Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

CarSenseAI: A Deep-Learning Pipeline for Real-Time Vehicle Damage Detection, Severity Assessment, and Repair-Cost Estimation

Mithun R
Dept. of CSE
Atria Institute of Technology
Bengaluru, India
1AT23CS087
1at23cs087@visioneer.atria.edu

Prof. Pavithra SS
Dept. of CSE
Atria Institute of Technology
Bengaluru, India
Assistant Professor
pavithra_cse@atria.edu.in

Katti Gururaj
Dept. of CSE
Atria Institute of Technology
Bengaluru, India
1AT23CS066
1at23cs066@visioneer.atria.edu

Lucky Mahawar
Dept. of CSE
Atria Institute of Technology
Bengaluru, India
1AT23CS078
1at23cs078@visioneer.atria.edu

Muhammad Rayyan Zabi
Dept. of CSE
Atria Institute of Technology
Bengaluru, India 1AT23CS093
1at23cs093@visioneer.atria.edu

***Abstract*—Vehicle damage assessment is traditionally performed through manual inspection by insurance surveyors, workshop technicians, and fleet-management personnel, a process that is slow, subjective, and difficult to scale as accident volumes and insurance-claim loads grow. We present CarSenseAI, an end-to-end, camera-based mobile system that automates damage detection, localization, severity classification, and repair-cost estimation from ordinary smartphone photographs. We give a formal statement of the assessment task as a composition of four learned maps, and specify each stage by its objective: a YOLOv8 detector trained with a CIoU-based composite loss, a Mask R-CNN segmentation stage supplying pixel-level damage area, a CNN severity classifier over a continuous severity score, and a cost regressor whose output is projected onto ontology-derived bounds so that predictions cannot leave the range domain knowledge admits. Synthetic data augmentation via GANs and diffusion models compensates for the scarcity of labelled examples in underrepresented damage categories. The system is delivered as a Flutter mobile application backed by a Node.js/Flask API and a Firebase/MySQL data layer, with GPS-based mechanic discovery, SOS emergency alerts, and service reminders layered on the core pipeline. A review of 41 studies published between 2019 and 2026 shows that most existing systems address only one stage of the assessment workflow rather than a complete pipeline; CarSenseAI is designed to close that gap. We describe the architecture, the stage-wise mathematical formulation, the requirements, and the planned evaluation protocol, and outline the open problems that remain: simultaneous multi-damage handling, on-device deployment cost, dataset diversity, and fraud detection.**

***Index Terms*—vehicle damage detection, deep learning, YOLOv8, Mask R-CNN, severity classification, repair-cost estimation, insurance technology, mobile computer vision**

I. Introduction

The automotive industry's rapid expansion has driven a corresponding rise in the number of vehicles on the road, and with it a rise in accidents and damage-related incidents. This has created sustained demand for damage-assessment systems that are efficient, reliable, and capable of handling claims at scale. The conventional route, manual inspection by insurance surveyors, workshop technicians, and fleet-management personnel, remains widely used but is time-consuming, subjective, and hard to scale when claim volumes are high.

CarSenseAI is proposed as a modern alternative. It is an AI-driven mobile application that identifies, categorizes, and evaluates vehicle damage from photographs captured directly on a smartphone. Using convolutional neural networks together with the YOLOv8 object-detection architecture, the system performs damage analysis with dependable accuracy, recognizing scratches, dents, cracks, and structural distortions; locating the affected regions; grading severity; and producing an estimated repair cost. Beyond the core assessment function, the platform layers on GPS-enabled nearby-mechanic suggestions, SOS alerts with live location sharing, maintenance reminders, and general repair guidance.

The goal is to close the gap between the growing need for fast, unbiased damage assessment and the limitations of manual inspection, via a complete pipeline that converts a raw vehicle photograph into a structured, actionable report without specialized hardware or controlled imaging conditions.

A. Motivation

Three forces make this the right moment for an automated pipeline. First, smartphone camera quality has reached a point where ordinary consumer photographs carry enough resolution and dynamic range for fine-grained damage cues, such as hairline scratches, shallow dents, and small cracks, to be visible without specialized rigs or fixed lighting. Second, the object-

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

detection and segmentation literature has matured to where real-time, mobile-deployable models (YOLOv8, lightweight Mask R-CNN variants) achieve accuracy that was previously the preserve of server-side, batch-processed pipelines. Third, insurance and fleet operators face rising claim volumes and rising expectations of near-instant turnaround, which manual inspection structurally cannot provide: a surveyor visit, a written report, and a manual cost lookup routinely take days, during which a vehicle may sit idle and a claim may stall. Automating the visual-assessment step, without removing a human from the final approval decision, is therefore both technically feasible and operationally valuable.

