Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Train–Test Leakage in Pothole Detection Benchmarks: Evidence from a Video-Disjoint Re-Split of BharatPotHole

Pathmanaban Pugazhendi

princepathu@gmail.com

Easwari Engineering College

Research Article

Keywords: Data leakage, Road defect detection, Pothole, Benchmark evaluation, Object detection, Reproducibility

Posted Date: October 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11338135/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Train–Test Leakage in Pothole Detection Benchmarks: Evidence from a Video-Disjoint Re-Split of BharatPotHole

P. Pathmanaban Pugazhendi[1*]

[1]Department of Automobile Engineering, Easwari Engineering College, Chennai, Tamil Nadu, India

*Corresponding author. E-mail: princepathu@gmail.com

Abstract

Detection results reported on public road-defect datasets guide research and the procurement of automated inspection systems. This study examines how train–test leakage affects such results on the BharatPotHole dataset. All 7,074 released images were audited using exact hashing, perceptual hashing with structural-similarity confirmation, export metadata and video identifiers. Of the released images, 86 were exact duplicates and 2,769 were export copies of other frames (40.4% in total), and only 1,333 of 4,196 positive images were unique frames. In the published split, 38.8% of test images had a near-duplicate in training and 99.4% came from training videos. A deduplicated, video-disjoint split was built, and YOLO26 detectors were evaluated under both splits with 95% confidence intervals from a bootstrap over videos. For the same detector, AP_{50} on positive test images was 0.619 on the published test set and 0.467 on its near-duplicate-free subset (difference 0.152, 95% CI 0.090–0.234). AP_{50} decreased with perceptual distance between a test image and its nearest training image. In a controlled contamination test, replacing unseen test images with copies of training frames raised the AP_{50} of a video-disjoint model from 0.133 to 0.494, whereas copies of unseen frames left it near 0.15. On positive images from RDD2022 (India), the model trained on the published split reached AP_{50} = 0.072, against 0.097–0.099 for models trained on the video-disjoint split. Frame-level splits of video-derived road imagery therefore give optimistic estimates of detection performance on new roads. The audit procedure, the corrected split and a reporting checklist are provided.

Keywords: Data leakage · Road defect detection · Pothole · Benchmark evaluation · Object detection · Reproducibility

1 Introduction

Potholes damage vehicles, raise crash risk for two-wheelers and increase maintenance cost when repairs are delayed. Many road agencies still inspect pavements manually, which is slow and covers networks unevenly. Camera-based detection with deep object detectors offers

a lower-cost alternative, and dashboard or smartphone video is now a common data source for such systems [1–3].

Public datasets are the main basis for comparing these detectors. A reported score is read by other researchers as a reference value and by agencies as an expectation of field performance. That reading is valid only if the test images are independent of the training images in the same way that a deployed system meets new roads. The same requirement applies to benchmark studies in transportation data science more generally [4].

Video-derived datasets make this condition easy to violate. Consecutive frames from one drive are nearly identical, and export tools such as Roboflow can add augmented copies of each frame. When frames are split into training and test sets at random, a test image often has a near-copy in the training set, and almost every test image comes from a video that also supplied training images. The test score then partly reflects recognition of seen scenes rather than detection on unseen roads. This is a form of data leakage, which has been documented in many fields that use machine learning [5–8].

This study examines the effect for one recent dataset, BharatPotHole, released with the iWatchRoadv2 system [3]. The dataset contains 7,074 images from dashboard video of Indian roads and a frame-level split described as 70:20:10. Three questions were addressed:

1. How much of the released dataset is duplicated, and how much do the training and test sets overlap?
2. How large is the difference in measured detection performance between test images with and without near-duplicates in training, and how does performance respond when leakage is introduced under control?
3. Does performance measured under the published split carry over to images from other roads?

The contributions are as follows:

- An audit procedure for video-derived detection datasets that separates exact duplicates, export copies, perceptual near-duplicates and shared videos.
- A deduplicated, video-disjoint split of BharatPotHole with no shared videos, source frames or near-duplicate groups between subsets.
- Three complementary analyses: a same-model comparison of test subsets with and without near-duplicates, a similarity gradient and a controlled contamination test with a matched control.

- An external test on positive images from RDD2022 (India) and a reporting checklist for road-defect benchmarks.

Section 2 reviews related work. Section 3 describes the data, audit, protocols and evaluation. Section 4 reports the results, Section 5 discusses their implications and limitations, and Section 6 concludes.

2 Related work

2.1 Evaluation leakage in machine learning

Leakage occurs when information that would not be available at deployment reaches the model during training or evaluation [5]. Kapoor and Narayanan [6] listed non-independence between training and test data as one of the main forms of leakage in machine-learning-based science. When test samples are correlated with training samples, a model can score well by matching the test sample to a seen one, so the test error underestimates the error on new data [9, 10]. Overviews describe how such leakage enters supervised pipelines [7], how tool-driven ("push the button") workflows and transfer learning add to the risk [8], and which questions and checks can detect it before results are reported [11, 12]. Removing contaminated test samples is one remedy for train–test overlap [13].

The size of the effect has been measured by comparing a leaky evaluation with a corrected one. Gerz and Jelali [14] re-evaluated a tomato-disease detector whose images had been augmented before splitting; with augmentation applied after splitting, AP_{50} fell from 0.987 to 0.328. In a review of 64 biochar studies, 98% used random point-level splits, and re-analysis showed accuracy inflation of up to 86% [15]. Naive splits also produced over-optimistic results in hard-disk failure prediction [16]. For image benchmarks, Barz and Denzler [17] found near-duplicates of test images in the training sets of CIFAR-10 and CIFAR-100, and Recht et al. [18] showed that classifiers lose accuracy on newly collected test sets built with the original protocol. These studies show that a benchmark score depends on how the test set was built, not only on the model.

2.2 Duplicate and correlated imagery

Images can be related to training images in several distinct ways: as byte-identical files, as exported or augmented copies of one source frame, as perceptually similar images, or as frames from the same recording. Each relation needs a different check. Exact duplicates are found with cryptographic hashes. Near-duplicates are found with perceptual hashes, such as

the discrete-cosine-transform hash (pHash), which map similar images to codes with a small Hamming distance [19–21]. Perceptual hashes are fast and robust to compression and small photometric changes, but less robust than learned embeddings to geometric changes [22, 23]; combined schemes use the hash as a filter and a second measure for confirmation [24]. Thresholds need high specificity, because false matches grow with collection size [25].

Duplicates of these kinds have been found in public image collections, including across the training and test sets of the ISIC skin-lesion datasets [26], and perceptual hashing has been proposed as a routine step before splitting public imaging datasets [27]. Correlated images that are not duplicates raise the same problem. Splitting three-dimensional scans at the slice level raised the test accuracy of brain MRI classifiers by 29–55% [28], and similar effects were reported for Parkinson's disease MRI [29], sequential CT slices [30], consecutive optical coherence tomography images [31] and tiles of pathology slides [32]; feature selection guided by validation results is a further source [33]. Reviews of Alzheimer's disease studies found that more than half of the surveyed papers may have suffered from leakage [34], and that studies with high leakage risk reported 95–99% accuracy against 66–90% for studies with subject-wise splits [35].

The standard remedy is grouped partitioning, in which all samples from one subject, scene or recording are kept in one subset. Grouped and random splits gave clearly different results in EEG decoding [36], Wi-Fi channel-state action recognition [37, 38], keystroke dynamics [39], wearable activity recognition [40], repeated-measures biomechanics [41] and frailty classification [42]. The same holds for spatial and temporal data, such as neighbouring pixels in hyperspectral images [43, 44], photovoltaic forecasting [45] and spatiotemporal air-quality models [46].

2.3 Automotive and road-damage benchmarks

In automotive perception, Babu et al. measured train–test similarity in six driving datasets with perceptual hashing and deep features: 25% of KITTI and 5% of ZOD test images had a near-identical training image, while the other datasets had almost none [47]. They also proposed D-LeDe, a statistical test that flags leakage from the change in performance when similar images are removed [48, 49]. For a drone traffic dataset, Lepin et al. [50] showed that splits built from one video gave higher scores than splits that separate videos or camera views.

