International Journal of Science, Strategic Management and Technology
Volume **02** Issue **08** August- **2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

RoadGuard: An Automated Road Damage Detection and Assessment System Using YOLOv8

Subhrajeet Ghosh[1], Anurag Das[1], Souradeep Roy[1], Tathagata Roy Chowdhury[2],Supantha Mondal[2]

1. Department of Computer Science and Engineering, Techno Institute of Engineering and Management, Formerly Known as Techno Engineering College, Banipur, India

2.,Mentor, Department of Computer Science and Engineering,
Techno Institute of Engineering and Management, Formerly Known as Techno Engineering College, Banipur, India

Corresponding Author Email: anuragbabla2@gmail.com

https://doi.org/10.55041/ijsmt.v2i8.046

Cite this Article: Ghosh, S., Das, A., Roy, S. & Chowdhury, T. R. (2026). RoadGuard: An Automated Road Damage Detection and Assessment System Using YOLOv8. International Journal of Science, Strategic Management and Technology, *02*(8), 1-9. https://doi.org/10.55041/ijsmt.v2i8.046

License: This article is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting use, distribution, and reproduction in any medium, provided the original author(s) and source are properly credited.

Abstract—Road surface defects such as potholes and cracks can reduce driving safety and increase maintenance costs when they are not identified early. This paper presents RoadGuard, an automated road-damage detection and assessment prototype that combines YOLOv8-based object detection with interpretable severity and repair-priority analysis. The system accepts a road image through a Flask web interface, stores and previews the selected file, performs YOLOv8 inference, and processes each detected object using separate severity and priority modules. Severity is estimated from the relative bounding-box area and damage type, while repair priority combines severity, damage importance, and detection confidence. In a representative end-to-end test, the system detected one pothole with 84.2% confidence and two cracks with 38.4% and 27.2% confidence. Their computed severity scores were 26.03, 25.06, and 6.83, respectively, while the corresponding repair-priority scores were 53.25, 38.31, and 25.67. The prototype demonstrates how object-detection outputs can be transformed into maintenance-oriented information through a lightweight web application. The reported confidence values are per-detection inference scores, not overall model accuracy. The system is intended as a decision-support prototype and can be extended with segmentation, depth estimation, GPS mapping, larger benchmark evaluation, and field calibration.

Index Terms—Road damage detection; YOLOv8; computer vision; pothole detection; crack detection; severity assessment; repair priority; Flask; intelligent transportation systems

I. INTRODUCTION

Road infrastructure is a critical component of safe mobility, logistics, economic activity, and public services. Surface defects such as potholes and cracks develop through traffic loading, moisture infiltration, material deterioration, and environmental exposure. If these defects are not identified and addressed in time, they may affect ride quality, vehicle safety, and maintenance planning. The increasing availability of road imagery creates an opportunity to automate part of this inspection process using computer vision.

Traditional road-condition surveys rely heavily on field personnel and repeated visual inspection. Although expert inspection remains necessary for engineering decisions, manual image review is time-consuming and difficult to scale across large road networks. Recent datasets and benchmark challenges have demonstrated that deep-learning detectors can localize several types of road damage from ordinary images [1], [2].

RoadGuard is developed as a lightweight decision-support prototype that connects visual detection with interpretable maintenance information. The system uses YOLOv8 for object localization and classification, then applies separate severity and repair-priority modules to each detected object. A Flask web application provides the user interface for uploading road images, previewing the selected file, running analysis, and presenting the results.

The main contributions of this work are:

1. An end-to-end image-to-decision road-damage workflow.
2. Integration of YOLOv8 inference with a browser-based Flask application.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

3. An explainable severity calculation based on relative detected area and damage type.
4. An interpretable repair-priority calculation.
5. An actual prototype demonstration using a representative road image and documented output screenshots.

II. PROBLEM STATEMENT AND OBJECTIVES

A road-damage detector normally answers two questions: what type of visible defect is present and where it occurs in the image. For maintenance planning, however, an additional question is required: which detected defect should receive attention first? A detection label alone does not provide this operational prioritization. RoadGuard therefore treats object detection as the first stage of a broader decision-support pipeline.

A. Objectives

The objectives are to detect potholes, cracks, and manholes from road images; obtain confidence values and bounding-box information from YOLOv8; estimate the relative visual area occupied by each detected object; calculate an interpretable severity score from 0 to 100; calculate a repair-priority score; classify the severity and priority into human-readable levels; and present the results through a simple web interface.

The prototype is deliberately positioned as a visual decision-support system rather than an engineering measurement tool. Its severity score is a proxy derived from image geometry and damage type. It does not directly measure physical depth, volume, structural integrity, or pavement strength.

