: Clean image

Figure 3.5: Image with fog

Figure 3.6: Image with rain

Figure 3.7: Image with snow

Figure 3.8: Image with blur

# Chapter 4

# Evaluations

## 4.1 Experiment Setup

To evaluate the robustness of YOLOv8 against the common corruptions mentioned above, experiments were conducted by training and testing the YOLOv8 object detection model on the SemanticKITTI dataset. This included both a clean dataset and a synthetically corrupted dataset, respectively. Additionally, testing was performed on models trained with the clean dataset but tested on the corrupted dataset to provide a reference set. To ensure the consistency of results, all experiments were performed with the same number of images in the training and testing sets, with 5984 images and 1497 images respectively. The Intersection over Union (IoU) threshold was set to 0.7, the training epochs to 10, and the batch size to 16. All experiments were conducted on an NVIDIA RTX 4090 Laptop GPU [23].

## 4.2 Experiment Results

From the confusion matrices below, we can see that when the models are both trained and tested on the clean data, or both on the same corrupted data, their performances are significantly better than when they are trained on different groups of data. This suggests that adding corruption would significantly affect the data distribution and thus YOLO's performance.

### 4.2.1 Accurcy

The detailed results of the average accuracy of each model are shown in Table 4.1. Training and testing the YOLOv8m model on clean data sets a benchmark, yielding an accuracy of 72.14%. This performance serves as a baseline for evaluating the impact of various corruptions on model accuracy. Training and validating on foggy conditions resulted in an accuracy drop to 65.00%. This decrease highlights the model's struggles with obscured visibility, a common issue in foggy environments where the distinction between objects and the background can be significantly diminished. Similar to fog, rain introduced visual noise and dynamic changes, with the model achieving a slightly higher

accuracy of 67.10%. The uniform whiteness and potential for occlusion and alteration of object contours in snow significantly hamper the model's detection capabilities. The model faced its most significant challenge with snow, with accuracy plummeting to 61.13%. Representing a general reduction in image clarity, blur led to an accuracy of 61.25%, closely mirroring the difficulties observed with snow. This underlines the model's difficulties in handling images where details are smeared or obscured, affecting its ability to discern and classify objects accurately.

The first row of the table, revealing how the YOLOv8m model trained on clean data falters when faced with environmental corruptions such as fog, rain, snow, and blur, starkly highlights the model's limitations in adapting to varied visual disturbances. The accuracy drops from a baseline of 72.14% in clean conditions to 41.50% with fog, descending further to 34.25% with rain, plummeting to 16.88% under snow conditions, and reaching its nadir at 13.50% when confronted with blur. This descending trend in accuracy underscores a critical vulnerability of the model: its performance is significantly compromised as the visual clarity of its input deteriorates. Among these, the challenges posed by blur—which effaces fine details critical for object identification—and by snow—which alters object appearances and the environment drastically—prove particularly debilitating. Conversely, fog and rain, despite reducing visibility and introducing dynamic visual noise, result in less dramatic declines.

The marked decline in model accuracy, when transitioning from clean to corrupted conditions or vice versa, points to a significant generalization gap. For instance, when the model trained on clean data is tested on corrupted datasets (fog, rain, snow, blur), there's a stark performance drop. It suggests that the model heavily relies on the pristine conditions of the training dataset, which lacks the complexity and variability of real-world scenarios.

Table 4.1: Results in Accuracy(%)

| Model | clean | fog | rain | snow | blur |
|---|---|---|---|---|---|
| Clean KITTI model | 72.14 | 41.50 | 34.25 | 16.88 | 13.50 |
| Fog model | - | 65.00 | - | - | - |
| Rain model | - | - | 67.1 | - | - |
| Snow model | - | - | - | 61.13 | - |
| Blur model | - | - | - | - | 61.25 |

### 4.2.2 Confusion Matrix

The thorough analysis of the YOLOv8m model, examining its performance trained on clean data and tested across various corruptions, reveals key trends and gaps in its detection capabilities. The confusion matrices in Figure 4.1 show that under conditions like blur, the model's accuracy significantly drops. This visual interference results in an increase in both false positives and false negatives across several classes. For example, vehicles such as 'Cars' and 'Vans,' which the model usually distinguishes well in clear

conditions, are often mixed up when blurred, leading the model to confuse one for the other or to miss them altogether.

