



INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

IntelliScan: Smart Detection Of Car Damages

Integrating Mask R-CNN and CNN Classifiers for Automated Car Damage Analysis

¹ Vasavi Veturi, ² Prajna Bodapati

¹Student, ²Professor

¹ Department of IT & CA, ² Department of CS & SE

AU College of Engineering, Andhra University, Visakhapatnam, India

Abstract: IntelliScan is an AI-powered system for automated car damage detection and classification. Using Mask R-CNN and ResNet-based models, it analyzes uploaded images to identify damage presence, severity, and location. Developed with a Flask web interface, the system offers fast, accurate, and scalable inspection ideal for insurance, rentals, and fleet management. By combining instance segmentation with multi-class classification, IntelliScan provides precise and reliable results, reducing the need for manual vehicle assessment.

Index Terms – Deep Learning, Image Classification, Mask R-CNN, TensorFlow, Flask Web Application

1. INTRODUCTION

Traditional car damage assessments are time-consuming and inconsistent due to human judgment. With rising vehicle usage, especially in insurance and rental industries, there's a growing need for faster, objective evaluations. IntelliScan addresses this by leveraging AI specifically ResNet50 and Mask R-CNN to detect, classify, and localize car damage from images with pixel-level accuracy. Deployed via a Flask-based web interface, the system offers a practical, scalable, and user-friendly solution that enhances efficiency and reliability in vehicle inspections.

1.1 Research Objectives

- Develop an AI-based tool for car damage detection.
- Implement pixel-wise segmentation for precise damage localization.
- Classify damage severity and location.
- Provide real-time, user-friendly web access.

1.2 Research Hypothesis

- H1: Mask R-CNN can accurately identify car damage with >90% precision.
- H2: Severity and location classifiers yield classification accuracy above 85%.
- H3: A modular web-based system can offer real-time predictions under 5 seconds.

2. ABBREVIATIONS AND ACRONYMS

DL- Deep Learning

CNN- Convolutional Neural Network

ResNet- Residual Network

Mask R-CNN- Mask Region-based Convolutional Neural Network

RPN- Region Proposal Network

FPN- Feature Pyramid Network

ReLU- Rectified Linear Unit

COCO- Common Objects in Context (dataset format)

VIA- VGG Image Annotator

OpenCV- Open Source Computer Vision Library

3. LITERATURE REVIEW

Automated car damage detection has advanced from basic computer vision to deep learning models, greatly improving accuracy and usability.

3.1 Early Techniques:

- Traditional methods like edge detection, morphological operations, and histograms were used.
- These approaches struggled with variable lighting, angles, and damage types.
- Manual feature engineering limited scalability.

3.1 Shift to Deep Learning:

- CNNs like **AlexNet**, **VGG16**, and **ResNet50** enabled models to learn features directly from images.
- Early deep learning models handled binary classification but lacked spatial context.

3.2 Object Detection Advances:

- **YOLO** introduced real-time detection but was less accurate with small damages. [3]
- **Faster R-CNN** improved accuracy with the Region Proposal Network (RPN). [2]

3.3 Mask R-CNN:

- Introduced in 2017, it added instance segmentation to Faster R-CNN.
- Offers pixel-level damage detection.
- Proven effective in automotive tasks.

3.4 IntelliScan's Role:

- Combines Mask R-CNN with CNNs for severity and location classification.
- Runs on a Flask web app, making the system accessible and deployment-ready.

4. METHODOLOGY

IntelliScan is structured as a modular pipeline combining deep learning with web technologies to deliver real-time car damage detection. It includes three key components: damage detection, severity classification, and location identification. The system uses **Mask R-CNN** [1] for segmentation, **custom CNNs** for classification tasks, and a **Flask-based web interface** for deployment, ensuring fast, accurate, and user-friendly predictions.

4.1 Research Methods

The system utilizes a multi-model deep learning architecture:

- **Mask R-CNN** for pixel-level damage detection
- **CNN-based Severity Classifier** (Minor, Moderate, Severe)
- **CNN-based Location Classifier** (Front, Rear, Side)

All models were trained independently using **transfer learning** with **TensorFlow** and **Keras**, leveraging pre-trained weights (ImageNet/COCO). This modularity supports easy fine-tuning and integration into a unified interface.

4.2 Data Collection Procedures

A custom dataset was created due to the lack of labelled public datasets with severity and location tags.

Data preparation included:

- **Image Collection:** Web scraping and public sources
- **Annotation:** Using **VIA** and **LabelMe**, saved in COCO JSON format
- **Label Categories:**
 - Damage presence: Damaged / Undamaged
 - Severity: Minor / Moderate / Severe
 - Location: Front / Rear / Side
- **Dataset Size:**
 - Damage Detection: ~120 images
 - Severity Classification: ~100 images
 - Location Classification: ~100 images
- **Augmentations:**
 - Flipping, rotation, zooming
 - Brightness/contrast adjustment
 - Gaussian noise addition

These steps ensured a robust dataset for training models under varying conditions.