B. Proposed System

CarSenseAI ingests vehicle images and returns structured damage reports, severity grades, and repair-cost estimates in real time. Unlike systems that address only one stage of the pipeline, it combines YOLOv8-based detection, Mask R-CNN segmentation, CNN-driven severity analysis, and regression-based cost prediction in one framework, targeting insurance claim verification, service workshops, used-vehicle marketplaces, and fleet management. Synthetic data augmentation via diffusion models improves robustness on uncommon damage categories, and the camera-only capture path removes any dependency on specialized sensors.

The remainder of the paper is organized as follows. Section II surveys prior work; Section III states the problem and the research gap; Section IV gives the formal, stage-wise formulation of the pipeline; Section V and Section VI describe the architecture and processing methodology; Section VII specifies requirements; Section IX sets out the evaluation protocol and threats to validity; and Section X concludes.

II. Related Work

We reviewed 41 studies published between 2019 and 2026, spanning four themes: general damage detection, fine-grained dent and scratch detection, repair-cost estimation, and segmentation or transformer-based methods. The 21 works most directly relevant to the design decisions below are cited here.

A. General vehicle damage detection

Early work relied on edge detection, texture analysis, and hand-engineered features, which performed acceptably in controlled settings but degraded under real-world lighting, viewpoint, and colour variation. Van Ruitenbeek [7] established an early deep-CNN baseline on more than 5,000 images. Later work layered on transfer learning with ResNet, VGG, and MobileNet, multi-scale YOLO-family detectors, and Mask R-CNN-based instance segmentation. Hasan et al. [2] found AI-based methods consistently outperform manual inspection on speed and consistency, several exceeding 90% classification accuracy; Amodu et al. [3] compared CNN, YOLO, and Mask R-CNN and reported 92% accuracy on a standardized benchmark, a figure we treat as a modern reference point. Verma et al. [4] paired CNN feature extraction with SVM classification. Mohamad et al. [6] showed that separating detection from severity regression improves overall accuracy, a design choice we adopt in (9)–(10). Lee et al. [5] introduced a three-quarter-view car-damage dataset and showed that viewpoint-consistent capture protocols materially improve classification accuracy over unconstrained photographs, a finding that shaped our decision to normalize orientation during preprocessing (FR-01). Tran et al. [1] survey the broader field and note that most published pipelines are still evaluated on small, single-institution datasets, which limits confidence in reported accuracy figures generalizing to new fleets or regions.

B. Dent- and scratch-specific detection

Fine surface damage is hard to detect because of subtle appearance and scarce labelled data. Qu et al. [8] used GAN-based augmentation to synthesize scratch patterns for underrepresented classes; Strietzel et al. [9] used ControlNet diffusion models for the same purpose with a measurable gain on real evaluation data, which is the basis for the mixing ratio in (16). Baig et al. [10] built a dedicated dent-detection dataset and showed strong YOLOv8 precision and recall for real-time localization. Zhou et al. [11] target surface-quality inspection in industrial settings and demonstrate that classical computer-vision pipelines remain competitive for narrowly scoped, single-defect-type inspection tasks even as deep learning dominates the general case. Yang et al. [12] extend this line with a broader body-surface-defect classifier, reinforcing that dent and scratch detection benefits from dedicated, defect-specific model heads rather than a single generic classifier.

C. Repair-cost estimation

Zhang et al. [13] were among the first to link ResNet50 damage features to repair-cost ranges via transfer learning. Sailaja et al. [14] unified detection and cost prediction in a single framework to cut end-to-end latency. Ahaggach et al. [15] combined ontology-based reasoning with regression to impose domain constraints on predicted costs, an approach we adopt directly and formalize as the projection in (13). DeepClaim [16] integrated severity analysis and cost estimation end to end. Perez-Zarate et al. [17] report that automated cost estimates track certified appraiser estimates closely for common, well-represented damage types but diverge for rare or compound damage, echoing the multi-damage limitation identified in Section III.

D. Segmentation and transformer-based approaches

Yusuf et al. [18] applied real-time instance segmentation to identify damaged components individually. Panboonyuen et al. introduced SLICK [19], using selective localization and instance calibration for precise boundary detection, and ALBERT [20], applying bidirectional transformer encoders to part-level segmentation. Khan et al. [21] fused IMU, audio, and camera data via a multi-modal autoencoder to catch minor damage that vision alone misses, illustrating the value of sensor fusion where visual evidence is weak. Across this theme the pattern is consistent: transformer-based segmentation delivers the strongest boundary precision reported in

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

TABLE I
LITERATURE SURVEY SUMMARY

Theme	Representative work	Limitation
General detection	Amodu et al. [3]; Verma et al. [4]	No severity or cost stage
Dent & scratch	Qu et al. [8]; Baig et al. [10]	Single damage type only
Cost estimation	Zhang et al. [13]; Ahaggach et al. [15]	No detection integration
Segmentation / transformer	SLICK [19]; ALBERT [20]; Yusuf et al. [18]	High compute cost
Insurance / claims	DeepClaim [16]; Sailaja et al. [14]	Narrow scope, no fraud check

