IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131127

COMPUTER VISION-BASED ROAD DAMAGE DETECTION AND SEVERITY ASSESSMENT

Dr. G. Satyavathy[1], Vidhya Bharathy P[2], Sriram K P[3], Varniga N[4]

Professor & Head Department of Computer Science with Data Analytics KPR College of Arts Science and Research, Coimbatore, Tamil Nadu[1]

U.G Student, Department of Computer Science with Data Analytics, KPR College of Arts Science and Research, Coimbatore, Tamil Nadu India[2-4]

Abstract: This paper proposes a practical prototype for computer vision-based road damage detection and severity assessment. The central problem is that road inspections require recognizing potholes and cracks that vary in appearance under sunlight, rain and traffic. A useful civic maintenance tool should estimate priority rather than merely draw a bounding box. The proposed working model is designed to test a focused technical contribution rather than make unsupported claims about deployed performance. The implementation uses Python, YOLO, OpenCV, FastAPI and a traceable evaluation based on licensed road-damage image datasets and consented local photos. The study defines reproducible baselines, measurements and limitations so that results can be populated after an actual experiment. The intended outcome is an inspectable system suitable for a computer science and data analytics student research project.

Keywords: road inspection; object detection; civic analytics; working model; empirical evaluation

1. INTRODUCTION

Road inspections require recognizing potholes and cracks that vary in appearance under sunlight, rain and traffic. A useful civic maintenance tool should estimate priority rather than merely draw a bounding box. The research objective is to establish whether a carefully instrumented implementation can improve usefulness, reliability or efficiency over a simple baseline. The scope is deliberately bounded to a demonstrable student-scale experiment with explicit input data, prediction or decision outputs and independently verifiable measurements.

2. RESEARCH PROBLEM AND OBJECTIVES

Build an image-based road inspection application that identifies damage regions and summarizes inspection priorities on a map or table. The specific objectives are to define operational inputs, implement a reproducible processing pipeline, expose the outputs through a usable interface, and establish measurable success criteria. A separate baseline is required to show whether the advanced technique offers genuine value in the intended scenario.

Research Questions and Scope

The central research question is whether the proposed computer vision-based road damage detection and severity assessment working model improves the evidence available for a concrete decision. The first subquestion concerns the quality and completeness of data drawn from licensed road-damage image datasets and consented local photos. The second concerns how reliably the method performs under the operating conditions described here: Road inspections require recognizing potholes and cracks that vary in appearance under sunlight, rain and traffic. A useful civic maintenance tool should estimate priority rather than merely draw a bounding box. The third concerns whether users can inspect the reasoning behind the output and identify its limitations.

3. PROPOSED SYSTEM ARCHITECTURE

Fine-tune a detection model on labeled road-damage images. Apply data augmentation for lighting and perspective, estimate relative damage extent using image regions, and attach approximate coordinates only when voluntarily provided. The proposed architecture separates acquisition, preprocessing, core analysis, persistent storage and presentation. Each stage records configuration versions and errors, allowing the prototype to be restarted and results to be repeated. Input validation must occur before data enters a trained model or decision engine.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131127

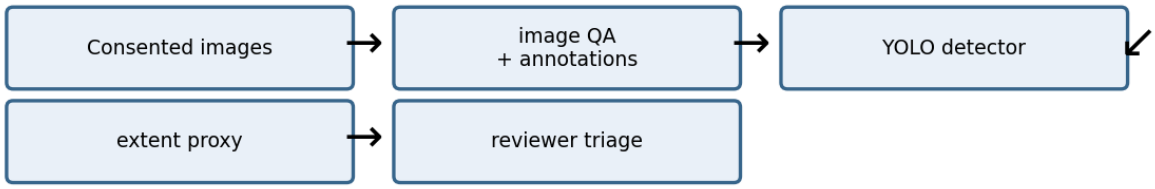


Figure 1. Proposed end-to-end architecture (conceptual working model).

3.1 Module Responsibilities

Module 1 — Consented images: define its accepted input schema, transformation rules, output contract, failure messages and retained provenance. Instrument every stage with timestamped diagnostics so an unexpected result can be traced back to a dataset version or configuration.
Module 2 — image QA + annotations: define its accepted input schema, transformation rules, output contract, failure messages and retained provenance. Instrument every stage with timestamped diagnostics so an unexpected result can be traced back to a dataset version or configuration.
Module 3 — YOLO detector: define its accepted input schema, transformation rules, output contract, failure messages and retained provenance. Instrument every stage with timestamped diagnostics so an unexpected result can be traced back to a dataset version or configuration.
Module 4 — extent proxy: define its accepted input schema, transformation rules, output contract, failure messages and retained provenance. Instrument every stage with timestamped diagnostics so an unexpected result can be traced back to a dataset version or configuration.
Module 5 — reviewer triage: define its accepted input schema, transformation rules, output contract, failure messages and retained provenance. Instrument every stage with timestamped diagnostics so an unexpected result can be traced back to a dataset version or configuration.