In conditions like fog, the confusion matrix displays a similar, but less severe, pattern compared to blur. This may be because fog, while it hides details, doesn't alter the shapes of objects as much as blur does. The model manages to hold onto the basic shapes, resulting in slightly better performance in fog than in blur, but still makes many mistakes, especially with less distinct object categories.

The problem becomes worse with rain and snow, both of which add dynamic and static visual noise, respectively. In snow, the uniform visual input results in a higher rate of object misclassification, with the confusion matrices showing a notable rise in false negatives. This suggests that objects covered or surrounded by snow often blend into the background. Rain, while also challenging, doesn't impact the model as much, possibly because the visual distortions caused by rain are more temporary and less overwhelming than the blanket effect of snow.

When looking at the model's performance on matched conditions, where it is trained and tested on the same type of corruption, the matrices reflect greater expertise, suggesting that the model has learned to recognize and predict patterns specific to each corruption type. However, this skill does not generalize well across conditions, which is critical for real-world applications where variability is common.

A closer examination of the matrices for each condition shows the model's varying response to different classes under corruption. For 'Car' and 'Van' categories, despite some confusion, the model still achieves a moderate true positive rate. However, for categories like 'Cyclist,' 'Truck,' and 'Misc,' the true positive rates are significantly lower across all corruption types. This inconsistency in performance could be due to the different levels of visual complexity and representation in the training data, particularly for categories like 'Tram' and 'Person sitting,' where the model's performance is particularly poor in corrupted conditions.

Notably, the 'Cyclist' class shows a dramatic drop in detection in corrupted conditions, which is concerning for safety in autonomous driving scenarios. Similarly, the 'Truck' class, likely because of its larger size and unique shape, faces high misclassification rates, especially in snow conditions where it might be mistaken for part of the environment.

The main trend across all corrupted conditions is the model's significant difficulty in accurately detecting less common and smaller objects, suggesting a potential overfitting to more common features in the training data. This indicates a need to improve the model's training approach, using a broader and more balanced dataset that includes adequate representation of all object classes under various conditions.

### 4.2.3 F1 score

Looking at the F1-confidence in Figure 4.2curves alongside the confusion matrix offers a comprehensive view of the YOLOv8m model's operational strength under different environmental conditions. The curves, which show the trade-off between precision and recall at different confidence levels, reveal the model's certainty in its predictions and

the balance between detecting as many true positives as possible while minimizing false positives.

In clear conditions, the F1 curves for classes like 'Cars' and 'Trucks' reach high levels, showing strong model performance with high certainty in its predictions. This high confidence is supported by high true positive rates and low false positive rates in the confusion matrix, showing the model's skill in identifying these categories in clear settings, likely due to their well-defined and distinct features in the training data.

When conditions change, like in images affected by blurring, there's a noticeable drop in the F1 score for almost all classes. The curves go down, showing a decrease in both precision and recall. This indicates a significant drop in the model's confidence in its predictions when faced with obscured or softened object edges. The confusion matrix supports this, showing increased confusion between similar classes like 'Cars' and 'Vans' and an overall rise in mistakes, highlighting the model's struggle to use detailed features from visually altered input.

Under foggy conditions, the gradual decrease in the F1 curves suggests a better preservation of shape recognition, as fog tends to maintain the outer contours of objects to some extent. However, the confusion matrix shows an increased difficulty in distinguishing between objects with similar shapes, like 'Trucks' and 'Vans,' which can blend into the foggy background.

With rain and snow, the F1 curves show significant variation for different classes. For example, the 'Cyclist' curve drops sharply, especially in snow, emphasizing the model's challenge with smaller, less contrasted figures against a chaotic background. This is concerning, given the safety implications in real-world autonomous navigation. The confusion matrix reflects these concerns, indicating a rise in false negatives where 'Cyclists' and potentially dangerous objects like 'Person sitting' are frequently overlooked or mistaken for harmless elements.

