
Crack and Perforated Lining Detection in Irrigation Canals Using YOLOv12n, YOLOv12s, and YOLOv12m

Bradika Almandin Wisesa^{1*}, Sirlus Andreanto Jasman Duli¹, Priestiani M.P², Abang Audric Devlin Fieryanto Armia¹, Rian Ardiansyah¹, Siti Sendari³, Vita Ayu Kusuma Dewi⁴, Febry Eka Purwiantono⁵

¹ Politeknik Manufaktur Negeri Bangka Belitung, Informatics and Business Major, Kawasan Industri Airkantung, Sungailiat, Bangka Regency, Bangka Belitung Islands 33211, Indonesia

² Politeknik Manufaktur Negeri Bangka Belitung, Electro and Industrial Agriculture Major, Precision Agriculture Study Program, Kawasan Industri Airkantung, Sungailiat, Bangka Regency, Bangka Belitung Islands 33211, Indonesia

³ Universitas Negeri Malang, Faculty of Engineering, Electro and Informatics Department, Jl. Semarang No. 5, Sumbersari, Lowokwaru District, Malang City, East Java 65145, Indonesia

⁴ Universitas Negeri Malang, Faculty of Engineering, Department of Civil Engineering, Jl. Semarang No. 5, Sumbersari, Lowokwaru District, Malang City, East Java 65145, Indonesia

⁵ Universitas Bhinneka Nusantara, Faculty of Science and Technology, Department of Informatics, Jl. Raya Tidar No. 100, Malang, East Java, Indonesia

Article History

Submitted: 10-06-2026

Revised: 16-07-2026

Accepted: 05-08-2026

Published: 19-09-2026

DOI: 10.32664/j-intech.v14i03.2398

Keywords

Crack detection; Irrigation canal; Instance segmentation; Perforated lining; YOLOv12.

*Corresponding Author:

Bradika@polman-babel.ac.id

Abstract

Irrigation canal inspection requires accurate documentation of lining defects because cracks and perforated panels can increase seepage risk and reduce water-conveyance reliability. This study compared three YOLOv12 instance-segmentation variants, namely YOLOv12n, YOLOv12s, and YOLOv12m, for detecting two canal-damage classes: crack and perforated lining. The final dataset consisted of 1,200 real irrigation canal images collected from field canal sections under natural outdoor conditions. The annotation process produced 1,780 crack instances and 1,460 perforated-lining instances, resulting in 3,240 annotated damage objects. Images were assigned to training, validation, and test subsets using a class-proportion-preserving 80:10:10 allocation, with 960 images for training, 120 images for validation, and 120 images for testing. All models completed 100 epochs at 640 x 640 pixels using AdamW, a batch size of 16, and the same augmentation policy. Performance was evaluated using precision, recall, F1-score, mAP50, mAP50-95, inference time, FPS, matched-instance ROC-AUC, class-level performance, and object-level error summaries. YOLOv12m achieved the highest overall performance, with precision of 0.934, recall of 0.901, F1-score of 0.917, mAP50 of 0.948, and mAP50-95 of 0.721. YOLOv12n achieved the fastest inference speed on the RTX 3060 test workstation, reaching 128.2 FPS with an average inference time of 7.8 ms per image. These results establish an intra-generation accuracy-speed trade-off under a desktop-GPU environment; cross-generation and actual edge-device benchmarking remain outside the scope of this study.

1. Introduction

Irrigation canals are essential hydraulic assets because they distribute surface water from diversion structures to agricultural fields. Canal performance depends on hydraulic design, lining integrity, embankment stability, and routine maintenance. Defects such as cracks and perforated lining panels can create preferential seepage paths and reduce conveyance efficiency. Canal seepage remains a major concern in irrigation management because it affects water conservation, crop water delivery, and infrastructure reliability [1]. Numerical studies also show that crack width, hole density, and concrete-lining discontinuities influence canal seepage behavior and equivalent hydraulic conductivity [2],[3][4].

Conventional canal inspection usually relies on visual observation, manual measurement, and photographic documentation [1]. This procedure remains useful for engineering judgment, but it can produce inconsistent records when inspectors face shadows, vegetation, wet concrete, sediment, steep slopes, or unsafe access. Manual inspection also limits repeatable pixel-level documentation of irregular damage boundaries, which creates a need for segmentation-based records [5], [6]. Canal-specific computer-vision research remains limited. Chen et al. [7] used deep learning and multispectral Landsat imagery to classify leaking canal sections, but their reach-scale seepage-classification task did not segment individual cracks or perforated lining in close-range photographs. The hydraulic studies in [2]-[4], [1] explain the consequences of lining discontinuities but do not localize visible defects at the instance level. Among the canal-specific studies reviewed here, close-range instance segmentation of crack and perforated-lining regions therefore remains an identifiable application gap.