Road-damage datasets were first built from smartphone images [1] and extended to several countries, including India, in RDD2022 [2]. One-stage YOLO detectors are a common choice

for pothole and pavement-defect detection because they run in real time on embedded hardware [51, 52]. BharatPotHole adds dashboard video frames from Indian roads under varied lighting [3]. These studies report detector performance; they do not examine how the train–test split affects it, and a high score is not by itself evidence of leakage.

2.4 Research gap

Leakage from duplicated and correlated images is well documented in image classification and medical imaging, and train–test similarity has been measured for driving datasets. For BharatPotHole, however, neither the extent of duplication and train–test overlap nor the resulting difference in measured detection performance under alternative partitioning protocols has been reported. This study provides both, separates the leakage mechanisms involved, and does so at low computational cost.

3 Materials and methods

Table 1 lists the notation. The workflow has four steps (Fig. 1): audit the released data, build a video-disjoint split, train detectors under both splits, and evaluate them on internal and external test sets.

Table 1 Notation and abbreviations used throughout

Symbol	Definition
AP_{50}	Average precision at IoU ≥ 0.50, all-point interpolation
$AP_{50:95}$	Mean average precision over IoU thresholds 0.50 to 0.95 in steps of 0.05
ΔAP_{50}	Paired difference in AP_{50} between two test subsets for the same model (AP_{50} gap)
R	Recall at the confidence threshold that maximises F_1 (IoU ≥ 0.50)
d_H	Hamming distance between 64-bit perceptual hashes (pHash) of two images
ρ	Share of positive C-test images replaced by export copies of other frames
N_{img}, N_{pos}, N_{gt}	Number of images, positive images and ground-truth boxes
Protocol A	Training and evaluation on the published frame-level split
Protocol C	Training and evaluation on the deduplicated, video-disjoint split
A-test	Published test set (Protocol A)
Leaked	A-test images with a confirmed near-duplicate ($d_H \leq 4$, or $5 \leq d_H \leq 8$ with SSIM ≥ 0.80) or a shared source frame in training
B-clean	Near-duplicate-free subset: A-test images with no confirmed near-duplicate and no shared source frame in training; videos may be shared
B-strict	Stricter variant of B-clean that also excludes 66 images with confirmed near-duplicates in the validation or test sets (sensitivity check)
C-test	Test set of the video-disjoint split
Train copies	Export copies of training frames of the video-disjoint split
RDD-India	Pothole (D40) images of RDD2022, India subset (external test, positive images only)

3.1 Dataset

BharatPotHole [3] contains 7,074 images of 640 × 640 pixels with a single class, pothole, in YOLO format. The images are frames from dashboard videos recorded in India and exported through Roboflow. The release provides a training/validation/test split of 5,067/1,345/662 images (71.6/19.0/9.4%). File names encode a video identifier, a frame index and, for exported copies, a Roboflow hash. There are 162 distinct video identifiers.

3.2 Audit of duplicates and overlap

Four relations between images were defined and checked separately:

- **Exact duplicate:** two files with identical MD5 hashes.
- **Export copy:** two files with the same source-frame name after removal of the Roboflow hash suffix (".rf.<hash>"). Export copies may differ in pixels, because the export can apply augmentation.
- **Near-duplicate:** two images whose 64-bit pHash [19] values differ by a Hamming distance $d_H \leq 8$ and that pass a confirmation rule. pHash was computed from the full image after grey-scale conversion and resizing to 32 × 32 pixels, using the 8 × 8 lowest-frequency discrete-cosine-transform coefficients. Pairs with $d_H \leq 4$ were accepted directly; pairs with $5 \leq d_H \leq 8$ were accepted only when the structural similarity index (SSIM) [53] of 128 × 128 grey-scale thumbnails was at least 0.80.
- **Shared video:** two images with the same video identifier.

For each validation and test image, d_H to its nearest training image was recorded, and the image was flagged for each relation it had with at least one training image. A perceptual match is evidence of shared visual content, not proof that a model has memorised that image. In the published test set, 39.1% of images had a confirmed near-duplicate in training and 38.8% had a training image at $d_H \leq 4$. The criteria overlap strongly: all 242 test images (36.6%) that shared a source frame with a training image were also confirmed near-duplicates, 17 images were confirmed near-duplicates without a shared source frame, and only two were confirmed through the SSIM rule at $5 \leq d_H \leq 8$.

3.3 Deduplication, label cleaning and video-disjoint split

Fig. 1 and Table 2 summarise the steps. Exact duplicates (86 images) were removed first. For each source frame with several export copies, one representative was kept, namely the copy whose mean brightness was closest to the median of its copies; 2,769 copies were removed.

All removed files were positive images. The counts refer to files that share a source frame; they do not imply that every copy was produced by a separate augmentation.

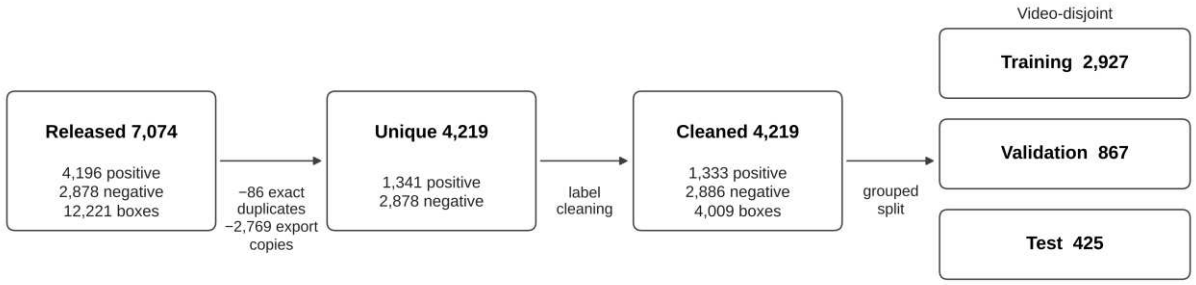

Fig. 1 Audit, deduplication and splitting workflow. Counts are images unless stated otherwise

Labels were then cleaned with the rules used by the evaluator for every protocol: polygon rows were converted to the enclosing box (minimum and maximum vertex coordinates), boxes were clipped to the image, boxes narrower or shorter than 2 pixels were removed, and identical boxes were merged. Eight images lost all their boxes in this step and became negative images, which is why the number of unique negative images (2,886) exceeds the released number (2,878). After cleaning, the 4,219 unique images hold 4,009 boxes.

Table 2 Composition of the released BharatPotHole dataset before and after deduplication

Quantity	Released	Unique
Images	7,074	4,219
Positive images	4,196	1,333
Negative images	2,878	2,886[a]
Annotated boxes	12,221	4,009[b]
Exact duplicates (MD5)	86	0
Augmentation copies	2,769	0

[a]Includes images whose only boxes were malformed or degenerate and were removed during label cleaning
[b]After removal of duplicate, malformed and degenerate (< 2 px) boxes
The unique images were split video-disjointly into 2,927 training, 867 validation and 425 test images

The unique images were grouped with a union–find structure that linked confirmed near-duplicates, copies of one source frame and frames of one video. Whole groups were then assigned to training, validation and test sets in a 70:20:10 ratio. Assignment used a randomised greedy search over 3,000 orderings that balanced the shares of images, positive images, boxes and small boxes ($\sqrt{\text{area}} < 32$ px). The resulting split has 2,927/867/425 images, and no video, source frame, MD5 hash or near-duplicate group is shared between subsets. This follows the subject-, scene- and video-level partitioning used in other domains [36, 43, 50]. Even a video-disjoint split cannot rule out every dependence, for example between videos recorded on the same road or with the same camera.

3.4 Evaluation protocols and test subsets

- **Protocol A** trains, validates and tests on the published split. Its test set is A-test.
- **Leaked** is the subset of A-test images that have a confirmed near-duplicate or a shared source frame in training (259 of 662 images).
- **B-clean** is the near-duplicate-free subset: the remaining 403 A-test images, with no confirmed near-duplicate and no shared source frame in training. It is source-frame-disjoint from training, but its images still come from videos used in training; it is therefore not free of all leakage.
- **B-strict** is a stricter variant of B-clean (337 images) used as a sensitivity check. It also excludes 66 positive images that have no near-duplicate in training but have confirmed near-duplicates in the validation set (38 images) or within the test set.
- **Protocol C** trains, validates and tests on the video-disjoint split. Its test set is C-test.