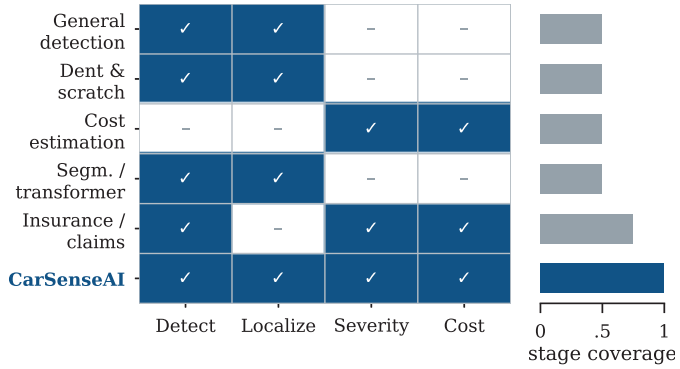

Fig. 1. Pipeline-stage coverage of the surveyed themes (Table I). A filled cell means the representative works of that theme address that stage. The bars give the fraction of the four stages covered. No surveyed theme covers all four; closing that gap is CarSenseAI's design target.

the survey, but at a computational cost that the reviewed papers themselves acknowledge is difficult to reconcile with real-time mobile inference, which is why CarSenseAI treats segmentation as a server-side stage rather than an on-device one (Section IX).

E. Comparative summary

Table I summarizes the four themes, their representative methods, and their principal limitation, and Figure 1 renders the same information as stage coverage. The recurring pattern is that most systems solve one stage of the pipeline rather than the whole workflow, and that the most accurate segmentation and transformer methods are too compute-heavy for direct mobile deployment.

III. Problem Statement and Research Gap

A. Problem statement

Vehicle damage assessment today depends largely on manual inspection by insurance surveyors, workshop technicians, and fleet-management personnel. This approach is naturally subjective, since different evaluators can reach different conclusions on the same damage; it is time-consuming; it is location-dependent, since it requires a trained inspector to be physically present; and it is susceptible to both honest human error and, in the insurance context, fraudulent misreporting. With millions of road accidents occurring worldwide each year and insurers processing correspondingly large claim volumes, the demand for an automated, unbiased, and scalable damage-evaluation system is increasingly acute.

B. Problem definition

The system must accept vehicle photographs and produce a complete damage-assessment result without a human evaluator in the visual-analysis step. Concretely it must (i) identify and classify the damage type or types present in a single image; (ii) localize the damaged region using bounding boxes and segmentation masks; (iii) grade severity on a consistent, reproducible scale; and (iv) convert the visual assessment into an estimated repair-cost range, across varying image quality and vehicle models, at latency low enough for interactive use. Section IV states this formally.

Given the survey in Section II, six recurring gaps motivate the design.

1) *Simultaneous multi-damage handling.* Most systems assume one dominant damage category per image, whereas real vehicles often exhibit several damage types at once.
2) *Incomplete pipelines.* Few studies connect detection, severity grading, and cost prediction into one deployable system.
3) *Dataset diversity.* Public datasets under-represent vehicle models, lighting conditions, viewpoints, and damage categories.
4) *Real-time deployability.* The most accurate architectures, particularly transformer-based ones, are too compute-heavy for mobile or edge deployment without significant optimization.
5) *Fraud detection.* Authenticity verification for detected damage is largely absent from existing insurance-oriented systems.
6) *Benchmark standardization.* The lack of public, universally accepted benchmarks makes cross-study comparison unreliable.

CarSenseAI is scoped to address gaps 1–4 in its current design; gaps 5 and 6 are flagged as open problems (Section IX).

IV. Formal Problem Formulation

Let $I \in \mathbb{R}^{H\times W\times 3}$ be a captured image and $m = (\mathit{make}, \mathit{model}, \mathit{year}, \mathit{region})$ the vehicle metadata supplied by the client. The assessment task is the composition

$$F = \Pi \circ g \circ f_{\text{sev}} \circ f_{\text{seg}} \circ f_{\text{det}}, \qquad F(I, m) = (\mathcal{D}, \mathcal{S}, [\hat{c}_-, \hat{c}_+]), \tag{1}$$

where $\mathcal{D}$ is the detection set, $\mathcal{S}$ the per-region severity grades, and $[\hat{c}_-, \hat{c}_+]$ the reported cost interval. Figure 2 shows the stages and the intermediate quantity each one produces.

