Thesis for The Degree of Licentiate of Engineering

Handling Data Leakage in Automotive Datasets for Object Detection

Md Abu Ahammed Babu

Division of Interaction Design & Software Engineering
Department of Computer Science & Engineering
Chalmers University of Technology and University of Gothenburg
Gothenburg, Sweden, 2025

"There is only one way to see things, until someone shows us how to look at them with ***different eyes****"*
- Pablo Picasso

Abstract

Background: Object detection is a central component of automotive perception systems that supports safe operation of autonomous driving technologies. The performance of such models is typically evaluated using large-scale image datasets, where choices in dataset construction and splitting strategies strongly influence the ability to detect objects correctly and accurately. Specifically, image similarity between training and test sets can unintentionally create data leakage, a situation where information from the test set is indirectly accessible during training, leading to overly optimistic performance estimates and threatening the reliability of evaluation results. This can have serious consequences in safety-critical domains such as autonomous driving, where a robust and trustworthy object detection model is a prerequisite for deployment.

Objective: The overall aim of this thesis is to investigate the problem of data leakage in automotive object detection research. Specifically, it seeks to understand how different dataset splitting strategies impact the evaluation performance of models, and also to establish methods for detecting data leakage in existing train-test splits of frequently used automotive datasets to enable a more reliable assessment of object detection models.

Method: The research follows an empirical approach and is structured with four studies. Papers A and B adopt a quantitative, experimental design to investigate the extent of various data splitting strategies of automotive image datasets and the overall impact on the model performance. Papers C and D follow a design science methodology, introducing the D-LeDe method for detecting data leakage in any existing data split, and evaluating its effectiveness on multiple automotive datasets and object detection model architectures.

Findings: The results in paper A demonstrate that image similarity can have a measurable impact on reported performance in object detection models. Results in paper B show that splitting data based on semantic similarity can significantly enhance the overall performance. The proposed D-LeDe method is presented in paper C to detect data leakage in any existing data split. Paper D presented how effectively the D-LeDe method performs across different datasets and with multiple models to identify which of the splits have data leakage in them.

Conclusions: This thesis demonstrates that data leakage caused by image similarity across dataset partitions is a tangible and non-trivial problem in automotive object detection. Even modest overlap between training and test sets can inflate model performance, leading to overly optimistic conclusions about generalisation. In larger or highly redundant datasets, the effect can be even stronger. The findings confirm that data leakage is a systematic risk that can occur in many renowned datasets and can compromise the reliability of models benchmarked using the default splits of such datasets. Recognising this problem and practicing caution to avoid data leakage is therefore essential to ensure the reliability of automotive perception models.

Keywords Automotive Perception, Object Detection, Data Leakage, Image Similarity, Dataset Splitting

Acknowledgment

First and foremost, I would like to express my deepest gratitude to the Almighty for giving me the strength, health, and patience to complete this journey. Without His blessings, none of this would have been possible.

I am profoundly grateful to my supervisor, Professor Miroslaw Staron, for his continuous guidance, insightful feedback, and encouragement throughout my research. His expertise and dedication have been instrumental in shaping this work.

I would also like to sincerely thank my co-supervisors, Dr. Darko Durisic and Dr. András Bálint, for their valuable advice, constructive comments, and constant support. My special thanks also go to Dr. Ashok Chaitanya Koppisetty, who has guided me with patience and shared his expertise generously during this journey. I am equally thankful to my mentor Dr. Sushant Kumar Pandey, whose feedback, discussions, and encouragement have enriched this research.

I want to extend my sincere appreciation to my colleagues at Volvo Cars for their support and collaboration throughout this research. Their technical insights, constructive discussions, and practical perspectives greatly enriched the industrial relevance of this work.

I am very grateful to all the wonderful friends and colleagues at IDSE, who constantly supported with their knowledge, experience, and wisdom. Their unwavering support, encouragement, and friendship have created an inspiring and enjoyable work environment that made this tough journey not only productive but also truly memorable. I can never thank enough for all the supportive words I receive from you.

Finally, and most importantly, I want to express my heartfelt gratitude to my family. I dedicate this work to the memory of my beloved mother, whom I lost during the course of this work and whose love, wisdom, and sacrifices continue to guide me in every step of life. To my father, whose strength and resilience amid illness have been a constant source of inspiration for me. To my siblings, for their unwavering encouragement and faith in me. To my loving wife, for her patience, understanding, and steadfast support throughout this long journey — this achievement is as much yours as it is mine. And to my son, Alfid, who turned one during this time - your smile has been my greatest source

of motivation, even from afar. I hope one day you will understand that every moment spent apart was for building something that I hope will make you proud.

This research was conducted within the DeVeLoP research project which is funded by Volvo Cars, as part of the Volvo Industrial PhD Program (VIPP), and by the Swedish Governemnt Agency for Innovation (VINNOVA) under the grant number: 2018-02725.

List of Publications

Appended publications

This thesis is based on the following publications:

[A] **M. A. A. Babu**, S. K. Pandey, D. Durisic, A. C. Koppisetty, and M. Staron.
"Impact of Image Data Splitting on the Performance of Automotive Perception Systems"
Proceedings of the International Conference on Software Quality 2024.

[B] **M. A. A. Babu**, S. K. Pandey, D. Durisic, A. C. Koppisetty, and M. Staron.
"Exploring Image Similarity-Based Splitting Techniques in Automotive Perception Systems."
Proceedings of the 17th International Conference on the Quality of Information and Communications Technology (QUATIC) 2024.

[C] **M. A. A. Babu**, S. K. Pandey, D. Durisic, A. C. Koppisetty, and M. Staron.
"D-LeDe: A Data Leakage Detection Method for Automotive Perception Systems."
Proceedings of the 11th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS) 2025.

[D] **M. A. A. Babu**, M. Staron, S. K. Pandey, D. Durisic, and A. Bálint.
"Evaluation of the D-LeDe Method for Image Data Leakage Detection in Automotive Datasets."
Submitted to the Journal of Machine Vision and Applications 2025.

Other publications

The following publications were published during my PhD studies. However, they are not appended to this thesis, due to overlapping contents of appended publications or contents not related to the thesis:

[A] A. Barua and **M. A. A. Babu**.
"AI Data Analytics for Air Quality Modeling."
Proceedings of the 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT) 2024.

[B] **M. A. A. Babu**, S. K. Pandey, D. Durisic, A. Bálint, and M. Staron.
"Cirrus++: Scalable 2D Annotations for Multi-Modal AD Research."
Proceedings of the 18th International Conference on the Quality of Information and Communications Technology (QUATIC) 2025.

Research Contribution

The CRediT (Contribution Roles Taxonomy) model [1] was employed to determine the authors' contribution to the appended papers of this thesis. I bear the primary responsibility of planning, designing, and writing of the thesis. I contributed to co-conceptualizing, co-designing the methodology and experiments, co-execution of the experiments, data gathering, data analysis, and authoring the papers A, B, C, and D. Table 1 outlines my contributions to the appended papers.

Role	**Papers**			
	A	**B**	**C**	**D**
Conceptualization	✓	✓	✓	✓
Methodology	✓	✓	✓	✓
Software	✓	✓	✓	✓
Validation	✓	✓	✓	✓
Formal Analysis	✓	✓	✓	✓
Investigation	✓	✓	✓	✓
Resources	✓	✓	✓	✓
Data curation	✓	✓	✓	✓
Writing – Original Draft	✓	✓	✓	✓
Writing – Review & Editing				
Visualization	✓	✓	✓	✓
Supervision				
Project Administration				
Funding acquisition				

Table 1: The author's contribution to the appended papers of the thesis.

List of Abbreviations

ACC	Adaptive Cruise Control
AD	Autonomous Driving
ADAS	Advanced Driver Assistance Systems
AEB	Autonomous Emergency Braking
AI	Artificial Intelligence
APS	Automotive Perception Systems
AV	Autonomous Vehicle
CAN	Controller Area Network
CRediT	Contribution Roles Taxonomy
CNN	Convolutional Neural Network
CV	Computer Vision
DAP	Data Analysis Plan
DL	Deep Learning
D-LeDe	Data Leakage Detector
DNN	Deep Neural Network
ECU	Electronic Control Unit
GPS	Global Positioning System
HMI	Human Machine Interaction
ITS	Intelligent Transportation Systems
LiDAR	Light Detection and Ranging
LKA	Lane Keeping Assist
mAP	mean Average Precision
ML	Machine Learning
NN	Neural Network
OD	Object Detection
ODD	Operational Design Domain
OEM	Original Equipment Manufacturer
OOD	Out Of Distribution
OTA	Over-the-Air
SAE	Society of Automotive Engineers
V2X	Vehicle-to-Everything
YOLO	You Only Look Once