Comparing one Protocol A model on A-test and on B-clean measures the part of the gap that is linked to near-duplicates and shared source frames. The model, weights and training data are identical; only the test images differ. The comparison does not measure the effect of shared videos, which affect A-test and B-clean alike, and B-clean differs from A-test in composition as well as in leakage status. The difference is therefore reported as an observed gap, not as the total causal effect of leakage. A-test images come from 93 videos (53 with positive images), B-clean images from 79 (36) and C-test images from 10 (9). Only four A-test images come from videos not used for training.

Protocol A and Protocol C also differ in training data. The Protocol A training set has 5,067 images from 153 videos, with 3,062 positive images, 8,777 boxes and 1,157 unique positive source frames; the Protocol C training set has 2,927 images from 136 videos, with 934 positive images (all unique) and 2,836 boxes. Protocol A therefore trains on about 24% more unique positive frames, in addition to their copies. Comparisons between Protocol A and Protocol C models therefore combine the effects of leakage, training-set size and composition, and are interpreted descriptively.

3.5 Detectors and training

YOLO26n and YOLO26s [51] were initialised from the same COCO-pretrained weights in every run. All runs used Ultralytics 8.4.162 with the library's default recipe: 100 epochs, early-stopping patience of 30, input size 640 px, batch size 16, automatic optimiser selection

(AdamW, learning rate 0.002), mixed precision and default augmentation including mosaic. The random seed was 0. Training ran on one NVIDIA Tesla T4 GPU.

YOLO26n was trained under Protocol A (2.7 h). YOLO26n and YOLO26s were trained under Protocol C. The checkpoint with the highest validation fitness (the library's weighted combination of AP_{50} and $AP_{50:95}$) was used for testing. Protocol A used the released labels; Protocol C used the cleaned labels. During Protocol A training, the library skipped 21 images whose label files mixed polygon and box rows. This affects training only: the evaluator reads both formats and applies the same cleaning rules to every test image.

3.6 External test set

The India subset of RDD2022 [2] was used as an external test. The 1,530 annotated images (720 × 720 px) with pothole (D40) boxes were kept, giving 3,187 boxes; other damage classes were ignored. No RDD2022 image was within $d_H \leq 4$ of a BharatPotHole image, and 31 were within $d_H \leq 8$. All 1,530 images contain at least one pothole, so this set measures detection of potholes on other roads but not false alarms on clean road.

3.7 Metrics and statistical analysis

Predictions were generated at a confidence threshold of 0.001, a non-maximum-suppression IoU of 0.7 and at most 300 detections per image. At each IoU threshold t, detections were matched greedily to ground-truth boxes in order of confidence. Average precision was computed by all-point interpolation of the precision–recall curve,

$$\mathrm{AP}_t = \sum_{k=1}^{K} (r_k - r_{k-1}) \max_{\tilde{r} \geq r_k} p(\tilde{r}) \tag{1}$$

where r_k is recall after the k-th detection, p is precision and K is the number of detections. AP_{50} is AP_t at $t = 0.50$, and

$$\mathrm{AP}_{50:95} = \frac{1}{10} \sum_{t \in \{0.50, 0.55, \ldots, 0.95\}} \mathrm{AP}_t \tag{2}$$

Recall R is reported at the confidence threshold that maximises $F_1 = 2PR/(P + R)$ at IoU ≥ 0.50, where P is precision. One evaluator was used for every model and test set, so values are comparable across rows; they may differ slightly from the library's own validation output. AP_{50} is a detection metric and is reported as such; the word "accuracy" is not used for it.

The AP_{50} gap for one model is the paired difference

$$\Delta\mathrm{AP}_{50} = \mathrm{AP}_{50}(\text{A-test}) - \mathrm{AP}_{50}(\text{B-clean}) \tag{3}$$

and the relative gap is ΔAP_{50} divided by AP_{50}(B-clean). Absolute differences are reported with every relative value, because a relative value is large when the reference score is low.

Confidence intervals (95%) were obtained with a cluster bootstrap [54] in which videos, the unit of dependence, were drawn with replacement until the number of drawn videos equalled the number in the test set. All images of each drawn video were kept, and a video drawn twice contributed its images twice. AP was recomputed on each resample, and the 2.5th and 97.5th percentiles of the 1,000 resampled values gave the interval. Resamples without ground-truth boxes in a subset were discarded. For paired differences, both subsets were computed from the same resample. RDD2022 images were resampled individually because they carry no video identifiers. A check with 5,000 instead of 1,000 resamples, made for the A-test–B-strict difference on positive images, changed the interval from 0.098–0.283 to 0.092–0.274; no resample was discarded, and 0.02% of resamples had $\Delta AP_{50} \leq 0$. These intervals describe variation between test videos; they do not include variation from model initialisation or training, which was not measured.

A-test and B-clean differ in their share of positive images (56% and 31%). Because negative images add false positives and lower precision, the comparison is reported on positive images only, with all-image results as a complementary analysis.

3.8 Similarity gradient and controlled contamination test

For the similarity gradient, positive A-test images were grouped by d_H to the nearest training image (0–4, 5–8, 9–12, > 12), and AP_{50} of the Protocol A model was computed per group. As a sensitivity analysis, B-clean was redefined with pure distance thresholds $d_H > \tau$ for $\tau \in \{2, 4, 6, 8\}$, again excluding images with a shared source frame.

The contamination test uses the Protocol C models, which have not seen any C-test image or video. A share ρ of the 134 positive C-test images was replaced by export copies of training frames, drawn without replacement from a pool of 726 positive copies (one copy per training source frame). Each substituted image kept its own exported annotation, so ground truth always matches the image shown. Negative images were left unchanged, so the numbers of images and of positive images stay constant. The levels were $\rho \in \{0, 10, 20, 30, 40, 50, 75, 100\}$%, with 20 random draws at each level above 0.

Replacing test images also changes their content, box counts and box sizes. As a matched control, the same procedure was repeated with export copies of C-test frames, which have the same export processing but come from unseen videos (112 copies, one per source frame). If

the rise in AP_{50} came from the export processing or from easier images rather than from leakage, the control would show the same rise. The three pools are similar in box count and size: 2.84, 3.08 and 2.88 boxes per image, and 56.4%, 57.4% and 54.3% of boxes under 32 px, for the original positive C-test images, the training copies and the C-test copies. The test is a controlled stress test of contamination; it shows how far leakage can raise the score, not how much it raised the score in any published study.

3.9 Reproducibility

All steps ran in Google Colab. Per-image predictions were cached, so the analysis can be repeated without inference. Only three detectors were trained; all other results reuse their cached predictions. The released code covers the audit (MD5, pHash and SSIM checks with their thresholds), label cleaning, the split construction, training configurations, the evaluator and the bootstrap.

4 Results

4.1 Duplication in the released dataset

Of the 7,074 released images, 2,855 (40.4%) were exact duplicates (86) or export copies (2,769) (Table 2, Fig. 1). All 2,855 were positive images. The release therefore contains 4,196 positive images but only 1,333 unique positive frames, an average of 3.1 files per frame. After cleaning, the unique images hold 4,009 boxes, compared with 12,221 in the release.

This duplication has two consequences. First, the released positive-to-negative ratio (59:41) differs from that of the underlying frames (32:68). Second, copies of one frame can fall into different subsets of a random split. Duplicates of this kind were also reported for skin-lesion and brain-tumour image collections [26, 27].

4.2 Train–test overlap

In the published split, 38.1% of validation images and 38.8% of test images had a training image at $d_H \leq 4$, and about 35% shared a source frame with a training image (Table 3). Nearly all validation and test images (99.4%) came from videos that also supplied training images. The median d_H to the nearest training image was 10 for validation and 8 for test images.