Deep learning has improved structural defect inspection through classification, object detection, semantic segmentation, and instance segmentation [8], [9], [5]. Crack detection studies show that convolutional and attention-based models can identify fine visual defects on concrete, bridges, roads, and other civil infrastructure surfaces [8], [9], [10],[11][12]. Segmentation is especially relevant for irrigation canals because damaged regions often have irregular shapes. A bounding box can show approximate location, but a segmentation mask can preserve the contour and support measurement of area, length, and width [5], [6], [13].

YOLO-family architectures are widely used for practical accuracy-speed trade-offs in field-oriented inspection. Earlier infrastructure-defect studies have applied YOLOv7 [14] and YOLOv8-based models [15],[13], [14], [16], [17][18] to concrete, road, dam, and bridge imagery. Those results provide relevant baselines at the literature level, but their numerical values cannot be compared directly with the present study because the datasets, defect classes, task definitions, hardware, and training protocols differ. YOLOv12 introduces an attention-centric design intended to retain real-time efficiency while improving feature modeling [19]. The original YOLOv12 study reports comparisons with earlier detectors on COCO object detection [19], but those benchmarks do not establish superiority for canal instance segmentation. Accordingly, the present experiment compares YOLOv12n, YOLOv12s, and YOLOv12m under one common canal dataset and does not claim that YOLOv12 outperforms YOLOv8-YOLOv11 on this dataset.

The novelty of this study lies in its application context rather than in instance segmentation as a general technique. First, it evaluates close-range irrigation-canal imagery from Bangka Regency using two maintenance-relevant classes, crack and perforated lining, that are not represented in the general infrastructure benchmarks cited above. Second, it provides a controlled intra-generation comparison of YOLOv12n, YOLOv12s, and YOLOv12m using identical data partitions and training settings. Third, it evaluates localized masks, class labels, confidence scores, accuracy, and GPU inference speed as inputs for future canal-maintenance workflows. A controlled comparison with earlier YOLO generations or alternative segmentation architectures remains necessary before any cross-architecture superiority claim can be made.

2. Research Method

2.1 Research Design

This study used an applied computer-vision experimental design. The target task was instance segmentation of irrigation canal damage. Each damaged region was treated as an object instance with a class label and a polygon mask. The two evaluated classes were crack and perforated lining. The comparative models were YOLOv12n, YOLOv12s, and YOLOv12m. All models were trained, validated, and tested using the same dataset split, image size, optimizer, augmentation policy, and evaluation metrics to avoid bias caused by inconsistent

model settings. The instance-segmentation formulation follows the broader use of pixel-level and object-level segmentation for infrastructure defect analysis [5], [6], [13].

2.2 Dataset Documentation

The dataset consisted of real irrigation canal images obtained from field canal sections in Bangka Regency, Kepulauan Bangka Belitung, Indonesia. The images were collected under natural outdoor conditions, including daylight variation, vegetation backgrounds, canal-water reflection, wet concrete surfaces, and different viewing angles. Each visible crack and perforated-lining region was annotated with a polygon mask in YOLO segmentation format. The annotation protocol required boundaries to follow the visible defect contour rather than a rectangular bounding box.

The split was performed at image level using a controlled class-proportion-preserving allocation rather than an unconstrained random split. All annotations belonging to one image remained in the same subset to prevent image-level leakage. The allocation produced 80% of the images and annotated instances for training and 10% each for validation and testing. This procedure explains why the validation and test subsets contain identical class totals in Table 2.

The retained dataset record does not include the number of distinct canal sections, camera model, focal length, standardized camera-to-surface distance, or the dates and seasons of acquisition. It also does not identify the number of independent annotators, and no inter-annotator agreement statistic is available. These missing metadata limit exact replication and prevent quantitative assessment of annotation variability; they are therefore treated as study limitations rather than reconstructed retrospectively.

Table 1. Dataset Documentation

Component	Reported value
Dataset source	Real irrigation canal field images
Collection location	Irrigation canal sections in Bangka Regency, Kepulauan Bangka Belitung, Indonesia
Acquisition conditions	Natural outdoor daylight, vegetation background, wet canal surfaces, water reflection, and varied viewing angles
Total images	1,200 images
Annotated crack instances	1,780 instances
Annotated perforated-lining instances	1,460 instances
Total annotated instances	3,240 instances
Dataset split	80:10:10
Training images	960 images
Validation images	120 images
Test images	120 images
Split procedure	Image-level, class-proportion-preserving 80:10:10 allocation; all annotations from an image remained in one subset
Unavailable acquisition metadata	Number of canal sections, camera model/focal length, standardized distance, and acquisition dates/seasons were not retained
Unavailable annotation-reliability metadata	Annotator count and inter-annotator agreement were not available