The consistency of the model under matched conditions—such as being trained and tested in fog—shows a resilient F1 curve, maintaining higher scores across the board. Yet, the true challenge lies in cross-condition robustness, where the model shows notable weakness, as seen in the steep drop of the F1 scores when, for instance, a model trained in clear conditions is tested on snowy images.

### 4.2.4 Perception-Recall curve

In ideal, clear conditions, the Precision-Recall (PR) curves Figure 4.3 for several classes such as 'Car' and 'Truck' rise towards high precision levels, reflecting the model's accuracy in predicting these object classes with fewer mistakes—a key feature in autonomous systems. These results align with the high mean Average Precision (mAP) scores, suggesting strong detection capability for classes that are commonly encountered and have distinct features in the training set.

However, the model's precision noticeably declines in the PR curves when faced with blur-corrupted environments. Such conditions significantly reduce the model's effectiveness, particularly for 'Cyclist' and 'Person sitting' classes, which show a steep

drop in precision. This highlights the model's sensitivity to loss of edge information—a critical issue when the clarity of contours is essential for class differentiation. This decline is consistent with the analysis of the F1-confidence curve, where a lower confidence level leads to a decreased F1 score due to more mistakes and missed detections.

Fog has a less noticeable but still significant effect on precision. Classes with unique shapes like 'Tram' manage to keep relatively high precision, as their distinct silhouettes remain visible. However, for others, such as 'Pedestrian' and 'Cyclist,' the curves indicate an increasing challenge, with precision decreasing more quickly as recall increases. This matches the trends seen in the confusion matrix and F1 curves, where the model's difficulty in distinguishing fine details under foggy conditions becomes apparent.

Rain and snow have varied impacts across the classes. While 'Truck' and 'Tram' keep some level of high precision, showing the model's ability to recognize larger objects despite visual challenges, 'Cyclist' and 'Person sitting' classes face significant difficulties. The curves suggest that dynamic elements like rain and static disruptions like snow greatly affect the model's ability to maintain a high true positive rate without increasing the number of false positives, as shown by the PR curve's steeper descent.

A detailed look at PR curves under matched corruption conditions (e.g., trained and tested in rain) shows an increased mAP, indicating the model's specialized adaptation to specific disturbances. Yet, the significant drop in precision across untrained conditions reveals a crucial flaw in the model's ability to generalize—a problem that could hinder real-world deployment where environmental consistency is not assured.

## 4.3 Measure the model confidence

### 4.3.1 Confidence Definition

To evaluate the confidence of the model, we need to introduce a confidence function to measure the confidence of the model's predictions.

Given the predicted bounding box of a plot, let $C_t$ be the center pixel of the ground-truth bounding box, and $C_p$ be the center pixel of the predicted bounding box. We measure the correctness of the prediction and the ground truth as the negative exponential of the Euclidean distance between the predicted and ground-truth center pixels:

$$correctness(C_t, C_p) = \exp[-w.||C_t - C_p||_2] \tag{4.1}$$

where $||\cdot||_2$ represents the $L_2$ distance, and $w \in \mathbb{R}$ is some regularization constant. Here we use the exponential function because we want to use the property that the exponential function reduces monotonically as the distance $||C_t - C_p||_2$ grows. Moreover, it has the desirable property that when $x \geq 0$, $exp[x] \in (0, 1]$, which can be directly used to represent probability.

The reason that we selected this exponential function as a measure of correctness is to punish very bad predictions. In practice, if the predicted bounding box is too far away