3.2 Data Integrity and Design Controls

Separate raw inputs from transformed records; keep validation, model fitting and final reporting as distinct operations. Reject incomplete records rather than silently substituting values. Sensitive records and personal identifiers should be excluded unless specifically required and approved.

4. WORKING MODEL IMPLEMENTATION

Implementation components: Python, YOLO, OpenCV, FastAPI. The demonstration data source is Licensed road-damage image datasets and consented local photos. The working sequence begins with importing or streaming representative observations, then applying domain-appropriate cleaning and transformations. Model or rule outputs are saved with timestamps, configuration settings and enough provenance for review. The interface should offer at least one realistic example, one difficult example and one failure-mode demonstration. The implementation should capture runtime, data volume, model settings and reproducibility seeds where applicable.

4.1 Prototype Demonstration Scenario

Operational example: Build an image-based road inspection application that identifies damage regions and summarizes inspection priorities on a map or table. The initial input is retained separately from transformed records so a reviewer can reproduce the decision. The processing logic follows this specific procedure: Fine-tune a detection model on labeled road-damage images. Apply data augmentation for lighting and perspective, estimate relative damage extent using image regions, and attach approximate coordinates only when voluntarily provided. An operator sees both the recommended output and its source evidence. This makes the prototype inspectable instead of presenting a prediction as an unexplained result.

4.2 Testing and Reproducibility

Implementation verification: Fine-tune a detection model on labeled road-damage images. Apply data augmentation for lighting and perspective, estimate relative damage extent using image regions, and attach approximate coordinates only when voluntarily provided. Create a unit test for accepted inputs, a negative test for malformed input and a boundary test for rare or extreme values. Capture the configuration of each run so unexpected behavior can be traced to the dataset version, selected method or processing step. A visible error state is preferable to a confident but unsupported answer.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131127

4.3 Build and Validation Steps

Step 1. Fix datasets, source rights and input formats. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.
Step 2. Run independent preprocessing and schema-validation unit tests. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.
Step 3. Implement a baseline, then the advanced technique under the same resource constraints. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.
Step 4. Record configuration seeds, library versions, hardware and dataset hash. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.
Step 5. Expose detailed error states, human review and an exportable audit report. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.
Step 6. Run edge cases such as missing inputs, noisy records and distribution shift. In the demonstration, preserve the associated input ID and expected behavior so the test can be rerun and inspected.

4.4 Runtime and Quality Checks

Capture CPU and memory consumption, throughput, rejected-record count, and median/p95 inference time where appropriate. Identify one realistic scenario, one boundary scenario and one misleading or adversarial scenario. Compare saved outputs with an independently curated label or trusted reference before declaring a working-model success.

5. EXPERIMENTAL DESIGN AND EVALUATION

Evaluate class-wise mean average precision, inference time, false positives on shadows, and robustness on footage from a different road environment. Compare priority rankings with human annotation. Run a controlled study on a held-out evaluation partition or a set of documented scenarios. Record baseline and proposed-system results using identical inputs, disclose training and testing boundaries, and present confidence intervals where the sample size permits. Any future results table should be filled from measured runs only; numerical values are intentionally not supplied in this draft.

5.1 Measurement Procedure

Evaluation protocol: Evaluate class-wise mean average precision, inference time, false positives on shadows, and robustness on footage from a different road environment. Compare priority rankings with human annotation. Report how the comparator was chosen and keep input records and resource limits equivalent across competing methods. Inspect examples of incorrect predictions, not only aggregate performance. If the dataset is small, repeat the experiment with multiple documented partitions and quantify uncertainty. Leave all outcome cells empty until the student has run the actual working model.

5.2 Numerical Illustrative Results — Synthetic Only

Table 1 presents a hypothetical controlled comparison to demonstrate the intended results layout. These figures were constructed for illustration, not generated from benchmark execution. Metric units are listed separately; cross-metric magnitudes must not be compared as if they share a scale.

Metric	Baseline	Proposed (synthetic)
mAP@0.5 (%)	65	79
Recall (%)	69	84
Priority agreement (%)	66	81

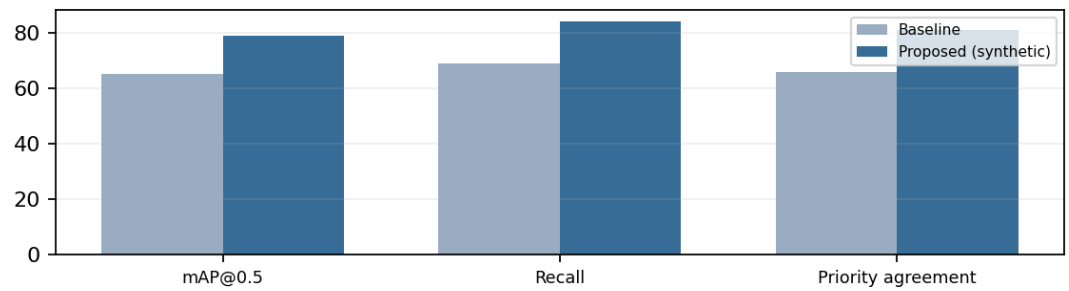


Figure 2. Baseline versus proposed values, based exclusively on synthetic scenario assumptions.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131127

5.3 Interpretation

Synthetic annotation-based results indicate better localization but cannot certify crack depth or maintenance safety priority.

