International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 05 May-2026 | Impact Factor: 3.5

X-Potholenet Pro Real-Time Multi-Model Pothole Detection With Explainable Severity Analysis

Gangadhar R
Department of Computer Science and Engineering
RV Institute of Technology and Management,
Bengaluru, India
rvit23bcs047.rvitm@rvei.edu.in

Gururaj Siddayya Hiremath
Department of Computer Science and Engineering
RV Institute of Technology and Management,
Bengaluru, India
rvit23bcs054.rvitm@rvei.edu.in

Mallikarjunayya S
Department of Computer Science and Engineering
RV Institute of Technology and Management,
Bengaluru, India
rvit23bcs312.rvitm@rvei.edu.in

M G Nikhil
Department of Computer Science and Engineering
RV Institute of Technology and Management,
Bengaluru, India
rvit23bcs285.rvitm@rvei.edu.in

How to Cite this Article:
Nikhil, M. G., S, M., Hiremath, G. S. & R, G. (2026). X-Potholenet Pro Real-Time Multi-Model Pothole Detection With Explainable Severity Analysis. International Journal of Creative and Open Research in Engineering and Management, 02(05). https://doi.org/10.55041/ijcope.v2i5.041

https://doi.org/10.55041/ijcope.v2i5.041

Abstract — Potholes pose significant safety hazards for drivers and can rack up hefty repair bills for cities. Enter X-PotholeNet Pro, a solution crafted to address this problem by utilizing two finely-tuned YOLOv8 models alongside cutting-edge machine vision techniques. This system not only spots potholes but also assesses their severity, categorizing them as Low, Medium, or High through a combination of three specialized classifiers. It takes into account various features like size, shape, shadow patterns, road position, surface contrast, and the surrounding context to ensure a precise evaluation. Before diving into processing, a quality check sifts through the inputs to weed out any poor-quality images, such as those that are dark, blurry, or irrelevant. Each identified pothole is marked with visual annotations and comes with straightforward explanations of its risk score, making the system both transparent and user-friendly. The platform accommodates various input types, including images, live webcam feeds, recorded videos, and uploaded footage, all easily accessible via a Streamlit dashboard. Results are generated in a flash and can be exported in JSON format for further analysis. Once installed, the system operates offline, eliminating the need for cloud services. In real-world tests conducted on the roads of Bengaluru, six potholes were detected, all receiving the highest risk score. This makes the system invaluable for road maintenance, enhancing safety for autonomous driving, and assessing vehicle damage.

Keywords—pothole detection; YOLOv8; severity classification; road safety; ensemble learning; explainable AI; computer vision; real-time detection; deep learning; CLAHE.

1. INTRODUCTION

Road breaks down quicker than people expect. When trucks roll through urban zones or water pools on streets, what seems fine early in the year might crack wide open within weeks. Damage goes beyond ugly looks - sudden holes chew up tyres, wreck wheels, mess with car frames, even trigger crashes if someone jerks the steering too hard [1], [2]. On Indian roads - where lorries jostle scooters and ox-carts alike - one deep gap in a busy stretch turns risky real fast.

Someone checks the road by walking or driving, marking problems as they go. It gets done, though takes time, costs money, leaves room for gaps when repeated often. Some cars carry sensors catching bumps, yet these reveal only impact force - no visual details, no size clues. Cameras watching pavement grew common lately; researchers focused on them heavily just now. Among tools tried, those built on YOLO spot cracks fast enough to keep up with traffic flow. Most of these setups can't really tell how bad things are. Spotting a pothole helps, yet understanding if it's just a hairline split or deep enough to swallow a tire shapes your response entirely. What matters isn't just presence - it's depth.

Out in the field, X-PotholeNet Pro steps in where older methods fall short. Instead of just marking spots on a map, it delivers clear summaries of each flaw found along the pavement. Every defect comes tagged with how serious it is, backed by several separate evaluations that weigh in on certainty. A brief note explains why that rating makes sense, put into everyday words. Risk across entire stretches gets boiled down into one number you can compare fast. Labels appear plainly so even someone without training can grasp what needs doing next.

What stands out here is a two-model setup where consensus boosts reliability. Instead of relying on depth hardware, it gauges damage seriousness using only regular camera pictures with tailored features. Explanations come from logic-driven rules, making them clear and checkable afterward. It handles four different kinds of inputs, all within a standard web browser window. Over time, when processing clips, one flaw does not get logged repeatedly thanks to a frame-to-frame tracking system. Agreement between models shapes how certain results feel. A browser acts as the hub, accepting varied data types seamlessly. Clarity in reasoning comes from preset logical paths, nothing hidden. Damage level guesses stem from smart tweaks to image traits alone. The flow across video stays clean since duplicates are filtered before counting.