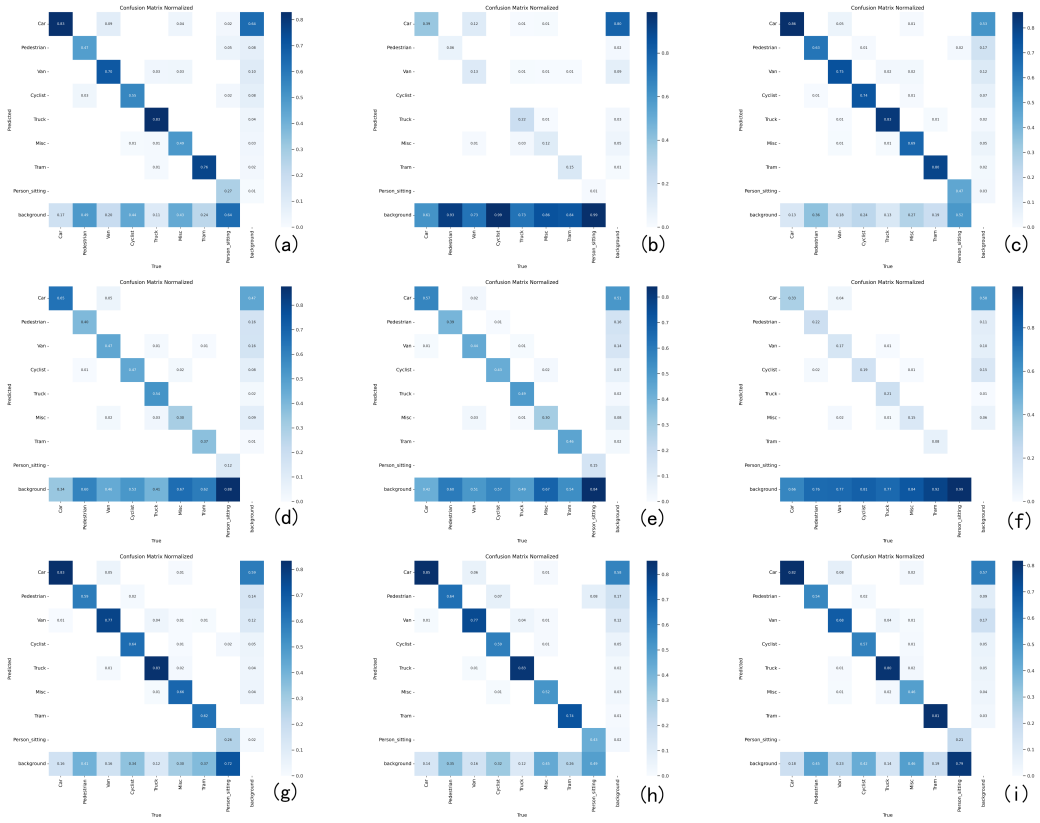


Figure 4.1: Confusion Matrixs (a)blur_blur,(b)clean_blur,(c)clean_clean,(d)clean_rain, (e)clean_snow,(f)fog_fog,(h)rain_rain,(i)snow_snow