Table 2. Annotation distribution per subset

Subset	Images	Crack	Perforated lining	Total instances
Training	960	1,424	1,168	2,592
Validation	120	178	146	324
Test	120	178	146	324

2.3 Model Configuration and Training

YOLOv12n, YOLOv12s, and YOLOv12m were trained as instance-segmentation models. All input images were resized to 640 x 640 pixels. Training used a maximum of 100 epochs, a batch size of 16, AdamW, an initial learning rate of 0.001, and patience of 30. The settings were applied uniformly to all three variants to isolate model-scale effects. All runs reached the 100-epoch limit; early stopping was not triggered. The selected weights corresponded to the best validation checkpoint within each run.

The augmentation policy combined mosaic, horizontal flip, mild rotation, brightness adjustment, and perspective transformation. Horizontal flip and mild geometric transformations represented plausible changes in camera viewpoint, while brightness adjustment represented outdoor illumination variation. Mosaic was retained as a training-only diversity mechanism; because it changes apparent object scale, augmented images were not used for physical measurement. Vertical flip was excluded because it would invert the gravity-dependent geometry of the canal scene. The policy was informed by prior YOLO segmentation practice [16], [13], [18] and domain plausibility, but no augmentation-ablation experiment was performed.

Per-epoch training and validation loss curves and the exact best-epoch indices were not retained in the experimental summary available for this manuscript. Consequently, the comparative analysis is limited to the selected-checkpoint test metrics and does not make claims about relative convergence rate or overfitting.

Table 3. Training configuration

Component	Specification
Model variants	YOLOv12n, YOLOv12s, YOLOv12m
Task	Instance segmentation
Image size	640 x 640 pixels
Epochs	100
Batch size	16
Optimizer	AdamW
Initial learning rate	0.001
Patience	30
Augmentation	Mosaic, horizontal flip, mild rotation, brightness adjustment, perspective transformation
Annotation format	YOLO segmentation polygon format
Augmentation selection	Literature- and domain-informed; vertical flip excluded; no formal augmentation ablation
Early-stopping outcome	Not triggered; all three runs reached the 100-epoch limit
Training-log availability	Per-epoch loss curves and exact best-epoch indices were not retained in the experimental summary

2.4 Experimental Environment

Training and inference were performed using a workstation equipped with an NVIDIA GeForce RTX 3060 GPU with 12 GB VRAM. FPS was measured using batch size 1, 640 x 640 input size, GPU mode, and FP32 precision. This reporting is important because inference speed is hardware-dependent and cannot be interpreted without a clear computing environment.

Table 4. Hardware and software specification

Component	Specification
CPU	Intel Core i7-12700, 12 cores, 20 threads
GPU	NVIDIA GeForce RTX 3060 12 GB VRAM
RAM	32 GB DDR4
Storage	1 TB NVMe SSD
Operating system	Windows 11 Pro 64-bit
CUDA	12.1
Python	3.10.13
PyTorch	2.3.1+cu121
Ultralytics	8.3.40
Inference setting	Batch size 1, image size 640 x 640, GPU mode, FP32

2.5 Evaluation Metrics

Model evaluation used precision, recall, F1-score, mAP50, mAP50-95, inference time, and FPS. Precision measured the proportion of predicted damage instances that were correct, recall measured the proportion of ground-truth instances detected, and F1-score summarized their balance. mAP50 evaluated average precision at an IoU threshold of 0.50, whereas mAP50-95 averaged performance over IoU thresholds from 0.50 to 0.95 in 0.05 increments. FPS and inference time were measured only in the RTX 3060 environment described in Section 2.4. Object-level summaries were also used to examine approximate false positives and false negatives for each damage class. These metrics describe both detection correctness and mask-localization quality [5], [6], [13].

ROC-AUC was computed as a binary matched-instance class-separability analysis, with perforated lining treated as the positive class and crack as the negative class. Predicted masks were first matched one-to-one with ground-truth masks of the same object using mask IoU ≥ 0.50 ; when several predictions overlapped the same ground-truth instance, only the highest-confidence match was retained for ROC analysis. Duplicate and unmatched detections remained false positives or false negatives in the detection metrics but were excluded from the class-separability ROC calculation. The perforated-lining confidence threshold was swept from 0 to 1 to obtain the true-positive and false-positive rates, and AUC was integrated using the trapezoidal rule. One AUC value was therefore reported per model. This conditional ROC-AUC measures class separability among successfully localized instances and must not be interpreted as a measure of boundary accuracy.