Table 3 Train–test overlap in the published split and in the video-disjoint split

Subset	N_{img}	$d_H \leq 4$ (%)[a]	Shared source frame (%)	Shared video (%)	Median d_H[b]
Published split					
Validation	1,345	38.1	35.0	99.4	10
Test	662	38.8	36.6	99.4	8
Video-disjoint split					
Validation	867	0.0	0.0	0.0	12
Test	425	0.0	0.0	0.0	14

[a]Share of images with a training image at $d_H \leq 4$
[b]Median distance to the nearest training image

In the video-disjoint split, no validation or test image had a training image at $d_H \leq 4$, and no video or source frame was shared. The median distance rose to 12 and 14 (Table 3). Fig. 2 shows the full distributions: the published split has a concentration of held-out images at $d_H \leq 4$ that is absent from the video-disjoint split. The share of near-identical test images (38.8%) is higher than the 25% reported for KITTI and the 5% reported for ZOD [47].

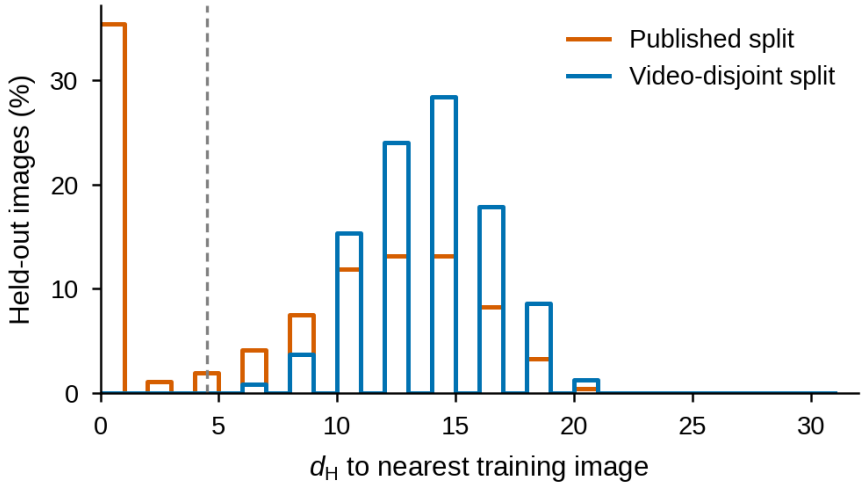

Fig. 2 Distribution of d_H from each held-out image (validation and test) to its nearest training image under the published split and under the video-disjoint split. The dashed line marks the near-duplicate threshold ($d_H \leq 4$)

Fig. 3 shows examples. At $d_H \leq 4$ the nearest training image is a near-copy of the test image, with the same potholes in the same positions; at larger distances it shows a different scene, and SSIM falls to 0.3–0.4. Further pairs and the export copies of one source frame are shown in Supplementary (Figs. S1 and S2).

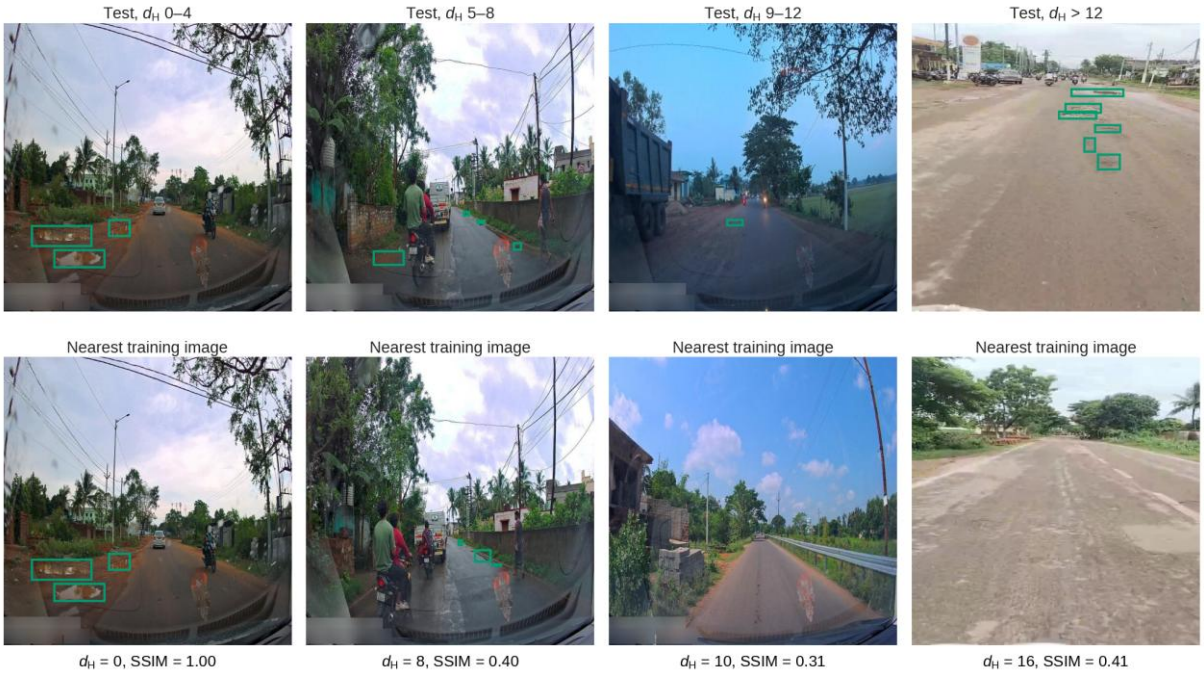

Fig. 3 One positive test image of the published split per d_H range (top) and its nearest training image (bottom). Green boxes: ground truth. d_H: Hamming distance between perceptual hashes; SSIM computed on 128 × 128 grey thumbnails

4.3 Same-model comparison of test subsets

The YOLO26n model trained under Protocol A reached AP_{50} = 0.615 on A-test with all images (Table 4). On positive images only, it reached 0.619 on A-test, 0.739 on the Leaked subset and 0.467 on B-clean (Table 5, Fig. 4a). The paired difference between A-test and B-clean was ΔAP_{50} = 0.152 (95% CI 0.090–0.234), a relative gap of 33%. The difference between Leaked and B-clean was 0.272 (95% CI 0.163–0.365), or 58%. Both intervals exclude zero.

Table 4 Detection performance by training protocol and test set (all images)

Test set	N_{img}	N_{pos}	N_{gt}	AP_{50} [95% CI]	$AP_{50:95}$ [95% CI]	R
YOLO26n, Protocol A						
A-test	662	371	1,148	0.615 [0.555, 0.690]	0.257 [0.223, 0.310]	0.551
B-clean	403	125	505	0.457 [0.389, 0.519]	0.147 [0.108, 0.177]	0.493
Leaked	259	246	643	0.739 [0.646, 0.817]	0.352 [0.292, 0.413]	0.661
RDD-India[b]	1,530	1,530	3,187	0.072 [0.062, 0.085]	0.021 [0.018, 0.025]	0.108
YOLO26n, Protocol C						
C-test	425	134	381	0.133 [0.006, 0.228]	0.042 [0.001, 0.073]	0.189
Train copies[a]	726	726	2,235	0.541	0.220	0.495
RDD-India[b]	1,530	1,530	3,187	0.099 [0.087, 0.113]	0.032 [0.028, 0.037]	0.157
YOLO26s, Protocol C						
C-test	425	134	381	0.135 [0.007, 0.229]	0.045 [0.002, 0.078]	0.171
Train copies[a]	726	726	2,235	0.471	0.178	0.468
RDD-India[b]	1,530	1,530	3,187	0.097 [0.086, 0.110]	0.031 [0.028, 0.037]	0.150

95% confidence intervals from 1,000 bootstrap resamples of videos (of images for RDD-India), percentile method. All models: input size 640 px, confidence threshold 0.001

[a]All 726 export copies of training frames at once; no confidence interval computed

[b]Contains positive images only

The all-image comparison gave a similar gap: ΔAP_{50} = 0.158 (95% CI 0.095–0.246), or 35% (Table 4). For $AP_{50:95}$ the absolute gap was 0.109 (95% CI 0.068–0.176) on positive images (0.259 vs 0.150) and 0.110 (0.070–0.175) on all images (0.257 vs 0.147), or 73% and 75% in relative terms; the larger relative values reflect the low $AP_{50:95}$ of B-clean as well as better box localisation on leaked images. Recall showed the same pattern (0.551 on A-test and 0.493 on B-clean, positive images). Example detections on both subsets are shown in Supplementary (Fig. S3).