A. Detection

The detector returns n triples

$$\mathcal{D} = \{(b_i, y_i, p_i)\}_{i=1}^{n}, \quad b_i \in \mathbb{R}^4,\ y_i \in \mathcal{Y},\ p_i \in [0,1], \tag{2}$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

with $\mathcal{Y} = \{scratch, dent, crack, structural\}$. Boxes are regressed with the complete-IoU objective, which adds a centre-distance and an aspect-ratio term to plain IoU so that non-overlapping predictions still receive gradient:

$$\mathcal{L}_{\mathrm{CIoU}} = 1 - \mathrm{IoU}(b,\hat{b}) + \frac{\rho^2(\mathbf{b},\hat{\mathbf{b}})}{d^2} + \alpha v, \tag{3}$$

where ρ is the centre distance, d the diagonal of the smallest enclosing box, $v = \frac{4}{\pi^2}\big(\arctan\frac{w}{h} - \arctan\frac{\hat{w}}{\hat{h}}\big)^2$ the aspect-ratio penalty, and $\alpha = v/(1-\mathrm{IoU}+v)$ its adaptive weight. The training objective is the usual composite

$$\mathcal{L}_{\mathrm{det}} = \lambda_b\mathcal{L}_{\mathrm{CIoU}} + \lambda_c\mathcal{L}_{\mathrm{cls}} + \lambda_d\mathcal{L}_{\mathrm{DFL}}, \tag{4}$$

with $\mathcal{L}_{\mathrm{cls}}$ a binary cross-entropy over classes and $\mathcal{L}_{\mathrm{DFL}}$ the distribution focal loss on the box-distance distribution. Overlapping predictions are reduced by non-maximum suppression: box b_j is discarded when

$$\exists\, b_k :\ p_k > p_j \ \wedge\ \mathrm{IoU}(b_j,b_k) > \tau_{\mathrm{nms}}, \qquad \tau_{\mathrm{nms}} = 0.45. \tag{5}$$

Detection quality is reported as mean average precision averaged over IoU thresholds,

$$\mathrm{mAP}_{50:95} = \frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}}\frac{1}{10}\sum_{t\in\{0.50,0.55,\ldots,0.95\}}\mathrm{AP}_y(t). \tag{6}$$

B. Segmentation and damage extent

Mask R-CNN is trained with the standard multi-task objective

$$\mathcal{L}_{\mathrm{seg}} = \mathcal{L}_{\mathrm{cls}} + \mathcal{L}_{\mathrm{box}} + \mathcal{L}_{\mathrm{mask}}, \tag{7}$$

where $\mathcal{L}_{\mathrm{mask}}$ is the per-pixel binary cross-entropy of the mask branch, applied only to the ground-truth class channel. For detection i lying on vehicle part k, the normalized damage extent is the mask area as a fraction of the part area,

$$\hat{a}_i = \frac{|M_i|}{|P_k|} \in [0,1], \qquad |M_i| = \sum_{u,v}\mathbf{1}[M_i(u,v)=1]. \tag{8}$$

Normalizing by part area rather than image area is what makes $\hat{a}_i$ comparable across capture distances and vehicle sizes.

C. Severity

Severity is first computed as a continuous score combining extent, estimated depth $\hat{d}_i$, and boundary irregularity $\hat{r}_i$ (perimeter over the perimeter of an equal-area disc, a proxy for tearing versus smooth deformation):

$$\sigma_i = w_1\hat{a}_i + w_2\hat{d}_i + w_3\hat{r}_i, \qquad \sum_{j=1}^{3} w_j = 1,\ w_j \ge 0. \tag{9}$$

The reported grade is the induced three-tier rule

$$s_i = \begin{cases} \textit{minor}, & \sigma_i < \tau_1, \\ \textit{moderate}, & \tau_1 \le \sigma_i < \tau_2, \\ \textit{severe}, & \sigma_i \ge \tau_2, \end{cases} \tag{10}$$

with τ_1, τ_2 calibrated on a validation split rather than fixed a priori. Figure 3 plots (9) against $\hat{a}$ for three depth values and shows the bands (10) induces. The CNN head that predicts s_i directly is trained with class-weighted cross-entropy, because minor damage dominates naturally collected data:

$$\mathcal{L}_{\mathrm{sev}} = -\sum_i\sum_c \omega_c\, y_{i,c}\log\hat{y}_{i,c}, \qquad \omega_c = \frac{N}{3N_c}, \tag{11}$$

where N_c is the number of training regions of class c and N the total.

D. Cost estimation under ontology constraints

The regressor maps severity and metadata to a raw cost, $\hat{c}_i^{\mathrm{raw}} = g(\sigma_i, y_i, m; \theta)$, trained with the Huber loss so that a handful of very expensive repairs do not dominate the gradient:

$$\mathcal{L}_\delta(e) = \begin{cases} \tfrac{1}{2}e^2, & |e| \le \delta_H, \\ \delta_H\big(|e| - \tfrac{1}{2}\delta_H\big), & |e| > \delta_H, \end{cases} \qquad e = \hat{c}^{\mathrm{raw}} - c. \tag{12}$$