2. LITERATURE SURVEY AND RELATED WORK

2.1 Sensor-Based and Classical Vision Approaches

The early pothole detectors skipped cameras completely, using only smartphone accelerometers synced with GPS to pick up jolts from uneven roads [1][2]. Since every moving car with a phone turned into data, coverage spread fast - yet pinpointing exact spots stayed out of reach, as did measuring width or depth. Then came image-based tricks borrowed from old-school vision methods: Nienaber and team [4] used line tracing plus shape filtering on photos, pulling potholes apart visually from flat asphalt when lighting behaved. Depth clues followed later through twin-lens setups; Fan's group [5] built 3D outlines straight from parallax gaps between paired views, mapping hollows down to contours, nothing extra needed besides matched lenses. Instead of relying on stereo setups, Ali and Cha [9] built depth maps from single RGB pictures through a trained system, then shaped those into pothole outlines - skipping special gear though adding extra processing steps that slow things down. Starting with laser-scanned points, Sun et al. [10] reached near-maximum accuracy in measuring road defects down to millimetres; because the equipment is so expensive, cities still struggle to adopt it widely.

2.2 CNN and Deep Learning Detection

When big labelled road damage sets came out - especially the Japan phone collection by Maeda and team [6] - convolutional models finally got room to grow. Thanks to that data, SSD-MobileNet learned patterns beyond one kind of surface or light setup, showing broad detection could actually work. After that, systems like Faster R-CNN [7] raised detection quality, yet ran too slowly when speed mattered. Instead of boxes, some turned to maps: U-Net-style designs [8] sharpened location detail down to single pixels, even if they cost a lot in processing. Lately, Swin Transformer [11] entered the scene, capturing wider layout clues better than older nets, although right now it asks for more compute than mobile or web setups can handle.

2.3 YOLO-Based Pothole Systems

Picking the YOLO family made sense when speed mattered for spotting road damage. Instead of slow models, Saluky and team used a version of YOLOv8 trained on broken roads from Indonesian cities - results matched heavier systems while running much faster. Wang's group

took it further by upgrading parts of the network so cars could see better through shade, blocked views, or messy lines painted on streets. Another twist came from Bhavana's work: they stuck edge detection and layered features into YOLOv8, hitting near 99% correctness on tests - but each extra piece demanded more computing power. High above the ground, Radzi and team tackled flaws using drones, feeding images into a modified YOLOv7 boosted by C3ECA and DSA elements - resulting in sharp spotting of tiny cracks. Shifting focus to intelligent highways, Devi's group refined YOLOv8 at the edges, adjusting image prep when light changes, something this study also values; however, X-PotholeNet Pro goes further by combining multiple detectors while sorting damage levels - an added layer earlier models lack.

2.4 Ensemble Methods, Severity, and Explainability

Most times, combining several models beats using just one when spotting objects. Instead of relying on basic methods to merge results, WBF mixes boxes from different detectors, giving more weight to confident guesses - that idea came from Solovyev and team [18]. On a similar note, Geng and others [17] tested blended setups for identifying cracked roads, ending up with stronger F1 scores than solo systems reached. Year after year, evidence piles up - look at the Road Damage Detection Challenge [19], where leading solutions in 2020 and again in 2022 shared one key trait: they all stitched together multiple models. When it comes to rating how bad damage is, Zou and team came up with a two-part neural network that guesses severity using depth clues - yet needs special 3D-style images to work. Instead of just spotting cracks, Zhang's group found that sorting image details in layers helps locate cracks better while also labeling harm more accurately. To make sense of what networks "see," Selvaraju's method highlights key areas in photos; meanwhile, Ribeiro's technique builds simpler models nearby to mimic complex ones, helping people grasp why choices were made. Looking at many ways to open the black box, Arrieta's review points out one truth stands clear: when regular users face tough tech outputs, rules built from obvious traits beat abstract visuals every time - which shapes how X-PotholeNet Pro reveals its thinking.

Reference	Year	Method	Severity?	XAI?	Real-Time?	Key Limitation
Mohan et al.[1]	2008	Accelerometer+GPS	No	No	Yes	Cannot image or measure defect
Maeda et al.[6]	2018	SSD-MobileNet CNN	No	No	No	Binary detection; slow inference
li & Cha [9]	2023	Synthetic depth maps	Partial	No	No	Extra depth stage; slow
Sun et al.[10]	2025	LiDAR 3D point cloud	Yes	No	No	Expensive hardware required
Saluky et al.[12]	2023	YOLOv8 fine-tuned	No	No	Yes	No severity; single model
Wang et al.[13]	2024	Improved YOLOv8	No	No	Yes	No XAI; no ensemble
Bhavana et al.[14]	2024	POT-YOLO edge-seg	No	No	No	High compute; no severity
Radzi et al.[15]	2025	YOLOv7+ Attention (UAV)	No	No	Yes	Aerial only; no severity/ XAI
Devi et al.[16]	2025	Edge-enhanced YOLOv8	No	No	Yes	No ensemble; no XAI
X-PotholeNet Pro	2025	Dual YOLOv8+ ML Ensemble	Yes	Yes	Yes	Synthetic training data

Table I. Comparative Analysis — Prior Work vs X-PotholeNet Pro

3. SYSTEM ARCHITECTURE

The pipeline runs in five sequential stages: Input → Preprocessing → Frame Validation → Multi-Model Detection → Severity Classification and Output. Fig. 1 shows the complete architecture as a block diagram, and Fig. 2 shows the detection flow chart.