Contents

Abstract iii

Acknowledgement v

List of Publications vii

Research Contribution ix

List of Abbreviations xi

1 Introduction 1
- 1.1 Background . . . 1
  - 1.1.1 Automotive Software Systems . . . 1
  - 1.1.2 Automotive Perception Systems . . . 3
  - 1.1.3 Object Detection in Automotive Systems . . . 4
  - 1.1.4 Automotive Image Datasets . . . 5
  - 1.1.5 Data Splitting and Its Role in Model Evaluation . . . 7
  - 1.1.6 Image Similarity . . . 8
  - 1.1.7 Data Leakage in Image Datasets . . . 10
- 1.2 Research Design . . . 10
  - 1.2.1 Research Goals and Questions . . . 11
  - 1.2.2 Research Methodology . . . 12
    - 1.2.2.1 Research Approach . . . 12
    - 1.2.2.2 Research Methods . . . 13
- 1.3 Related Work . . . 15
  - 1.3.1 Data Splitting Strategies . . . 15
  - 1.3.2 Image-similarity measures . . . 16
  - 1.3.3 Similarity-based splitting and deduplication methods . . . 17
  - 1.3.4 Data Leakage Detection and Mitigation . . . 18
- 1.4 Summary of Findings . . . 19
  - 1.4.1 Impact of Data Splitting on Model Evaluation (RG1) . . . 19
  - 1.4.2 Effectiveness of Similarity-based Splitting (RG2) . . . 20
  - 1.4.3 Development of the D-LeDe (Data Leakage Detector) Method (RG3) . . . 21
  - 1.4.4 Cross-dataset and Cross-architecture Validation of the D-LeDe Method (RG4) . . . 22
- 1.5 Threats to Validity . . . 23
- 1.6 Discussion . . . 25

1.7 Conclusions and Future Work 26

2 Paper A **29**
2.1 Introduction 29
2.2 Background and Related Work 31
2.3 Experiment Design 33
2.3.1 Data Preparation 33
2.3.2 Treatments – Independent variable 34
2.3.2.1 Treatment 1: Sequence-based split (ThreeTwoTwo) 35
2.3.2.2 Treatment 2: Sequence-based split (FiveOneOne) 35
2.3.2.3 Treatment 3: Frame-based split (4 frames - 1 frame) 36
2.3.2.4 Treatment 4: Object-based split 36
2.3.2.5 Treatment 5: Similarity-based split 36
2.3.2.6 Treatment 6: Dissimilarity-based split 37
2.3.3 Pipeline 37
2.3.4 Tasks 38
2.3.5 Dependent variable 38
2.3.6 Hypotheses 38
2.3.7 Analysis methods 39
2.3.8 Experimental setup 39
2.4 Evaluation Results 39
2.4.1 Treatment 1: Sequence-based split (ThreeTwoTwo) . . . 40
2.4.2 Treatment 2: Sequence-based split (FiveOneOne) 41
2.4.3 Treatment 3: Frame-based split 41
2.4.4 Treatment 4: Object-based split 42
2.4.5 Treatment 5: Similarity-based split 42
2.4.6 Treatment 6: Dissimilarity-based split 42
2.4.7 Performance Comparison of all treatments 42
2.5 Discussion 44
2.6 Threats to Validity 44
2.7 Conclusion and Future work 46

3 Paper B **47**
3.1 Introduction 47
3.2 Background and Related work 49
3.3 Experiment design 51
3.3.1 Data Selection 51
3.3.2 Data Preparation 51
3.3.3 Independent Variable - Treatments 52
3.3.3.1 **Treatment 0 - Random split** 52
3.3.3.2 **Treatment 1 - Visual similarity-based split** 52
3.3.3.3 **Treatment 2 - Object similarity-based split** 53
3.3.3.4 **Treatment 3 - MajorClass similarity-based split** 53
3.3.3.5 **Treatment 4 - AllClass similarity-based split** 54
3.3.4 Dependent variables 54

3.3.5 Hypotheses . . . 55
3.3.6 Pipeline . . . 55
3.3.7 Task . . . 55
3.3.8 Analysis method . . . 56
3.3.9 Experiment setup . . . 56
3.4 Results . . . 56
3.5 Discussion . . . 58
3.6 Validity Evaluation . . . 60
3.7 Conclusion and future work . . . 60

4 Paper C **63**
4.1 Introduction . . . 63
4.2 Background and related work . . . 64
4.3 Research design . . . 67
4.4 Results . . . 68
4.5 D-LeDe method . . . 70
4.6 Evaluation . . . 72
4.7 Discussion . . . 75
4.8 Threats to validity . . . 76
4.9 Conclusion and future work . . . 78

5 Paper D **79**
5.1 Introduction . . . 79
5.2 Background and Related Work . . . 81
5.3 Methodology . . . 83
5.3.1 The D-LeDe Method . . . 83
5.3.2 Evaluation Datasets . . . 84
5.3.2.1 Zenseact Open Dataset (ZOD) . . . 86
5.3.2.2 ONCE Dataset . . . 86
5.3.2.3 SODA10M Dataset . . . 87
5.3.2.4 BDD100k Dataset . . . 87
5.3.2.5 Proprietary Front-Camera Dataset (Proprietary-FC) . . . 87
5.3.3 Object Detection models . . . 88
5.3.3.1 YOLOv7 . . . 89
5.3.3.2 Faster R-CNN . . . 89
5.3.4 Similarity Analysis with pHash . . . 90
5.3.5 Experimental Setup . . . 91
5.4 Evaluation Results . . . 91
5.5 Discussion . . . 94
5.6 Threats to Validity . . . 96
5.7 Conclusion and Future Scope . . . 97

Bibliography **99**

Chapter 1

Introduction

The automotive industry is undergoing a profound transformation, driven by advances in software and artificial intelligence (AI). Modern vehicles are no longer only mechanical products but complex cyber-physical systems in which software determines safety, efficiency, and user experience [2, 3]. Among the many software-intensive subsystems, perception plays a central role. By processing signals from cameras, Light Detection and Ranging (LiDAR), radar, and other sensors, perception systems create an understanding of the environment by detecting surrounding objects, other road users, etc. This understanding is essential for both advanced driver-assistance systems (ADAS) and autonomous driving (AD) functions to operate safely and correctly [4, 5].

1.1 Background

1.1.1 Automotive Software Systems

A contemporary vehicle can be described as a large-scale cyber-physical system that integrates mechanical components, electronics, and software into a tightly coupled whole. It is estimated that a modern premium car contains over 100 million lines of code distributed across dozens of electronic control units (ECUs) that manage everything from engine timing to driver assistance features [2]. A functional representation of such a system with multiple interconnected software components is shown in Figure 1.1.

Each of these components encapsulates a set of related functions and services that together enable efficient, connected, and autonomous vehicle operation. The main software components are summarized in Table 1.1, which outlines their core responsibilities and representative examples. These domains do not operate independently; rather, they exchange data continuously through in-vehicle networks and cloud interfaces, forming an integrated automotive software ecosystem.

The complexity arises not only from the sheer volume of software but from the heterogeneity and interdependence among these subsystems. Each ECU often operates under different real-time constraints, developed by separate suppliers using distinct hardware and software platforms, yet all must interact seamlessly through in-vehicle networks such as Controller Area Network (CAN)

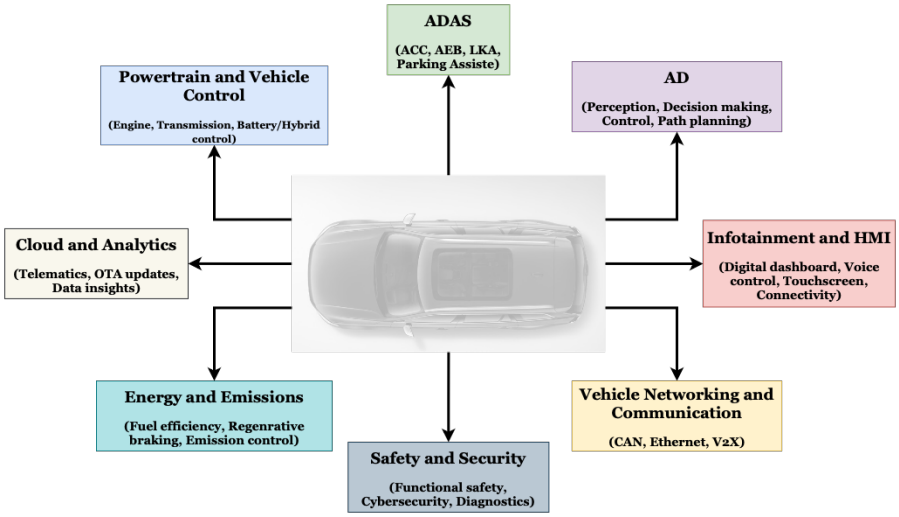

Figure 1.1: A functional view of different software components of a modern automotive software system.

or Ethernet. Changes in one subsystem can have cascading effects on others, making integration, testing, and verification highly challenging. Moreover, as vehicles evolve toward higher levels of automation and connectivity, the interaction between perception, control, and cloud-based services further amplifies this complexity. As a result, software integration and lifecycle management have become as central in automotive research and development as traditional mechanical engineering.

The rise of autonomous and semi-autonomous driving adds further demands on software systems in the vehicle. Perception, planning, and control layers must work together in real time, and failures in any one of them may compromise safety. This is why standards such as ISO 26262 define strict processes for ensuring functional safety in automotive software development [6].

Another ongoing trend is the shift from distributed ECUs toward more centralized computing platforms. Centralization simplifies software management, enabling over-the-air updates and deployment of data-driven functions [7]. Among these, *perception software* plays a crucial role by processing sensor data from cameras, radar, and lidar to detect and classify objects, recognizing lanes and traffic signs, and interpreting the vehicle's surroundings [4, 8].

While artificial intelligence (AI) broadly refers to systems capable of reasoning and decision-making, *machine learning (ML)* — and more specifically, deep learning — forms the foundation of modern perception systems. ML models learn directly from large datasets rather than relying on hand-crafted rules, making them adaptable but also sensitive to data quality and bias. In practice, perception algorithms based on ML are integrated into AI-driven architectures that govern higher-level tasks such as planning and control. This makes perception software a safety-critical component whose reliability depends on both the robustness of its models and the soundness of the data and evaluation methods used to train them [9].

Table 1.1: Short descriptions of different software components in modern automotive software systems.