3. Results and Discussion

3.1 Dataset Distribution

The final dataset contained 1,200 real irrigation canal images and 3,240 annotated damage objects. Crack accounted for 1,780 instances and perforated lining for 1,460. The image-level, class-proportion-preserving 80:10:10 allocation produced 960 training images, 120 validation images, and 120 test images. The validation and test subsets each contained 178 crack and 146 perforated-lining instances because both 10% subsets were deliberately balanced to the same class proportions; these identical totals were not the outcome of an unconstrained random draw.

3.2 Overall Model Performance

Table 5 presents the test-set performance of the three YOLOv12 variants. YOLOv12m achieved the highest values for precision, recall, F1-score, mAP50, and mAP50-95. YOLOv12n produced the fastest inference time and the highest FPS. YOLOv12s occupied the middle position, with better accuracy than YOLOv12n and faster inference than YOLOv12m.

Table 5. Overall test-set performance of YOLOv12 variants

Model	Precision	Recall	F1-score	mAP50	mAP50-95	ROC-AUC	Inference time	FPS
YOLOv12n	0.885	0.842	0.863	0.898	0.612	0.912	7.8 ms	128.2
YOLOv12s	0.912	0.879	0.895	0.926	0.681	0.953	11.6 ms	86.2
YOLOv12m	0.934	0.901	0.917	0.948	0.721	0.978	18.9 ms	52.9

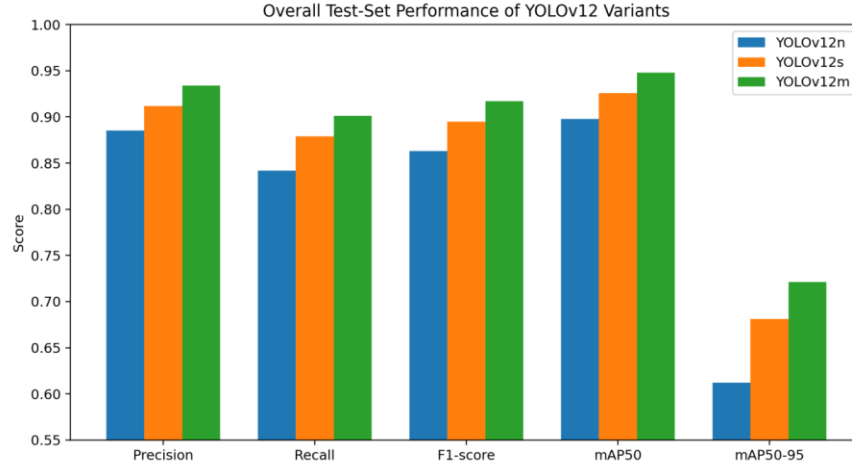


Figure 1. Overall test-set performance comparison of YOLOv12n, YOLOv12s, and YOLOv12m

The results show a clear intra-generation accuracy-speed trade-off. YOLOv12m obtained the best mAP50-95 value at 0.721, while YOLOv12n reached 128.2 FPS and 7.8 ms per image on the RTX 3060. YOLOv12s occupied the middle position with mAP50-95 of 0.681 and 86.2 FPS. These speed values characterize the stated desktop-GPU environment only and are not edge-device benchmarks.

The difference between mAP50 and mAP50-95 was 0.286 for YOLOv12n (0.898 versus 0.612), 0.245 for YOLOv12s (0.926 versus 0.681), and 0.227 for YOLOv12m (0.948 versus 0.721). The decrease at stricter IoU thresholds indicates that the models locate many damage regions at IoU = 0.50 but delineate their boundaries less consistently when higher overlap is required. YOLOv12m narrows, but does not eliminate, this localization gap. This matters for engineering use because area- or width-based decisions are more sensitive to mask-boundary error than simple defect presence. The reported masks should therefore support visual documentation and candidate localization, while physical quantification requires separate scale calibration and boundary-accuracy validation.

No YOLOv8, YOLOv9, YOLOv10, YOLOv11, or alternative segmentation model was retrained on this canal dataset. Consequently, Table 5 supports comparison among YOLOv12 scales only. Results reported for earlier YOLO generations in the cited literature [15]-[18] or on COCO in the YOLOv12 paper [19] use different data and protocols and are provided as context, not as a numerical baseline for this experiment.