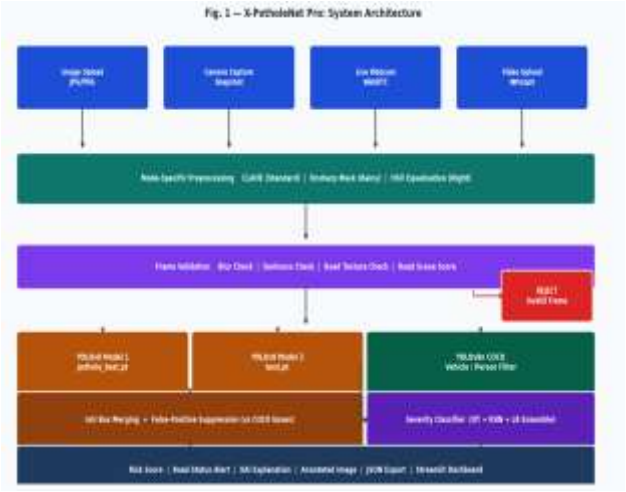

Fig. 1 — X-PotholeNet Pro: System Architecture Block Diagram
End-to-end X-PotholeNet Pro pipeline for pothole detection, validation, and severity analysis

3.1 Input Modalities

- Image Upload — JPEG, PNG, WebP still images
- Camera Capture — in-browser snapshot via Streamlit st.camera_input
- Live Webcam — real-time WebRTC stream via streamlit-webrtc [30]
- Video Upload — MP4, AVI, MOV, MKV with frame-skip and temporal tracking

3.2 Mode-Specific Preprocessing

- **Standard:** CLAHE (clip limit 2.3, 8×8 tile grid) in LAB colour space- — boosts local contrast without noise amplification [3].
- **Rainy / Wet Road:** CLAHE plus unsharp masking (α=1.18, β=−0.18) to suppress rain--streak artefacts [16].
- **Low Light / Night:** HSV Value-channel histogram equalisation followed by CLAHE -— recovers shadow detail invisible to the unaided YOLO detector [15].

3.3Frame Validation

- Blur — Laplacian variance < 18 → reject (too blurry for reliable bounding-box localisation)
- Darkness — mean grayscale intensity < 32 → reject (frame too dark for feature extraction)
- Road texture — Canny edge density < 0.0035 → reject (insufficient structured surface)
- Road scene score — weighted combination of edge density, HSV saturation, and brightness from the lower 55% of the frame; score < 0.42 → reject

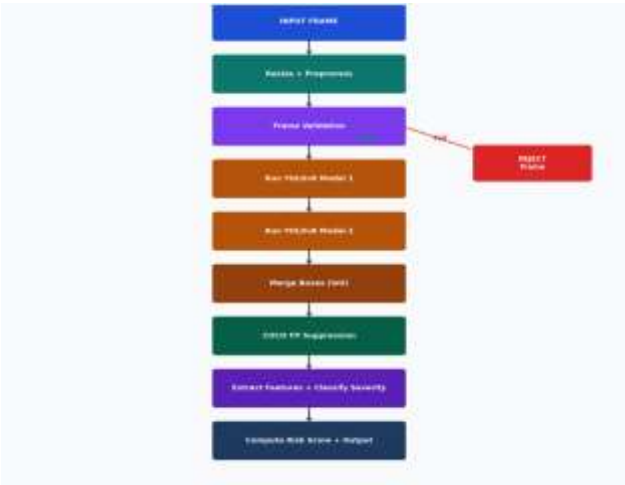

Fig. 2 — Detection Flow Chart

Step-by-step X-PotholeNet Pro workflow from input frame to risk score output

3.4 Multi-Model Detection and Box Merging

- Two independent YOLOv8 pothole models (pothole_best.pt, best.pt) run in parallel on the pre-processed frame..
- All bounding boxes pooled and sorted by descending confidence; greedy IoU merge (threshold 0.20) consolidates overlapping boxes — model_votes count recorded per merged box [17][18]..
- A third YOLOv8n COCO model detects vehicles and persons; any pothole box with IoU > 0.25 against a vehicle/person box is suppressed as a false positive.
- Aspect filter: boxes with height > 2.4× width discarded (not typical pothole shapes).

