Evaluating the Robustness and Generalizability of D-LeDe Using Multiple Automotive Datasets

Downloaded from: https://research.chalmers.se, 2026-10-16 02:14 UTC

Citation for the original published paper (version of record):

Babu, M., Staron, M., Durisic, D. et al (2026). Evaluating the Robustness and Generalizability of D-LeDe Using Multiple Automotive Datasets. SN Computer Science, 7(7). http://dx.doi.org/10.1007/s42979-026-05289-7

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

SN Computer Science (2026) 7:704
https://doi.org/10.1007/s42979-026-05289-7

ORIGINAL RESEARCH

Evaluating the Robustness and Generalizability of D-LeDe Using Multiple Automotive Datasets

Md Abu Ahammed Babu[1,2] · Miroslaw Staron[2] · Darko Durisic[1] · András Bálint[1] · Sushant Kumar Pandey[3]

Received: 10 September 2025 / Accepted: 8 August 2026
© The Author(s) 2026

Abstract

Data leakage, where semantically or visually similar samples exist across training and test splits, continues to threaten the reliability of object detection benchmarks. The D-LeDe method was recently proposed as a statistical technique for detecting data leakage, showing promise in initial applications. This paper extends the D-LeDe method to evaluate its robustness and generalizability through a focused experimental study. We revisit the KITTI dataset which was previously identified as leakage-prone and introduce a new application of D-LeDe on SODA10M. The core investigation centers on whether visually similar images, measured using perceptual hashing, are the primary cause of data leakage indications captured by D-LeDe. To this end, we progressively remove visually similar image pairs from the test sets of both datasets and observe changes in the Relative Increase Rate, the key decision metric of D-LeDe. Results show that even after removing about 50% of similar images from the KITTI test set, D-LeDe continues to detect data leakage, suggesting other forms of redundancy or latent factors causing the leakage. In contrast, SODA10M consistently remains leakage-free across all levels of image pair removal. These findings reinforce the reliability of D-LeDe in detecting non-obvious forms of data leakage, underscoring its applicability as a diagnostic tool for dataset integrity in object detection workflows.

Keywords Data leakage detection · Object detection · YOLOv7 · Kitti · Soda10m · Automotive perception systems

Introduction

In recent years, deep learning (DL) has become the backbone of perception systems in autonomous driving (AD), with object detection (OD) models playing a critical role in understanding the vehicle's surroundings. As these models continue to improve, so does our reliance on their accuracy for safe navigation, planning, and real-time decision-making. But there is a foundational assumption that often goes unchecked which is the belief that the training and test data used to evaluate these systems are truly independent. In reality, that assumption is surprisingly easy to violate and the consequences are more serious than they might first appear.

This issue, known as data leakage occurs when information from the test set unintentionally influences the model during training [5]. Unlike intentional cheating or malicious behavior, most data leakage is subtle and accidental that may occur as a result of careless dataset splitting, overlooked temporal sequences, or duplicate images slipping through pre-processing. An example of such a case is illustrated in Fig. 1. Unfortunately, when data leakage is present, performance metrics like mean average precision (mAP) can become misleadingly optimistic. A model might appear to perform exceptionally well, but only because it has already seen parts of the test data in some form during training. In safety-critical domains like AD, this is a risk that we cannot afford.

✉ Md Abu Ahammed Babu
md.abu.ahammed.babu@volvocars.com

Miroslaw Staron
miroslaw.staron@gu.se

Darko Durisic
darko.durisic@volvocars.com

András Bálint
andras.balint@volvocars.com

Sushant Kumar Pandey
s.k.pandey@rug.nl

1 Volvo Cars, Gothenburg, Sweden

2 University of Gothenburg and Chalmers University of Technology, Gothenburg, Sweden

3 University of Groningen, Groningen, The Netherlands

Published online: 20 August 2026

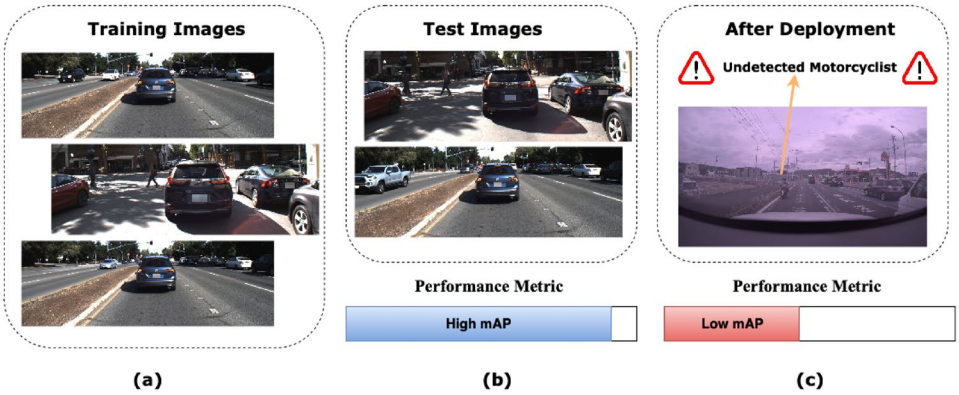


Fig. 1 An illustrative example of how data leakage during a split artificially inflates the performance of the model. The split shown in **a** and **b** results in sequential image frames being spread between training and test data which leads to high performance during test phase and gives false confidence in the model, leading to its deployment. A potentially hazardous consequence is illustrated in **c** showing a situation in which motorcyclist in front of the vehicle is undetected

Beyond benchmarking concerns, inflated performance estimates due to unnoticed data leakage can have tangible implications in intelligent transportation systems. For instance, vision-based traffic violation detection frameworks relying on conventional Closed-Circuit Television (CCTV) infrastructure depend on reliable object detection performance for enforcement and monitoring tasks [21]. Similarly, large-scale traffic sensing and network planning systems use data-driven models to inform infrastructure and sensor deployment decisions [2]. In such contexts, overestimated validation metrics may lead to unjustified confidence in system robustness and operational reliability. Ensuring the integrity of training and evaluation splits is therefore essential not only for methodological soundness but also for trustworthy deployment in transportation applications.

Data leakage can manifest in various ways such as data augmentation overlap, or inadvertent reuse of visually similar images in both train and test sets. In image-based datasets for AD, which often contain sequences of near-duplicate frames or geographically clustered samples, data leakage is especially difficult to detect. Moreover, many perception datasets, particularly in the automotive domain, do not provide explicit documentation or procedures to ensure the removal of near-duplicate or temporally adjacent frames when constructing training and test splits. This lack of formal de-duplication or temporal separation strategies can lead to inadvertent data leakage. As a result, even widely used benchmark datasets may contain overlapping content across splits if not handled carefully.

To address this, the D-LeDe (Data Leakage Detector) method was recently proposed to detect the presence of data leakage in object detection datasets [4]. Rather than relying on metadata, filename matching, or perceptual similarity alone, D-LeDe quantifies data leakage by injecting controlled amounts of test data into the training set and observing the relative increase rate (RIR) in model performance metrics such as the mean Average Precision (mAP). If performance fails to improve significantly with small data leakage increments (e.g., after adding 10, respectively 20% of test data to the training set), this suggests that the model had already benefited from similar data during training, indicating that leakage was present in the original split.

In the original work [4], D-LeDe was applied to the KITTI object detection benchmark [10] and a proprietary automotive dataset ("Cirrus") [27], where it successfully revealed leakage in KITTI despite a lack of exact duplicate images. The method's simplicity, and reliance on quantifiable performance metrics make it an appropriate tool for data leakage auditing in the automotive image datasets.

However, an open question remains: To what extent can visual similarity, as captured by perceptual hash (pHash) distances [32] or similar metrics, serve as a proxy for data leakage? Intuitively, one might assume that filtering out highly similar image pairs based on perceptual similarity should reduce leakage and thus render methods like D-LeDe as redundant. In this extended study, we challenge that assumption through a systematic application of D-LeDe in the presence and absence of similar image pairs.