Component	Description
Autonomous Driving (AD)	Implements perception, prediction, planning, and control functions for higher levels of driving automation (SAE Level 3+), integrating AI-based decision-making and sensor fusion.
Advanced Driver Assistance Systems (ADAS)	Provides partial automation and driver-support features such as adaptive cruise control, lane keeping, collision avoidance, and parking assistance.
Infotainment and HMI	Manages driver and passenger interfaces, including digital dashboards, navigation, voice assistants, and entertainment systems.
Powertrain and Vehicle Control	Governs propulsion, braking, and energy management across internal combustion, hybrid, and electric platforms.
Vehicle Networking and Communication	Enables secure and reliable data exchange between ECUs and with external systems using CAN, Ethernet, and V2X communication technologies.
Safety and Security	Incorporates functional safety mechanisms (ISO 26262) and cybersecurity frameworks (ISO/SAE 21434) to ensure reliable and protected operation.
Energy and Emissions	Focuses on improving efficiency and environmental performance through emission monitoring, regenerative braking, and eco-driving strategies.
Cloud and Analytics	Provides connectivity to backend systems for over-the-air updates, telematics, and data-driven analytics that enable predictive maintenance and fleet optimization.

1.1.2 Automotive Perception Systems

At the core of autonomous driving lies the vehicle's ability to perceive its environment. Perception systems integrate information from different sensors such as cameras, LiDAR, radar, ultrasonic sensors, and transform raw signals into meaningful representations of the world around the vehicle. These systems are responsible for detecting other road users, recognizing traffic signs and lane markings, and estimating free drivable space. In practice, perception acts as the first stage in the decision-making pipeline, providing the necessary input for planning and control software [10].

In practice, the perception process in modern vehicles follows a logical workflow, as illustrated in Figure 1.2. Sensors first capture data from the surrounding environment, producing continuous streams of camera images, LiDAR point clouds, or radar echoes. These heterogeneous data are then processed by perception algorithms that detect and track dynamic objects, recognize traffic signs, and estimate lane and road geometry. The output of this stage is a structured and semantically rich representation of the driving

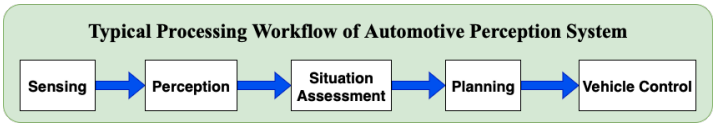

Figure 1.2: A simplified illustration of how the automotive perception system works.

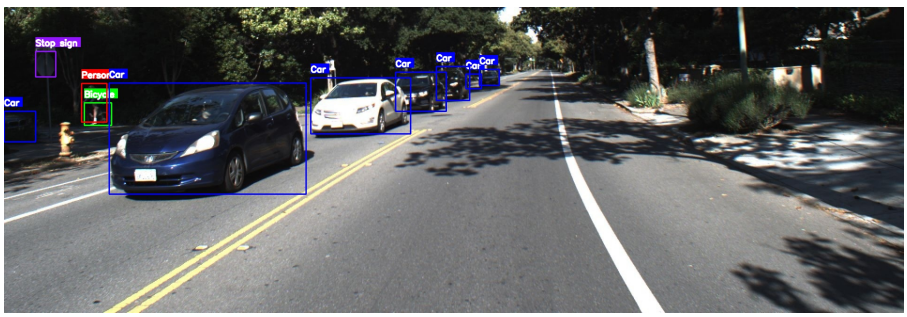

Figure 1.3: An example of Object Detection with the bounding boxes identified by the perception system. The image shows the detected cars, a person with a bicycle, and a stop sign, identified in bounding boxes with different colours.

scene, which abstracts the complexity of raw sensor data. This situational understanding is then passed to the planning and vehicle control modules, enabling context-aware decision-making and motion execution [11]. Common perception tasks include object detection and tracking, semantic segmentation, road and lane detection, free-space and occupancy estimation, and localization and mapping [12–14].

1.1.3 Object Detection in Automotive Systems

Object detection is one of the core tasks in Computer Vision (CV) and plays a central role in automotive perception. The goal is to locate and classify objects of interest, such as vehicles, pedestrians, and traffic signs, within an image (see example in Figure 1.3). Early approaches relied on hand-crafted visual features combined with traditional machine learning classifiers, such as Histogram of Oriented Gradients (HOG) or Scale-Invariant Feature Transform (SIFT), combined with classical machine learning classifiers. The Deformable Parts Model (DPM) [15] was one of the most influential of these early methods — it represented objects as a collection of parts with geometric relationships between them, trained using support vector machines. Although such models worked well on constrained datasets, they struggled to generalize to the complex and dynamic conditions encountered in real driving environments, such as varying illumination, motion blur, occlusions, weather effects, and viewpoint changes [16–18]. These limitations later motivated the shift toward deep learning-based methods, which could learn both features and object representations directly from data rather than relying on manual feature engineering.

The field changed dramatically with the introduction of deep convolutional neural networks (CNNs), first shown on a large scale in the ImageNet challenge [19]. Instead of designing features by hand, CNNs learn rich visual representations directly from pixels, enabling much higher robustness and accuracy. In the context of object detection, two main families of deep CNN models became dominant. Two-stage detectors, such as Faster R-CNN [20], separate the process into two steps: first generating region proposals that likely contain objects, and then classifying each region. Single-stage detectors, such as YOLO [21], perform both tasks in a single pass over the image, directly predicting bounding boxes and object labels. This design makes them faster and better suited for real-time applications like autonomous driving, while two-stage detectors often achieve slightly higher performance scores [22] in more controlled settings. However, they are more computationally expensive, which is not always feasible for real-time automotive systems. Single-stage detectors, in contrast, are faster and lighter, which makes them suitable for scenarios where speed is critical. Modern versions, such as YOLOv7, manage to combine good accuracy with the low latency needed for deployment in cars.

In a perception workflow, the outputs of object detection are rarely the endpoint. The bounding boxes and classes are passed to tracking and motion estimation modules, which then provide input to planning and control. This means that errors at the detection stage - missed objects, false alarms, or poor localization - can propagate through the entire system and affect safety. For this reason, evaluating detection models under realistic conditions and with careful handling of training and test data is an essential step for building trustworthy perception systems.

1.1.4 Automotive Image Datasets

In machine learning, a *dataset* refers to a structured collection of data samples that are used to train, validate, and evaluate models [23]. For computer vision tasks, datasets typically consist of images or video frames, often with annotations such as bounding boxes, segmentation masks, or class labels. The choice of dataset distribution has a strong influence on the performance and generalization ability of a model, since it defines what the model is exposed to during training and how it is later judged in evaluation.

In the automotive domain, image datasets are the foundation of a perception system. They provide large-scale, annotated collections of driving scenes that capture the complexity of real traffic. Some of the well-known examples include KITTI [16], Cityscapes [17], BDD100K [24], and nuScenes [8]. A detailed list of a few of the major datasets is presented in Table 1.2 with the release year, number of citations [1], driving scenarios covered, and their main uses. These datasets contain diverse conditions such as urban and rural roads, day and night driving, and variations in weather. Their annotations cover objects of interest to driving, including vehicles, pedestrians, cyclists, traffic lights, and road signs. The diversity and quality of these datasets are critical, since perception models trained on narrow or biased data may fail in unseen real-world situations.

A unique property of automotive image datasets is that they often consist of *sequences of frames* captured by cameras mounted on vehicles. This introduces

[1]The citation count is reported from Google Scholar, accessed on September 23, 2025

Table 1.2: Summary of major autonomous driving image datasets with release years, citation counts, driving scenarios, and main tasks.

Dataset	**Year**	**Citations**	**Driving Scenarios**	**Tasks**
KITTI [25]	2012	12020	Urban, rural, and highway driving scenarios	2D/3D detection, stereo, tracking, odometry
Oxford Robot-Car [26]	2015	1962	Driving scenarios from different weather, seasons, and times of day	Localization, mapping, visual odometry
Cityscapes [17]	2016	513	City driving scenarios in daytime and diverse traffic conditions	Semantic segmentation, detection
ApolloScape [27]	2018	1361	Urban driving scenarios	Segmentation, lane, 3D car instances
BDD100K [24]	2018	3196	Highways and urban driving scenarios, combination of both day and night driving and various weather conditions	Detection, segmentation, tracking, drivable area
nuScenes [8]	2019	8347	Urban driving scenarios	3D detection, tracking, segmentation
Argoverse [28]	2019	1908	Urban driving scenarios	Motion forecasting, 3D tracking
Waymo Open Dataset [18]	2020	4205	Urban, sub-urban, highway driving scenarios with varied weather condition	2D/3D detection, tracking, motion forecasting
SODA10M [29]	2021	96	Varied driving scenarios	2D/3D Object detection, semi/self-supervised
ONCE [30]	2021	378	Diverse city driving scenarios from both day and night, as well as in different weather conditions	2D/3D object detection
Argoverse v2 [31]	2023	906	Driving scenarios from multiple cities	3D detection, motion forecasting
Cirrus [32]	2021	82	Highway and urban road driving scenarios	2D/3D object detection, LiDAR model adaptation
Zenseact Open Dataset (ZOD) [33]	2023	100	Highways, city, and rural driving scenarios in diverse weather conditions	2D/3D detection, segmentation, road classification

both opportunities and challenges. On the one hand, sequential data enables tasks such as tracking or motion prediction. On the other hand, consecutive frames are often very similar, which means that it becomes an important issue how the dataset is split into training and test sets.