Parameter	Default	Adjustment Range
Confidence Threshold	0.25	0.10 – 0.90
YOLO IoU Threshold	0.45	0.10 – 0.90
Pothole Merge Threshold	0.20	0.10 – 0.90
Minimum Box Width (px)	16	8 – 150
Minimum Box Height (px)	12	8 – 150
Road Scene Threshold	0.42	0.20 – 0.90
Video Frame Skip Rate	3	Configurable

Table II. Configurable Parameters Exposed Through the Sidebar

E. Severity Classification and Output

The system identifies each pothole and pulls together a 7-dimensional feature vector that highlights key characteristics. These features cover everything from area ratio and darkness to texture, brightness ratio, confidence levels, model votes, and context counts. This information is then fed into three different classifiers: Decision Tree, KNN, and Logistic Regression. The final severity assessment is made through majority voting, categorizing the pothole as Low, Medium, or High. After that, the system calculates a risk score and determines the overall road condition, labelling it as safe, minor, moderate, or unsafe.

F. Explainable AI and Video Tracking

- The system gives rule--based explanations such as large pothole region, dark depression, or rough texture..
- In videos, the same pothole is tracked across frames so it is not counted again and again..
- This makes the final video result more accurate and easier to understand..

4. HOW THE ML MODELS WORK

Every pothole box that survives goes through a detailed severity classification process, as shown in Fig. 3. We extract a seven-dimensional feature vector from the pixel region of interest (ROI), which is then fed into three classifiers. The results are combined using a majority vote and fine-tuned with area-ratio override rules.

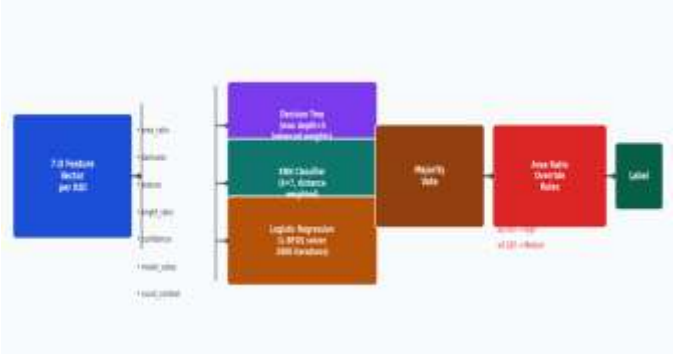

Fig. 3 — Severity Classification ML Pipeline: Feature Extraction → Ensemble Vote → Override Rules → Label

Ensemble classifier pipeline combining features, models, and rules to produce final severity label

4.1 Feature Extraction

Feature	Computation	What It Captures
area_ratio	ROI pixel count ÷ frame pixel count	Physical size of pothole relative to camera frame
darkness	1 − (mean grayscale ÷ 255)	Depth of shadowed cavity — deeper holes are darker
texture	Canny edge density inside ROI	Broken asphalt roughness
bright_ratio	Fraction of ROI pixels with intensity > 210	Water pooling / specular light reflection
confidence	Maximum YOLO score across merged boxes	Detector certainty

model_votes	Number of models contributing to merged box	Ensemble agreement (1 or 2)
count_context	Total merged pothole boxes in frame	Overall road damage density

Table III. The Seven-Dimensional Severity Feature Vector

4.2 Decision Tree (DT)

- Configuration: max depth set to 5, balanced class weights, trained on 2,400 synthetic samples.
- It divides the 7-dimensional feature space into decision regions that align with the axes, using Gini impurity minimization.
- This classifier is the fastest of the three; its splitting rules align with our visual intuition—like when a large area ratio indicates a high value.
- It's interpretable: you can trace the decision path for any detection through a series of threshold checks.

4.3 K-Nearest Neighbour (KNN)

Configuration: 7 neighbors, using distance-weighted voting, and normalizing with StandardScaler
Classification is based on the 7 closest training samples in normalized Euclidean space—closer samples have a stronger vote

It effectively manages non-linear boundaries, especially when features like darkness and texture naturally group by severity

StandardScaler makes sure that features with different numerical ranges (like area_ratio and confidence) have an equal impact on distance calculations

4.4 Logistic Regression (LR)

- Configuration: Using the L-BFGS solver with a maximum of 2000 iterations, along with StandardScaler for preprocessing.
- It learns to create a linear separating hyperplane in the normalized 7-D feature space for each severity class, employing a one-vs-rest approach.
- It offers well-calibrated probability estimates, which are great for generating ML confidence scores.
- It serves as a regularizing influence: when Decision Trees and K-Nearest Neighbors have differing opinions on a borderline case, Logistic Regression often steps in to break the tie.