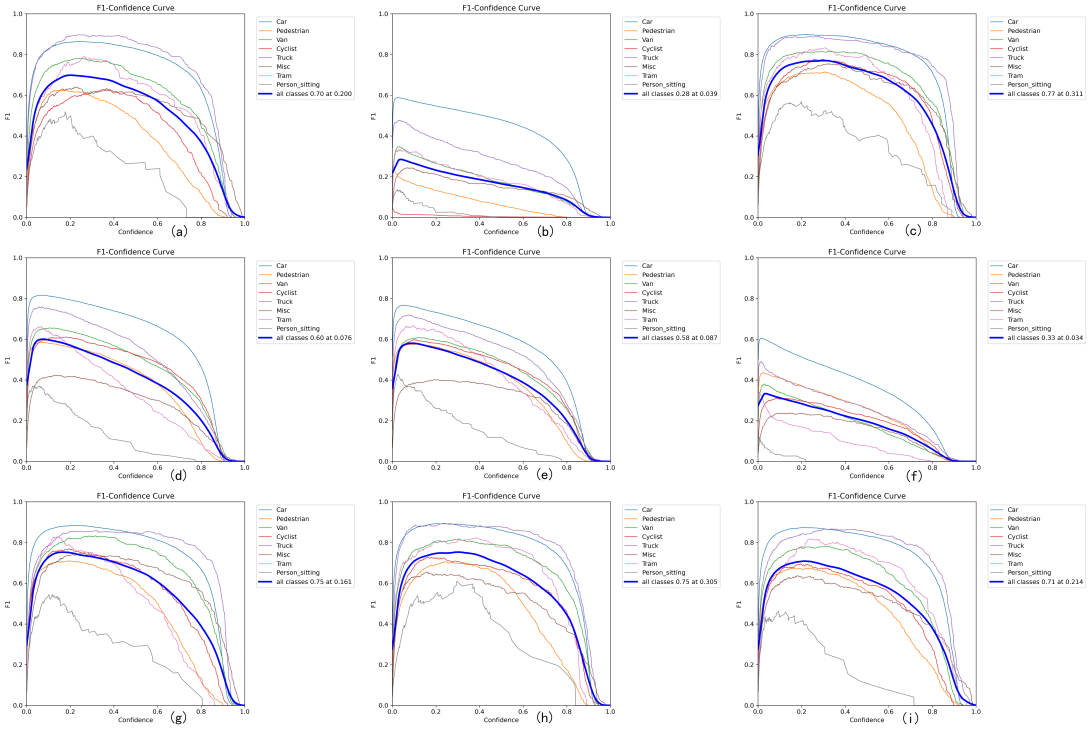


Figure 4.2: F1-curves (a)blur_blur,(b)clean_blur,(c)clean_clean,(d)clean_rain, (e)clean_snow,(f)fog_fog,(h)rain_rain,(i)snow_snow

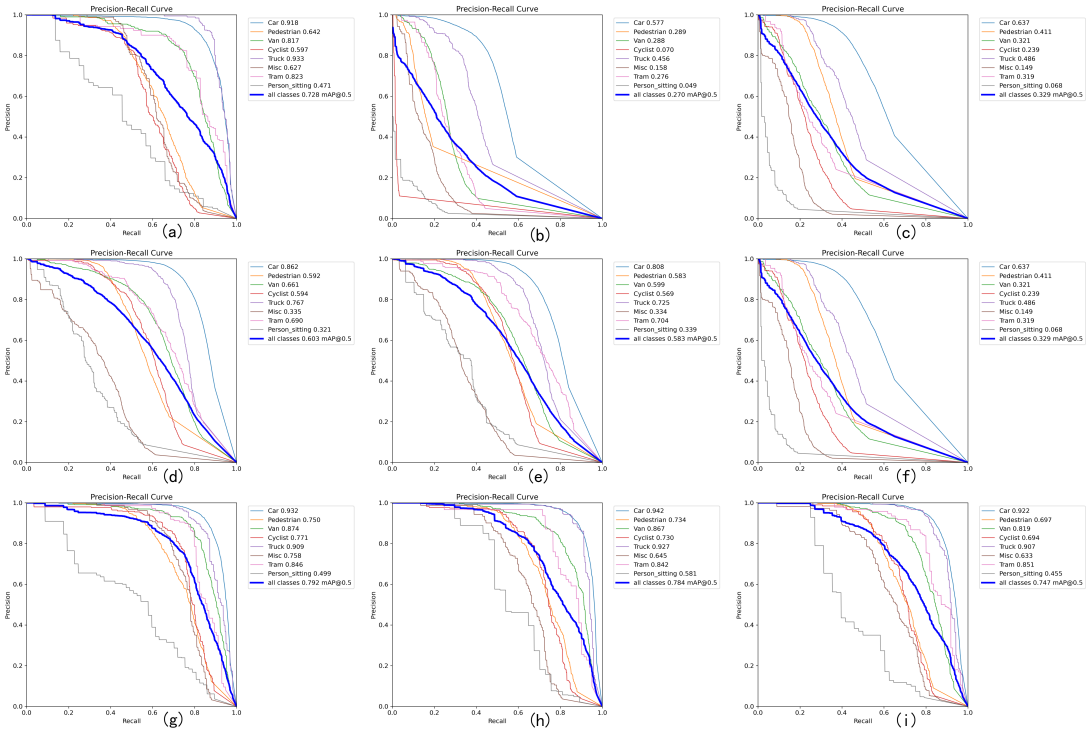


Figure 4.3: PR-curves (a)blur_blur,(b)clean_blur,(c)clean_clean,(d)clean_rain, (e)clean_snow,(f)fog_fog,(h)rain_rain,(i)snow_snow

from the ground truth one, it is better to consider it a false positive, instead of one that has a relatively small accuracy. So, ideally, we want the confidence function to be very small when the distance is big. Thus, the exponential function is just right for this job.