Automotive datasets also differ in scale and annotation detail. For example, the KITTI dataset contains around 8,000 annotated images. It has been highly influential due to its early introduction and well-structured benchmarks for key tasks such as object detection and depth estimation. In contrast, more recent datasets like BDD100K or SODA10M are larger in volume and provide more diverse annotations of different object categories, lane marking, and drivable area segmentation, reflecting the real-world driving complexities. Proprietary datasets collected by industry further extend this landscape by focusing on region-specific scenarios or edge cases, such as rare traffic situations or unique environmental conditions not well represented in public datasets.

1.1.5 Data Splitting and Its Role in Model Evaluation

Supervised learning depends on a basic but crucial step: splitting or partitioning data into training, validation, and test sets [34], as shown in Figure 1.4. The idea is simple — teach the model on one subset, tune it on another, and report final performance on a held-out portion, which is meant as an estimate of the expected real-world performance — but the apparent simplicity hides an important assumption: that the samples used for training and evaluation are suitably independent and representative of the same underlying problem. In many computer vision applications, this assumption is weak [35], and in automotive image data, it is often violated because images are collected as sequences, from repeated routes, or under similar environmental conditions. When that happens, the meaning of a reported test score becomes ambiguous and does not clearly indicate whether the model is genuinely generalizing, or it is benefiting from accidental similarity between the partitions [36].

Different data splitting strategies make different assumptions about independence and about what kind of generalization is being tested. A few common paradigms are worth naming and distinguishing here. A *random split* [37] places images into partitions at random; it is simple and often used as a baseline, but it implicitly assumes sample independence. A *temporal* or sequence-based split [38] separates data along time (earlier frames for training, later for testing), which better reflects deployment where models are trained on past data and evaluated on future observations. A *geographic* or location-based split [39] divides data by capture location to probe spatial generalization. *Stratified* splits [40] preserve class or condition distributions across partitions (for example, balancing day/night or weather conditions), addressing class-imbalance concerns but not necessarily removing visual redundancy. Finally, *similarity-based* splits [41] group images by visual or contextual similarity and then partitions those groups; this approach is explicitly designed to avoid near-duplicate overlap between train and test sets. Each choice embodies a different view of what "unseen data" should mean in evaluation.

Object detection adds its own complications. Images are not independent collections of single labels — they contain multiple instances, overlapping objects, and spatio-temporal correlations. Frames from the same driving

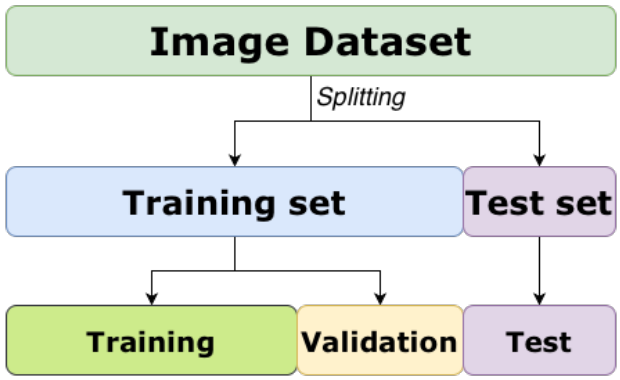

Figure 1.4: Illustration of data splitting. The image dataset is first partitioned into two subsets: training and test. The training set can be further divided into training, to train/teach the model, and validation, to tune hyperparameters. The test set is held out to test the model afterwards and report the final performance.

sequence may show the same vehicle from slightly different viewpoints or the same scene under subtly different lighting; these near-duplicates are particularly problematic because they can inflate the measured performance of detectors while hiding their true robustness to new scenes, occlusions, or rare object classes. In short, the splitting scheme determines the type of generalization being measured: spatial transfer, temporal robustness, class-wise balance, or resistance to near-duplicate memorization. Misalignment between the chosen split and the intended evaluation goal, therefore, leads to misinterpreted results. For these reasons, dataset splitting is a methodological decision that shapes conclusions.

1.1.6 Image Similarity

As noted above, the way a dataset is partitioned only makes sense once we agree on what it means for images to be 'different' or 'the same'. That is the role of *image similarity*, which defines whether two images should be treated as distinct or the same observation seen twice. Figure 1.5 illustrates a typical case from an automotive dataset, where two consecutive frames from a driving sequence appear near-identical to the human eye. Treating one of these frames as training data and the other as a held-out test sample can lead to misleadingly high performance scores, since the model is tested on data it has already "seen" before.

To be precise, practitioners typically talk about a few related concepts. Some of these concepts are adopted and used in this thesis. Near-duplicate or near-identical images are those that differ only by small visual changes (adjacent video frames, tiny viewpoint shifts, minor compression or lighting variation). Structural similarity between images means a shared global layout or shared textural patterns between images, even if pixel values differ (for example, the same scene under different exposure). Such measures attempt

(a) (b)

Figure 1.5: Example of a similar image pair where images 1.5a and 1.5b look near-identical. Both images are taken from the KITTI dataset [25].

to capture similarity beyond raw pixel values. Another concept, semantic similarity of images, means that the images share the same high-level content (for example, both contain a car and a pedestrian) but differ in appearance, background, or context.

These intuitive distinctions map to different measurement approaches, each with strengths and limitations. At the simplest level are *pixel-based* measures, which are easy to compute, but small lighting changes or minor camera motion can produce large differences in pixel values even when images look similar. A widely used alternative is the Structural Similarity Index (SSIM), which compares local luminance, contrast, and structure and aligns better with human perceptual judgements [42]. SSIM can be useful to detect structural changes, but it is relatively sensitive to larger viewpoint changes and does not capture semantic similarity. Perceptual hashing (for example, pHash [43]), on the other hand, reduces an image to a compact fingerprint such that near-duplicates have a small Hamming distance between their hashes. pHash is commonly used for large-scale near-duplicate detection since the calculation process is fast and easy to scale.

At a higher, semantic level, similarity can be measured using feature embeddings obtained from deep convolutional networks (pretrained classifiers or self-supervised models). These embeddings place images in a space where proximity tends to reflect shared semantic content rather than pixel-level likeness [44, 45]. Metrics built on such embeddings (for instance, cosine similarity or learned perceptual metrics like Learned Perceptual Image Patch Similarity (LPIPS)) capture more abstract similarity but come at a comparatively higher computational cost and require careful selection of which layer and model to use.

Which measure is most relevant depends on the specific leakage risk one wants to capture. For the thesis work, the central concern is *near-duplicate overlap*: cases where a model effectively sees the same scene in training and testing. For this reason, a lightweight and scalable near-duplicate detector such as pHash is a practical first line of analysis. That choice is pragmatic since pHash is efficient for screening millions of image pairs and produces an interpretable Hamming-distance score that can be thresholded to flag pairs above a certain level of similarity. pHash thresholds must be decided according to the dataset and the particular application. However, pHash is not sensitive to semantic similarity, which limits its applicability. Complementary methods such as SSIM or embedding-based distances can be useful to help characterize borderline cases and could be used in the future.

1.1.7 Data Leakage in Image Datasets

Data leakage is a well-known challenge in machine learning [46], but its impact is often underestimated, particularly in computer vision. In simple terms, leakage occurs whenever information that should be unavailable at prediction time is, directly or indirectly, used during model training or validation [47, 48]. The consequence is always the same — evaluation metrics shows optimistic results and appears to be a reliable guide to real-world performance. Data leakage can take several technical forms — for example, features that encode the target (target leakage), inadvertent reuse of test examples during training (train–test contamination), or preprocessing steps that inadvertently expose information from test sets into the training procedure — but the unifying theme is that the development pipeline has been contaminated by information that should remain hidden.

One of the most frequent sources of leakage in computer vision tasks is the presence of near-duplicate or overlapping samples in both training and test sets. For instance, consecutive frames extracted from a driving sequence may show near-identical scenes, with only minor changes in object position. If such frames are distributed across the split, the model can achieve high test accuracy by memorizing visual patterns rather than learning transferable features [49]. This is particularly critical in safety-related applications, where overestimated performance may give a false impression of robustness.

Another common form of leakage arises when metadata or dataset structure unintentionally carries information that the model can exploit. For example, filenames or directory structures may implicitly encode class labels, and if not handled carefully, such artifacts can leak into the training process [47]. In the automotive context these risks are amplified because of how data are collected: continuous video from fixed rigs produces strong temporal and contextual correlations, and large industrial collections may emphasize particular routes or conditions. For these reasons, understanding the different types of leakage in image data and being able to characterize which one is present in a given dataset is a prerequisite for a credible evaluation of perception systems.

This thesis focuses on understanding and addressing data leakage in the image datasets for automotive object detection systems. In the next section, the research design is presented. It describes the goals, questions, and methods that guided the four studies included in this work, and explains how each study contributes to addressing the overall problem.

1.2 Research Design

The research design specifies the research goals (RGs) and research questions (RQs) that frame the investigation as well as the scope and direction of the thesis. Each goal targets a particular aspect of the overall problem of data leakage in automotive object detection systems, while the associated questions provide more concrete paths of inquiry.

1.2.1 Research Goals and Questions

The research is structured into four complementary goals. Each goal addresses a different aspect of the problem — from understanding how dataset splits influence model performance, to developing methods that detect data leakage in practice. Together, they form a progression from foundational analysis to applied methodological contribution. We derive the following research goals and research questions:

RG1: Identify data splitting techniques/strategies for image datasets and their impact on the performance of an automotive object detection (OD) model.

Rationale: Different splitting strategies may yield substantially different evaluation outcomes even when using the same dataset and model. Understanding this variability is essential to ensure that observed performance differences reflect model capability rather than experimental design.