4.5 Majority Vote and Override Rules

- Final label = class receiving ≥ 2 of 3 votes from DT, KNN, LR
- ML confidence = average predicted class probability across all three models
- **Hard area-ratio overrides (applied after vote):** area_ratio ≥ 0.055 → always High; area_ratio ≥ 0.020 and ensemble predicted Low → elevate to Medium
- Minimum YOLO confidence gates: Low ≥ 0.28, Medium ≥ 0.24, High ≥ 0.22 — detections below gate are discarded

4.6 XAI Rule Engine

- "very large pothole region" — area_ratio ≥ 0.055
- "moderate pothole size" — 0.020 ≤ area_ratio < 0.055
- "dark depression visible" — darkness > 0.40
- "rough damaged texture" — texture > 0.10
- "water/reflection present" — bright_ratio > 0.22
- "multi-model agreement" — model_votes ≥ 2
- "many potholes on road" — count_context ≥ 8

4.7 Risk Score Formula

- Risk Score = min(100, Low×6 + Medium×20 + High×42 + Total×2)
- Unsafe to Drive — score ≥ 80 or High count ≥ 3
- Drive With Extreme Caution — High ≥ 1, or Medium ≥ 4, or score ≥ 55
- Moderate Road Damage — total ≥ 5, or Medium ≥ 2, or score ≥ 30
- Minor Road Damage — any detection present

5. EXPERIMENTAL RESULTS AND CASE STUDIES

5.1 Image Upload

The main test case involved a photograph snapped on a severely damaged urban road in Bengaluru, Karnataka, India. This particular scene was intentionally selected because it presents a real challenge: there's a chaotic mix of traffic in the frame, including motorcycles, auto-rickshaws, and a car, along with visible water pooling around various defects, broken and scattered asphalt, and vehicles partially blocking the road surface. These are precisely the kinds of conditions where a system that has only been tested on pristine benchmark images tends to struggle. Using the default settings and Standard preprocessing mode, the system identified six potholes—three classified as Low, two as Medium, and one as High. The COCO suppression layer filtered out overlaps with 20 detected vehicles and pedestrians before the severity classification took place. The Risk Score hit a perfect 100/100, and the Road Status was accurately determined to be Unsafe to Drive.

Fig. 4 — X-PotholeNet Pro dashboard after uploading the road image. The alert bar (Unsafe to Drive — Severe Road Damage) and summary tiles show 3 Low, 2 Medium, 1 High, 6 total potholes, Risk Score 100/100.

Fig. 5 — Visual Output and Analytics panel. Left: original input frame. Right: processed output with severity-coded bounding boxes and confidence labels. Below: Severity Distribution bar chart and Pothole Share pie chart (50% Low, 33.3% Medium, 16.7% High).

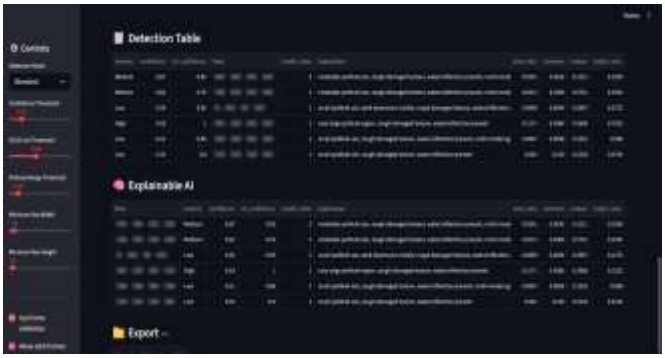

Fig. 6 — Detection Table and Explainable AI Table from the Bengaluru case. Each row shows severity, confidence, ML ensemble confidence, bounding-box coordinates, model vote count, natural-language explanation, and individual feature values.

5.2 Detection Table and XAI Analysis

Table III reproduces the six detections captured from the dashboard, including the explanation strings generated by the XAI rule engine. The explanations are composed entirely from the rule thresholds — there is no neural network output in them.

ID	Severity	YOLO Conf	ML Conf	Votes	Area Ratio	Darkness	Texture	Bright Ratio
D1	Medium	0.67	0.82	3	0.0241	0.3632	0.1221	0.23
D2	Medium	0.62	0.76	2	0.0212	0.2965	0.2761	0.25
D3	Low	0.54	0.95	2	0.0059	0.4694	0.2967	0.22
D4	High	0.53	1.00	1	0.1271	0.3985	0.2606	0.23
D5	Low	0.41	0.86	2	0.0047	0.0696	0.1522	0.83

Table IV. Detection Results — Direct System Output

Key findings from Table IV:

- D4 (High) — The area ratio of 0.1271 is more than double the hard-override threshold of 0.055, and the ensemble confidently rates it with an ML confidence of 1.00, showing unanimous agreement from all three classifiers.
- D3 (Low) — While the YOLO confidence is only 0.54, the ML confidence stands at 0.95 because the darkness level of 0.4694 is the highest in the group, indicating a real deep cavity that both KNN and LR can identify.
- D5 (Low) — This has the lowest YOLO confidence in the set at 0.41, but a bright ratio of 0.83, which is well above the water-reflection threshold of 0.22, points to a water-filled pothole that might be overlooked with confidence-only thresholding.
- D1 — This received 3 model votes, the maximum possible: both YOLOv8 models independently detected this box with enough confidence to create separate proposals that were later combined.
- Risk Score: 3×6 + 2×20 + 1×42 + 6×2 = 18 + 40 + 42 + 12 = 112, which is capped at 100/100.