Then, assuming the input features are with distribution $\mathcal{D}$, with ground-truth-bounding box centers $\mathcal{Y}$, where each $y \in \mathcal{Y}$ is a pixel on the image, formally $y \in \{1,...,W\}\mathbf{x}\{1,...,L\}$, where $W$ and $D$ represents the image width and length, respectively. Given a model $f : \mathcal{D} \to \mathcal{Y}$, we define the confidence of $f$ to be:

$$confidence(f) = \mathbb{V}_{\mathbf{x}\sim\mathcal{D}}[correctness(f(\mathbf{x}), y_{\mathbf{x}})] \tag{4.2}$$

where $y_{\mathbf{x}}$ is the ground-truth bounding box's center of $\mathbf{x}$. In intuitively, this metric measures the overall distances to which the predicted bounding boxes are different from the ground truth bounding boxes.

In practice, to estimate confidence on a sample test $\mathcal{S}$, we define the empirical confidence as follows,

$$confidence(f) = \frac{1}{\mathcal{S}} \sum_{\mathbf{x}\in\mathcal{S}} [correctness(f(\mathbf{x}), y_{\mathbf{x}}) - \overline{correctness(f(\mathbf{x}), y_{\mathbf{x}})}]^2 \tag{4.3}$$

where $\overline{correctness(f(\mathbf{x}), y_{\mathbf{x}})}$ is the sample mean of the correctness,

$$\overline{correctness(f(\mathbf{x}), y_{\mathbf{x}})} = \frac{1}{\mathcal{S}} \sum_{\mathbf{x}\in\mathcal{S}} [correctness(f(\mathbf{x}), y_{\mathbf{x}})] \tag{4.4}$$

### 4.3.2 Results

We applied the above definition of correctness and confidence on our trained models and calculated them on the trained datasets. The distribution of the correctness is given in Figure 4.4 According to the confidence metric larger values indicate that the centroid

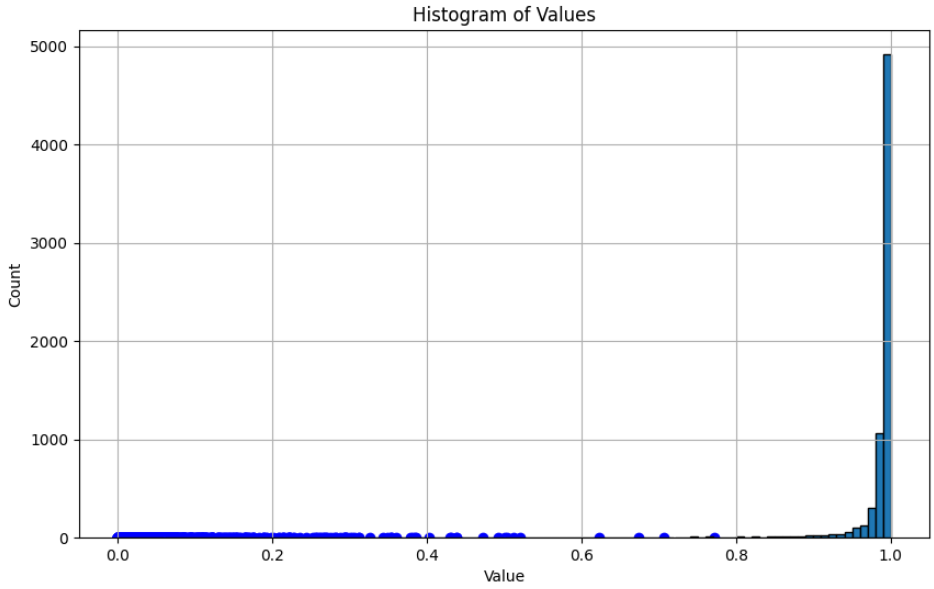


Figure 4.4: Distribution of bonding box predicted on fog model

of the detected bonding box is closer to the GT. The confidence result of the model should be close to normal distribution, Figure 4.4 reflects this well.