The gap was stable across near-duplicate definitions. With pure distance thresholds, ΔAP_{50} was 0.154 (95% CI 0.093–0.238), 0.152 (0.090–0.234), 0.159 (0.099–0.245) and 0.171 (0.100–0.304) for τ = 2, 4, 6 and 8, or 33–38% in relative terms. These subsets contain 99–127 positive images from 31–36 videos. The stricter B-strict subset (59 positive images from 29 videos) gave a slightly larger gap, 0.174 (0.098–0.283), or 39%. The result therefore does not depend on the exact definition of the near-duplicate-free subset.

Table 5 AP_{50} gap between test subsets for the same detector (YOLO26n, Protocol A), positive test images only

Subset	N_{pos}	N_{gt}	AP_{50} [95% CI]	$AP_{50:95}$	R	ΔAP_{50} vs B-clean [95% CI]
By leakage status						
Leaked	246	643	0.739 [0.642, 0.819]	0.352	0.661	+0.272 [0.163, 0.365]
A-test	371	1,148	0.619 [0.552, 0.702]	0.259	0.551	+0.152 [0.090, 0.234]
B-clean	125	505	0.467 [0.392, 0.538]	0.150	0.493	Reference
B-strict[b]	59	201	0.445 [0.358, 0.545]	0.141	0.418	–
By d_H to the nearest training image						
0–4[a]	246	643	0.739 [0.642, 0.819]	0.352	0.661	–
5–8	26	68	0.592 [0.464, 0.741]	0.181	0.515	–
9–12	41	86	0.466 [0.296, 0.623]	0.154	0.454	–
>12	58	351	0.453 [0.297, 0.550]	0.152	0.470	–

Confidence intervals from 1,000 paired bootstrap resamples of videos, percentile method
[a]Same positive images as the Leaked subset; groups 5–8, 9–12 and >12 together form B-clean
[b]Sensitivity check; A-test minus B-strict: ΔAP_{50} = +0.174 [0.098, 0.283]

4.4 Detection performance and similarity to the training set

AP_{50} decreased as test images became less similar to their nearest training image (Table 5, Fig. 4b). It was 0.739 for d_H 0–4, 0.592 for 5–8, 0.466 for 9–12 and 0.453 for d_H > 12. Most of the decrease occurred below d_H = 9; beyond that, AP_{50} was stable. The 0–4 group contains the same positive images as the Leaked subset, and the other three groups together form B-clean. The intervals of adjacent groups overlap, so the gradient is supported by the trend rather than by any single pair of groups.

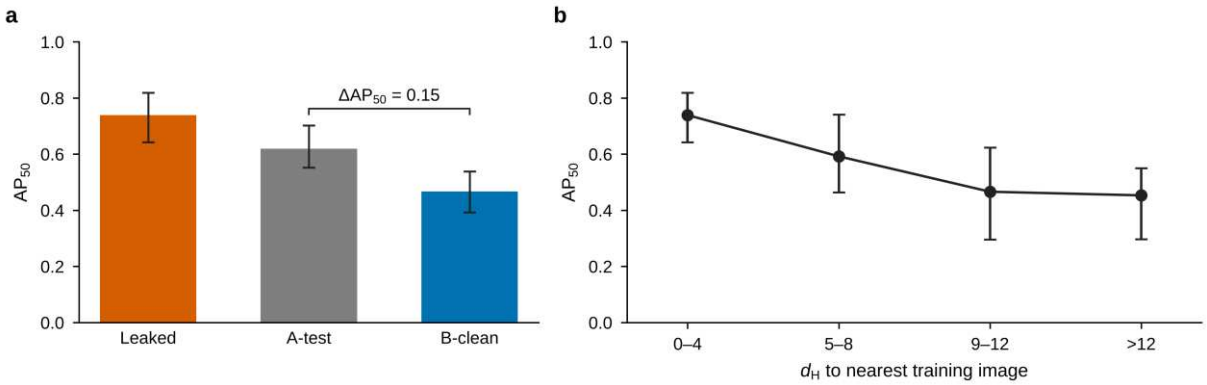

Fig. 4 Same-model comparison for YOLO26n trained under Protocol A, positive test images only. **a** AP_{50} on the Leaked subset, the full A-test and the near-duplicate-free B-clean subset; ΔAP_{50} is the paired difference between A-test and B-clean. **b** AP_{50} by d_H between each test image and its nearest training image. Error bars: 95% confidence intervals from 1,000 bootstrap resamples of videos

4.5 Controlled contamination test

On the unaltered C-test ($\rho = 0$), the Protocol C models reached $AP_{50} = 0.133$ (YOLO26n) and 0.135 (YOLO26s) (Table 6, Fig. 5). With copies of training frames, AP_{50} increased steadily with ρ: to 0.303 and 0.275 at $\rho = 50\%$, and to 0.494 ± 0.024 and 0.436 ± 0.030 at $\rho = 100\%$, or 3.7 and 3.2 times the value on the unaltered C-test. With copies of unseen C-test frames (examples in Supplementary , Fig. S4), AP_{50} changed little: 0.141 and 0.147 at $\rho = 50\%$, and 0.149 and 0.150 at $\rho = 100\%$. Because the two pools have similar box counts and sizes (Sect. 3.8), the rise is linked to the training origin of the substituted images, not to export processing or easier images. At $\rho = 100\%$, YOLO26n reached 80% of the 0.615 that the Protocol A model obtained on A-test, so leakage alone can produce most of the score reported under the published split.

Table 6 Controlled contamination test: AP_{50} of video-disjoint models on C-test with a share ρ of positive images replaced by copies of training frames or by copies of unseen C-test frames

ρ (%)	YOLO26n, training copies	YOLO26n, unseen copies	YOLO26s, training copies	YOLO26s, unseen copies
0	0.133	0.133	0.135	0.135
10	0.165 ± 0.015	0.137 ± 0.009	0.165 ± 0.015	0.136 ± 0.009
20	0.202 ± 0.020	0.136 ± 0.011	0.200 ± 0.024	0.139 ± 0.013
30	0.246 ± 0.028	0.142 ± 0.010	0.226 ± 0.025	0.141 ± 0.013
40	0.286 ± 0.025	0.140 ± 0.014	0.258 ± 0.031	0.145 ± 0.017
50	0.303 ± 0.031	0.141 ± 0.009	0.275 ± 0.035	0.147 ± 0.016
75	0.387 ± 0.033	0.147 ± 0.012	0.370 ± 0.028	0.147 ± 0.010
100	0.494 ± 0.024	0.149 ± 0.032	0.436 ± 0.030	0.150 ± 0.021

Values are mean ± standard deviation of AP_{50} over 20 random draws; $\rho = 0$ is the unaltered C-test. Training copies: export copies of frames used to train the model (pool of 726). Unseen copies: export copies of C-test frames (pool of 112)