To do so, we considered a large-scale, diverse object detection dataset, SODA10M [12], which unlike KITTI, contains thousands of frames from a broad range of environmental conditions, geographical locations, and temporal windows. Applying D-LeDe to SODA10M revealed no detectable data leakage in its standard split. We then removed visually similar image pairs—based on pHash distance measurements from both KITTI and SODA10M incrementally, starting from the lowest pHash distance (indicating highest similarity), and observed how the RIR patterns evolved. This progressive removal led to consistent RIR values below the threshold in KITTI, confirming leakage, whereas SODA10M's RIR also remained consistently above the leakage threshold, reinforcing the conclusion that no critical data leakage was present in SODA10M's default split.

To guide our study, we formulate the following research questions:

RQ1: How does the D-LeDe method perform when applied to the SODA10M dataset, beyond its initial evaluation on KITTI?

RQ2: What is the impact of removing visually similar images on the relative increase rate used in D-LeDe?

RQ3: To what extent can perceptual similarity analysis alone be used as a substitute for the D-LeDe method?

The contribution of this paper can be summarized in the following highlights:

- We evaluate D-LeDe on both KITTI and SODA10M datasets, demonstrating its generalizability to diverse datasets.
- We perform a controlled filtering of similar image pairs using pHash distance in both KITTI and SODA10M to test D-LeDe's dependence on visual similarity.
- We show that D-LeDe's detection signal persists even after removal of similar images, confirming it captures a broader spectrum of leakage beyond perceptual similarity.

Together, these contributions extend D-LeDe's original findings and validate its utility as a practical tool for identifying and auditing data leakage in real-world object detection pipelines.

The rest of the paper is structured as follows. In section "Related Work", we review related work on data leakage detection, dataset splitting, and visual similarity filtering. Section "D-LeDe Method" briefly summarizes the D-LeDe method. Section "Research Design" describes the datasets and experimental setup, including the pHash distance-based filtering strategy. In section "Results", we present results from both KITTI and SODA10M, with and without filtering, and analyze the impact on relative increase rates. Section "Discussion" discusses the implications of the results for benchmark design and dataset hygiene. Finally, section "Threats to Validity" discusses different validity threats considered in this research and section "Conclusion and Future Work" concludes the paper and outlines directions for future research.

Related Work

Data leakage occurs when a feature inadvertently reveals information about the target variable–either because the same data points appear in both training and testing sets or due to preprocessing methods such as mean imputation, seasonal adjustments, or correlated variables [5, 13, 25]. While this issue is well-known in structured ML [15, 24], its manifestation in image-based tasks, particularly object detection for autonomous vehicles is often overlooked. In computer vision pipelines, "data leakage" is not only about exact duplicates but also arises when non-identical yet highly similar frames—such as consecutive camera captures—find their way into both train and test splits [14]. These data leakage can possibly lead to object detection models that depend on the background of the images or contextual cues rather than the intended object representations. This can result in inflated test performance and potentially unsafe behavior in production, especially in safety-critical domains like autonomous driving [16].

Additionally, ML experts have identified data leakage as one of the hidden contributors to the reproducibility crisis [24]. As Götz and colleagues argue [11], data leakage can manifest at multiple stages of the machine learning pipeline—even before model training begins—such as through unintended overlaps or information flow between development phases like validation and test data. A wide array of data leakage cases have been documented: Yagis et al. [29] demonstrate a 40–55% inflation in brain MRI classification results due to improper slice-level splitting, while Bussola et al. [7] find up to 41% performance inflation in histopathology when leakage is not addressed.

Such evidence underscores that even a small extent of data leakage can lead to substantial performance distortions. Yet, existing detection methods remain largely ad hoc or manually overseen. Apicella et al. [3] categorize various forms of data leakage in ML and highlight their consequences for model reliability, but note a lack of practical detection mechanisms—especially for image data. On the code analysis side, static analyzers such as the one developed by Yang et al. [30] can detect data leakage related to pre-processing issues in Jupyter notebooks, including label leakage, improper target encoding, and data normalization across training and test sets, achieving 97.6% precision. However, these tools are designed primarily for structured data workflows and do not address dataset-level, image-based leakage, such as duplicated or overly similar images appearing in both training and test sets.

In color and texture-rich domains like object detection, "Clever-Hans" patterns can emerge [17]. This leads to the unsettling possibility that a model may perform well on test data, not because it generalizes but because leakage has given it an unfair preview of contextually similar examples. Previous efforts to address data leakage in visual data have centered largely on pixel-wise de-duplication. Techniques such as perceptual hashing (pHash) compute compact image fingerprints to weed out duplicates [6]. However, these approaches are fundamentally narrow, targeting only pixel-level redundancy. They cannot reliably detect semantic overlap, scene or object co-occurrence, or variations introduced by different sensors or lighting. Moreover, hashing methods can be circumvented through adversarial perturbations or

lead to false negatives in semantically overlapping images that differ visually [22]. Recognizing these limitations, the original D-LeDe method introduced a model-based solution to data leakage detection [4].

Parallel to this work, recent benchmarks in autonomous perception have addressed dataset diversity and redundancy. SODA10M dataset includes labeled frames across 32 cities, various weather conditions, and multiple sensor types, offering 10 million unlabeled and 20,000 labeled images [12]. Its design intent was clear: reduce visual redundancy and improve robustness [8, 18]. Yet, until now, no systematic data leakage analysis of its labeled splits has been done.

This paper addresses this gap by a stepwise process. First, we apply D-LeDe to SODA10M for the first time, testing whether the labeled splits suffer from data leakage. Second, we introduce controlled filtering using pHash to remove 0–50% of visually similar image pairs from the test set, imitating common de-duplication workflows. Third, we re-apply D-LeDe at each filtering stage to assess whether the assessment of the presence or absence of data leakage persists.

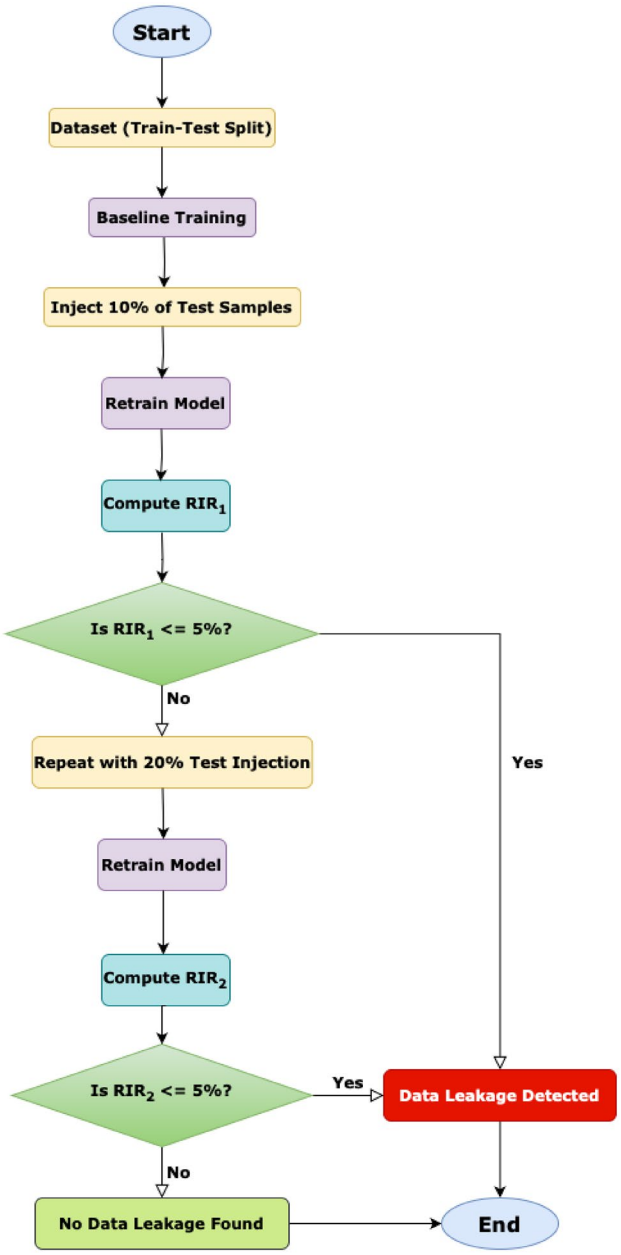


Fig. 2 Flowchart of the D-LeDe method. Leakage is inferred when injecting test data into the training set fails to significantly improve test performance. The method includes two stages of injection (10% and 20%) and evaluates the corresponding RIR

D-LeDe Method

As described in the Introduction, **D-LeDe** is a model-driven approach that detects potential data leakage by analyzing the change in performance when a subset of test data is injected into the training set. It computes a metric called the Relative Increase Rate (**RIR**), which quantifies the marginal performance gain after this injection. A disproportionately low gain may suggest that the model has already been implicitly exposed to similar data, indicating leakage.