III. LITERATURE REVIEW

Recent road-damage research has moved from manual surveys toward image-based computer vision, supported by increasingly diverse benchmark datasets. RDD2022 contains 47,420 road images from six countries and more than 55,000 annotated damage instances, while CRDDC'2022 established a common evaluation setting for international road-damage detection [1], [2]. UAV-based datasets have expanded viewpoints and pavement conditions, including UAV-PDD2023 with more than 11,150 distress instances [3]. Studies using YOLOX, YOLOv5, YOLOv7, and convolutional networks demonstrate that one-stage detectors can reduce inspection effort while retaining practical inference speed [4]–[7]. These studies also show sensitivity to image quality, damage scale, background complexity, and geographic variation.

From 2023 onward, research increasingly focused on lightweight architectures, attention mechanisms, feature fusion, and improved localization. LE-YOLOv5 targeted efficient road-damage detection for resource-constrained deployment, while DenseSPH-YOLOv5 combined enhanced feature extraction with a transformer prediction head for multi-scale detection [8], [9]. Other work explored multi-distress classification and localization, showing the value of combining classification with object detection for pavement management [10]. YOLOv8 became prominent as a practical training and inference framework [11]. RDD-YOLO, YOLOv8-PD, and improved YOLOv8s models reported gains through attention, feature extraction, and lightweight modifications [12]–[14]. Crack-specific research confirms that low-resolution defects, complex backgrounds, and elongated crack structures remain difficult to detect reliably [15].

Recent studies address a limitation shared by many 2-D detectors: confidence scores do not directly represent physical severity. Mask R-CNN approaches improve localization, while YOLOv8 combined with point-cloud or depth information can estimate geometric properties such as area and depth [16], [17]. Other work investigates improved YOLOv8 crack detection and specialized detection strategies for practical deployment [18]. Research in 2026 further explores UAV-based detection and real-world datasets containing potholes, cracks, and manholes [19], [20]. RoadGuard emphasizes a simple, reproducible decision-support pipeline: YOLOv8 provides localization, while rule-based modules convert detections into interpretable severity and repair-priority scores. This bridges object detection and maintenance-oriented reporting without replacing engineering inspection.

IV. METHODOLOGY

The implemented methodology consists of five connected stages: image acquisition, YOLOv8 inference, detection record generation, severity analysis, and repair-priority analysis. The final structured records are rendered by the Flask web application.

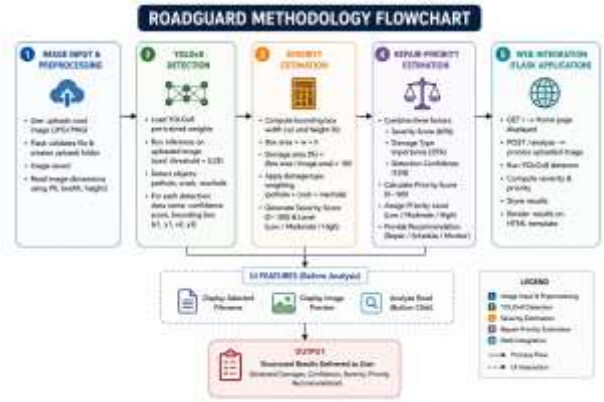

Fig. 1. End-to-end workflow of the proposed RoadGuard system for automated road-damage detection, severity assessment, and repair-priority estimation.

A. Image Input and Preprocessing

The user selects a JPG, JPEG, or PNG road image through the browser. The Flask route validates the uploaded file, creates the application uploads directory when required, saves the image, and reads its dimensions using the Python Imaging Library (PIL). The saved image path is then supplied to the detector.

B. YOLOv8 Detection

The detector module loads the project YOLOv8 weights and performs inference on the uploaded image. The implementation uses a confidence threshold of 0.25. For every accepted detection, the system records the class name, confidence value, and bounding-box coordinates. The project interface identifies three target damage classes: pothole, crack, and manhole.

C. Severity Estimation

For each detection, bounding-box width and height are obtained from the predicted coordinates. The bounding-box area is divided by the complete image area and multiplied by 100 to obtain a relative damage-area percentage. The severity module combines this visual-area measure with a damage-type weighting scheme. The implemented weighting assigns greater importance to potholes than cracks and manholes, allowing the output to be converted into a 0–100 severity score and a qualitative level.

D. Repair-Priority Estimation

The priority module combines three interpretable factors: severity contributes 60%, damage type contributes 25%, and detection confidence contributes 15%. Damage-importance values are defined for potholes, cracks, and manholes. The resulting score is mapped to a qualitative repair-priority level and an associated

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

recommendation. This separation makes the system easier to modify than a single opaque scoring model.