- **RQ 1** Is there a statistically significant performance difference of the object detection model for different data splitting techniques being applied?

RG2: Assess image similarity from both visual and semantic perspectives, and use them for data splitting to improve object detection performance of automotive perception systems.

Rationale: Image similarity defines how "different" two samples are. By examining both visual and semantic similarity, this goal aims to identify measurable criteria that can guide more meaningful dataset splitting and help avoid hidden data overlap.

- **RQ 2.1** How to measure image similarity for object detection tasks in the automotive context?
- **RQ 2.2** What semantic information from images can be used to split automotive image datasets to improve object detection model performance?

RG3: Establish a method to detect data leakage in existing data splits.

Rationale: Quantifying how incremental leakage alters model performance metrics allows us to understand its real-world impact and to design diagnostic indicators for its detection. This knowledge is key to building reliable evaluation pipelines.

- **RQ 3.1** How does incrementally introduced data leakage impact the overall model performance?
- **RQ 3.2** How can the model performance increase rate be used to identify data leakage in automotive image dataset?

RG4: Detect data leakage in existing data splits of automotive image datasets to improve the precision of automotive object detection.

Rationale: Evaluating a few of the current public and one proprietary datasets for leakage in their default split demonstrates the practical utility

of the proposed approach. Consistent detection across diverse models and datasets would indicate that the method generalizes and can improve future dataset design.

- **RQ 4.1** Which dataset(s) among the chosen ones have data leakage in their default splits?
- **RQ 4.2** How consistently does the method detect leakage across diverse automotive datasets and different OD model architectures?

Figure 1.6 maps the research goals and research questions to the appended papers. Each paper focuses on achieving separate research goals and relevant research question(s) related to that specific goal were answered accordingly.

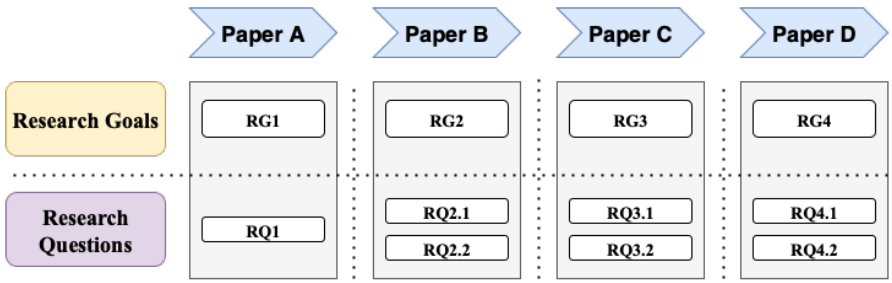

Figure 1.6: Mapping of the research goals and research questions of the thesis to the appended papers. Each paper investigates a RG by answering the corresponding RQ(s).

1.2.2 Research Methodology

The credibility of any research depends not only on the results obtained but also on the transparency of the process through which they were reached. In empirical software engineering research, methodological choices play a central role in shaping the validity of findings. This thesis adopts the research decision-making structure proposed by Wohlin and Aurum [50] to make clear how the research goals were addressed and how the four appended papers relate to one another.

1.2.2.1 Research Approach

The thesis builds on several methodological references that have shaped empirical software engineering research. Among these, the ACM SIGSOFT *empirical standards* for software engineering research [51] highlight the need for clarity in research questions, transparency in reporting, and careful justification of design choices. These principles are reflected in how the thesis makes its research questions explicit, documents experimental setups in detail, and acknowledges known limitations such as dataset imbalance or the reliance on certain model architectures.

Wohlin and Runeson's guidance on *industry–academia collaboration* [52] is also relevant to the problem studied here — data leakage in automotive

perception — which may have a direct impact on industrial practice. One of the seven datasets used in this research is proprietary (called Proprietary_FC in Paper D) and stems from ongoing industrial collaborations, while benchmark datasets such as KITTI and BDD100k are included to ensure broader reproducibility. This balance between industrial context and public validation reflects the aim of both industrial relevance and academic rigor.

Easterbrook et al. [53] emphasize that methodological choices should be tailored to the maturity of the research question. This view is visible in the structure of the thesis: early studies (Papers A and B) are exploratory and experimental, aimed at understanding the scope of the problem, while later studies (Papers C and D) adopt a design science orientation to develop and evaluate a concrete solution (the D-LeDe method). In this way, the methodology evolves in step with the research questions.

While these different perspectives offer valuable guidance, the primary organizing framework for this thesis is Wohlin and Aurum's decision-making structure [50]. This model organizes research design into three levels — strategic, tactical, and operational, and provides a systematic way to connect high-level research aims with concrete methodological choices. The following subsection applies this structure to describe the specific research methods of the thesis.

1.2.2.2 Research Methods

Following the decision-making structure proposed by Wohlin and Aurum [50], the research design of this thesis can be described at three levels: strategy, tactics, and operations. Each level represents a different layer of methodological choices, ranging from the overall orientation of the work to the detailed procedures followed in individual studies. Figure 1.7 provides a compact overview, and the following paragraphs elaborate on how the four appended papers connect to each level.

Strategy Phase At the strategic level, the research in this thesis is positioned as applied, positivist, and largely explanatory in nature. Applied research means that the work is motivated by a practical problem — in this case, the risk of incorrect performance assessment due to data leakage in automotive object detection — and aims to provide both understanding and actionable solutions. A positivist stance underpins all four papers: knowledge is sought through systematic observation, measurement, and analysis of empirical data [53]. Papers A, B, and C follow a deductive logic, where hypotheses about data leakage and its effects are tested in controlled settings, while Paper D adopts an inductive stance in its evaluation of the proposed method across diverse datasets and model architectures. The research purposes also vary across papers: Papers A–C are explanatory, focusing on uncovering causal relationships between data similarity, dataset splits, and model performance; Paper D shifts toward evaluation, assessing how well the proposed solution performs in practice. Together, these choices reflect a coherent strategy that moves from identifying and explaining a phenomenon to developing and validating a solution, in line with the guidance that research design should evolve with the research questions rather than remain fixed [50–52].

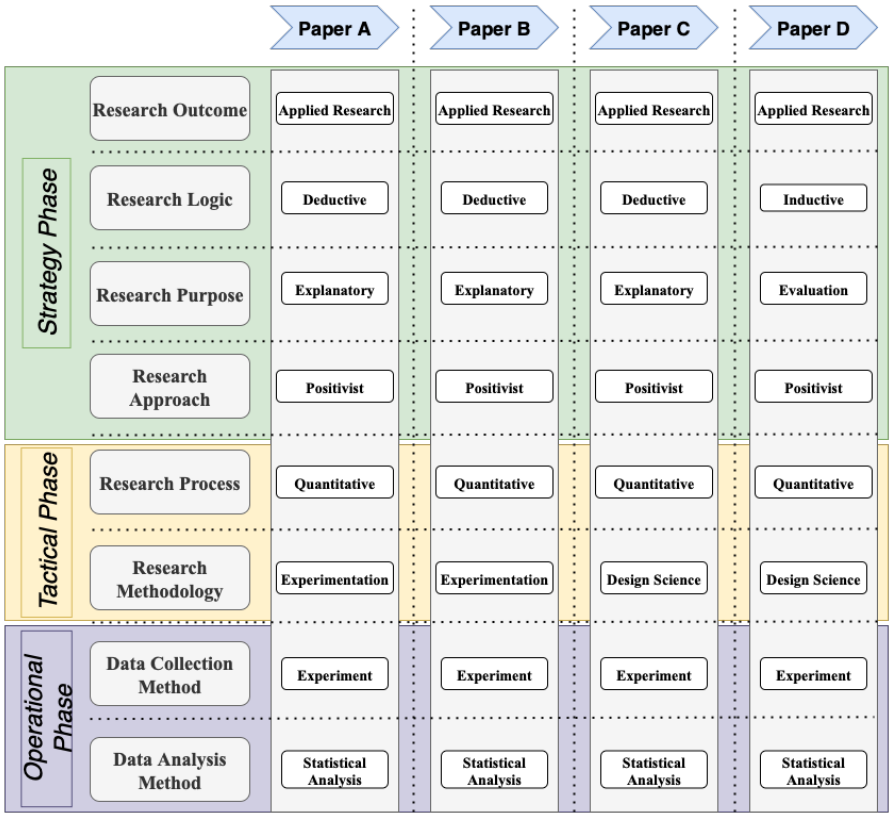

Figure 1.7: Research design of the appended papers, according to the decision-making structure proposed by Wohlin and Aurum [50].

Tactical Phase At the tactical level, the research is structured by deciding on which research processes (quantitative or qualitative) and methodologies (experimentation or design science) were selected that support the overall aims of the research. Across all papers, the process is quantitative, with an emphasis on objective measurement and statistical evaluation. Papers A and B employ experimentation: controlled computational studies in which dataset splitting strategies and similarity measures are manipulated while object detection performance is observed. Such experiments make it possible to draw causal inferences about the impact of dataset design choices [54]. In contrast, Papers C and D adopt a design science methodology. Here, the focus shifts to the development and validation of a concrete artifact, the D-LeDe method, intended to detect data leakage in practice. Following design-science principles, Paper C introduces the method and demonstrates its feasibility, while Paper D extends the evaluation to broader and more realistic conditions. This combination of experimentation and design science provides complementary perspectives: controlled studies to build causal understanding, and artifact-based studies to offer practical solutions.