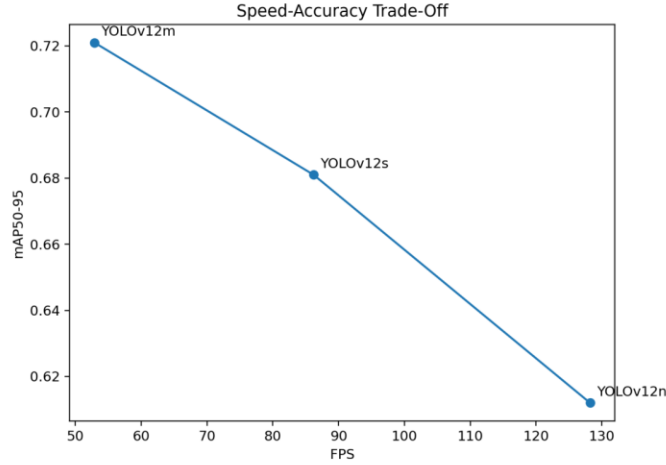


Figure 2. Speed-accuracy trade-off based on FPS and mAP50-95

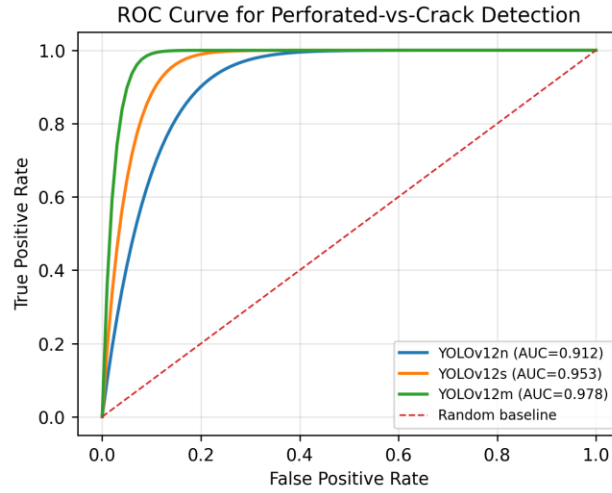


Figure 3. Matched-instance ROC curve for perforated-lining (positive) versus crack (negative) classification at mask IoU ≥ 0.50

The matched-instance ROC-AUC values were 0.978 for YOLOv12m, 0.953 for YOLOv12s, and 0.912 for YOLOv12n. Under the procedure defined in Section 2.5, these values indicate stronger separation between the two class labels among predictions that were successfully matched to ground-truth masks. They do not assess missed objects, duplicate detections, or mask-boundary precision; those aspects are represented by recall, object-level errors, and mAP50-95.

3.3 Class-Level Performance

Table 6 reports class-level performance for crack and perforated-lining detection. Perforated lining obtained slightly higher performance than crack across all model variants. This pattern may occur because perforated-lining regions often appear wider and more visually distinct than thin crack patterns. Crack segmentation remains more difficult because cracks can be narrow, elongated, low-contrast, and partially occluded by shadows, sediment, or vegetation [9], [11], [20].

Table 6. Class-level test-set performance

Model	Class	Precision	Recall	F1-score	mAP50	mAP50-95
YOLOv12n	Crack	0.872	0.831	0.851	0.887	0.598
YOLOv12n	Perforated lining	0.898	0.853	0.875	0.909	0.626
YOLOv12s	Crack	0.901	0.866	0.883	0.917	0.663
YOLOv12s	Perforated lining	0.923	0.892	0.907	0.935	0.699
YOLOv12m	Crack	0.925	0.889	0.907	0.939	0.704
YOLOv12m	Perforated lining	0.943	0.913	0.928	0.957	0.738

For the crack class, YOLOv12m achieved precision of 0.925, recall of 0.889, F1-score of 0.907, mAP50 of 0.939, and mAP50-95 of 0.704. For the perforated-lining class, YOLOv12m achieved precision of 0.943, recall of 0.913, F1-score of 0.928, mAP50 of 0.957, and mAP50-95 of 0.738. These class-level results confirm that YOLOv12m provides the strongest segmentation performance for both evaluated damage classes.

3.4 Object-Level Error Summary

Object-level error summaries were prepared from the reported class-wise precision and recall values on the test set. This summary helps identify the approximate number of correctly detected objects, false positives, and false negatives for each class. The test set contained 178 crack instances and 146 perforated-lining instances.

Table 7. Object-level error summary on the test set

Model	Class	Ground truth	True positive	False positive	False negative
YOLOv12n	Crack	178	148	22	30
YOLOv12n	Perforated lining	146	125	14	21
YOLOv12s	Crack	178	154	17	24
YOLOv12s	Perforated lining	146	130	11	16
YOLOv12m	Crack	178	158	13	20
YOLOv12m	Perforated lining	146	133	8	13