The D-LeDe decision criterion was originally derived from controlled experiments conducted on a clean split of the Cirrus dataset [27]. In that setting, ground-truth partition integrity was known, and synthetic leakage was introduced incrementally by injecting predefined proportions of evaluation samples into the training set. The resulting RIR trends were analyzed to characterize how performance increases as a function of leakage magnitude. Based on these empirical observations, a 5% RIR cutoff and a fixed number of leakage injection steps were established as practical decision parameters. It is important to note that the original study focused on trend characterization and empirical threshold selection rather than formal false positive/false negative rate estimation. The present work builds upon those findings and evaluates the consistency of D-LeDe results across additional datasets while incorporating statistical robustness analysis.

As illustrated in Fig. 2, the methodology begins with training a baseline model using a designated clean training set. After convergence, the model is evaluated on the corresponding test set to obtain a reference performance metric, such as mAP in the case of object detection. This establishes the initial benchmark. Next, a small, randomly selected subset of the test set—typically 10%—is injected into the training data, simulating a data leakage scenario. The model is re-trained or fine-tuned with this modified training set, and the evaluation metric is computed again. The first-stage RIR is calculated as the percentage increase in test performance compared to the baseline.

If the RIR is found to be less than a threshold value (empirically set to 5%), this is taken as an indicator that the injected samples did not contribute significant new information. In other words, the model likely had access to similar patterns during its initial training, pointing toward leakage.

If RIR was above the threshold value, a second-stage analysis is performed: 20% of the test set is injected into the training data, and the model is retrained once more. If the performance gain in this round also remains below the threshold, D-LeDe strengthens its inference of leakage.

In line with the original D-LeDe design, we employ a two-step injection strategy to measure RIR at two points, after injecting 10% and then 20% of the test set into the training data. A low RIR at the 10% step is sufficient to indicate potential data leakage, while the second step serves to verify results in cases where the initial RIR is ambiguous or close to the decision boundary. The RIR cutoff (set at 5%) and injection thresholds follow the original study's configuration [4] and were selected empirically to balance sensitivity and robustness. While not re-tuned here, these parameters remain adjustable depending on the application context.

A distinguishing feature of D-LeDe is its semantic granularity. Rather than detecting exact duplicates (as is done in traditional leakage detection via perceptual hashing or checksum comparison), D-LeDe is able to capture leakage caused by nuanced correlations—e.g., two images showing the same object in slightly different lighting or angles. This makes it especially valuable for real-world datasets in autonomous driving, surveillance, or medical imaging, where near-duplicates are common and pixel-level similarity can be misleading.

Another practical advantage is D-LeDe's model-centric design. It does not rely on handcrafted similarity metrics or dataset-specific heuristics. Instead, it functions as a plug-in diagnostic module that can be applied post-hoc to any supervised training pipeline. Because it evaluates leakage using the model's own capacity to learn, it is implicitly tailored to the domain and the task-specific decision boundary. This property makes D-LeDe applicable to not only automotive but other domains also. For domains such as autonomous driving, healthcare diagnostics, and financial risk modeling, D-LeDe offers an interpretable and reproducible tool to validate model robustness.

Research Design

This section outlines the experimental setup used to evaluate the D-LeDe method under controlled conditions. All evaluations were carried out on established object detection datasets (section "Datasets") using two types of detection model (section "Object Detection Models"), and a progressive filtering strategy was applied to test for the contribution of visually similar images to potential leakage indications.

Datasets

We conducted experiments on two publicly available datasets commonly used in the context of object detection: KITTI [10] and SODA10M [12]. These datasets were selected for their contrasting characteristics in terms of size, and content diversity.

KITTI

The KITTI dataset is a widely adopted benchmark in the field of autonomous driving and object detection. It consists of high-resolution RGB images captured from a forward-facing camera mounted on a vehicle navigating urban environments. The dataset includes annotations for several object categories, such as Car, Pedestrian, and Cyclist, along with 2D bounding boxes. In this work, we focus exclusively on the object detection task using the 2D annotations.

SODA10M

It is a large-scale driving dataset designed to support training and evaluation of perception models in diverse real-world scenarios. It comprises more than 10 million unlabeled images and a subset of 20,000 images with human annotations across 9 object classes. The dataset includes diverse environments and conditions, including varying weather, time-of-day, traffic density, and geographical locations. SODA10M is particularly relevant for assessing the scalability and generalizability of D-LeDe, due to its size and diversity of the images in it.

Dataset Preparation and Consistency

Both datasets were used in their original form without any image-level preprocessing or filtering. The only modification involved converting the annotations into YOLO format, which was necessary for compatibility with the object detection model employed in all experiments. To maintain consistency with prior work and to isolate the effect of dataset characteristics on the detection of leakage, the same preprocessed version of each dataset was used throughout the study, during initial leakage assessment, progressive filtering, and re-evaluation stages.

Dataset Splits

The KITTI object detection benchmark provides 7481 labeled training images and 7518 unlabeled test images. Since test annotations are not publicly available and evaluation is conducted via the official server, we did not use the official test split for quantitative analysis in this study. Instead, all experiments were conducted exclusively on the labeled training set. Following common practice, we constructed an internal split by partitioning the 7481 training images into a training subset (70%) and a test subset (30%). The test subset was held out during training and used for mAP computation. However, as only the official test images are available, we used them to do a comprehensive pHash distance analysis of the official KITTI train-test images as a whole at the end to investigate the true level of image redundancy on that official split. For the SODA10M dataset, we have used the officially provided train/test split of the labeled data.

Object Detection Models

For all experiments in this study, we used the YOLOv7 [26] and RT-DETR [33] object detection models as representatives of single-stage and transformer-based detectors families respectively. In the original D-LeDe study, the method was first introduced and validated using YOLOv7. Retaining the same architecture was crucial for preserving the integrity of the data leakage analysis across both the original and the revised datasets. The addition of one more model family enhances the generalizability aspects of the findings.

YOLOv7 is a state-of-the-art single-stage detector that balances speed with detection accuracy, making it well-suited for large-scale evaluation tasks such as those presented in this study. Its proven robustness across a range of benchmark datasets, including those with diverse visual distributions, also supports its continued use in this extended investigation.

RT-DETR is another state-of-the-art real-time object detector that leverages the strengths of transformer-based architectures to achieve an effective balance between high detection accuracy and efficient inference speed. By eliminating the need for traditional post-processing steps, it streamlines the detection pipeline while maintaining competitive performance. Its strong generalization across diverse benchmark datasets [34], particularly those with complex scenes and varied object scales [1], further demonstrates its suitability for large-scale evaluation tasks in this study.

Experimental Setup

The training and evaluation protocols used in this work mirror those from the original D-LeDe paper, with no modifications or task-specific tuning applied. This consistency is central to ensuring the validity of the comparative results, especially when evaluating the method's assessments on progressively revised datasets.