Operational Phase Finally, at the operational level, the design translates into the specific instruments and procedures used. The studies employ well-known object detectors (YOLOv7 and Faster R-CNN) to ensure that results are not tied to a single family of models. Additionally, both public benchmarks (KITTI, BDD100K, ONCE, SODA10M, ZOD) and a proprietary Original Equipment Manufacturer (OEM) dataset (called as Proprietary_FC dataset in the paper) are used to balance reproducibility with industrial relevance. Image similarity is operationalised through perceptual hashing (pHash), which provides a computationally efficient way of identifying near-duplicate images, complemented where necessary by sanity checks and alternative measures. Evaluation relies on established metrics — mAP for precision/recall trade-offs and F1-score for balanced performance — and results are reported as averages over multiple runs to reduce noise from random initialisation [55, 56]. To increase reproducibility, experiments were run under controlled and documented conditions, with fixed preprocessing, consistent hyperparameters, and version-controlled code and data splits. Where proprietary data could not be shared, transparent reporting of procedures and complementary experiments on public datasets were used to mitigate this limitation of reproducibility. In this way, the operational layer ties the overall strategy and tactics to a set of concrete, transparent practices that ensure validity and comparability across the four papers.

1.3 Related Work

The section 1.1 outlined the conceptual background of data splitting, image similarity, and their relevance to object detection in automotive perception. Building on that foundation, this section reviews existing research addressing these challenges from a methodological perspective particularly focused on the works directly related to this thesis: (i) different data splitting strategies used in practice in different research fields, (ii) similarity measures proposed for identifying visual and semantic relationships between images, (iii) techniques that use such measures to construct or refine dataset splits, and (iv) implications of data leakage detection and mitigation strategies in practice. Together, these provide the methodological context and highlight the gaps that motivate the approaches developed in this thesis.

1.3.1 Data Splitting Strategies

Reliable evaluation of machine learning models critically depends on how data are divided into training and testing sets. In the context of automotive perception, improper data splitting can easily lead to information leakage [57], inflating model performance and providing a misleading sense of reliability. As detailed below, the methodological implications of data splitting have been increasingly recognized across multiple domains.

Previous empirical studies have demonstrated that the choice of splitting strategy can substantially influence model performance. For instance, Lyu et al. [58] examined the impact of different splitting approaches on AI for Information Technology (IT) operations (AIOps) and found that random splits consistently yielded significantly higher performance metrics than time-based

or baseline splits (Wilcoxon test, $p < 0.05$). The authors attributed this to unintended data leakage across partitions, concluding that time-based splitting better preserves temporal independence between training and evaluation sets. However, their findings were limited to prediction tasks with tabular data, making the direct transferability to image-based object detection uncertain.

Similarly, Meng et al. [59] evaluated several user-based and transaction-based splitting strategies for recommender systems, observing strong variability in recall depending on the chosen approach and dataset. Temporal global splitting was identified as the most realistic, underscoring again that evaluation bias can stem from how datasets are partitioned, even when the model and data remain constant. While informative, these results stem from recommendation and forecasting tasks, where sequential or transactional continuity of data points are used rather than spatial or semantic relationships between samples.

To address imbalance and small-sample issues in classification, Doan et al. [60] proposed a within-class stratified clustering (WICS) method combined with Synthetic Minority Oversampling Technique for nominal and categorical data (SMOTE-NC) sampling. This approach improved classifier stability on a missile impact damage dataset, achieving an F1-score of 0.821 and AUC of 0.949. Nevertheless, its applicability to complex multi-class scenes such as traffic scenario images is limited, as multiple object types may appear in a single frame - for example, an image may contain vehicles, pedestrians, and other potential road users. Such object variability with spatial correlations cannot be easily stratified or oversampled.

Earlier methodological work by Wu et al. [61] systematically compared splitting techniques on hydrological forecasting datasets and demonstrated that each method introduces distinct bias and variance characteristics. The DUPLEX method [62], which assigns samples based on Euclidean distance to maximize coverage, was shown to reduce bias but produce more conservative estimates. While promising for regression contexts, its suitability for high-dimensional visual data remains unexplored.

In the domain of autonomous driving, widely used benchmark datasets such as KITTI [25] and nuScenes [8] are typically released with predefined train–test splits. However, the underlying rationale or similarity constraints guiding these splits are rarely documented. As these datasets were collected from various locations and reflect different driving conditions, their default partitions may not accurately represent the diversity and correlation structure required to develop modern perception systems.

1.3.2 Image-similarity measures

A range of image-similarity measures has been proposed and used for tasks such as deduplication, image retrieval, and examining dataset hygiene. These measures differ by the level of visual abstraction they capture and by their computational cost; both aspects matter when applying them to large automotive data collections.

Pixel-level and structural-level measures are the simplest types. Pixel-wise distances and signal-based indices such as mean squared error (MSE) or peak signal-to-noise ratio (PSNR) are straightforward to compute but brittle to illumination, viewpoint, and compression changes. The Structural Similarity

Index (SSIM) was introduced to address perceptual aspects of image quality by comparing local luminance, contrast, and structure; SSIM often aligns better with human judgements than raw pixel metrics and is useful for detecting structural changes between near-identical frames [42].

Perceptual hashing methods (for example, pHash) create compact fingerprints intended to map visually near-identical images to similar binary strings; comparison is done by calculating the Hamming distance between two such strings [43]. Hashing is attractive for large-scale screening because it is fast, storage-efficient, and provides interpretable thresholds for near-duplicate detection. Its only weakness is semantic insensitivity: pHash may flag very similar low-level patterns but misses deeper semantic similarity (different viewpoints or severe appearance changes) which might still produce leakage in tasks sensitive to object identity.

Embedding-based and learned perceptual measures operate at a higher semantic level. Features from pretrained convolutional networks (e.g., ResNet) or learned perceptual metrics (LPIPS) place images in continuous spaces where proximity tends to reflect object-level or scene-level similarity [44, 45]. More recently, large-scale multimodal embeddings (e.g., Contrastive Language-Image Pre-Training, CLIP) have shown strong ability to capture semantic likenesses across diverse imagery [63]. Embedding-based distances are powerful for grouping semantically-related images, but they are computationally heavier and sensitive to the choice of backbone and embedding layer.

When selecting a similarity measure for dataset assessment or splitting, two practical trade-offs dominate: (1) whether the goal is to group near-duplicates (favor hashing methods) or to capture semantic closeness (favor deep embeddings), and (2) scalability and interpretability (favor hashing or low-dimensional embeddings). For automotive OD, where both near-duplicate frames and semantic repeats (same scene under different weather) matter, a hybrid approach is commonly recommended in the literature: use hashing for fast screening and embeddings for deeper verification [43, 45].

1.3.3 Similarity-based splitting and deduplication methods

Several practical methods have been proposed to exploit image-similarity information when splitting datasets or removing redundancy. The literature can be grouped into (A) explicit deduplication / near-duplicate detection, (B) similarity-based clustering for split construction, and (C) embedding-driven split strategies.

(A) *Near-duplicate detection* techniques typically apply hashing or locality-sensitive hashing to identify near-identical frames and remove or group them. These methods are well-established in large-scale image pipelines because they are inexpensive and effective at eliminating trivial redundancy [43]. Papers applying such techniques report immediate reductions in artificial performance inflation caused by frame-level overlap in datasets with strong temporal redundancy [64].

(B) *Similarity-based clustering* uses pairwise similarity (via hashing or embeddings) to cluster visually related images and then assigns whole clusters to partitions. This prevents clusters' members from data leakage across train/test

boundaries. Approaches differ in clustering algorithm (k-means on embeddings, agglomerative clustering, or graph-based connected components using a thresholded similarity matrix) and in how they control balance across classes and conditions [41]. The main challenge here is maintaining representative splits: clustering can create partitions with different class distributions or environmental coverage, which complicates fair evaluation.

(C) *Embedding-driven split strategies* directly exploit neural embeddings (from supervised or self-supervised models) to define similarity. These methods can capture semantic similarity more reliably than hashing, and they have been used both for dataset curation and for constructing more rigorous data splits [45, 63]. The literature highlights two recurring trade-offs: embedding approaches better capture semantic similarity but require more compute time and careful calibration; clustering on embeddings helps avoid near-duplicate leakages but must be paired with balancing strategies to maintain experimental validity (for example, equal distribution of class instances).

Across these strands, most prior work focuses on classification [65, 66] or retrieval tasks [67, 68] rather than multi-instance object detection. Object detection brings extra complications — images contain multiple annotated instances with spatial relations — so, splitting strategies must consider instance-level duplication (same object appearing in multiple frames) as well as image-level similarity. This nuance is often missing in generic deduplication pipelines and motivates detection-specific methods such as instance-aware clustering or track-aware deduplication, which are discussed only sparsely in the literature [69–71].

1.3.4 Data Leakage Detection and Mitigation

Data leakage has long been recognized as a persistent problem in machine learning and deep learning [46], especially for computer vision tasks where training and test samples may look alike [72]. Leakage may happen in several ways. Sometimes the same image, or one captured only a few seconds apart, ends up in both training and testing sets. At other times, preprocessing operations or shared statistical information cause hidden information flow between data splits [73, 74]. Even when it is believed that the partitioning is correct, small correlations in the background, lighting, or viewpoint can leak information. Earlier studies have shown that such leakage can inflate accuracy or F1-scores by as much as 20–50%, depending on dataset size and split procedure [75, 76]. These findings emphasize that careful data management is as important as model design. Methods for detecting and reducing data leakage can be divided into data-centric and model-centric approaches. Data-centric checks work directly on the dataset before training. Perceptual-hashing (pHash) and structural-similarity (SSIM) measures are popular first steps because they are simple and fast to compute [42, 43]. They help to find near-duplicates or consecutive frames with minimal visual differences. However, they focus on perceptual similarity and may fail to capture deeper semantic overlaps such as the same object seen from different angles or under different lighting.