The ontology contributes lower and upper bounds $\ell(k,y,m)$ and $u(k,y,m)$ for a given part, damage type, and vehicle, taken from labour-rate and parts tables. The reported estimate is the projection of the raw output onto that interval,

$$\hat{c}_i = \Pi_{k,y,m}\big(\hat{c}_i^{\mathrm{raw}}\big) = \min\big(\max(\hat{c}_i^{\mathrm{raw}},\ \ell(k,y,m)),\ u(k,y,m)\big), \tag{13}$$

which guarantees a prediction can never fall outside what domain knowledge admits, whatever the regressor extrapolates (Figure 4). For an image with several detections, part-level overlap is discounted so that two damages on one panel are not billed twice:

$$\hat{C} = \sum_{i=1}^{n}\hat{c}_i - \sum_k \gamma_k\Big(\sum_{i\in\mathcal{I}_k}\hat{c}_i - \max_{i\in\mathcal{I}_k}\hat{c}_i\Big), \tag{14}$$

where $\mathcal{I}_k$ indexes detections on part k and $\gamma_k \in [0,1]$ is the shared-labour fraction for that part. The interval finally shown to the user is $[\hat{C}(1-\delta),\ \hat{C}(1+\delta)]$ with δ set from validation residuals. Cost accuracy is reported against appraiser figures as

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\big|\hat{C}_i - C_i\big|, \qquad \mathrm{MAPE} = \frac{100}{N}\sum_{i=1}^{N}\frac{|\hat{C}_i - C_i|}{C_i}. \tag{15}$$

E. Augmentation and latency budget

Training data mixes real and synthetic images at a per-class ratio that depends on how under-represented the class is,

$$\beta_c = \min\left(\beta_{\max},\ \frac{N_{\max} - N_c}{N_c}\right), \tag{16}$$

so that scarce categories such as cracks receive proportionally more GAN/diffusion samples [8], [9] while common ones are left close to their natural distribution, with $\beta_{\max}$ capping the synthetic share.

End-to-end response time is additive over the stages,

$$T = T_{\mathrm{net}} + T_{\mathrm{pre}} + T_{\mathrm{det}} + T_{\mathrm{seg}} + T_{\mathrm{sev}} + T_{\mathrm{cost}} + T_{\mathrm{db}}, \tag{17}$$

and, since the AI layer is stateless per request, c inference workers with service rate μ under offered load λ give the response time

$$W = \frac{1}{\mu}\cdot\frac{1}{1-\rho}, \qquad \rho = \frac{\lambda}{c\mu} < 1, \tag{18}$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

which is what fixes the worker count needed to hold T under the two-second budget of Section VII at a target claim rate (Figure 6).

V. System Architecture

CarSenseAI follows a modular, layered architecture with four layers: Presentation, API Gateway, Processing, and Data (Figure 5). The layering is deliberate: it lets the mobile client, the AI pipeline, and the persistence layer evolve independently, so the detection model can be retrained and redeployed without a client release, and the client can add UI flows without touching inference code.

The Presentation Layer is the Flutter mobile application: it captures images, displays assessment reports, and exposes GPS-based mechanic discovery and SOS alerts. The API Gateway Layer, built on Node.js/Flask, routes requests, authenticates users, validates input, and formats responses. The Processing Layer hosts the AI pipeline and returns a structured report. The Data Layer, on Firebase/MySQL, stores user data, vehicle records, historical assessments, and the reference data behind $\ell(\cdot)$ and $u(\cdot)$ in (13), and drives push notifications through Firebase Cloud Messaging.

A. Implementation rationale

Each choice in Table III follows from a requirement rather than a default. Flutter was chosen over separate native codebases because it halves the maintenance surface for a two-platform target while still compiling to native ARM code, which matters for camera-preview and image-upload responsiveness. Node.js/Flask was chosen for the gateway because both have mature middleware for authentication, rate limiting, and request validation, which is what the gateway is responsible for. Firebase was chosen for authentication and messaging specifically, not as the system of record; MySQL remains the system of record for structured relational data such as historical assessments and cost reference tables, where transactional consistency and query flexibility matter more than real-time sync. YOLOv8 and Mask R-CNN were chosen over end-to-end transformer detectors because Section II shows transformer-based segmentation, while more accurate on boundary precision, is not yet practical for the latency budget in (17); this is a deliberate accuracy/latency trade-off, not an oversight.

VI. Methodology

Data collection and augmentation. Real-world images are supplemented with GAN- and diffusion-model-generated synthetic samples [8], [9] at the per-class ratio of (16), offsetting the scarcity of examples for uncommon damage categories.

Detection backbone. YOLOv8 performs damage detection and coarse classification under (4), chosen for fast real-time inference and strong small-object performance, consistent with Baig et al. [10].

Segmentation module. Mask R-CNN produces instance-level masks under (7), supplying the area measure $\hat{a}_i$ of (8) that severity and cost both depend on [18].