All models were trained using the default YOLOv7 training configuration: 100 epochs, a learning rate of 0.001, batch size of 16, and image resolution fixed at 640×640 pixels. The optimizer (SGD), data augmentation routines, and learning rate scheduling were also left unchanged. The same hyperparameter set was also used to run the RT-DETR model trainings to ensure the fairness of the comparison of results across different model architectures. Dataset annotations were converted into a YOLO-compatible format, with no additional pre-processing or filtering applied prior to training.

The experimental runs were executed on the same hardware environment used in the original D-LeDe paper [4], comprising a server equipped with an Intel Core-i7 CPU running at 3.70 GHz and 32.0 GB of RAM with an extra NVIDIA GeForce RTX 4090 GPU. The PyTorch 1.13 with CUDA 11.7 framework was used for the model training. Evaluation of the trained models followed the exact same pipeline as previously established. Specifically, mAP was computed across the test set and used to compute the RIR after each stage of data injection.

By ensuring that the training environment, model architecture, and evaluation procedures remained stable across all experiments, we can isolate the effects of dataset revisions and assess the robustness of the D-LeDe method under varying degrees of visual redundancy.

Perceptual Hashing

Perceptual hashing (pHash) is a robust method for computing the similarity between images based on their visual appearance. Unlike cryptographic hash functions such as MD5 [9] or SHA-256 [31], which are highly sensitive to even a single pixel difference, pHash is designed to capture the perceptual content of images, making it tolerant to minor transformations such as resizing, slight color variations, or compression artifacts.

In the context of this study, pHash was employed to measure the degree of similarity between training and test images in OD datasets. The motivation stems from the nature of data leakage: when test images are perceptually similar to those seen during training, the model may exhibit artificially high performance that does not generalize to unseen data. Therefore, it is natural to expect that an accurate and

robust image similarity metric could validate whether data leakage exists.

The underlying mechanism of pHash involves several stages that transform an image into a compact binary representation:

1. **Grayscale Conversion:** The image is first converted into grayscale to eliminate color-related variations that are not critical for structure-based similarity.
2. **Resizing:** The grayscale image is resized to a fixed low resolution of 32×32, allowing a reduction in computational complexity while preserving essential structural information.
3. **Discrete Cosine Transform (DCT):** A two-dimensional DCT is applied to the resized image, capturing the spatial frequency components. DCT effectively highlights the prominent structural patterns of the image while discarding high-frequency noise.
4. **Low-Frequency Component Extraction:** The top-left 8×8 submatrix of the DCT coefficients, which contains the lowest-frequency components, is selected. These coefficients represent the broad structural characteristics of the image.
5. **Hash Generation:** The median of the selected DCT values is computed, and each coefficient is then compared to this median. A binary hash of 64 bits is produced by assigning a 1 if the coefficient is above the median and 0 otherwise.

The final binary hash serves as a compact fingerprint of the image. To compare two images, the Hamming distance between their hashes is computed (i.e., the number of locations where the pHash values differ). A lower distance indicates higher perceptual similarity. An example of how the pHash generation process works is given in Fig. 3.

The key advantage of pHash distance lies in its ability to approximate human visual similarity judgments, making it particularly suitable for image-based object detection tasks. Its computational efficiency, coupled with robustness to minor distortions, justifies its use as a supporting analysis tool in the present study.

Progressive Dataset Revision and Re-evaluation

In this study, we carried out a progressive dataset revision process aimed at gradually removing visually similar images from the test sets to understand whether the assessments by D-LeDe regarding data leakage change in this process.

The core idea was to filter out similar images from the test data using the pairwise pHash distance values of all train-test image pairs. For each dataset, we computed the pHash distance between every test image to all training images. Once sorted, these distances allowed us to identify the most visually redundant test samples.

We adopted a stepwise removal approach. Starting with the most identical image pairs (minimum pHash distance), we progressively filtered out test images, first removing images having pHash distance = 0 with any training image, then 2, and so on, before we reached a point where more than 50% of the test images were eliminated. This stopping criterion was chosen because removing more than half the test set compromises the reliability of performance metrics and may introduce evaluation bias. After each step, we performed the D-LeDe method to see if it detected data leakage and whether this assessment changed during the process.

It is important to emphasize that this filtering process was entirely automatic and did not involve human judgment. Each revised test set was paired with the original training set (unchanged throughout) and fed through the same D-LeDe evaluation process. That is, at each revision step, a fresh model was trained on the original training data and evaluated on the full test set (to establish a new baseline), and the fresh model was trained again after injecting a portion (10%

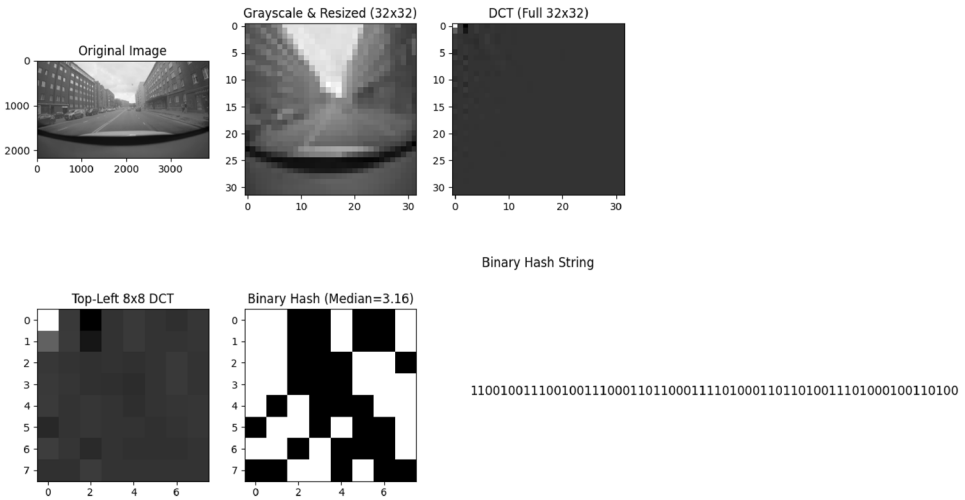


Fig. 3 An illustration of how the pHash of an image is generated. The final outcome of this process is the generated pHash as a binary string of length 64

Table 1 Initial evaluation results of the KITTI and SODA10M datasets

Dataset	Steps	Avg. mAP	RIR (%)	Leakage detected?
KITTI	Step 0	0.852	0.00	**Yes**
	Step 1	0.856	0.47	
	Step 2	0.866	1.17	
SODA10M	Step 0	0.554	0.00	No
	Step 1	0.643	12.28	
	Step 2	0.684	6.25	

D-LeDe indicated data leakage for KITTI but not for SODA10M

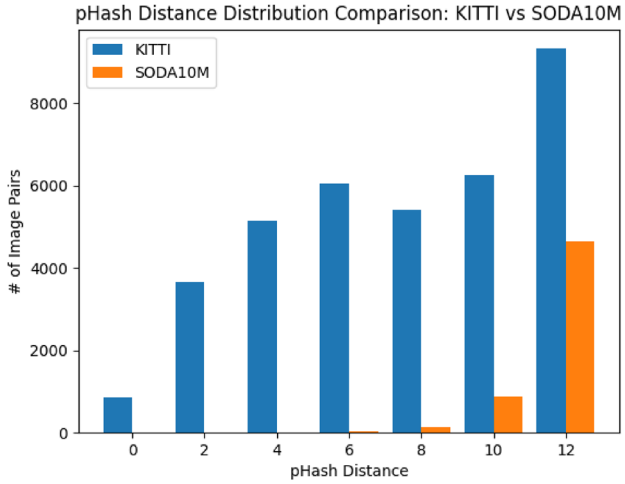


Fig. 4 A histogram of the pHash distances for image pairs in the KITTI and SODA10M datasets up to pHash distance ≤ 12

and then 20%) of the revised test set into the training data and evaluated on the same test data to compute RIR.