To go beyond these limits, embedding-based similarity search has become common. Pretrained networks such as Residual Networks (ResNet) or CLIP map images into vector spaces where similar scenes stay close together [44, 63].

Using approximate nearest-neighbour libraries like Facebook AI Similarity Search (FAISS) or the hierarchical navigable small world library (hnswlib), developers can quickly find test images that resemble those in training [77, 78]. These approaches detect more subtle forms of leakage but depend on how embeddings are extracted and how thresholds are set, which can vary from one dataset to another.

For object detection tasks, where the same object may appear in many frames, track-based methods may help to reduce redundancy. Algorithms such as Simple Online and Realtime Tracking (SORT), DeepSORT, or multi-object tracking using ByteTrack link bounding boxes across frames so that dataset splits can be made at the object level rather than at the image level [79–81]. This reduces the risk that the same car or pedestrian appears in both training and testing.

The success of such tracking, however, depends on consistent annotations and stable detection across frames. Model-centric approaches look at the learning process itself. Influence-function analysis can estimate how much each training example affects a test prediction [82]. Training-dynamics studies track the loss or confidence of each sample during training to find cases that the model learns unusually easily, which might indicate duplicated or leaked data [83]. These methods are slower to run and cannot confidently denote the presence of data leakage straight away.

Despite the development, major gaps remain in data leakage detection for automotive datasets. Hashing and SSIM cannot identify semantic or instance-level similarity, embedding methods require a specific choice of pretrained models and high computation time, and tracking can fail under scenarios with occlusion or inconsistent labeling of objects. Moreover, few studies measure how detected similarity actually relates to evaluation inflation in object detection models. Developing a practical and general method that can identify the presence of data leakage across diverse automotive datasets and their existing splits remains an open problem.

1.4 Summary of Findings

This section summarizes the findings of the appended papers with respect to the research goals and research questions formulated within the thesis.

1.4.1 Impact of Data Splitting on Model Evaluation (RG1)

The first research goal (RG1) was to investigate how the choice of data splitting strategies influences the performance of object detection models in automotive perception.

To address this, Paper A examined six splitting techniques using a modern dataset of driving sequences called Cirrus (details in Table 1.2), focusing on how these strategies affected the training and evaluation of a YOLOv7 object detector model. Cirrus is one of the most recent datasets containing high-quality images separated as sequences from different driving scenarios into folders. The uniqueness of Cirrus also adds novelty to this research. The study revealed statistically significant performance differences across the splitting methods chosen: for example, similarity-based splitting, where the images were

grouped by pairwise similarities, leads to a mean average precision (mAP) of 0.943, while frame-based splitting, where consecutive frames spread over train and test sets, leads to 0.841. The object-based splitting, which separated images of the same object to form the train set and all other images were put into the test set, lead to the worst performance with an mAP of just 0.118.

These findings demonstrated that the choice of splitting strategy is a central factor shaping the apparent performance of perception systems. In particular, they highlighted the risk that commonly used approaches may incorrectly inflate model performance depending on how we split the images into training and test sets. This insight provided the foundation for RG2, which further examined whether similarity-aware methods could provide a more reliable way of constructing splits.

Answer to RQ1: Choice of data splitting strategies have significant impact on the evaluation outcomes of the model because they alter the degree of visual and semantic overlap between training and test data.

1.4.2 Effectiveness of Similarity-based Splitting (RG2)

The second research goal (RG2) focused on assessing image similarity from both visual and semantic perspectives and exploring how these measures can be applied to construct more meaningful dataset splits for object detection in automotive perception. Building on the findings of Paper A, which showed that hidden similarities between images can distort evaluation results, Paper B aimed to identify which aspects of similarity, pixel-level visual resemblance or higher-level semantic context, are most relevant for achieving robust model evaluation.

In Paper B, four different measures of similarity were explored: one based on visual features and three based on semantic information available in the dataset. Visual similarity captures low-level correlations between images, such as similar lighting, textures, or viewpoints, but it proved insufficient to fully account for overlaps that affect model generalization. In contrast, semantic similarity, defined by object-level attributes such as class distribution, object size, position, and orientation, provided a more discriminative basis for dataset partitioning.

Answer to RQ2.1: Image similarity can be measured in two ways, visual and semantic. Visual similarities are calculated from pixel representations, while semantic similarity can be measured using intrinsic and contextual information from the images.

The results showed that semantic similarity-based splits generally produced higher and more stable performance scores than purely visual approaches. These results indicate that incorporating semantic understanding of scenes allows for constructing splits that better reflect real-world variation and lead to more reliable performance evaluation. In particular, the “AllClass” similarity measure, where the intrinsic information of all instances in an image constitutes the similarity vector, led to improvements in mean average precision (mAP)

compared to visual similarity-based split by 47%. These findings suggest that taking semantic context into account helps not only to avoid the leakage that arises when similar images are placed across training and test sets but also to improve overall model performance keeping the train and test data as identical as possible.

These findings demonstrated that not all types of similarity contribute equally to have fair model evaluation. Semantic measures, which consider contextual information such as object counts, sizes, and positions, proved more effective for constructing meaningful dataset splits than purely visual comparisons.

Answer to RQ2.2: Contextual information, such as the number of different objects and their properties (width, height, position, etc.) can be used to measure semantic similarity between images and split them using these measures. Applying semantic similarity-based splitting, which incorporates richer contextual cues from images, yields higher detection performance than visual similarity-driven splitting.

1.4.3 Development of the D-LeDe (Data Leakage Detector) Method (RG3)

The third research goal (RG3) was to establish a method for detecting data leakage in existing dataset splits by analyzing the relative performance increase rate (RIR). This goal builds directly on the insights from RG1 and RG2, which showed that evaluation outcomes can be distorted by hidden similarities between training and test data. RG3 aimed to quantify how incremental leakage affects model performance and to use this relationship as a diagnostic indicator for detecting potential leakage in automotive perception datasets.

The D-LeDe method (Data Leakage Detector) introduced in Paper C, is designed to identify potential leakage by quantifying how model performance changes when controlled amounts of test data (10% in each step) are gradually leaked into the train set. The underlying hypothesis was that if the original split already contains leakage, the relative performance increase rate (RIR) would remain low even as more leakage is introduced, since part of the test data is already familiar to the model. In contrast, a clean split would make the model to show a steeper rise in performance with incremental leakage, indicating the sensitivity of the model to newly exposed data. The D-LeDe method was validated on the KITTI dataset, a widely used automotive benchmark. The analysis revealed a low RIR in the default split of KITTI, suggesting the presence of existing leakage. To verify this finding, pairwise similarity analysis using perceptual hashing (pHash) was conducted and the outcome confirmed that a significant number of near-identical images were present across the training-test partition.

Answer to RQ3.1: Incrementally introducing data leakage leads to increased model performance, demonstrating a measurable link between leakage and inflated evaluation results. The rate of this increase, the Relative Increase Rate (RIR), provides a quantitative indicator of how sensitive model performance is to leakage, forming the basis for leakage detection.

The main contribution of this study was to show that data leakage is a practical risk that can be systematically discovered. By providing a method for leakage detection, this work bridged the gap between recognizing the problem (RG1 and RG2) and enabling dataset users to detect it in practice.

Answer to RQ3.2: The trajectory of performance growth differs between clean and contaminated splits which can be measured using Relative Increase Rate (RIR) as per the outcome of **RQ3.1**. It was observed that RIR increases by more than 5% with every 10% leakage introduction if the initial split does not contain leakage. In contrast, when leakage is already present in the data split, RIR increases by less than or equal to the 5% threshold. This difference allowed the development of the D-LeDe method, which uses the RIR as an indicator to detect possible leakage in datasets.

1.4.4 Cross-dataset and Cross-architecture Validation of the D-LeDe Method (RG4)

The fourth research goal (RG4) was to detect data leakage in existing splits of multiple automotive datasets, including a proprietary one, using the proposed D-LeDe method applied on multiple model architectures. While RG3 demonstrated that the method could detect leakage in a single benchmark dataset, it remained an open question whether it can consistently detect data leakage in other datasets also which contain images of more diverse driving conditions and scenarios. In particular, for a detection method to be useful in practice, it must work consistently regardless of the dataset or object detection model employed.

Paper D addressed this by applying D-LeDe to five automotive datasets — BDD100k, ONCE, SODA10M, ZOD, and a proprietary dataset using two different object detection architectures: YOLOv7 (single-stage) and Faster R-CNN (multi-stage). The evaluation showed a consistent pattern: the default splits of three datasets (BDD100k, Proprietary-FC, and ZOD) were flagged to contain data leakage by D-LeDe across both OD models, while ONCE and SODA10M were not. These results were further supported by pHash analysis, which revealed higher inter-set similarity scores in the default splits where D-LeDe identified leakage.

Answer to RQ4.1: Several widely used automotive datasets, including KITTI and BDD100K, contain data leakage in their default training–test splits. Other datasets, such as ONCE and SODA10M, do not have data leakage in their default splits.

The findings reinforced that D-LeDe is capable of identifying data leakage consistently across varied datasets and detection model architectures. This consistency highlights its practical relevance for practitioners and researchers aiming to avoid misleading evaluation results.

Answer to RQ4.2: The D-LeDe method consistently detected leakage across multiple datasets with their default data splits and across different object detection model architectures, including single-stage (YOLO) and multi-stage (Faster R-CNN) architectures. This indicates that the method can be applied reliably in diverse experimental setups.