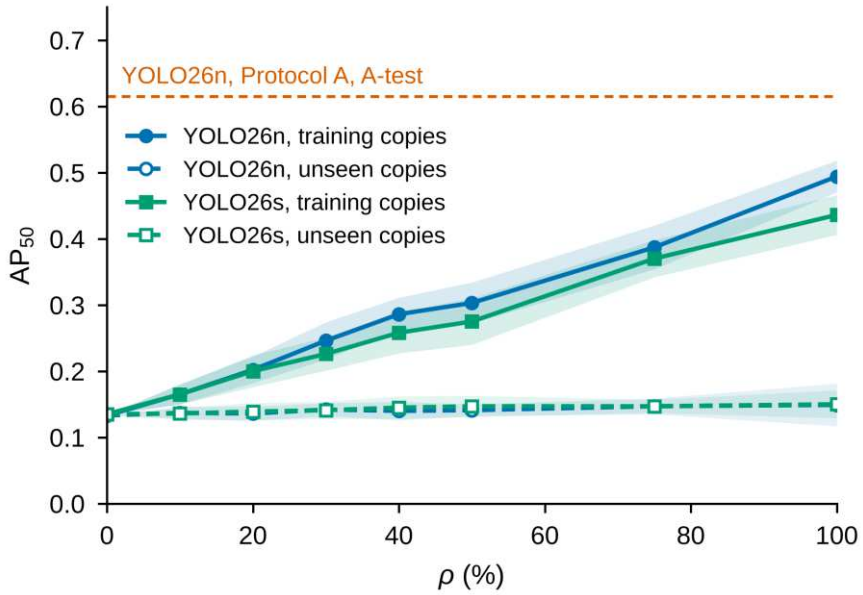

Fig. 5 Controlled contamination test. AP_{50} of the video-disjoint models (Protocol C) on C-test as a share ρ of positive test images is replaced by export copies of training frames (solid lines, filled markers) or of unseen C-test frames (dashed lines, open markers). Lines: mean of 20 random draws; bands: ±1 standard deviation. Horizontal line: YOLO26n trained and tested under Protocol A (AP_{50} = 0.615)

The C-test intervals are wide (AP_{50} 95% CI 0.006–0.228 for YOLO26n; Table 4). C-test contains 134 positive images from nine videos, one of which contributes 57, and AP_{50} varied strongly between videos.

4.6 External test

On RDD-India, all models scored low (Table 4, Fig. 6). The Protocol A model reached AP_{50} = 0.072 (95% CI 0.062–0.085). The Protocol C models reached 0.099 (0.087–0.113) and 0.097 (0.086–0.110). The Protocol A model was trained on 1.7 times as many images, yet its AP_{50} on these positive images was 0.025–0.027 lower, and its interval does not overlap those of the Protocol C models. The higher internal score under Protocol A therefore did not come with better detection on this external set. Because the two protocols also differ in training-set size and composition, and RDD-India differs from BharatPotHole in image characteristics, this result does not isolate the effect of leakage.

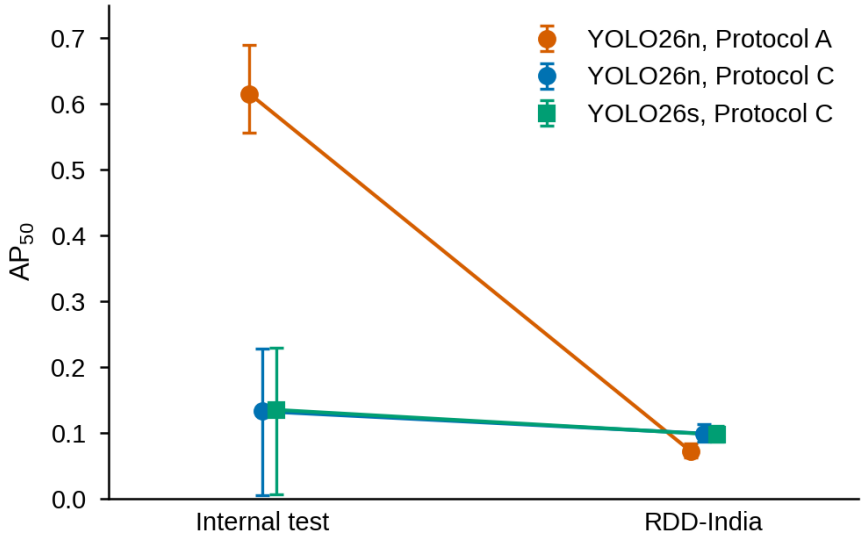

Fig. 6 AP_{50} on the internal test set (A-test for Protocol A, C-test for Protocol C) and on the positive images of RDD-India. Error bars: 95% bootstrap confidence intervals

5 Discussion

5.1 What the three analyses show

The three analyses differ in what they control. The same-model comparison holds the model fixed and shows that test images with a near-duplicate or shared source frame in training scored 0.272 AP_{50} higher than the near-duplicate-free subset. It is observational: the subsets also differ in composition. The similarity gradient shows that AP_{50} rises as test images approach their nearest training image. The contamination test manipulates the leaked share directly while holding the model, the negative images and the number of positive images fixed, and its matched control rules out export processing as the cause of the rise. It provides the most direct evidence that contamination raises AP_{50}, but as a stress test rather than an estimate for the published split. Taken together, the analyses indicate that near-duplicates in the published split raise the measured AP_{50}, and that the 0.152 gap between A-test and B-clean is a plausible estimate of that component.

The relative gap measured here (33% for the published test set, 58% for its leaked part) is of the same order as the inflation reported for other correlated image data, such as 29–55% for slice-level MRI splits [28], 5–30% for OCT images [31] and 12–24 percentage points for CT slices [30]. It is smaller than the 66-point AP_{50} drop reported for augmentation before splitting in plant-disease detection [14]. These values use different metrics and tasks and are not directly comparable, but they reflect the same mechanism: when test samples share a source with training samples, the score reflects memorisation as well as generalisation [6, 10].

The total effect of leakage in the published split is likely to be larger than the A-test–B-clean gap. B-clean images are not near-copies of training images, yet they come from the same drives, cameras, road surfaces and lighting. The Protocol A model scored 0.457 on B-clean (all images), against 0.133 for a Protocol C model on C-test. This difference is consistent with video-level dependence, as observed for subject-level overlap in other domains [37, 41, 42], but it also reflects the larger and different Protocol A training set, so it is not treated as a measured effect of leakage. Separating these factors would require a training-size-matched comparison of random and video-disjoint splits.

5.2 Why performance without leakage is low

AP_{50} on the video-disjoint test set was about 0.13, and AP_{50} on RDD-India was below 0.10. Diagnostics on the Protocol C YOLO26s model help to explain this. On its own training images it reached $AP_{50} = 0.47$, against 0.27 on validation, so even the training labels were hard to fit. Of 433 pairs of near-identical frames that both carried labels, 224 differed by two or more boxes, which indicates inconsistent annotation (examples in Supplementary , Fig. S5). AP_{50} on validation fell from 0.42 at IoU $\geq$ 0.1 to 0.03 at IoU $\geq$ 0.75, so box extents are ambiguous. Recall was 0.04 for boxes under 16 px and 0.65 for boxes of 96 px or more, and 56% of boxes in the unique set are under 32 px. AP_{50} across validation videos ranged from 0.02 to 0.63.

These factors limit any detector trained on this dataset. They do not change the overlap counts, but they do affect AP_{50} and can therefore change the size of the measured gaps. Because all subsets and protocols were scored with the same cleaned labels, the comparisons are internally consistent; with more consistent annotation the absolute values, and possibly the gaps, would differ.

5.3 Implications for practice

For researchers, a frame-level split of video data can give AP_{50} values that are tens of percent higher in relative terms than those on test images without near-duplicates in training. Comparisons between detectors on such splits may partly rank models by their ability to recognise seen scenes. Improvements of a few points in AP_{50}, as reported for new detector designs [52], are small compared with the gaps measured here. For road agencies, a reported score is a reasonable expectation of field performance only when the test roads were not used in training. Procurement specifications for automated inspection could require evaluation on routes withheld from training, preferably from another city or season, and on images without

defects to measure false alarms. External validation is also rare in other fields [35], which suggests that this requirement needs to be stated explicitly.

Table 7 summarises the checks recommended when a road-defect dataset or result is published. Each check uses information that dataset authors already hold, and several follow guidance from other fields [7, 11, 12, 26, 47].

Table 7 Reporting checklist for road-defect detection datasets and results

Item	Recommended practice
Unit of splitting	Split by video, route or capture session, never by frame
Export augmentation	Export unaugmented frames; apply augmentation only during training
Deduplication	Report exact (MD5) and near-duplicate (pHash with d_H threshold) counts before splitting
Overlap statistics	Report the share of test images with a near-duplicate, source frame or video in training
Class balance	Report unique positive and negative frames, not only file counts
Uncertainty	Report confidence intervals from a bootstrap over videos or routes, and repeat training with several seeds
External test	Report performance on at least one dataset from other roads, including images without defects
Split release	Release the split manifest with video identifiers so that results can be reproduced

5.4 Limitations

The study examines one dataset and two detectors of one family, each trained with one seed. The leakage mechanism does not depend on the detector, but the size of the gaps will vary with dataset, architecture and training procedure, and the values reported here should not be generalised to all road-defect detectors. Seed variance was not measured; the bootstrap over videos captures variation between test videos only. The same-model comparison was made for YOLO26n alone, and no training-size-matched comparison of Protocols A and C was made.