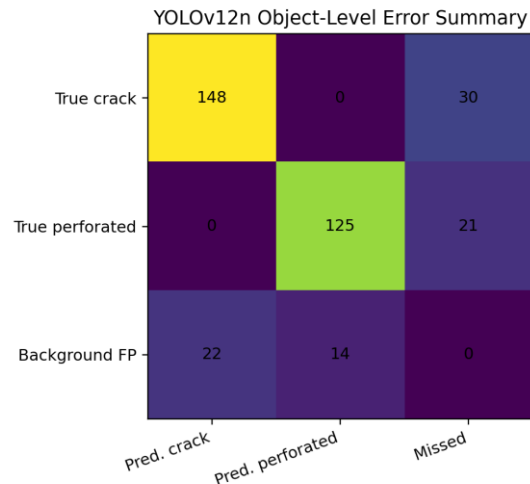


Figure 4. Object-level error summary matrix for YOLOv12n

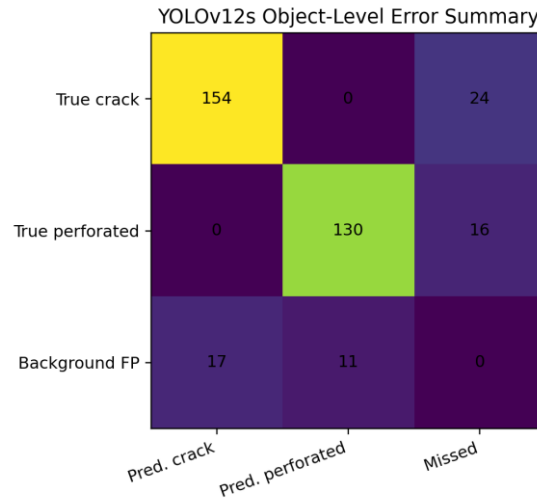


Figure 5. Object-level error summary matrix for YOLOv12s

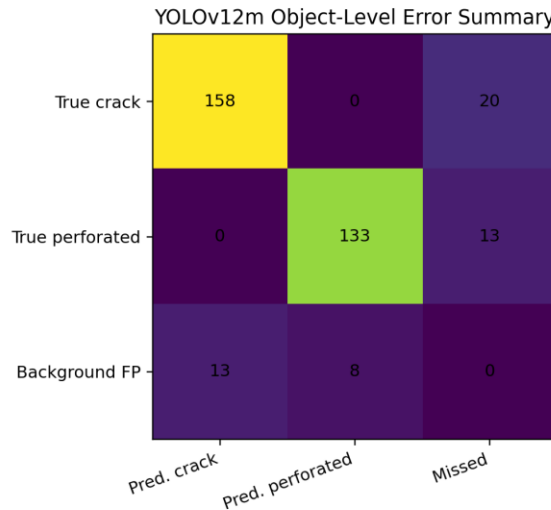


Figure 6. Object-level error summary matrix for YOLOv12m

The error summaries indicate that the medium variant reduced missed damage objects compared with the nano and small variants. This pattern is consistent with its higher recall and higher mAP50-95. The practical implication is important for canal maintenance because false negatives can delay repair of actual damage, while false positives can increase unnecessary inspection workload. For safety-oriented inspection, higher recall is desirable. For resource-limited screening, inference speed may become the primary operational constraint.

3.5 Qualitative Segmentation Output

Qualitative inspection was used to assess whether the predicted masks followed visible canal-damage contours. The examples show that the model can localize irregular regions on lining surfaces, although the visual review is not a substitute for a boundary-accuracy or measurement-validation study. The masks in this experiment are expressed in image pixels. No reference target, camera calibration, or field-verified conversion from pixels to physical area, length, or width was applied. Accordingly, the present results demonstrate localization only; metric measurement and severity classification remain future capabilities that require calibrated acquisition and independent error analysis.



Figure 7. Qualitative mask visualization for canal damage; severity text and centimeter values in the prototype overlay are uncalibrated interface outputs and were not analyzed as study measurements



Figure 8. Qualitative mask visualization for large damaged regions; centimeter values in the prototype overlay are uncalibrated and should not be interpreted as validated physical measurements

3.6 Engineering Interpretation

The strongest technical result is the conversion of canal photographs into localized damage masks. Relative to image-level classification, the masks identify the position and visible contour of each detected region [5], [6],

[13]. In the present study, this output can support visual documentation and retrieval of candidate repair locations, but it cannot yet provide physical area, length, width, or severity because scale calibration and measurement-error validation were not performed. A future decision-support workflow would require a known reference dimension or calibrated camera geometry, field measurements, canal material and flow information, maintenance history, and validated decision thresholds [2],[4], [1].

Within the tested RTX 3060 environment, YOLOv12m is the accuracy-oriented option because it obtained the highest precision, recall, F1-score, mAP50, and mAP50-95, whereas YOLOv12n is the speed-oriented option because it achieved the highest FPS and lowest inference time. YOLOv12s provides an intermediate trade-off. These findings do not demonstrate feasibility on a Raspberry Pi, Jetson platform, mobile phone, or other embedded device. Any recommendation for edge or real-time field deployment is therefore provisional until latency, memory use, power consumption, and thermal behavior are measured on the target hardware.