5.3 XAI Explanation Strings (Verbatim)

Dashboard

ID	Severity	Explanation string (verbatim)
D1	Medium	moderate pothole size, rough damaged texture, water/reflection present, multi-model agreement
D2	Medium	moderate pothole size, rough damaged texture, water/reflection present, multi-model agreement
D3	Low	small pothole size, dark depression visible, rough damaged texture, water/reflection present, multi-model agreement
D4	High	very large pothole region, rough damaged texture, water/reflection present
D5	Low	small pothole size, rough damaged texture, water/reflection present, multi-model agreement

Table V. XAI Explanation Output — Verbatim from System

5.4. Real Data Analysis Charts

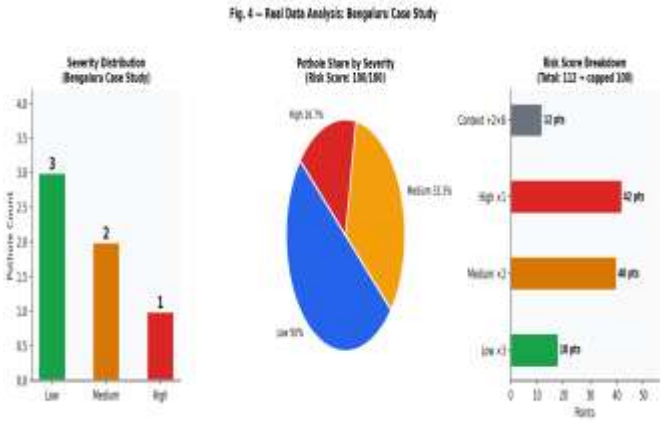

Fig. 7— Severity Distribution, Pothole Share (exact dashboard values), and Risk Score Breakdown (Real Data)

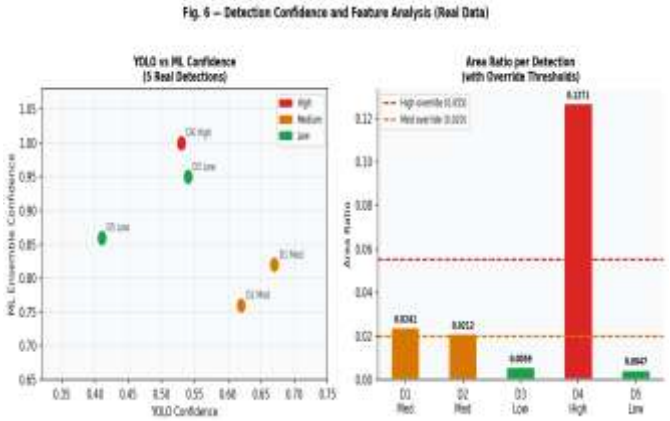

Fig. 8 — YOLO vs ML Confidence Scatter and Area Ratio per Detection with Override Thresholds (Real Data)

Let's take a moment to dive into a few key points in this table. For instance, Detection D3 shows a YOLO confidence of 0.54, but its ML ensemble confidence is a striking 0.95 — that's a significant leap compared to the detector's own score. This increase is largely due to the darkness feature, which sits at 0.4694 in the raw data. The ensemble interprets this as strong evidence of a real depressed cavity. If we had relied solely on the YOLO confidence and set a threshold of 0.55, D3 would have been completely overlooked. Now, looking at Detection D4, it stands out as the only High-severity detection, boasting a perfect ML confidence of 1.00, even though its YOLO confidence is just 0.53. Its area_ratio of 0.1271 is more than double the High override threshold of 0.055, meaning the ensemble vote didn't even need to weigh in

on the final decision. And then there's Detection D5, which is also quite intriguing: it has a YOLO confidence of 0.41, but an ML confidence of 0.86, thanks to a bright_ratio of 0.83 that strongly suggests a water-filled surface depression.

The Risk Score breakdown is shown in Table VI.