E. Web Integration

The Flask application provides a browser interface for upload and result visualization. The GET route renders the home page, while the POST /analyze route validates the uploaded file, stores it, invokes YOLOv8, calculates severity and priority, and returns the structured results to the HTML template. The user interface also displays the selected filename and image preview before analysis.

V. SYSTEM IMPLEMENTATION

Python is used as the primary implementation language. YOLOv8 inference is handled through the Ultralytics framework, Flask provides the web layer, HTML and CSS construct the interface, and PIL is used for image-dimension extraction. The project separates model inference, severity calculation, priority calculation, and web presentation into modular files. This structure allows the detector or scoring logic to be changed independently.

The detector module isolates the object-detection operation. The severity module converts bounding-box information into an area percentage and severity score, while the priority module combines severity, damage type, and confidence to produce a maintenance-oriented recommendation. The web module coordinates these components and sends the final result objects to the HTML template.

The resulting application provides an end-to-end workflow from image selection to interpretable results. The screenshots document the implemented user journey: application launch, landing page, upload interface, file selection, image preview, analysis request, and final detection dashboard.

Fig. 2. RoadGuard home page showing the AI-powered road inspection interface.

Fig. 3. Upload interface before image selection, including YOLOv8 model and three damage-class indicators.

Fig. 4. File-selection stage used to choose a road image from the local test-image directory.

Fig. 5. Selected road image displayed in the RoadGuard upload interface before analysis.

Fig. 6. Uploaded image preview with the selected filename and Analyze Road action.

VI. RESULTS AND DISCUSSION

The prototype was evaluated through the implemented Flask interface using a representative road image selected from the project's test-image directory. The complete workflow executed successfully: image selection, filename display, image preview, YOLOv8 inference, detection-result generation, severity estimation, and repair-priority calculation. The final dashboard reported three detections in the supplied example: one pothole and two cracks.

The pothole produced the strongest detection confidence at 84.2%. Its relative bounding-box area was 2.60%, and the severity module assigned a score of 26.03/100, classified as Moderate. Its repair-priority score was 53.25/100, classified as High, with the recommendation that repair should be scheduled soon.

The first crack was detected with 38.4% confidence and a relative area of 3.58%. Its severity score was 25.06/100, classified as Moderate, and its repair-priority score was 38.31/100, classified as Moderate. The second crack had 27.2% confidence, a relative area of 0.98%, a severity score of 6.83/100, and a repair-priority score

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

of 25.67/100. It was classified as Low severity and Moderate repair priority.

These outputs demonstrate the principal value of the proposed pipeline: the system does not stop at object labels. It converts each detection into a structured record containing damage type, confidence, relative area, severity, priority, and a recommended action. This makes the result easier for a user to interpret than a raw detector output.

Table I :Representative Roadguard Prototype Inference Results

Damage	Confidence	Area	Severity Score	Severity	Priority Score	Priority
Pothole	84.2%	2.60 %	26.03	Moderate	53.25	High
Crack	38.4%	3.58 %	25.06	Moderate	38.31	Moderate
Crack	27.2%	0.98 %	6.83	Low	25.67	Moderate

Fig. 7. Detection-results dashboard showing the pothole and two crack detections with confidence, severity, and repair priority.

The reported confidence values should not be interpreted as overall model accuracy. They are individual inference confidence values for the detections shown in one representative image. Similarly, the severity and priority values are generated by the project's rule-based modules and should be treated as prototype decision-support indicators. A statistically valid accuracy claim requires a documented held-out test set and standard object-detection metrics such as precision, recall, F1-score, mAP@0.5, and mAP@0.5:0.95.

VII. COMPARISON WITH RECENT WORK

Recent systems increasingly improve the detector itself through attention mechanisms, feature fusion, lightweight backbones, improved loss functions, segmentation, and depth information [12]–[20]. Such approaches are valuable when the primary objective is maximizing detector-level metrics or estimating physical geometry. RoadGuard follows a complementary approach. Instead of modifying the YOLOv8 architecture, it focuses on the software pipeline around a detector and introduces an interpretable layer for severity and maintenance priority.

This distinction is useful for an undergraduate research prototype because the complete processing chain can be inspected and reproduced. The detector answers “what and where is the visible damage?”, while the severity module answers “how visually significant is the detection?” and the priority module answers “how urgently should it be considered for maintenance?” The separation also makes the system suitable for future integration with stronger detectors, segmentation models, depth sensors, or geographic information systems.