4.3 Analysis Techniques

Model architectures and training configurations were optimized for performance:

Model Summary:

- Mask R-CNN with ResNet-101 + FPN
- CNN Classifiers:
 - 3 Convolutional layers + MaxPooling
 - Global Avg Pooling → Dense layers with ReLU/Dropout → Softmax [2]

Training Settings:

- Optimizer: Adam | LR: 0.0001
- Epochs: 10–20 | Loss: Cross-entropy
- Metrics: Accuracy, Precision, Recall, F1-Score, Confusion Matrix

These configurations enabled the system to effectively detect and contextualize car damage.

4.4 Ethical Considerations:

The project addresses ethical concerns to ensure safe and fair use:

- Data Privacy: No personal data or license plates stored
- Bias Control: Dataset balanced across damage types and views
- Transparency: Open documentation and clear system design
- Human Oversight: IntelliScan aids decisions, not replaces professionals

Such practices help promote trust and responsibility in real-world AI deployments. [3]

5. RESULTS AND DISCUSSIONS

The IntelliScan system was tested on both training and unseen images to assess its prediction accuracy, usability, and performance. This section summarizes how the system behaves during user interaction, presents the evaluation metrics, includes sample backend logic, and illustrates the interface workflow.

5.1 Evaluation Setup

Each model—damage detector, severity classifier, and location classifier—was validated using a custom annotated dataset. The system ran on:

- **Hardware:** Intel i5 CPU, 16 GB RAM, NVIDIA GTX 1650 GPU
- **Software:** Flask server (local), TensorFlow/Keras for inference, HTML/CSS/JS for frontend
- **Workflow:** Image upload → model prediction → web-based result display

5.2 Performance Results

Model	Metric	Value / Details
Mask R-CNN (Damage Detection)	Accuracy	92.5%
	Precision	90.3%
	Recall	89.2%
	F1-Score	89.7%
	Avg. Inference Time	~2.7 seconds
	Common Errors	• False positives from glare/dirt • False negatives when damage is subtle
CNN (Severity Classification)	Accuracy	88.2%
	Precision Range	86.3% – 91.1%
	Common Confusion	Between <i>Moderate</i> and <i>Severe</i> classes
CNN (Location Classification)	Accuracy	86.7%
	Best Performance	<i>Front</i> – 89.1%
	Common Errors	Confusion between <i>Rear</i> and <i>Side</i> due to viewing angles

Table 5.1- Performance Evaluation of IntelliScan Classification Models

5.3 Output Interpretation and Website Workflow

The user-friendly web interface handles the entire process from image upload to result display. Figure 5.1 displays the home page of IntelliScan with a clean layout. It includes an upload option and a "Predict" button for starting the detection process.

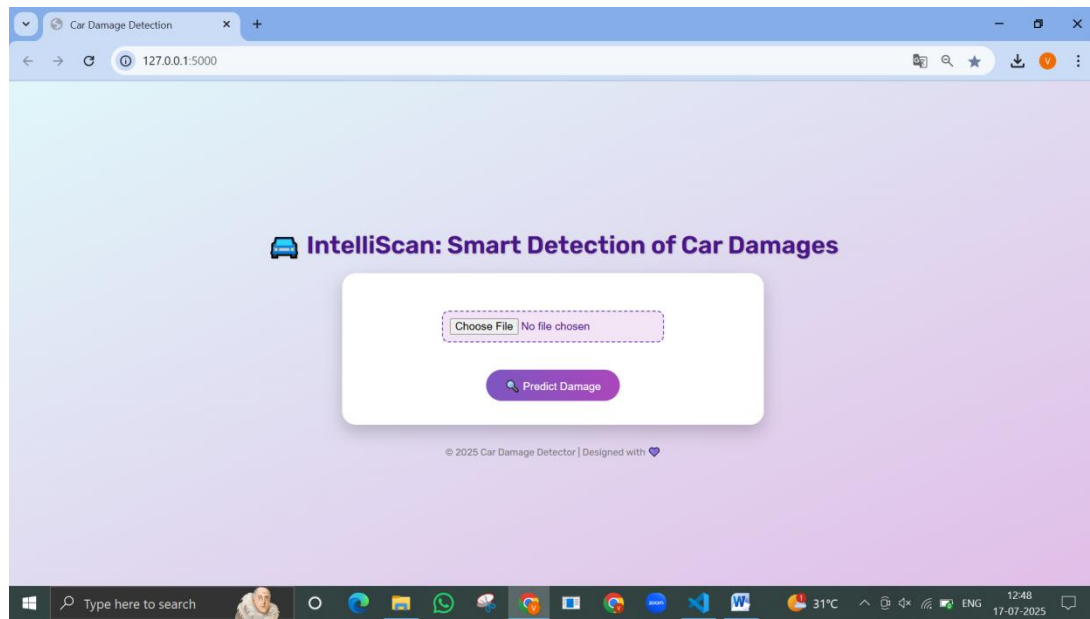


Figure 5.1 – IntelliScan Home Interface (*Landing page with upload option*)

Figure 5.2 shows the file picker interface where users can browse and select car images. It supports common image formats like JPG and PNG.