The perceptual-hash thresholds are a choice; the sensitivity analysis covers a range of thresholds, but pHash is less sensitive to geometric changes than learned embeddings [22, 23]. C-test covers few videos, so its confidence intervals are wide. RDD-India contains only positive images and differs from BharatPotHole in image size, blur and object scale (14.2% of its boxes are small, against 56.4% in BharatPotHole; examples in Supplementary , Fig. S6), so its results reflect domain shift as well as training differences. Finally, annotation inconsistencies were not corrected, and they lower all absolute values reported here.

6 Conclusions

This study measured how train–test leakage affects detection performance on a public road-defect benchmark. The main findings are as follows:

- Of 7,074 released BharatPotHole images, 86 were exact duplicates and 2,769 were export copies, and only 1,333 positive frames were unique.
- In the published split, 38.8% of test images had a near-duplicate in training, and 99.4% came from videos used in training.
- For the same detector, AP_{50} on positive images was 0.619 on the published test set and 0.467 on its near-duplicate-free subset, an absolute gap of 0.152 (95% CI 0.090–0.234) and a relative gap of 33%; the gap was 0.152–0.174 across five definitions of that subset.
- Substituting copies of training frames into an unseen test set raised the AP_{50} of a video-disjoint model from 0.133 to 0.494, while copies of unseen frames left it near 0.15.
- The model trained on the published split did not perform better on external positive images (AP_{50} 0.072 vs 0.097–0.099).

Scores reported on frame-level splits of video-derived road imagery should not be read as estimates of field performance. Splitting by video, deduplicating before splitting and testing on external roads are simple measures that make reported results more reliable. Future work should repeat the analysis with several seeds and detector families, match training-set sizes across protocols, extend the audit to other road-damage datasets, and quantify the effect of annotation consistency.

Declarations

Funding : No specific funding was received for this study.

Competing interests: The author declares no competing interests.

Ethics approval and consent to participate: Not applicable. The study uses publicly available road images and involves no human participants.

Consent for publication: Not applicable.

Data availability

BharatPotHole and RDD2022 are publicly available

BharatPotHole : https://www.kaggle.com/datasets/surbhisaswatimohanty/bharatpothole

RDD2022: https://www.kaggle.com/datasets/aliabdelmenam/rdd-2022

The audit tables and the cached predictions are available at https://github.com/pathmanaban86/bharatpothole-leakage

Author contributions P. Pathmanaban Pugazhendi: conceptualisation, methodology, software, formal analysis, investigation, writing – original draft, writing – review and editing.

References

1. Maeda H, Sekimoto Y, Seto T, Kashiyama T, Omata H (2018) Road damage detection and classification using deep neural networks with smartphone images. Comput-Aided Civ Infrastruct Eng 33(12):1127–1141. https://doi.org/10.1111/mice.12387
2. Arya D, Maeda H, Ghosh SK, Toshniwal D, Sekimoto Y (2024) RDD2022: a multi-national image dataset for automatic road damage detection. Geosci Data J 11(4):846–862. https://doi.org/10.1002/gdj3.260
3. Sahoo RR, Mohanty SS, Mishra S (2025) iWatchRoadv2: pothole detection, geospatial mapping, and intelligent road governance. arXiv:2510.16375. https://doi.org/10.48550/arXiv.2510.16375
4. Somvanshi S, Dutta A, Das S (2026) Benchmarking advanced tabular AI models for classifying SAE automation levels in crash data. Data Sci Transp 8(2):19. https://doi.org/10.1007/s42421-026-00162-8
5. Kaufman S, Rosset S, Perlich C, Stitelman O (2012) Leakage in data mining: formulation, detection, and avoidance. ACM Trans Knowl Discov Data 6(4):15. https://doi.org/10.1145/2382577.2382579
6. Kapoor S, Narayanan A (2023) Leakage and the reproducibility crisis in machine-learning-based science. Patterns 4(9):100804. https://doi.org/10.1016/j.patter.2023.100804
7. Sasse L, Nicolaisen-Sobesky E, Dukart J, Eickhoff SB, Götz M, Hamdan S, Komeyer V, Kulkarni A, Lahnakoski JM, Love BC, Raimondo F, Patil KR (2025) Overview of leakage scenarios in supervised machine learning. J Big Data 12(1):135. https://doi.org/10.1186/s40537-025-01193-8
8. Apicella A, Isgrò F, Prevete R (2025) Don't push the button! Exploring data leakage risks in machine learning and transfer learning. Artif Intell Rev 58(11):339. https://doi.org/10.1007/s10462-025-11326-3
9. Kernbach JM, Staartjes VE (2022) Foundations of machine learning-based clinical prediction modeling: part II—generalization and overfitting. Acta Neurochir Suppl 134:15–21. https://doi.org/10.1007/978-3-030-85292-4_3
10. Elangovan A, He J, Verspoor K (2021) Memorization vs. generalization: quantifying data leakage in NLP performance evaluation. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp 1325–1335. https://doi.org/10.18653/v1/2021.eacl-main.113
11. Bernett J, Blumenthal DB, Grimm DG, Haselbeck F, Joeres R, Kalinina OV, List M (2024) Guiding questions to avoid data leakage in biological machine learning applications. Nat Methods 21(8):1444–1453. https://doi.org/10.1038/s41592-024-02362-y

12. Albanchez JS (2026) Beyond accuracy: a reproducible framework for detecting data leakage and shortcut learning in machine learning pipelines toward trustworthy, reproducible, and sustainable AI systems. Int J Artif Intell Tools 2650021. https://doi.org/10.1142/S0218213026500211
13. Gupta M, Christopher J, Ramisetty K (2025) Mitigation strategies for data leakage in machine learning system design. In: Proceedings of the IEEE Region 10 Symposium (TENSYMP 2025). https://doi.org/10.1109/TENSYMP63728.2025.11144995
14. Gerz F, Jelali M (2026) Towards reliable data augmentation in machine learning: practices to prevent data leakage. Smart Agric Technol 15:102462. https://doi.org/10.1016/j.atech.2026.102462
15. Jang HJ, Ng W, Chen S, Setia R, Minasny B (2026) Illusions of accuracy: widespread data leakage distorts machine learning results in biochar research. GCB Bioenergy 18(9):e70173. https://doi.org/10.1111/gcbb.70173
16. Grillmeyer D, Hadry M, Lesch V, Borst V, Leppich R, Bauer A, Kounev S (2025) Quantifying data leakage in failure prediction tasks. In: Proceedings of the 16th ACM/SPEC International Conference on Performance Engineering (ICPE), pp 334–346. https://doi.org/10.1145/3676151.3719368
17. Barz B, Denzler J (2020) Do we train on test data? Purging CIFAR of near-duplicates. J Imaging 6(6):41. https://doi.org/10.3390/jimaging6060041
18. Recht B, Roelofs R, Schmidt L, Shankar V (2019) Do ImageNet classifiers generalize to ImageNet? In: Proceedings of the 36th International Conference on Machine Learning (ICML), PMLR 97:5389–5400
19. Zauner C (2010) Implementation and benchmarking of perceptual image hash functions. Master's thesis, Upper Austria University of Applied Sciences, Hagenberg
20. Thyagharajan KK, Kalaiarasi G (2021) A review on near-duplicate detection of images using computer vision techniques. Arch Comput Methods Eng 28(3):897–916. https://doi.org/10.1007/s11831-020-09400-w
21. Chum O, Philbin J, Zisserman A (2008) Near duplicate image detection: min-Hash and tf-idf weighting. In: Proceedings of the British Machine Vision Conference (BMVC). https://doi.org/10.5244/C.22.50
22. Mahmud MF, Nusrat Z, Pan WD (2026) Comparative evaluation of perceptual hashing and deep embedding methods for robust and efficient image deduplication. Electronics 15(7):1493. https://doi.org/10.3390/electronics15071493
23. Truong T, Jush FK, Lenga M (2024) Benchmarking pretrained vision embeddings for near- and duplicate detection in medical images. In: Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI). https://doi.org/10.1109/ISBI56570.2024.10635550
24. Jakhar Y, Borah MD (2025) Effective near-duplicate image detection using perceptual hashing and deep learning. Inf Process Manag 62(4):104086. https://doi.org/10.1016/j.ipm.2025.104086
25. Morra L, Lamberti F (2019) Benchmarking unsupervised near-duplicate image detection. Expert Syst Appl 135:313–326. https://doi.org/10.1016/j.eswa.2019.05.002
26. Cassidy B, Kendrick C, Brodzicki A, Jaworek-Korjakowska J, Yap MH (2022) Analysis of the ISIC image datasets: usage, benchmarks and recommendations. Med Image Anal 75:102305. https://doi.org/10.1016/j.media.2021.102305
27. Saifullah M, Baldauf D, Sakumura Y (2025) Reliable brain tumor classification without metadata: a step-by-step guideline with duplicate removal. In: Proceedings of the IEEE