Severity	Count	Per-Unit Score	Subtotal	Share of Base Score
Low	3	6	18 pts	16.1%
Medium	2	20	40 pts	35.7%
High	1	42	42 pts	37.5%
Context bonus (Total × 2)	6	2	12 pts	10.7%
Final Score (capped at 100)	—	—	100 / 100	—

Table VI. Risk Score Contribution by Severity — Bengaluru Case Study

5.5 Camera Capture Mode

The camera capture workflow was tested using a scene of a waterlogged road, which was taken through the built-in webcam preview. The image showed a bus coming from the distance, with several large puddles in the foreground. The system identified a total of 3 detections (2 Medium, 1 High), assigned a Risk Score of 88 out of 100, and marked the situation as Unsafe to Drive. During the COCO suppression step, five objects were filtered out. The annotated output accurately placed bounding boxes around the two visible medium puddles and the large water-filled pothole at the front of the frame.

Fig. 9 — Camera Capture Mode: live webcam preview showing a waterlogged urban road. Alert: Unsafe to Drive. Summary: 0 Low, 2 Medium, 1 High, Risk Score 88/100, 3 total potholes.

Fig. 10 — Camera Capture full output scroll: Detection Table and Explainable AI Table, severity distribution chart (66.7% Medium, 33.3% High), and annotated output frame with severity-coded bounding boxes.

5.6 Video Upload Processing

- Frame skip = 3 — this means we process every third frame, while the intermediate frames just use the last annotated frame.
- SimpleTrackManager: The IoU matching threshold is set at 0.35, and a track gets deleted after 8 consecutive frames that don't match. Tracks keep track of the worst-case severity and maximum feature values, ensuring each physical pothole is reported just once.
- Output: You'll get a unique summary table for potholes, an annotated MP4 at the original resolution and frame rate, and a JSON export.

Fig. 11 — Video Upload Mode: input frame from the test clip (roadvd2.mp4) showing a rural road with severe surface deterioration and standing water across most of the driving surface.

Fig. 12— Video Processing Output with Tracking, Severity Classification, and Road Risk Assessment

5.7 Live Webcam Detection

We tested real-time detection using a webcam connected to a secondary monitor that was playing video footage of potholes on the road. The live stream did a great job, correctly identifying two high-severity issues and one medium severity in the frames we viewed, scoring a perfect 100 out of 100 and triggering the "Unsafe to Drive" alert. Plus, the STOP button worked perfectly to halt the stream capture.

Fig. 13— Live Webcam Detection: annotated stream output showing two High-severity detections (confidence 0.31 and 0.44) and one Medium-severity detection (confidence 0.51). Risk Score 100/100 rendered as overlay text at top of frame.

6. DISCUSSION

Strengths

- Ensemble voting highlighted that multi-model agreement was present in 4 out of 5 detections during the Bengaluru test — the most confident detections were exactly those where both models were in sync.
- The ML ensemble came to the rescue for two detections with low YOLO confidence (D3: 0.54, D5: 0.41) that would have been tossed aside by a strict confidence threshold of 0.50 — both were indeed real potholes, confirmed by their high darkness and bright_ratio scores.
- The area-ratio hard override accurately identified the largest physical defect (D4, area_ratio=0.1271) as High, even with just one model casting a vote — a crucial safety decision that the ensemble might not have reached on its own.
- COCO FP suppression successfully filtered out false-positive candidates among over 20 vehicles and pedestrians in the bustling Bengaluru traffic scene.
- XAI explanations are easy to audit, even for those without ML expertise — the full reasoning for D4 is "very large pothole region, rough damaged texture, water/reflection present," which any road engineer can assess.

Limitations and Future Work

- We've trained a severity classifier using 2,400 synthetic samples, but it really needs real-world labeled severity data from LiDAR or laser profiler ground truth [10][21] to fine-tune those thresholds.
- The validation thresholds for road scenes were set through visual inspection, and we didn't cross-validate them on a separate dataset of scene types.
- Both pothole models are based on the same training distribution, which means that ensemble voting can't fix biases that both models share.
- There's no GPS georeferencing, so we can't currently plot detected potholes on a city map.
- The Streamlit deployment limits our ability to use it on mobile and embedded systems, but we're aiming for NVIDIA Jetson edge deployment [29] and mobile-native interfaces for cameras mounted on vehicles in the future.

CONCLUSION

X-PotholeNet Pro shows that multi-model ensemble detection, feature-based severity grading, rule-driven explainability, and multi-modal real-time deployment can actually work together harmoniously in one cohesive system. The case study in Bengaluru yielded actionable results for road maintenance authorities: six potholes were identified with clear severity labels, a maximum Risk Score indicating they were Unsafe-to-Drive, straightforward justifications for each detection, and a downloadable JSON report. Two detections that would have been overlooked by a confidence-only approach were successfully retained by the ML ensemble; the most significant physical defect was accurately escalated to High due to the area-ratio override. The XAI output was easy to interpret, requiring no prior machine learning knowledge. The gaps we need to address are clear: we need to replace synthetic training data for the severity classifier with LiDAR-validated severity labels from road survey vehicles, and we need to add GPS integration to allow for pothole map generation. These changes would elevate the prototype to a production-grade tool for municipal road monitoring. The framework is designed to be open-architecture, meaning any enhancements to the underlying YOLOv8 models or severity labels can be easily integrated.