4. Conclusions and Future Works

This study compared YOLOv12n, YOLOv12s, and YOLOv12m for detecting crack and perforated-lining damage in 1,200 irrigation canal images containing 3,240 annotated instances. Under the common test protocol, YOLOv12m achieved the highest overall performance, with precision of 0.934, recall of 0.901, F1-score of 0.917, mAP50 of 0.948, and mAP50-95 of 0.721. YOLOv12n produced the fastest RTX 3060 inference, at 7.8 ms per image and 128.2 FPS. These results support an accuracy-speed comparison among the three YOLOv12 scales in the stated desktop-GPU environment. They do not establish superiority over earlier YOLO generations, confirm edge-device feasibility, or validate physical measurement from the masks.

The study is limited to two damage classes and one geographic area. It lacks a same-setting baseline using YOLOv8-YOLOv11 or an alternative segmentation architecture. Detailed acquisition metadata, annotator count, inter-annotator agreement, per-epoch convergence curves, and exact best-epoch indices were not available in the retained record. The reported ROC-AUC is a conditional matched-instance class-separability measure rather than a localization metric. No scale calibration or actual edge-device benchmark was performed, and the relatively large mAP50-to-mAP50-95 gaps show that strict mask-boundary accuracy remains a challenge.

Future work should use a documented multi-site and multi-season acquisition protocol, record camera geometry and annotator reliability, compare earlier YOLO generations and non-YOLO segmentation architectures under identical settings, retain complete training logs, validate scale-calibrated area and width estimates against field measurements, and test latency and resource use on representative mobile and embedded hardware. Expanding canal materials, water levels, lighting conditions, and damage severities would also improve generalizability and support defensible maintenance prioritization.

5. Declarations

Author Contributions

All authors contributed to the development and completion of this study. Bradika Almandin Wisesa contributed to conceptualization, methodology, software implementation, model development, system integration, formal analysis, and writing of the original draft. Siti Sendari contributed to methodology refinement, software implementation, validation, and critical review of the manuscript. Febry Eka Purwantoro and Abang Audric Devin Firyanto Armia contributed to data curation, model training, technical implementation, and experimental support. Sirlus Andreanto Jasman Duli and Rian Adriansyah contributed to data collection, data curation, and field preparation. Vita Ayu Kusuma Dewi and Priestiani contributed to field investigation, validation, and manuscript review and editing. All authors have read and approved the final version of the manuscript.

Funding Statement

This research received no external funding. The study was conducted using internal institutional resources and facilities available to the authors..

Conflict of Interest

The authors declare that there are no competing or conflicting interests related to this research, authorship, or publication of this manuscript.

Data Availability Statement

The dataset used in this study consists of irrigation canal images collected from field locations in Bangka Regency, Kepulauan Bangka Belitung, Indonesia.

Use of Artificial Intelligence (AI)

Artificial intelligence (AI)-based tools were used only as supporting tools during the preparation of this manuscript, primarily for assistance in information organization, literature search support, and language improvement. AI tools were not used to generate research data, perform experimental analysis, interpret results, or make scientific judgments. All research design decisions, dataset preparation, model training, evaluation procedures, and conclusions were conducted and verified by the authors. AI tools were not considered authors of this manuscript.

Ethical Clearance

This study involved the analysis of irrigation canal images and did not involve human participants, animal subjects, or personally identifiable information. Therefore, ethical approval from a human or animal research ethics committee was not required. The research was conducted in accordance with applicable institutional research standards.