This progressive evaluation enabled us to analyze how D-LeDe's data leakage assessment evolved as more visually similar samples were eliminated. If RIR values began increasing consistently across steps, it would suggest that the original leakage diagnosis was largely driven by visual redundancy. Conversely, persistent low RIRs would imply that data leakage was not merely due to perceptual similarity but potentially due to other forms of redundancy in the dataset. The same methodology was applied independently to both the KITTI and SODA10M datasets.

Results

This section presents the outcomes of applying the D-LeDe method to the KITTI and SODA10M datasets, followed by a progressive re-evaluation of both datasets after removing visually similar image pairs. As established earlier, the D-LeDe method indicates data leakage if the RIR remains below or equal to the 5% threshold in either Step 1 or Step 2.

Table 1 shows the initial evaluation results of the two datasets without any intervention. The results clearly indicate the presence of data leakage in KITTI, while there is no indication of data leakage for SODA10M, considering that RIR is well above the 5% threshold in both steps.

Table 2 The number of test images (and associated test set percentage) that have a minimum pHash distance with at least one training image. The bold rows indicate the pHash distances for which percentage of test sets exceeds 50%

Dataset	pHash distance	Number of test images	Percentage of test set (%)
KITTI	0	152	6.1
	2	379	15.2
	4	684	27.4
	6	1027	41.1
	8	**1371**	**54.8**
SODA10M	6	18	0.4
	8	106	2.1
	10	461	9.3
	12	1332	26.7
	14	**2590**	**52.0**

After the initial evaluation, a comprehensive similarity analysis using pHash distance calculation of all train-test image pairs for both datasets has been done. The results of this analysis are shown as a histogram plot in Fig. 4 as described in section "Progressive Dataset Revision and Re-evaluation". Using these results, a stepwise removal of visually similar images from the test set was conducted until we reached the point where 50% or more of the test images needed to be removed. This means if a test image has lower pHash distance with at least one training image, that test image is flagged to be removed during the stepwise removal process. The comprehensive results are summarized in Table 2, which highlights that 1027 test images from KITTI would be removed at maximum, which constitutes 41.1% of the test set and the highest pHash distance to any training image is 6. On the other hand, a total of 1332 test images of SODA10M would be removed throughout this process, which is 26.7% of the test set with the highest pHash distance of 12. Exceeding these will cause the removal of $\geq$ 50% of the test data for both datasets.

Tables 3 and 4 reports the evaluation outcome of the KITTI and SODA10M datasets with the help of two OD models, YOLOv7 and RT-DETR, representing the single-stage detectors and transformer-based detectors families. The tables contain the mean and standard deviations (SD) of the mAP values and RIRs at different steps and the 95% confidence intervals (CI).

The revised experiments on the KITTI dataset (results shown in Table 3) reveal that even after removing test images with a pHash distance of up to 6, which corresponds to removing 41.1% of test set images, the RIR remained below the 5% threshold for both models evaluations (2.17% using YOLOv7 and 1.47% using RT-DETR), thereby consistently

Table 3 D-LeDe evaluation of the **KITTI dataset** with YOLOv7 and RT-DETR detection models

pHash distance	0	2	4	6
YOLOv7				
mAP_0	0.844±0.0054	0.848±0.0034	0.846±0.0030	0.840±0.0033
95% CI_0	[0.837, 0.851]	[0.844, 0.852]	[0.842, 0.849]	[0.836, 0.844]
mAP_1	0.853±0.0065	0.856±0.0022	0.849±0.0024	0.858±0.0029
95% CI_1	[0.845, 0.861]	[0.853, 0.859]	[0.846, 0.852]	[0.855, 0.862]
RIR	*1.05±1.16*	*0.92±0.47*	*0.40±0.39*	*2.17±0.40*
95% CI_{RIR}	[0.39, 2.48]	[0.33, 1.51]	[0.08, 0.89]	[1.68, 2.66]
RT-DETR				
mAP_0	0.892±0.0026	0.892±0.0049	0.890±0.0015	0.883±0.0021
95% CI_0	[0.885, 0.899]	[0.880, 0.905]	[0.886, 0.893]	[0.877, 0.888]
mAP_1	0.901±0.0021	0.900±0.0015	0.902±0.0020	0.896±0.0047
95% CI_1	[0.895, 0.906]	[0.896, 0.903]	[0.897, 0.907]	[0.884, 0.907]
RIR	*0.97±0.47*	*0.82±0.43*	*1.39±0.24*	*1.47±0.68*
95% CI_{RIR}	[0.45, 1.35]	[0.33, 1.12]	[1.12, 1.58]	[0.79, 2.15]

The table shows the mean±SD and 95% confidence interval (CI) of mAPs and RIR. The RIR values italic confirm the presence of data leakage (RIR≤5%) in the KITTI dataset

Table 4 D-LeDe evaluation of the **SODA10M dataset** with YOLOv7 and RT-DETR detection models

pHash distance	6	8	10	12
YOLOv7				
mAP_0	0.558±0.0065	0.558±0.0054	0.552±0.0029	0.532±0.0011
95% CI_0	[0.550, 0.566]	[0.551, 0.565]	[0.549, 0.556]	[0.530, 0.533]
mAP_1	0.641±0.0074	0.638±0.0052	0.636±0.0011	0.633±0.0052
95% CI_1	[0.632, 0.650]	[0.631, 0.644]	[0.635, 0.638]	[0.627, 0.640]
mAP_2	0.686±0.0089	0.686±0.0055	0.676±0.0047	0.677±0.0055
95% CI_2	[0.675, 0.697]	[0.679, 0.693]	[0.671, 0.682]	[0.670, 0.684]
RIR_1	14.7±1.53	14.3±1.67	15.3±0.49	19.1±0.81
95% CI_{RIR1}	[12.82, 16.62]	[12.26, 16.42]	[14.65, 15.87]	[18.12, 20.12]
RIR_2	***7.16±1.48***	***7.73±1.38***	***6.29±0.74***	***6.92±1.32***
95% CI_{RIR2}	[5.32, 9.00]	[6.02, 9.44]	[5.36, 7.21]	[5.29, 8.56]
RT-DETR				
mAP_0	0.603±0.0010	0.607±0.0068	0.601±0.0010	0.599±0.0015
95% CI_0	[0.601, 0.605]	[0.590, 0.624]	[0.599, 0.603]	[0.595, 0.602]
mAP_1	0.673±0.0015	0.682±0.0072	0.688±0.0015	0.676±0.0021
95% CI_1	[0.669, 0.676]	[0.664, 0.700]	[0.684, 0.691]	[0.671, 0.682]
mAP_2	0.724±0.0084	0.739±0.0051	0.747±0.0103	0.720±0.0020
95% CI_2	[0.703, 0.744]	[0.726, 0.751]	[0.721, 0.772]	[0.715, 0.725]
RIR_1	11.57±0.38	12.27±0.21	14.43±0.40	12.97±0.57
95% CI_{RIR1}	[11.30, 12.00]	[12.10, 12.50]	[14.00, 14.80]	[12.50, 13.60]
RIR_2	***7.58±1.01***	***8.38±1.86***	***8.58±1.74***	***6.46±0.38***
95% CI_{RIR2}	[6.41, 8.17]	[6.23, 9.59]	[7.11, 10.50]	[6.06, 6.82]

The table shows the mean±SD and 95% confidence interval (CI) of mAPs and RIR. The RIR values bolditalic indicate absence of data leakage (RIR>5%) in the SODA10M dataset

indicating the presence of data leakage. In contrast, the evaluation outcome on SODA10M dataset, presented in Table 4, was also consistent throughout the progressive removal steps. The RIR_1 was always well-above the 5% threshold (between 14 and 19% using YOLOv7, 11–14% using RT-DETR). Even the RIR_2 values were found between 6.46–8.58%, also above the D-LeDe threshold of 5%, indicating the absence of data leakage.