1.5 Threats to Validity

This section reflects on potential aspects that may affect the validity of the findings. Identifying such factors allows for a critical evaluation of the research process and provides transparency regarding its limitations. To structure this reflection, we draw on the framework proposed by Wohlin et al. [54]. This section presents four commonly recognized categories of validity threats: conclusion, internal, construct, and external. These categories provide a systematic lens to assess the credibility, rigor, and applicability of the results, while also clarifying the measures taken to mitigate identified risks.

Conclusion Validity

Conclusion validity concerns the reliability of the relationships observed in the studies. A recurring issue in **Papers A, B**, and **C** is the dependence on mAP (and in some cases F1-score) as primary performance measures. **Paper C** further highlights that the choice of threshold for perceptual hashing (pHash) may not fully capture all degrees of image similarity. **Paper D** additionally notes that random factors such as weight initialization or batch ordering can influence mAP and RIR values; to mitigate this, results were averaged over multiple runs. Overall, while the chosen metrics are standard in the literature, their sensitivity to data imbalance and experimental design is inevitable, particularly while the examined datasets were studied in their default splits.

Internal Validity

Internal validity refers to factors that may confound the observed results. All four papers acknowledge risks related to training procedures. **Papers A, B**, and **C** describe a possible maturation threat, as models were initially trained for a fixed number of epochs (100), regardless of convergence. This was partly mitigated in later experiments by extending training to 500 epochs, though the gains from using more epochs were minimal. **Paper D** identifies risks stemming from differences in hyperparameter tuning between YOLOv7 and Faster R-CNN, which could affect performance comparisons. These were addressed by keeping the hyperparameters the same across both model architectures. Together, these

issues underscore that procedural variations could influence observed outcomes, though mitigation steps were taken by applying standardized training settings.

Construct Validity

Construct validity concerns whether the studies truly measure what they claim to measure. A key limitation arises from **mono-operation bias** in **Papers A, B**, and **C**, since the experiments focused primarily on object detection tasks. This could restrict the generalizability of findings to other perception functions such as semantic segmentation or tracking. To mitigate this, the chosen task was retained for its representativeness in automotive perception — object detection being a fundamental component of most perception pipelines — and the conclusions were framed accordingly, without overstating their scope. Future work is planned to test the findings in related tasks to confirm transferability. In **Paper C** and **Paper D**, the reliance on a single similarity threshold for pHash posed a risk of inadequately representing the construct of "image similarity". This was mitigated by performing sensitivity analyses using multiple threshold values and verifying the results with visual inspection of image pairs.

Papers A and **B** also noted potential **mono-method bias**, as mean Average Precision (mAP) alone does not capture all aspects of model behavior, particularly the balance between precision and recall. To mitigate this, complementary metrics such as the F1-score were incorporated into the analyses. This triangulation of metrics strengthened the validity of the conclusions by ensuring that observed performance differences were not artifacts of a single evaluation criterion.

External Validity

External validity relates to the generalizability of the findings. All four papers acknowledge limitations in model and dataset selection. **Papers A, B**, and **C** rely heavily on YOLOv7 which, even though justified by referring to the existing literature, may not represent other detection architectures. **Paper D** addresses this concern by adopting other model architectures. Dataset imbalance is highlighted in **Papers A–C**, with Cirrus and KITTI in particular introducing skewed class distributions. While this mirrors common challenges in benchmark datasets, it reduces the ability to generalize to more balanced scenarios. **Paper A** and **B** also stress that using a single OEM dataset limits applicability, and **Paper D** points out that restricting evaluation to automotive datasets constrains generalizability to other domains such as medical or aerial imaging.

Taken together, these limitations imply that the findings are most reliable within automotive object detection settings that exhibit similar data characteristics. Applying the results to other domains or model types should therefore be done with empirical validation, ensuring that both dataset composition and architectural assumptions align with those studied here.

In summary, the identified threats to validity reflect the complexity of evaluating data leakage in automotive image datasets. By making these limitations explicit, the thesis provides a basis for interpreting the findings with appropriate caution and for guiding future work to address remaining concerns.

1.6 Discussion

The results of this research collectively demonstrate that data leakage is a significant and quantifiable factor influencing the evaluation of object detection systems in automotive perception. By systematically analyzing different data splitting strategies and similarity measures, the work moves toward identifying the presence of leakage through the development of the D-LeDe (Data Leakage Detector) approach. The findings have implications not only for academic research in machine learning evaluation but also for industrial practices where dataset integrity directly affects product reliability and safety.

From a methodological perspective, the proposed D-LeDe method represents an important step forward in ensuring the trustworthiness of the dataset and evaluation of the data split used for model development. By using the Relative Increase Rate (RIR) as a diagnostic measure, it becomes possible to quantify whether and to what extent a dataset split suffers from data leakage. This approach provides a practical mechanism to detect contamination that traditional similarity filtering methods might overlook. The validation across multiple public automotive datasets and different model architectures confirmed that RIR patterns can consistently distinguish between clean and contaminated splits. This contributes to the academic state of the art by operationalising the detection of data leakage as a measurable phenomenon rather than a qualitative concern. In contrast to previous studies that primarily warned about the risk of duplication or hidden correlations, this work demonstrates a scalable and reproducible way of diagnosing such issues, linking them directly to performance distortions observed in object detection tasks.

For the automotive industry, the ability to systematically detect and quantify data leakage provides a tangible advantage. It allows for better dataset governance, where splits can be verified before model training and evaluation, reducing the risk of false estimation of the perception system accuracy. This contributes directly to safer and more reliable model deployment, which is essential in applications where perception errors can have safety-critical consequences. Integrating D-LeDe into the company's model development workflow offers additional value by shortening the development cycle, preventing costly re-training, which might be caused by leakage while detected in the validation processes before deployment. In an industry moving rapidly toward autonomous and assisted driving technologies, such systematic verification becomes a key element of both product quality and institutional credibility.

From a scientific standpoint, the research reinforces the growing recognition that dataset construction is as critical as model architecture in determining machine learning performance. The findings align with recent trends in data-centric AI, emphasizing that fairness and generalization cannot be guaranteed without ensuring clean and representative data splits [84]. By empirically linking similarity-induced overlap to measurable evaluation bias, this thesis bridges a conceptual gap in the literature: it translates the notion of "hidden data contamination" into a reproducible metric that can be applied across different domains. As such, the D-LeDe method not only strengthens evaluation reliability but also contributes a new tool to the scientific community for assessing the quality of data splits and benchmarking future research.

In practical terms, the studies collectively demonstrate that the reliability

of perception models depends not only on how well they are trained but also on how rigorously their data pipelines are controlled. The proposed method helps to identify and mitigate subtle forms of leakage before deployment. This capacity to systematically audit the used data split has implications for academic reproducibility as well, offering a pathway toward more transparent reporting standards in future machine learning studies.

Although the work primarily focuses on automotive perception, its broader relevance should not be overlooked. The principles and detection strategies developed here are applicable to other domains where image similarity and overlap are prevalent, such as medical imaging and aerial surveillance. In all these areas, uncontrolled data leakage can distort evaluation results, leading to overestimated model performance and potentially unsafe or unreliable real-world outcomes. The method introduced through this thesis thus offers a transferable framework for safeguarding the integrity of model evaluation across application areas.

Overall, this research contributes to a paradigm shift from assumption-based evaluation to evidence-based auditing of data splits. For both industry and academia, this means moving toward evaluation practices that are not only more transparent but also more defensible.

1.7 Conclusions and Future Work

This thesis set out to investigate the problem of data leakage in automotive object detection and to explore ways of detecting it. The work was motivated by the observation that evaluation practices in computer vision often lack an awareness that subtle overlaps and similarities between partitions may enhance reported results. By addressing this gap, the thesis contributes both to the scientific understanding of the phenomenon and to the practical toolkit available for engineers working in safety-critical domains.

The four papers included in the thesis form a coherent progression. Papers A and B provide evidence that data leakage is an issue that can systematically bias evaluation outcomes. They demonstrated that performance differences across splitting strategies are substantial, and that similarity between images plays a key role in shaping these outcomes. Papers C and D advanced the contribution further by introducing and validating the D-LeDe method, an approach to detecting leakage in an existing data split. Taken together, these studies not only diagnose the problem but also provide a pathway toward more reliable evaluation practices in automotive perception.

The results of this thesis underline the importance of careful data splitting, and systematically evaluating it should be a prerequisite for deploying trustworthy perception systems. For autonomous driving, where errors can have severe consequences, this insight is particularly necessary. Beyond the automotive domain, the lessons learned here might also be applicable to other areas of machine learning where similar data collection processes may occur as well.

Several directions for future research emerge from this thesis. First, while D-LeDe currently detects data leakage based on visual overlap between data partitions, its analytical scope can be extended by incorporating more ex-

pressive feature representations. For instance, leveraging image embeddings derived from self-supervised or foundation models could allow to capture deeper semantic similarities that go beyond visual similarity analysis. Second, further validation in real-world industrial settings would help confirm the practicality and scalability of the method, especially when applied to large proprietary datasets. Third, integrating similarity analysis and duplicate image detection directly into dataset creation and curation pipelines will create a shift from reactive to preventive approach. Such integration within data annotation or pre-processing frameworks could allow developers to eliminate potential leakage even before the training begins.

Finally, future research should also explore the relationship between data leakage and other sources of evaluation bias, such as distributional shifts or label noise. Investigating how these factors interact may help build a more comprehensive framework for reliable evaluation in safety-critical applications. By continuing along these lines, the community can work toward developing more accurate and more trustworthy machine learning systems.