TABLE II
Functional Requirements

ID	Description
FR-01	Capture/upload vehicle images (JPEG, PNG) with automatic lighting and orientation normalization.
FR-02	Detect and classify damage types at $\geq 85\%$ accuracy on the evaluation set.
FR-03	Localize damage via bounding boxes (2) and segmentation masks (8).
FR-04	Classify severity into minor, moderate, severe per (10).
FR-05	Generate repair-cost ranges under the ontology constraints of (13).
FR-06	Suggest nearby mechanics via GPS, with contact info, distance, and ratings.
FR-07	Support SOS alerts with live location sharing to pre-configured contacts.
FR-08	Send service reminders based on mileage and manufacturer intervals.
FR-09	Provide basic repair guidance for common damage types.

TABLE III
Software Stack

Category	Technology	Purpose
Mobile framework	Flutter SDK	Cross-platform UI
Language	Dart	App logic and state
Deep learning	PyTorch/TensorFlow	Training and serving
Detection	YOLOv8	Real-time damage detection
Segmentation	Mask R-CNN	Pixel-level boundaries
Backend API	Node.js/Flask	Request orchestration
Realtime DB	Firebase	Auth, cloud messaging
Relational DB	MySQL	Persistent storage
Location	Google Maps API	Mechanic discovery
Notifications	FCM	Push delivery

Severity classification. A dedicated CNN implements (9)–(11), following the separated detection/severity design of Mohamad et al. [6], which improves overall pipeline accuracy relative to a single joint model.

Cost estimation. A regression module trained under (12), combined with the ontology projection (13) [15], converts severity and vehicle metadata into a bounded estimate, keeping predictions consistent with domain knowledge rather than purely data-driven extrapolation.

Interface and delivery. The pipeline is exposed via a REST API; the Flutter client uploads images and renders structured reports, severity grades, and cost ranges, alongside the GPS, SOS, and reminder features.

VII. Requirements Specification

A. Functional requirements

Table II lists the core functional requirements, derived from the problem definition in Section III.

B. Non-functional requirements

Interface. The client is built in Flutter for a single Android/iOS codebase, using material-design widgets and gesture handling, with an interface designed to minimize the steps between capture and result.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

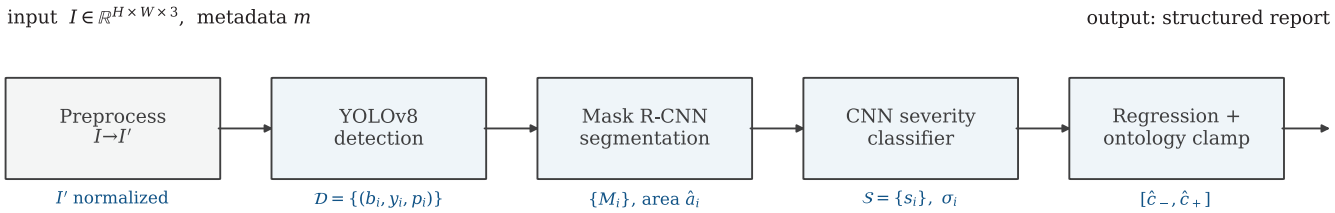

Fig. 2. Processing pipeline and the intermediate quantity produced by each stage, following the composition in (1).

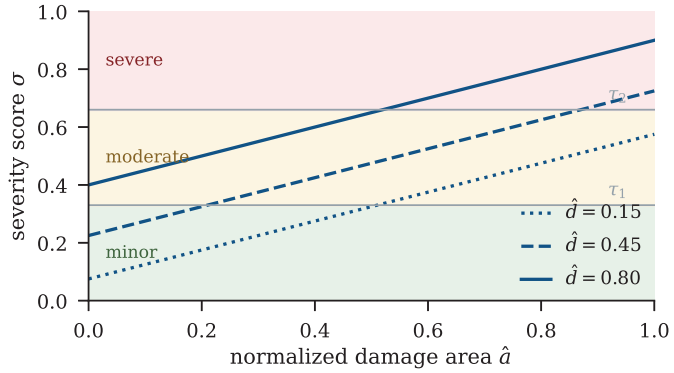

Fig. 3. Severity score σ of (9) against normalized damage area $\hat{a}$ for three depth values, with $w = (0.5, 0.3, 0.2)$ and, for illustration, $\hat{r} = \hat{d}$. Shaded bands are the decision regions of (10); τ_1 and τ_2 are calibrated, not fixed.

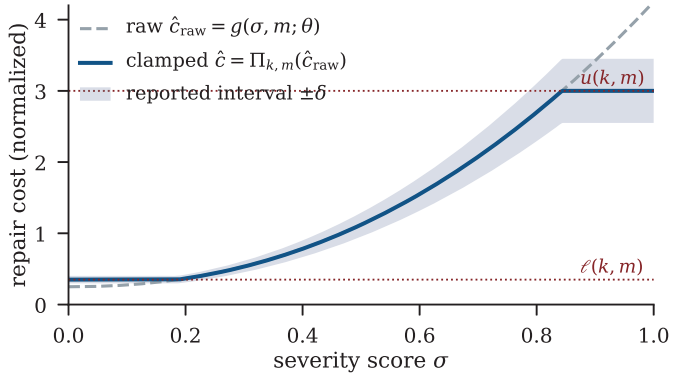

Fig. 4. Effect of the ontology projection (13). The raw regressor output is free to extrapolate; the reported estimate is confined to $[\ell(k, y, m), u(k, y, m)]$, and the shaded band is the interval finally shown to the user. Costs are normalized, not currency values.