VIII. LIMITATIONS

Several limitations should be considered. First, the current severity calculation uses bounding-box area as a visual proxy and therefore does not measure true damaged surface area. This is particularly important for cracks because a rectangular box may contain substantial undamaged pixels. Second, the scoring weights are heuristic and have not been calibrated against civil-engineering inspection standards or field measurements.

Third, the supplied demonstration represents an end-to-end prototype inference on a representative image rather than a complete statistical evaluation. The paper therefore avoids presenting the displayed 84.2%, 38.4%, and 27.2% confidence values as system accuracy. Fourth, the current web application is designed for local execution and does not yet include GPS mapping, database-backed historical tracking, or continuous vehicle-mounted video processing.

IX. FUTURE WORK

Future work should begin with a formal evaluation on a held-out test set using precision, recall, F1-score, mAP@0.5, mAP@0.5:0.95, per-class confusion analysis, and inference-time measurements. Dataset expansion should include different road materials, weather conditions, camera heights, lighting conditions, and geographic regions.

The severity component can be improved through instance segmentation so that the damaged pixels rather than the rectangular bounding box are measured. Depth cameras or monocular depth estimation could then be integrated to estimate pothole depth and three-dimensional geometry, following the direction of recent research [17]. GPS coordinates, road-segment mapping, historical defect tracking, and municipal maintenance records could convert RoadGuard into a larger road-condition monitoring platform.

The application can also be extended to video streams from vehicle-mounted cameras, edge-device deployment, automatic report generation, and cloud-based dashboards. These extensions would allow repeated detections to be aggregated over time and could support predictive maintenance rather than single-image inspection.

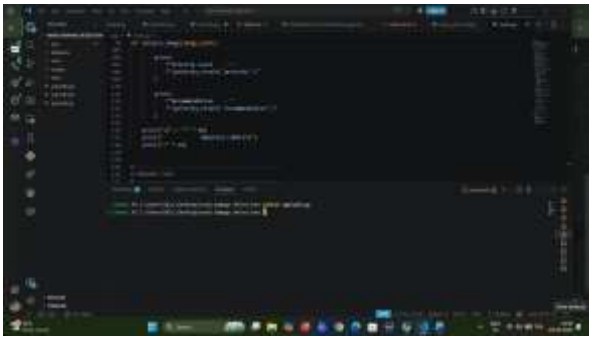

Fig. 8. RoadGuard analysis console showing the project execution environment and completion of the analysis workflow.

X. CONCLUSION

This paper presented RoadGuard, an automated road-damage assessment prototype that combines YOLOv8 object detection with interpretable severity and repair-priority scoring. The implemented system accepts road images through a Flask web interface, displays

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

the selected filename and preview, performs object detection, and converts the resulting detections into structured maintenance-oriented information.

The representative end-to-end test detected one pothole at 84.2% confidence and two cracks at 38.4% and 27.2% confidence. Their severity scores were 26.03, 25.06, and 6.83, while their repair-priority scores were 53.25, 38.31, and 25.67, respectively. These results demonstrate successful integration of the computer-vision and decision-support stages. The prototype is not presented as a substitute for engineering inspection; rather, it provides a practical bridge between automated visual detection and preliminary maintenance prioritization.

With larger benchmark evaluation, segmentation, depth estimation, field calibration, GPS integration, and deployment optimization, the RoadGuard architecture can be developed into a more comprehensive intelligent road-inspection platform.

ACKNOWLEDGMENT

The authors sincerely thank their mentor, Tathagata Roy Choudhury, for guidance and academic support during the development of the RoadGuard project and preparation of this manuscript.

REFERENCES

[1] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, and Y. Sekimoto, "RDD2022: A multi-national image dataset for automatic road damage detection," Geosci. Data J., vol. 11, no. 4, pp. 846–862, 2024, doi: 10.1002/gdj3.260.

[2] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, H. Omata, T. Kashiyama, and Y. Sekimoto, "Crowdsensing-based Road Damage Detection Challenge (CRDDC'2022)," in Proc. IEEE Int. Conf. Big Data, 2022, pp. 6378–6386, doi: 10.1109/BigData55660.2022.10021040.

[3] H. Yan and J. Zhang, "UAV-PDD2023: A benchmark dataset for pavement distress detection based on UAV images," Data Brief, vol. 51, Art. no. 109692, 2023, doi: 10.1016/j.dib.2023.109692.

[4] M. P. B. and S. K. C., "Enhanced pothole detection system using YOLOX algorithm," Auton. Intell. Syst., vol. 2, Art. no. 22, 2022, doi: 10.1007/s43684-022-00037-z.