Observing that the data leakage indication remains the same after removing images at low pHash distances indicates that the detected leakage is not solely caused by image similarities. The D-LeDe method repeatedly flags leakage even after such a substantial removal of visually similar content, which implies that other forms of data leakage might be present in the KITTI dataset, possibly arising from distributional overlap, temporal proximity, or scene redundancy not captured by image similarity analysis alone.

To determine whether the observed RIR values arise from stochastic training variability, we perform paired permutation tests with 95% confidence. For each dataset and configuration, we compute the distribution of mean differences by randomly flipping the sign of per-run paired differences between successive steps across 10,000 permutations. The resulting p-values were consistently well below the threshold (p-value $\ll 0.05$), allowing us to reject the null hypothesis and conclude that the observed RIR variations

are not related to random training fluctuations; rather, these results provide strong evidence that the measured RIR values reflect genuine effects of data leakage.

To assess the robustness of the D-LeDe decision criterion further with respect to the RIR threshold of 5% motivated in [4], we performed a sensitivity analysis by varying the threshold between 2% and 10%. Table 5 summarizes the resulting leakage assessments across different pHash filtering settings for both datasets and the evaluation models. For the KITTI dataset, the leakage verdict remains consistent across all evaluated thresholds and pHash configurations for both models except when test images with pHash distance ≤ 6 were removed and the threshold was set as low as 2%. For the SODA10M dataset, the leakage verdict is also stable (i.e., no leakage detected) for thresholds up to 5%, but begins to change at higher thresholds (8% and above). In fact, the verdict flipped across the models in two cases when the threshold was set to 8%.

Table 5 Sensitivity analysis of the RIR threshold across different pHash distance settings and the OD models used

pHash distance	RIR threshold (%)	Using YOLOv7	Using RT-DETR
KITTI			
0	2	Yes	Yes
	5	Yes	Yes
	8	Yes	Yes
	10	Yes	Yes
2	2	Yes	Yes
	5	Yes	Yes
	8	Yes	Yes
	10	Yes	Yes
4	2	Yes	Yes
	5	Yes	Yes
	8	Yes	Yes
	10	Yes	Yes
6	**2**	**Yes**	**No**
	5	Yes	Yes
	8	Yes	Yes
	10	Yes	Yes
SODA10M			
6	2	No	No
	5	No	No
	8	Yes	Yes
	10	Yes	Yes
8	2	No	No
	5	No	No
	8	**Yes**	**No**
	10	Yes	Yes
10	2	No	No
	5	No	No
	8	Yes	**No**
	10	Yes	Yes
12	2	No	No
	5	No	No
	8	Yes	Yes
	10	Yes	Yes

"Yes" indicates that data leakage is detected, while "No" indicates no leakage. The bold cells are the only cases where the verdict changed across different evaluation models applied while the threshold is very low or much higher than the D-LeDe's 5% default threshold

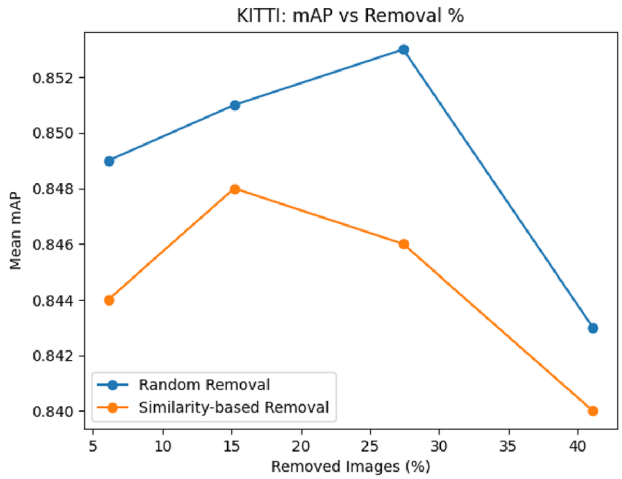
Overall, the results of the analysis indicate that the detection of leakage is sensitive with respect to the RIR cutoff threshold, and the results are less consistent across models when deviating too much from the originally defined cutoff value. These findings support the use of the 5% RIR threshold as defined in [4] as a balanced operating point, providing stable and interpretable decisions across datasets without being sensitive to minor variations in performance gains.

As per Table 1, the amount of perceptual similarity that can be removed while keeping at least 50% of the test set size differs significantly. For KITTI, this point is reached at pHash distance 6, while for SODA10M, the corresponding pHash distance threshold is 12.

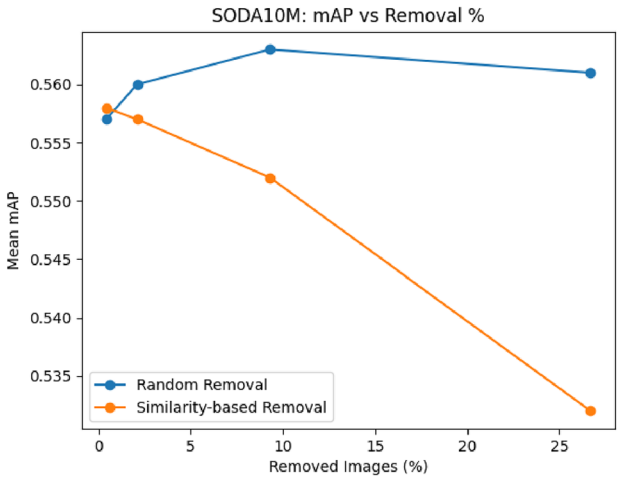
Another intriguing observation is that the RIR values remained relatively stable regardless of how much of the test data was removed. This suggests that D-LeDe assessments are robust, as long as the remaining data still represents the underlying distribution. The fact that the verdicts do not flip on the 5% cut-off even when less than half of the test data is eliminated indicates the method's reliability and stability. To further disentangle the effect of test set size from sample selection, we compared random data removal with image similarity-based (using pHash distances) data removal. As shown in Fig. 5, the mean mAP remains largely stable across varying levels of random removal for both datasets, indicating that test set shrinkage alone has minimal impact on performance. In contrast, similarity-based removal consistently results in lower mAP, with the performance gap becoming more evident at higher removal percentages for the SODA10M dataset.

Furthermore, to investigate the claim that the leakage in KITTI might have arisen not from visual similarity but other forms of overlaps, we have computed the train-test similarity in the semantic space using CLIP [23] embeddings in both datasets. The findings presented in Table 6 prove that the KITTI dataset has higher degrees of train-test similarity in the semantic space which was not captured using visual/perceptual similarity analysis. This provides further indications that the underlying semantic overlap between KITTI's training and test images causes data leakage which was captured by D-LeDe.

Even though we could not perform experimental evaluation of the KITTI official split due to unavailability of the test data labels, we conducted a comprehensive train-test similarity analysis using both pHash distance and CLIP. The pHash distance histogram plot shown in Fig. 6 has the



(a) KITTI dataset



(b) SODA10M dataset

Fig. 5 Effect of test set shrinkage on mAP. Random removal maintains stable performance across different removal percentages, whereas similarity-based removal leads to consistent degradation. This indicates that performance changes are driven by sample selection rather than test set shrinkage

Table 6 Similarity analysis results using CLIP embeddings

Dataset	KITTI	SODA10M
Test images with similarity≥0.95 (%)	2194 (97.5%)	1 (About 0%)
Test images with pHash distance≤6 (%)	1027 (41.1%)	18 (0.4%)

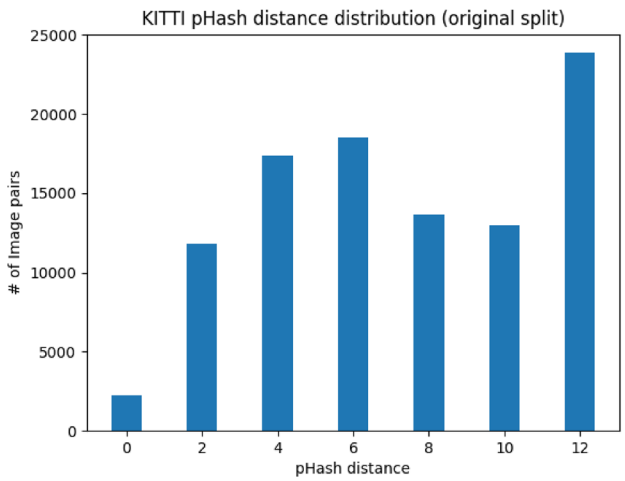


Fig. 6 The pHash distances histogram plot of all train-test image pairs of the official KITTI dataset split

similar pattern of growth as the split used in this research which confirms that the official split also has very strong probability of potential data leakage. In addition, 1733 (23.05%) test images were found using CLIP having similarity≤0.95 with any training image and 419 (5.6%) test images were found with pHash distance≤6. While these numbers may look smaller, they are still higher compared to what has been found for the SODA10M official/default split and thus might pose the leakage risk on the official KITTI dataset split.