Software stack. Table III lists the technology choices and their role.

Hardware. Development requires ≥ 16 GB RAM, a CUDA-capable GPU with ≥ 6 GB VRAM, 256 GB SSD storage, and a multi-core processor. Deployment targets an Android device with ≥ 4 GB RAM, a rear camera ≥ 8 MP, and Android 8.0+; the backend requires ≥ 8 GB RAM and a GPU with ≥ 4 GB VRAM per inference worker.

Latency. The end-to-end budget is $T \leq 2$ s at the 95th

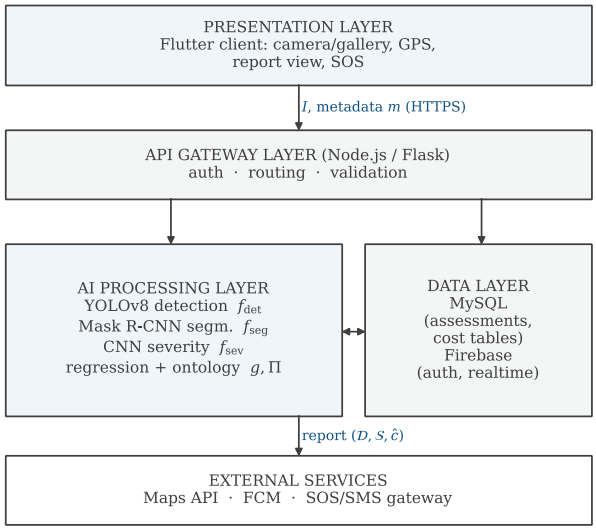

Fig. 5. Layered system architecture. The AI Processing Layer implements the maps of (1); it holds no per-request state, which is what permits the horizontal scaling modelled by (18).

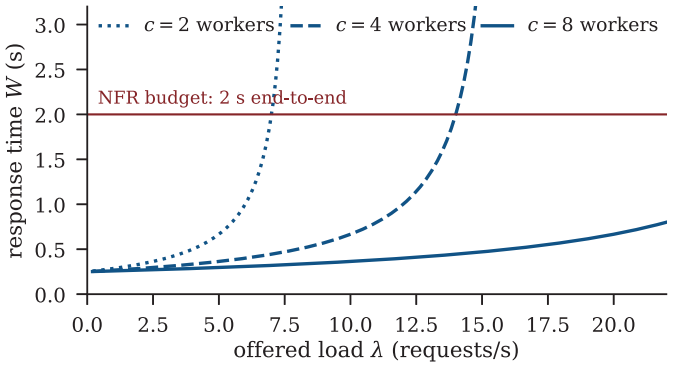

Fig. 6. Worker count required to hold the two-second response budget under (18), for a per-worker service rate of $\mu = 4$ requests/s. These are design-time budget curves from the queueing model, not measurements.

percentile, decomposed as in (17); Figure 6 shows the worker count (18) implies for a given claim rate.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

C. Security, privacy, and scalability

Because CarSenseAI handles vehicle photographs, location data, and, in the insurance-claim case, information tied to a claim record, three cross-cutting concerns shape the design. *Data protection:* images and location data travel over authenticated, encrypted API calls, and the Data Layer separates identity and auth (Firebase) from claim and assessment records (MySQL), limiting the blast radius of any single credential compromise. *Access control:* the gateway enforces per-user authorization on every request, so a client can retrieve only its own assessment history and cannot enumerate other users' records. *Scalability:* the AI Processing Layer is stateless per request, so inference workers scale horizontally behind the gateway during peak claim periods, for instance after severe weather, with the relation between load, worker count, and response time given by (18).

VIII. System Model

The principal use cases, each with its main actor and its pre/postcondition pair: registration and authentication (owner; valid email or phone $\rightarrow$ authenticated session); image capture and upload (owner; session and camera access $\rightarrow$ image queued); damage detection (system; valid image $\rightarrow$ labelled boxes (2)); severity evaluation (system; segmented regions $\rightarrow$ a grade per region (10)); repair-cost prediction (system; grade and metadata $\rightarrow$ bounded estimate (13)); nearby-mechanic discovery (owner; location permission $\rightarrow$ ranked list); SOS alert (owner; configured contacts $\rightarrow$ contacts notified with live location); service-reminder management (owner; mileage on record $\rightarrow$ scheduled notification); and repair-guidance access (owner; completed report $\rightarrow$ guidance for the detected damage type). Each is traceable back to a functional requirement in Table II.