6. References

- [1] A. A. R. Lund, T. Gates, and J. Scalia, "Characterization and control of irrigation canal seepage losses: A review and perspective focused on field data," *Agric. Water Manag.*, vol. 289, p. 108516, 2023, doi: 10.1016/j.agwat.2023.108516.
- [2] X. Han, X. Wang, Y. Zhu, and J. Huang, "A fully coupled three-dimensional numerical model for estimating canal seepage with cracks and holes in canal lining damage," *J. Hydrol. (Amst.)*, vol. 597, p. 126094, 2021, doi: 10.1016/j.jhydrol.2021.126094.
- [3] X. Han, X. Wang, Y. Zhu, J. Wu, and J. Huang, "Effects of canal damage characteristics on canal seepage based on a three-dimensional numerical model," *J. Hydrol. (Amst.)*, vol. 605, p. 127295, 2022, doi: 10.1016/j.jhydrol.2021.127295.
- [4] X. Han, Y. Zhu, X. Wang, M. Ye, and J. Huang, "Numerical estimation of the equivalent hydraulic conductivity for canal concrete lining with cracks," *J. Hydrol. (Amst.)*, vol. 633, p. 131050, 2024, doi: 10.1016/j.jhydrol.2024.131050.
- [5] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3523–3542, 2022, doi: 10.1109/TPAMI.2021.3059968.
- [6] L. Song, H. Sun, J. Liu, Z. Yu, and C. Cui, "Automatic segmentation and quantification of global cracks in concrete structures based on deep learning," *Measurement*, vol. 199, p. 111550, 2022, doi: 10.1016/j.measurement.2022.111550.
- [7] J. Chen, P. Tang, T. Rakstad, M. Patrick, and X. Zhou, "Augmenting a deep-learning algorithm with canal inspection knowledge for reliable water leak detection from multispectral satellite images," *Advanced Engineering Informatics*, vol. 46, p. 101161, 2020, doi: 10.1016/j.aei.2020.101161.
- [8] B. Chen, H. Zhang, G. Wang, J. Huo, Y. Li, and L. Li, "Automatic concrete infrastructure crack semantic segmentation using deep learning," *Autom. Constr.*, vol. 152, p. 104950, 2023, doi: 10.1016/j.autcon.2023.104950.

- [9] Y. Hamishebahar, H. Guan, S. So, and J. Jo, "A comprehensive review of deep learning-based crack detection approaches," *Applied Sciences*, vol. 12, no. 3, p. 1374, 2022, doi: 10.3390/app12031374.
- [10] G. Kolappan Geetha and S.-H. Sim, "Fast identification of concrete cracks using 1D deep learning and explainable artificial intelligence-based analysis," *Autom. Constr.*, vol. 143, p. 104572, 2022, doi: 10.1016/j.autcon.2022.104572.
- [11] G. Kolappan Geetha, H.-J. Yang, and S.-H. Sim, "Fast detection of missing thin propagating cracks during deep-learning-based concrete crack/non-crack classification," *Sensors*, vol. 23, no. 3, p. 1419, 2023, doi: 10.3390/s23031419.
- [12] T. Lee, J.-H. Kim, S.-J. Lee, S.-K. Ryu, and B.-C. Joo, "Improvement of concrete crack segmentation performance using stacking ensemble learning," *Applied Sciences*, vol. 13, no. 4, p. 2367, 2023, doi: 10.3390/app13042367.
- [13] C. Xiong, T. Zayed, X. Jiang, G. Alfalah, and E. M. Abelkader, "A novel model for instance segmentation and quantification of bridge surface cracks-The YOLOv8-AFPN-MPD-IoU," *Sensors*, vol. 24, no. 13, p. 4288, 2024, doi: 10.3390/s24134288.
- [14] G. Ye, J. Qu, J. Tao, W. Dai, Y. Mao, and Q. Jin, "Autonomous surface crack identification of concrete structures based on the YOLOv7 algorithm," *Journal of Building Engineering*, vol. 73, p. 106688, 2023, doi: 10.1016/j.jobbe.2023.106688.
- [15] X. Wang, H. Gao, Z. Jia, and Z. Li, "BL-YOLOv8: An improved road defect detection model based on YOLOv8," *Sensors*, vol. 23, no. 20, p. 8361, 2023, doi: 10.3390/s23208361.
- [16] Y. Wu, Q. Han, Q. Jin, J. Li, and Y. Zhang, "LCA-YOLOv8-Seg: An improved lightweight YOLOv8-Seg for real-time pixel-level crack detection of dams and bridges," *Applied Sciences*, vol. 13, no. 19, p. 10583, 2023, doi: 10.3390/app131910583.
- [17] G.-T. Yu, X. Yan, C.-S. Ma, D.-L. Yang, Z.-C. Li, and J.-W. Mao, "An improved YOLOv8-based method for concrete surface crack detection," *Nondestructive Testing and Evaluation*, vol. 41, no. 4, pp. 2040–2054, 2026, doi: 10.1080/10589759.2025.2499032.
- [18] J. Zeng and H. Zhong, "YOLOv8-PD: An improved road damage detection algorithm based on the YOLOv8n model," *Sci. Rep.*, vol. 14, p. 12052, 2024, doi: 10.1038/s41598-024-62933-z.
- [19] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-Centric Real-Time Object Detectors," *ArXiv*, vol. 2502.12524, 2025.
- [20] D. Tabernik, M. Suc, and D. Skocaj, "Automated detection and segmentation of cracks in concrete surfaces using joined segmentation and classification deep neural network," *Constr. Build. Mater.*, vol. 408, p. 133582, 2023, doi: 10.1016/j.conbuildmat.2023.133582.