International Conference on Tools with Artificial Intelligence (ICTAI), pp 1237–1244. https://doi.org/10.1109/ICTAI66417.2025.00180

28. Yagis E, Atnafu SW, García Seco de Herrera A, Marzi C, Scheda R, Giannelli M, Tessa C, Citi L, Diciotti S (2021) Effect of data leakage in brain MRI classification using 2D convolutional neural networks. Sci Rep 11:22544. https://doi.org/10.1038/s41598-021-01681-w

29. Veetil IK, Chowdary DE, Chowdary PN, Sowmya V, Gopalakrishnan EA (2024) An analysis of data leakage and generalizability in MRI based classification of Parkinson's disease using explainable 2D convolutional neural networks. Digit Signal Process 147:104407. https://doi.org/10.1016/j.dsp.2024.104407

30. Rojas F, Madanian S (2026) Bias in developing AI medical diagnosis: impact of sequential CT scan slices on data leakage in lung cancer. Inform Med Unlocked 65:101799. https://doi.org/10.1016/j.imu.2026.101799

31. Tampu IE, Eklund A, Haj-Hosseini N (2022) Inflation of test accuracy due to data leakage in deep learning-based classification of OCT images. Sci Data 9:580. https://doi.org/10.1038/s41597-022-01618-6

32. Bussola N, Marcolini A, Maggio V, Jurman G, Furlanello C (2021) AI slipping on tiles: data leakage in digital pathology. In: Pattern Recognition. ICPR International Workshops and Challenges. Lecture Notes in Computer Science, vol 12661. Springer, Cham, pp 167–182. https://doi.org/10.1007/978-3-030-68763-2_13

33. Samala RK, Chan H-P, Hadjiiski L, Koneru S (2020) Hazards of data leakage in machine learning: a study on classification of breast cancer using deep neural networks. In: Proc SPIE 11314, Medical Imaging 2020: Computer-Aided Diagnosis, 1131416. https://doi.org/10.1117/12.2549313

34. Wen J, Thibeau-Sutre E, Diaz-Melo M, Samper-González J, Routier A, Bottani S, Dormont D, Durrleman S, Burgos N, Colliot O (2020) Convolutional neural networks for classification of Alzheimer's disease: overview and reproducible evaluation. Med Image Anal 63:101694. https://doi.org/10.1016/j.media.2020.101694

35. Young VM, Gates S, Garcia LY, Salardini A (2025) Data leakage in deep learning for Alzheimer's disease diagnosis: a scoping review of methodological rigor and performance inflation. Diagnostics 15(18):2348. https://doi.org/10.3390/diagnostics15182348

36. Del Pup F, Zanola A, Tshimanga LF, Bertoldo A, Finos L, Atzori M (2025) The role of data partitioning on the performance of EEG-based deep learning models in supervised cross-subject analysis: a preliminary study. Comput Biol Med 196:110608. https://doi.org/10.1016/j.compbiomed.2025.110608

37. Varga D (2024) Exposing data leakage in Wi-Fi CSI-based human action recognition: a critical analysis. Inventions 9(4):90. https://doi.org/10.3390/inventions9040090

38. Varga D, Cao AQ (2025) Why partitioning matters: revealing overestimated performance in WiFi-CSI-based human action recognition. Signals 6(4):59. https://doi.org/10.3390/signals6040059

39. Ayotte B, Banavar MK, Hou D, Schuckers S (2021) Group leakage overestimates performance: a case study in keystroke dynamics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp 1410–1417. https://doi.org/10.1109/CVPRW53098.2021.00156

40. Rehman SU, Ali A, Khan AM, Okpala C (2024) Human activity recognition: a comparative study of validation methods and impact of feature extraction in wearable sensors. Algorithms 17(12):556. https://doi.org/10.3390/a17120556

41. Karbalaie A, Abtahi F, Häger CK (2026) Participant-aware model validation for repeated-measures data: comparative cross-validation study. JMIR AI 5:e87728. https://doi.org/10.2196/87728

42. Hughes CML, Zhang Y (2026) Bursting the bubble: data leakage and inflated deep learning accuracy in multivariate time-series frailty classification. Front Comput Sci 8:1700489. https://doi.org/10.3389/fcomp.2026.1700489

43. Nalepa J, Myller M, Kawulok M (2019) Validating hyperspectral image segmentation. IEEE Geosci Remote Sens Lett 16(8):1264–1268. https://doi.org/10.1109/LGRS.2019.2895697

44. Ahmad M, Mazzara M, Distefano S, Khan AM, Altuwaijri HA (2024) Improving generalization for hyperspectral image classification: the impact of disjoint sampling on deep models. Comput Mater Contin 81(1):503–532. https://doi.org/10.32604/cmc.2024.056318

45. Wedhasari T, Castro R (2026) A critical review and strategic roadmap of PV power forecasting (2016–2026): addressing temporal leakage and operational integration gaps. Energies 19(12):2937. https://doi.org/10.3390/en19122937

46. Just AC, Arfer KB, Rush J, Dorman M, Shtein A, Lyapustin A, Kloog I (2020) Advancing methodologies for applying machine learning and evaluating spatiotemporal models of fine particulate matter (PM2.5) using satellite data over large regions. Atmos Environ 239:117649. https://doi.org/10.1016/j.atmosenv.2020.117649

47. Babu MAA, Staron M, Bálint A, Durisic D, Pandey SK (2026) Evaluating train-test data leakage in automotive image datasets. In: Proceedings of the IEEE Intelligent Vehicles Symposium (IV), pp 201–207. https://doi.org/10.1109/IV66570.2026.11623865

48. Babu MAA, Pandey SK, Durisic D, Koppisetty AC, Staron M (2025) D-LeDe: a data leakage detection method for automotive perception systems. In: Proceedings of the International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS), pp 210–221. https://doi.org/10.5220/0013476700003941

49. Babu MAA, Staron M, Durisic D, Bálint A, Pandey SK (2026) Evaluating the robustness and generalizability of D-LeDe using multiple automotive datasets. SN Comput Sci 7(7):704. https://doi.org/10.1007/s42979-026-05289-7

50. Lepin F, Velastin SA, León R, García-Herrero J, Rojas-Martínez G, Espinosa-Oviedo JE (2026) UTUAV: a drone dataset for urban traffic analysis. Drones 10(1):15. https://doi.org/10.3390/drones10010015

51. Ultralytics (2025) Ultralytics YOLO, version 8.4.162 (YOLO26 models). https://github.com/ultralytics/ultralytics. Accessed 11 October 2026

52. Li Z, Zhang N, Ren Y, Li G, Ni H (2026) SHE-YOLO: real-time and low-cost road defect detection via semantic hypergraph enhancement. Eur Phys J Spec Top. https://doi.org/10.1140/epjs/s11734-026-02611-5

53. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612. https://doi.org/10.1109/TIP.2003.819861

54. Efron B, Tibshirani RJ (1993) An introduction to the bootstrap. Chapman & Hall, New York

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- Supplementary.docx