# Chapter 5

# Discussions

## 5.1 Limitations and Future Works

Despite the fact that this work included common corruptions in autonomous driving, only a single experiment for each type of corruption is conducted, whereas real-world environments often contain a mixture of several types of corruption at the same time. Potential improvements could be made focus on:

1. **Expand the Range of Corruption Types.** Investigate a broader spectrum of corruption scenarios, including those not covered in the current study, to gain a more comprehensive understanding of AV perception system vulnerabilities.
2. **Explore Mixed Corruption Scenarios** Design experiments that simulate real-world environments where multiple corruption types occur simultaneously, to assess the compound effects on AV system performance.
3. **Diversify Models and Datasets.** Utilize a variety of models and datasets in future experiments to ensure the findings are robust and broadly applicable across different AV technologies.
4. **Study Long-Term Adaptation Strategies.** Investigate long-term model adaptation strategies that allow AV systems to dynamically adjust to new and evolving corruption types without notable overfitting.

Future Research Directions Considering these limitations, future research should aim to:

1. **Expand the Range of Corruption Types/** Investigate a broader spectrum of corruption scenarios, including those not covered in the current study, to gain a more comprehensive understanding of AV perception system vulnerabilities.
2. **Explore Mixed Corruption Scenarios.** Design experiments that simulate real-world environments where multiple corruption types occur simultaneously, to assess the compound effects on AV system performance.
3. **Diversify Models and Datasets.** Utilize a variety of models and datasets in future experiments to ensure the findings are robust and broadly applicable across

different AV technologies.

4. **Study Long-Term Adaptation Strategies.** Investigate long-term model adaptation strategies that allow AV systems to dynamically adjust to new and evolving corruption types without significant overfitting.

## 5.2 Conclusions

To enhance the robustness of perception systems for autonomous vehicle (AV), a thorough investigation is taken including synthetic data generation and its influence on model resilience in corrupted environments. This evolution from focusing primarily on Perception Error Metrics (PEM) to a wider analysis of synthetic corruption and its effects on the durability of perception systems signifies a pivotal shift in the direction of this research.

**Synthetic Data Generation.** The creation of synthetic data, includes an array of corruption conditions from weather-induced alterations to sensor-specific distortions, has become the fundamentals of this thesis. Through detailed experimentation, we found that integrating such synthetically corrupted datasets into the training process significantly improves model resilience. This finding supports the research assumption that training models on a wide spectrum of potential real-world inaccuracies is crucial for achieving robustness.

**Evaluating Model Robustness.** A detailed evaluation of model performance under various corrupted conditions has revealed significant variability in the resilience of perception systems. The variation in performance across different types of corruption highlights the complexity of 'robustness,' indicating that it cannot be universally measured but is dependent on the specific nature of the encountered corruption. As such, the development of AV perception systems should emphasize adaptability, preparing models not only for diverse conditions but also for unseen environmental shifts.

This research emphasizes the critical role of synthetic data in narrowing the gap between controlled laboratory settings and the unpredictable real-world conditions. The insights from assessing model robustness against diverse corrupted datasets stress the importance of adopting a comprehensive approach to training, which prioritizes not just accuracy in optimal conditions but also resilience to corruption.

Moving forward, this thesis sets the stage for further studies into advanced synthetic data generation methods, the incorporation of adaptive algorithms in perception systems, and the enhancement of benchmarking protocols to more accurately measure robustness in AV applications.

# Bibliography


[1] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[2] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11679–11689, 2020.

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving, 2020.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] Carlos A. Diaz-Ruiz, Youya Xia, Yurong You, Jose Nino, Junan Chen, Josephine Monica, Xiangyu Chen, Katie Luo, Yan Wang, Marc Emond, Wei-Lun Chao, Bharath Hariharan, Kilian Q. Weinberger, and Mark Campbell. Ithaca365: Dataset and driving perception under repeated and challenging weather conditions. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21351–21360, 2022.

[6] Yinpeng Dong, Caixin Kang, Jinlai Zhang, Zijian Zhu, Yikai Wang, Xiao Yang, Hang Su, Xingxing Wei, and Jun Zhu. Benchmarking robustness of 3d object detection to common corruptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1022–1032, June 2023.