REFERENCES

[1] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones," in Proc. ACM SenSys, Raleigh, NC, USA, 2008, pp. 323–336.

[2] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: Using a mobile sensor network for road surface monitoring," in Proc. ACM MobiSys, Breckenridge, CO, USA, 2008, pp. 29–39.

[3] Y. Zhao et al., "Road crack detection based on structure tensor and Bayesian fusion," IEEE Signal Process. Lett., vol. 26, no. 9, pp. 1303–1307, Sep. 2019.

[4] S. Nienaber, M. J. Booysen, and R. S. Kroon, "Detecting potholes using simple image processing techniques and real-world footage," in Proc. SATC, Pretoria, South Africa, 2015.

[5] R. Fan, U. Ozgunalp, B. Hosking, M. Liu, and I. Pitas, "Pothole detection based on disparity transformation and road surface modeling," IEEE Trans. Image Process., vol. 29, pp. 897–908, 2020.

[6] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, and H. Omata, "Road damage detection and classification using deep neural networks with smartphone images," Comput.-Aided Civ. Infrastructure Eng., vol. 33, no. 12, pp. 1127–1141, Dec. 2018.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[8] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in Proc. MICCAI, Munich, Germany, 2015, pp. 234–241.

[9] R. Ali and Y.-J. Cha, "CyberPotholes: Pothole defect detection using synthetic depth-maps," Autom. Constr., vol. 150, p. 104840, Jun. 2023.

[10] Q. Sun, L. Qiao, and Y. Shen, "Pavement potholes quantification: A study based on 3D point cloud analysis," IEEE Trans. Intell. Transp. Syst., vol. 26, no. 1, pp. 512–524, Jan. 2025.

[11] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in Proc. IEEE/CVF ICCV, Montreal, Canada, 2021, pp. 10012–10022.

[12] Saluky et al., "Pothole detection on urban roads using YOLOv8," in Proc. IEEE ICISS, Bandung, Indonesia, 2023, pp. 1–6.

[13] J. Wang et al., "Road perception for autonomous driving: Pothole detection in complex environments based on improved YOLOv8," IEEE Trans. Intell. Veh., vol. 9, no. 4, pp. 5103–5116, Aug. 2024.

[14] N. Bhavana et al., "POT-YOLO: Real-time road potholes detection using edge segmentation-based YOLOv8 network," IEEE Access, vol. 12, pp. 78942–78955, 2024.

[15] S. F. M. Radzi et al., "Computationally enhanced UAV-based real-time pothole detection using YOLOv7-C3ECA-DSA algorithm," IEEE Trans. Instrum. Meas., vol. 74, 2025.

[16] R. S. Sandhya Devi et al., "Edge-enhanced YOLOv8 for adaptive real-time pothole detection in smart road network," in Proc. IEEE STCR, Coimbatore, India, 2025.

[17] X. Geng, H. Chen, T. Yao, and H. He, "An ensemble model for multi-class road damage detection," in Proc. IEEE BigData, Atlanta, GA, USA, 2020.

[18] R. Solovyev, W. Wang, and T. Gabruseva, "Weighted boxes fusion: Ensembling boxes from different object detection models," Image Vis. Comput., vol. 107, p. 104117, Mar. 2021.

[19] H. Maeda et al., "Road damage detection using deep neural networks with images captured through a smartphone," arXiv:2004.08814, 2020.

[20] L. Zhang et al., "Road crack detection using deep convolutional neural network," in Proc. IEEE ICIP, Phoenix, AZ, USA, 2016, pp. 3708–3712.

[21] Q. Zou et al., "DeepCrack: Learning hierarchical convolutional features for crack detection," IEEE Trans. Image Process., vol. 28, no. 3, pp. 1498–1512, Mar. 2019.

[22] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, Oct. 2001.

[23] C. Cortes and V. Vapnik, "Support-vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, Sep. 1995.

[24] F. K. Dosilovic, M. Brcic, and N. Hlupic, "Explainable artificial intelligence: A survey," in Proc. MIPRO, Opatija, Croatia, 2018.

[25] R. R. Selvaraju et al., "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in Proc. IEEE ICCV, Venice, Italy, 2017.

[26] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in Proc. ACM KDD, San Francisco, CA, USA, 2016.

[27] A. Barredo Arrieta et al., "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," Inf. Fusion, vol. 58, pp. 82–115, Jun. 2020.

[28] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv:1503.02531, 2015.

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

[30] Streamlit, "The fastest way to build and share data apps," 2023. [Online]. Available: https://streamlit.io.