IX. Evaluation Plan and Threats to Validity

A. Planned evaluation methodology

Because CarSenseAI is at the design-and-implementation stage, this paper specifies the protocol the system is to be assessed against rather than reporting field results. Detection and segmentation are to be evaluated with (6) and with mask-level IoU, against a held-out test split stratified by damage type, vehicle model, and lighting condition, so that accuracy is not dominated by whichever category is best represented in training. Severity is to be evaluated with per-class precision, recall, and F1, since the three classes are unlikely to be balanced in naturally collected data; the class weights of (11) address the same imbalance during training. Cost estimation is to be evaluated with (15) against certified appraiser estimates on a shared validation subset, which is how the reviewed literature [13], [17] validates automated cost models against a human baseline. FR-02's 85% accuracy target is the acceptance threshold for the detection stage before a build is eligible for pilot deployment.

B. Threats to validity

Multi-damage handling: the detection stage is not yet validated on images containing several concurrent damage types, and the overlap discount (14) is specified but its γ_k are not yet fitted. *Deployability:* transformer-based segmentation is the most accurate option surveyed but remains too compute-heavy for on-device inference, so CarSenseAI relies on server-side inference. *Dataset diversity:* synthetic augmentation mitigates but does not eliminate the shortage of labelled data. *Fraud detection:* authenticity verification of submitted images is not yet part of the pipeline and is a priority for the insurance use case. *Benchmarking:* planned evaluation will use internally curated data rather than a standardized public benchmark, which limits direct comparison with prior systems until such a benchmark exists.

Planned extensions are a multi-label detection head for concurrent damage types; model compression and quantization to move segmentation on-device; expanded training data across makes and imaging conditions; and an authenticity-verification module for fraud-resistant insurance workflows.

X. Conclusion

CarSenseAI is a complete, deployable pipeline for vehicle damage assessment that composes YOLOv8 detection, Mask R-CNN segmentation, CNN severity classification, and ontology-constrained regression cost estimation (1), wrapped in a Flutter client with GPS, SOS, and reminder features. Against a survey of 41 studies from 2019–2026, most existing systems address a single stage of this workflow; the contribution here is to integrate detection, localization, severity, and cost into one production-oriented, camera-only system with an explicit stage-wise formulation, in which the ontology projection (13) keeps cost predictions inside limits domain knowledge admits and the queueing relation (18) ties the accuracy/latency trade-off to a concrete deployment budget. Concurrent multi-damage detection, on-device deployment cost, dataset diversity, and fraud detection define the immediate next steps.

References

[1] D. H. Tran et al., "A survey on vehicle damage detection using deep learning," 2025.
[2] M. Hasan et al., "Vehicle damage detection using artificial intelligence: a systematic literature review," 2025.
[3] O. D. Amodu et al., "A comprehensive evaluation of deep learning models for vehicle damage detection," 2025.
[4] N. Verma et al., "Car damage detection analysis using deep learning and computer vision techniques," 2025.
[5] D. Lee et al., "Automated vehicle damage classification using the three-quarter view car damage dataset and deep learning approaches," 2024.
[6] A. T. Mohamad et al., "Car damage severity assessment using supervised deep learning," 2024.
[7] R. E. van Ruitenbeek, "Vehicle damage detection using deep convolutional neural networks," 2019.
[8] G. Qu et al., "Automotive scratch detection: a lightweight convolutional network approach augmented by generative adversarial learning," 2025.
[9] J. Strietzel et al., "AI scratching your car: using diffusion models for training data generation in automotive damage detection," 2024.
[10] D. Z. Baig et al., "Small dents, big impact: a dataset and deep learning approach for vehicle dent detection," 2025.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 09 , September - 2026

[11] Q. Zhou et al., "An automatic surface defect inspection system for automobiles," 2019.
[12] Z. Yang et al., "Research on detection and classification of automotive body surface defects," 2025.
[13] M. Zhang et al., "AI-based vehicle damage repair price estimation system," 2023.
[14] N. V. Sailaja et al., "Enhanced deep learning techniques for intelligent vehicle damage detection and cost estimation," 2025.
[15] H. Ahaggach et al., "Enhancing car damage repair cost prediction: integrating ontology reasoning with regression models," 2024.
[16] "An AI-driven framework for automated vehicle damage severity assessment and insurance cost estimation (DeepClaim)," 2025.
[17] S. A. Perez-Zarate et al., "Automated car damage assessment using computer vision," 2024.
[18] S. A. Yusuf et al., "Automotive parts assessment: applying real-time instance segmentation for automated vehicle damage assessment," 2022.
[19] T. Panboonyuen et al., "SLICK: selective localization and instance calibration for car damage segmentation," 2025.
[20] T. Panboonyuen et al., "ALBERT: advanced localization and bidirectional encoder representations for transformer-based car damage segmentation," 2025.
[21] S. Khan et al., "Robust anomaly detection through multi-modal autoencoder fusion for small vehicle damage detection," 2025.