[7] Yinpeng Dong, Shouwei Ruan, Hang Su, Caixin Kang, Xingxing Wei, and Jun Zhu. Viewfool: Evaluating the robustness of visual recognition to adversarial viewpoints, 2022.

[8] Shuo Feng, Yiheng Feng, Haowei Sun, Shan Bao, Yi Zhang, and Henry X.

Liu. Testing scenario library generation for connected and automated vehicles, part ii: Case studies. *IEEE Transactions on Intelligent Transportation Systems*, 22(9):5635–5647, 2021.

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[10] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2d2: Audi autonomous driving dataset, 2020.

[11] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations, 2019.

[12] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty, 2020.

[13] Junjie Hu, Chenyu Bao, Mete Ozay, Chenyou Fan, Qing Gao, Honghai Liu, and Tin Lun Lam. Deep depth completion from extremely sparse data: A survey, 2022.

[14] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023.

[15] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, November 2022.

[16] Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. imgaug. `https://github.com/aleju/imgaug`, 2020. Online; accessed 01-Feb-2020.

[17] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18963–18974, 2022.

[18] Guillaume Leclerc, Hadi Salman, Andrew Ilyas, Sai Vemprala, Logan Engstrom, Vibhav Vineet, Kai Xiao, Pengchuan Zhang, Shibani Santurkar, Greg Yang, Ashish Kapoor, and Aleksander Madry. 3db: A framework for debugging computer vision models, 2021.

[19] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

[20] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018.

[21] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8759–8768, 2018.

[22] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

[23] NVIDIA. GeForce RTX 40 Series Laptops. `https://www.nvidia.com/en-gb/geforce/laptops/`, 2023. Accessed: 2024-04-03.

[24] Andrea Piazzoni, Jim Cherian, Mohamed Azhar, Jing Yew Yap, James Lee Wei Shung, and Roshan Vijay. Vista: a framework for virtual scenario-based testing of autonomous vehicles. In *2021 IEEE International Conference on Artificial Intelligence Testing (AITest)*. IEEE, August 2021.

[25] Andrea Piazzoni, Jim Cherian, Justin Dauwels, and Lap-Pui Chau. Pem: Perception error model for virtual testing of autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 25(1):670–681, January 2024.

[26] M Pitropov, DE Garcia, J Rebello, et al. Canadian adverse driving conditions dataset. *The International Journal of Robotics Research*, 40(4-5):681–690, 2021.

[27] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[28] Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions, 2022.

[29] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla. A systematic review of perception system and simulators for autonomous vehicles research. *Sensors*, 19(3):648, 2019.

[30] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, Sep 2018.

[31] Yuan Shen, Shanduojiao Jiang, Yanlin Chen, and Katie Driggs Campbell. To explain or not to explain: A study on the necessity of explanations for autonomous vehicles, 2022.

[32] Jacob Solawetz and Francesco. What is yolov8? the ultimate guide. Roboflow Blog, Jan 2023. Available online: `https://blog.roboflow.com/whats-new-in-yolov8/`.

[33] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng

Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[34] Deepak Talwar, Sachin Guruswamy, Naveen Ravipati, and Magdalini Eirinaki. Evaluating validity of synthetic data in perception tasks for autonomous vehicles. In *2020 IEEE International Conference On Artificial Intelligence Testing (AITest)*, pages 73–80, 2020.

[35] Juan Terven, Diana-Margarita Córdova-Esparza, and Julio-Alejandro Romero-González. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716, November 2023.

[36] Peng Wang, Xinyu Huang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[37] Weisong Weisong Wen, Guohao Zhang, and Li-Ta Hsu. Gnss nlos exclusion based on dynamic object detection using lidar point cloud. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):853–862, 2021.

[38] Shizhe Zang, Ming Ding, David Smith, Paul Tyler, Thierry Rakotoarivelo, and Dali Kaafar. The impact of adverse weather conditions on autonomous vehicles: Examining how rain, snow, fog, and hail affect the performance of a self-driving car. *IEEE Vehicular Technology Magazine*, PP:1–1, 03 2019.