Discussion

The findings, as reported in the previous section, indicate the robustness of the D-LeDe method across two examined datasets of varying scale, diversity, and structure, along with two different OD model architectures. The KITTI dataset, evidenced in both initial and extended evaluations, exhibits persistent signs of data leakage, and progressively revised versions of the dataset do not substantially alter this verdict. These results highlight a critical insight: the data leakage in KITTI is unlikely to be solely attributed to image-level visual similarity. Specifically, the CLIP analysis at the end of the previous section indicates semantic similarity in the dataset. This shows that the D-LeDe method can capture data leakage caused by semantic level similarities that would be missed by pairwise image similarity analyses.

Answer to RQ1: Unlike KITTI, which was found to exhibit hidden data leakage, SODA10M showed no indication of leakage based on the relative increase rate (RIR) metric. This demonstrates D-LeDe's applicability across datasets, including two diverse object detection models, and confirms that it does not falsely report leakage in the examined settings.

In contrast, the evaluation of the SODA10M dataset presents a consistent data leakage-negative profile. Having such consistency in the results for both datasets across different model architectures during progressive removal of similar image pairs underlines potential robustness of the D-LeDe method in this context.

Answer to RQ2: Removing visually similar image pairs based on pHash distances had little to no impact on the RIR values for either dataset. For KITTI, the method still identified leakage even after removing highly similar images. For SODA10M, the method continued to report no leakage regardless of similarity filtering. This indicates that the D-LeDe method can capture forms of leakage that cannot be addressed by similarity filtering only.

A noteworthy contrast was reflected while CLIP embeddings were used to measure the level of train-test image similarities of the two datasets. This analysis indicated semantic similarities for KITTI that were not captured by visual similarity, but no such similarities were present for SODA10M. This discrepancy reflects intrinsic differences in dataset construction. KITTI's images are captured from a forward-facing monocular camera in urban driving scenarios, leading to high redundancy and overlap in test and training imagery. SODA10M, on the other hand, is comparatively larger and more diverse in terms of viewpoints, environments, and temporal sampling, which naturally reduces the likelihood of perceptual or semantic similarity-driven data leakage.

Answer to RQ3: Perceptual similarity analysis using pHash was not sufficient to replicate the leakage detection capability of D-LeDe. The results show that even after removing the most visually similar image pairs, D-LeDe's verdict remained unchanged in both datasets. This highlights that perceptual similarity alone cannot reliably identify data leakage, as it fails to capture semantic overlaps or other forms of redundancy.

As a limitation, while the original D-LeDe study established the RIR criterion through controlled leakage injection experiments on the Cirrus dataset, the present work does not include a dedicated benchmark for estimating false positives and false negative rates under systematically controlled leakage conditions. As such, the current evaluation should not be interpreted as a formal detection-theoretic validation of D-LeDe.

Moreover, a global sensitivity analysis involving the joint variation of injection ratios, decision thresholds, random seeds, and training hyperparameters was not conducted in the present study due to the substantial computational cost associated with repeated model training. Although a localized threshold sensitivity analysis was performed, broader variance-based sensitivity characterization remains an important direction for future investigation.

Overall, these findings reinforce that D-LeDe can detect potential data leakage that may not be easily discoverable through conventional image similarity measures, proving its value beyond the application of simpler methods.

Threats to Validity

The validity threat analysis was conducted in accordance with the framework proposed by Wohlin et al. [28].

a) Conclusion validity: This threat concerns the reliability of the inferences drawn from the observed experimental results. As D-LeDe's assessment depends on the relative increase rate in mAP scores, fluctuations in model training due to stochastic factors like weight initialization, data shuffling, or optimizer behavior may influence the outcome. To minimize this, we conducted permutation tests on each experiment's results using the obtained RIRs. The test confirmed that the observed RIR variations were only minimally affected by training stochasticity.

b) Internal validity: Internal validity threats arise from factors that may unintentionally affect the observed outcomes. One such threat involves the consistency of the training configuration across different experiments. Although we used YOLOv7 exclusively to maintain model consistency, minor variations in training duration, augmentation, or convergence behavior across datasets might influence the final mAP values, indirectly impacting the RIR calculation. We addressed this by using the same default configuration (e.g., 100 epochs, learning rate of 0.001, image size 640×640) for all training runs. Additionally, the entire training and evaluation pipeline was kept consistent to isolate the effect of data characteristics and not model behavior.

c) Construct validity: Construct validity refers to how well the experimental procedures and measures align with the theoretical concepts under study. In this work, the central constructs are *data leakage* and *visual similarity*. A key threat arises from the operationalization of "similarity" using pHash alone. While pHash is efficient and robust against pixel-level noise, it may not capture more complex semantic similarities or structural overlaps between images. To overcome this limitation, we have also shown the similarity analysis using CLIP which takes the semantic aspects of images into account to measure similarity.

d) External validity: External validity concerns the generalizability of the results beyond the experimental settings. The findings of this study are limited to two datasets: KITTI and SODA10M, both from the domain of autonomous driving. The extent to which D-LeDe would perform reliably across other domains (e.g., medical imaging, remote sensing) remains unexplored. While this choice ensures internal consistency and reduces confounding factors, it may limit generalizability across architectures that differ in design, such as two-stage models like Faster R-CNN. To address this threat, we have conducted a few primary experiments using Faster R-CNN but not detailed stepwise removal investigations in this study due to time constraints. In future work, we plan to replicate all the experimental steps on the Faster R-CNN model to better understand the D-LeDe's effectiveness across all model architectures.

Conclusion and Future Work

This study presents a systematic extension and evaluation of the D-LeDe method for detecting data leakage in object detection benchmark datasets. By leveraging YOLOv7 as the backbone model, keeping it consistent with the original derivation of the D-LeDe method, and also using another transformer-based model for evaluation, the study re-evaluates the data leakage assessments on the KITTI dataset and extends the method's applicability to an additional large-scale automotive dataset, SODA10M. Through a controlled and progressive removal of visually similar test images based on pHash distances, the robustness and reliability of the D-LeDe method were critically examined.

The findings confirm that the D-LeDe method consistently identifies data leakage presence in KITTI and its absence in SODA10M, even when a considerable amount of the most perceptually similar test images are removed. The consistent verdicts across varying levels of visual similarity threshold suggest that the D-LeDe method captures not only leakage driven by perceptual overlaps but also subtler and possibly non-visual but semantic correlations that may lead to data leakage.

This study further demonstrates the utility of combining statistical evaluation (e.g., RIR thresholds) with dataset-level interventions (e.g., perceptual de-duplication) as a practical framework for understanding model generalization and trustworthiness. Such analyses are particularly important for high-stakes applications in autonomous driving and surveillance, where test leakage can lead to inflated confidence and unrealistic deployment expectations.