[5] G. Guo and Z. Zhang, "Road damage detection algorithm for improved YOLOv5," Sci. Rep., vol. 12, Art. no. 15523, 2022, doi: 10.1038/s41598-022-19674-8.

[6] V. Pham, D. Nguyen, and C. Donan, "Road damages detection and classification with YOLOv7," arXiv:2211.00091, 2022.

[7] J. Zhu, J. Zhong, T. Ma, X. Huang, W. Zhang, and Y. Zhou, "Pavement distress detection using convolutional neural networks with images captured via UAV," Autom. Constr., vol. 133, Art. no. 103991, 2022, doi: 10.1016/j.autcon.2021.103991.

[8] Z. Diao, X. Huang, H. Liu, and Z. Liu, "LE-YOLOv5: A lightweight and efficient road damage detection algorithm based on improved YOLOv5," Int. J. Intell. Syst., 2023, Art. no. 8879622, doi: 10.1155/2023/8879622.

[9] A. M. Roy and J. Bhaduri, "A computer vision enabled damage detection model with improved YOLOv5 based on Transformer Prediction Head," arXiv:2303.04275, 2023.

[10] D. Li, Z. Duan, X. Hu, D. Zhang, and Y. Zhang, "Automated classification and detection of multiple pavement distress images based on deep learning," J. Traffic Transp. Eng. (English Ed.), vol. 10, no. 2, pp. 276–290, 2023, doi: 10.1016/j.jtte.2021.04.008.

[11] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," version 8.0.0, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[12] Z. Yang et al., "RDD-YOLO: Road damage detection algorithm based on improved You Only Look Once version 8," Appl. Sci., vol. 14, no. 8, Art. no. 3360, 2024, doi: 10.3390/app14083360.

[13] J. Zeng and H. Zhong, "YOLOv8-PD: An improved road damage detection algorithm based on YOLOv8n model," Sci. Rep., vol. 14, Art. no. 12052, 2024, doi: 10.1038/s41598-024-62933-z.

[14] J. Wang, R. Meng, Y. Huang, L. Zhou, L. Huo, Z. Qiao, and C. Niu, "Road defect detection based on improved YOLOv8s model," Sci. Rep., vol. 14, Art. no. 16758, 2024, doi: 10.1038/s41598-024-67953-3.

[15] H. Wu, L. Kong, and D. Liu, "Crack detection on road surfaces based on improved YOLOv8," IEEE Access, vol. 12, pp. 190850–190864, 2024, doi: 10.1109/ACCESS.2024.3517632.

[16] L. Li, J. Liu, J. Xing, Z. Liu, K. Lin, and B. Du, "Road pothole detection based on crowdsourced data and extended Mask R-CNN," IEEE Trans. Intell. Transp. Syst., vol. 25, no. 9, pp. 12504–12516, 2024, doi: 10.1109/TITS.2024.3360725.

[17] J. Zhong, D. Kong, Y. Wei, and B. Pan, "YOLOv8 and point cloud fusion for enhanced road pothole detection and quantification," Sci. Rep., vol. 15, Art. no. 11260, 2025, doi: 10.1038/s41598-025-94993-0.

[18] Z. Zhang, H. Zhang, and T. Zhang, "Enhanced YOLOv8-based pavement crack detection: A high-precision approach," PLoS One, vol. 20, no. 5, Art. no. e0324512, 2025, doi: 10.1371/journal.pone.0324512.

[19] H. Yang, Z. Ma, H. Zhu, and Q. Yu, "Road damage detection method based on UAV imagery and YOLO-SCX," Front. Built Environ., vol. 12, 2026, Art. no. 1681691, doi: 10.3389/fbuil.2026.1681691.

[20] Z. Tang, H. Wang, and R. Chamchong, "Enhanced YOLOv8 for efficient road damage detection with spatial-channel reconstruction and multi-scale attention," Sci. Rep., vol. 16, Art. no. 18205, 2026, doi: 10.1038/s41598-026-49498-9.

[21] Mandal, Supantha & Roy, Soumi & Das, Niladri & Roy, Ankur & Mandal, Tista & Gupta, Shivam & Roy Chowdhury, Tathagata. (2026). Quantsaas: An Intelligent Financial Analytics And Advisory Platform Using Predictive Intelligence And Ocr Automation. International Journal Of Multidisciplinary Research And Analysis. 2. 185-197. 10.1555/Ijrpa.6229.

[22] Roy Chowdhury, Tathagata & Mollah, Bahar. (2026). Artificial Intelligence Meets Healthcare Informatics: A Comprehensive Review Of Object Recognition For Clinical Decision Support And Smart Medical Systems. International Journal Of Research Publication And Reviews. 2. 185-197.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.