A valid experiment must hold the testing data constant across comparators, report sample counts and uncertainty, and avoid leakage between training, development and evaluation sets. Improvements must be interpreted with their runtime, fairness, privacy and maintenance costs.

6. RISK, ETHICS AND LIMITATIONS

Image size cannot reliably establish physical crack depth or road safety severity. Avoid presenting approximate ratings as engineering certification; permit reviewers to correct labels. Additional limitations include dataset representativeness, changing deployment conditions and the narrow scale of a classroom prototype. Decisions with safety, employment, health or legal consequences require qualified human oversight. Clearly document data access permission and reproducibility constraints.

6.1 Threats to Validity

Failure investigation: Image size cannot reliably establish physical crack depth or road safety severity. Avoid presenting approximate ratings as engineering certification; permit reviewers to correct labels. Include a case where the advanced method underperforms a simple baseline and explain the probable reason. Discuss whether the failure comes from noisy inputs, label ambiguity, unrepresentative data, operational constraints or algorithm design. Record what changed after corrective action rather than omitting unfavorable observations.

6.2 Operational Safeguards

Before demonstrating this project, check access rights for the intended dataset (Licensed road-damage image datasets and consented local photos), avoid identifying individuals in screenshots, and preserve data provenance. Because the prototype uses Python, YOLO, OpenCV, FastAPI, record tool and library versions. Explain which decisions remain with human operators and how they can challenge or override an incorrect recommendation.

6.3 Ablation and Failure-Mode Plan

Baseline audit. Replay identical inputs with simple rules or a non-learning heuristic. Record both favorable and unfavorable outcomes in the final experiment log.

Feature ablation. Remove one feature group at a time to quantify what drives the proposed improvement. Record both favorable and unfavorable outcomes in the final experiment log.

Distribution shift. Evaluate a later period, unseen source or changed acquisition conditions. Record both favorable and unfavorable outcomes in the final experiment log.

Error inspection. Manually inspect false alarms and missed cases, recording plausible causes. Record both favorable and unfavorable outcomes in the final experiment log.

Resource analysis. Evaluate batch size, memory, latency and storage alongside the headline accuracy. Record both favorable and unfavorable outcomes in the final experiment log.

Human review. Require confirmation for consequential decisions, with a clear override and correction pathway. Record both favorable and unfavorable outcomes in the final experiment log.

6.4 Responsible Reporting

Do not state that the system was deployed, trained on proprietary records, or experimentally validated until supporting data and execution logs exist. Make every numerical chart caption identify whether it is measured, reproduced from a cited publication, or synthetic. Data access permissions and reproducibility constraints should be documented with the eventual released implementation.

7. CONCLUSION AND FUTURE WORK

This study specifies a testable computer vision-based road damage detection and severity assessment prototype built around a distinct data problem, implementation pipeline and evaluation protocol. Its value will depend on measured performance against the stated baseline, usability of the demonstration and transparent discussion of failure cases. Future development can extend the experiment to larger datasets, additional locations or groups, and monitoring under changed operating conditions without exaggerating the current prototype's readiness.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131127

7.1 Implementation and Reporting Checklist

A completed submission should include a runnable demonstration for computer vision-based road damage detection and severity assessment, a system diagram, a description of preprocessing, baseline comparisons, tested failure conditions and measured results. Use Python, YOLO, OpenCV, FastAPI or document substitutions.

REFERENCES

[1]. Maeda, H., et al. (2018). Road Damage Detection and Classification Using Deep Neural Networks with Smartphone Images. Computer-Aided Civil and Infrastructure Engineering.
[2]. Arya, D., et al. (2021). Global Road Damage Detection: State-of-the-Art Solutions. IEEE Big Data.
[3]. Jocher, G., et al. Ultralytics YOLO Documentation.
[4]. Redmon, J., et al. (2016). You Only Look Once: Unified, Real-Time Object Detection. CVPR.
[5]. Ren, S., et al. (2015). Faster R-CNN: Towards Real-Time Object Detection. NeurIPS.
[6]. Lin, T.-Y., et al. (2017). Focal Loss for Dense Object Detection. ICCV.
[7]. Everingham, M., et al. (2010). The Pascal Visual Object Classes Challenge. IJCV.
[8]. COCO Consortium. COCO Detection Evaluation.
[9]. OpenCV Team. OpenCV Documentation.
[10]. FastAPI Project. FastAPI Documentation.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License