Building on these findings, several avenues remain open for exploration. First, the application of D-LeDe can be extended to other domains beyond autonomous driving, such as medical imaging, remote sensing, and industrial inspection, where data leakage might have different characteristics. Second, a comprehensive global sensitivity analysis jointly varying injection ratio, threshold selection, random seed, and training hyperparameters can be done to get a deeper understanding of parameter interactions and variance contributions. Such analyses are commonly performed using variance-based methods (e.g., Sobol indices) or screening approaches such as the Morris method, which have been applied already in other engineering domains to quantify model robustness and parameter influence [19, 20]. Conducting a full global sensitivity study in the context of D-LeDe would be a valuable direction for future work.

While the RIR threshold of the D-LeDe method was empirically calibrated using controlled leakage injection on a clean dataset, a formal theoretical evaluation including explicit estimation of false positive and false negative rates under varying leakage types remains an open research direction. By making the data leakage evaluation process more principled and reproducible, this study contributes a step forward in improving the integrity of benchmark-based research in image recognition and computer vision.

Author Contributions The first author conceived the study, designed the experiments, implemented the methodology, conducted the analysis, and wrote the manuscript. Other authors supervised the research, contributed to the interpretation of results, and reviewed and revised the manuscript. All authors read and approved the final manuscript.

Funding Open access funding provided by Chalmers University of Technology. This research is partially funded by VINNOVA under the grant no. 7000282463.

Data Availability The datasets used in this study are publicly available. The KITTI dataset is available at https://www.cvlibs.net/datasets/kitti/, and the SODA10M dataset is available at https://soda-2d.github.io/. Additional experimental scripts and analysis code can be made available by the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Informed Consent The authors provide consent for publication in this journal.

Research Involving Human and/or Animals This research does not involve any human participants or animals.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Alimov M, Meiramkhanov T. Domain generalization in autonomous driving: Evaluating yolov8s, rt-detr, and yolo-nas with the road-almaty dataset. 2024. arXiv preprint arXiv:2412.12349.
2. Almutairi A, Owais M. Active traffic sensor location problem for the uniqueness of path flow identification in large-scale networks. IEEE Access. 2024;12:180385–403.
3. Apicella A, Isgrò F, Prevete R. Don't push the button! exploring data leakage risks in machine learning and transfer learning. 2024. arXiv preprint arXiv:2401.13796
4. Babu MAA, Pandey SK, Durisic D, Koppisetty AC, Staron M. D-lede: A data leakage detection method for automotive perception systems. In: Proceedings of the 11th international conference on vehicle technology and intelligent transport systems

- VEHITS. 2025. pp. 210–221. INSTICC, SciTePress. https://doi.org/10.5220/0013476700003941
5. Baby A, Krishnan H. A literature survey on data leak detection and prevention methods. Int J Adv Res Comput Sci. 2017;8(5):
6. Biswas R, Blanco-Medina P. State of the art: Image hashing. 2021. arXiv preprint arXiv:2108.11794.
7. Bussola N, Marcolini A, Maggio V, Jurman G, Furlanello C. Ai slipping on tiles: data leakage in digital pathology. In: International conference on pattern recognition. Springer International Publishing Cham; 2021. pp. 167–82.
8. Cheng G, Yuan X, Yao X, Yan K, Zeng Q, Xie X, et al. Towards large-scale small object detection: Survey and benchmarks. IEEE Trans Pattern Anal Mach Intell. 2023.
9. Deepakumara J, Heys HM, Venkatesan R. Fpga implementation of md5 hash algorithm. In: Canadian Conference on Electrical and Computer Engineering 2001. Conference Proceedings (Cat. No. 01TH8555). 2001;2, pp. 919–924. IEEE
10. Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: the Kitti dataset. Int J Robotics Res. 2013;32(11):1231–7.
11. Götz-Hahn F, Hosu V, Saupe D. Critical analysis on the reproducibility of visual quality assessment using deep features. PLoS ONE. 2022;17(8):e0269715.
12. Han J, Liang X, Xu H, Chen K, Hong L, Mao J, Ye C, Zhang W, Li Z, Liang X, et al. Soda10m: A large-scale 2d self/semi-supervised object detection dataset for autonomous driving. 2021. arXiv preprint arXiv:2106.11118
13. Hussein EA, Ghaziasgar M, Thron C, Vaccari M, Jafta Y. Rainfall prediction using machine learning models: literature survey. Springer International Publishing, Cham, 2022. pp. 75–108.https://doi.org/10.1007/978-3-030-92245-0_4
14. Joshi P, Hasanuzzaman M, Thapa C, Afli H, Scully T. Enabling all in-edge deep learning: a literature review. IEEE Access. 2023;11:3431–60.
15. Kapoor S, Narayanan A. Leakage and the reproducibility crisis in machine-learning-based science. Patterns. 2023;4(9).
16. Kernbach JM, Staartjes VE. Foundations of machine learning-based clinical prediction modeling: Part II-generalization and overfitting. In: Machine learning in clinical neuroscience: foundations and applications. 2022. pp. 15–21.
17. Lapuschkin S, Wäldchen S, Binder A, Montavon G, Samek W, Müller KR. Unmasking clever hans predictors and assessing what machines really learn. Nat Commun. 2019;10(1):1096.
18. Liu M, Yurtsever E, Fossaert J, Zhou X, Zimmer W, Cui Y, et al. A survey on autonomous driving datasets: statistics, annotation quality, and a future outlook. IEEE Trans Intell Veh. 2024;
19. Owais M. Analysing witczak 1–37a, witczak 1–40d and modified hirsch models for asphalt dynamic modulus prediction using global sensitivity analysis. Int J Pavement Eng. 2023;24(1):2268808.
20. Owais M. Preprocessing and postprocessing analysis for hot-mix asphalt dynamic modulus experimental data. Constr Build Mater. 2024;450:138693.
21. Owais M, Shehata A, Shaband A, Moussa GS. A framework for establishing an automated traffic violation detection system in new assiut city using ordinary cctv units. Mansoura Eng J. 2025;50(3):11.
22. Qiu Z, Pan Y, Yao T, Mei T. Deep semantic hashing with generative adversarial networks. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval. 2017. pp. 225–234
23. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, et al. Learning transferable visual models from natural language supervision. In: International conference on machine learning. 2021. pp. 8748–63. PMLR
24. Sculley D, Holt G, Golovin D, Davydov E, Phillips T, Ebner D, et al. Hidden technical debt in machine learning systems. Adv Neural Inf Process Syst. 2015;28.
25. Silva GF, Fagundes TP, Teixeira BC, Chiavegatto Filho AD. Machine learning for hypertension prediction: a systematic review. Curr Hypertens Rep. 2022;24(11):523–33.
26. Wang CY, Bochkovskiy A, Liao HYM. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. 2022. arXiv preprint arXiv:2207.02696.
27. Wang Z, Ding S, Li Y, Fenn J, Roychowdhury S, Wallin A, Martin L, Ryvola S, Sapiro G, Qiu Q. Cirrus: a long-range bi-pattern lidar dataset. In: 2021 IEEE international conference on robotics and automation (ICRA). 2021. pp. 5744–50. IEEE
28. Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B, Wesslén A. Experimentation in software engineering. Springer. 2012
29. Yagis E, Atnafu SW, Seco G, de Herrera A, Marzi C, Scheda R, et al. Effect of data leakage in brain mri classification using 2d convolutional neural networks. Sci Rep. 2021;11(1):22544.
30. Yang C, Brower-Sinning RA, Lewis G, Kästner C. Data leakage in notebooks: Static detection and better processes. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering. 2022. pp. 1–12.
31. Yoshida H, Biryukov A. Analysis of a sha-256 variant. In: International workshop on selected areas in cryptography. Springer; 2005. pp. 245–60.
32. Zauner C. Implementation and benchmarking of perceptual image hash functions. 2010, https://phash.org/docs/pubs/thesis_zauner.pdf
33. Zhao Y, Lv W, Xu S, Wei J, Wang G, Dang Q, et al. Detrs beat yolos on real-time object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024. pp. 16965–74.
34. Zhou Y, Wei Y. UAV-detr: an enhanced rt-detr architecture for efficient small object detection in uav imagery. Sensors. 2025;25(15):4582.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.