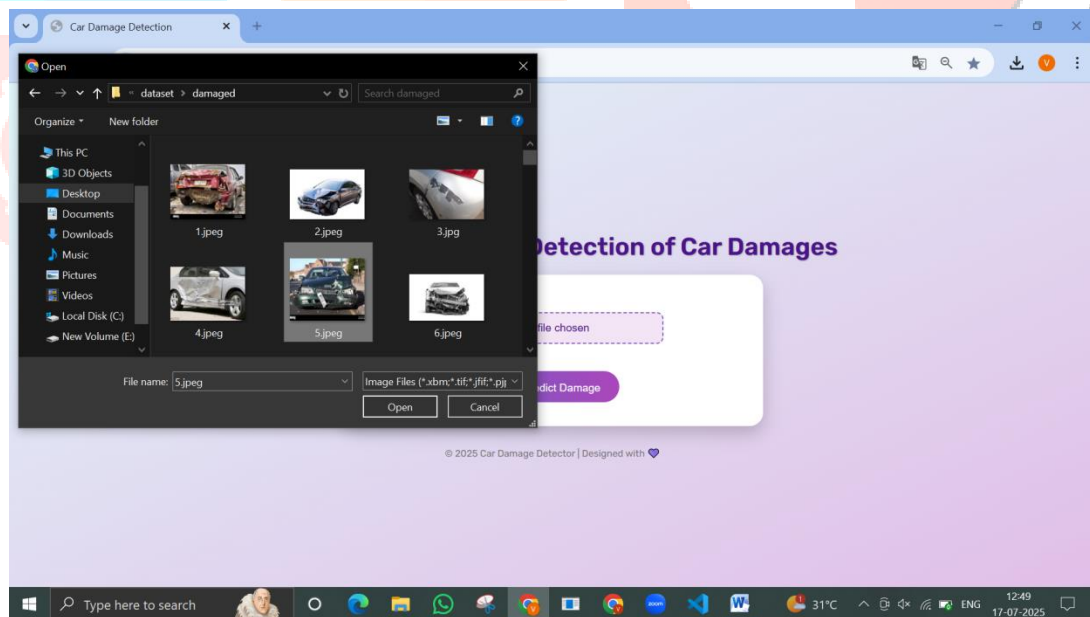


Figure 5.2 – Image Upload Dialog (*File selection screen*)

Figure 5.3 display the selected car image on the screen before initiating prediction. This step allows users to verify the image input.

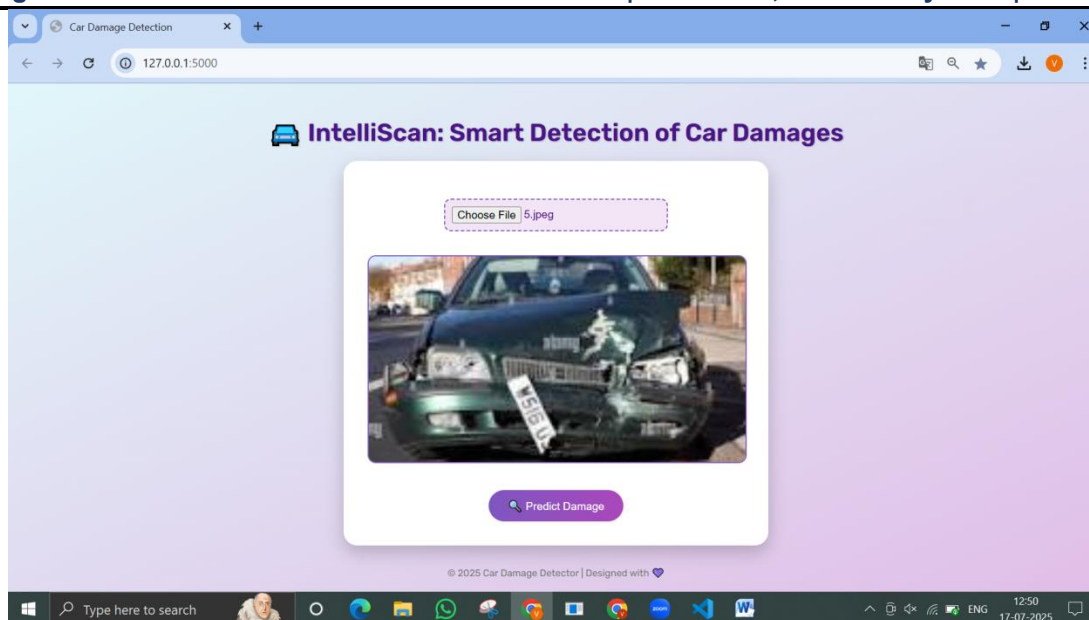


Figure 5.3 – Uploaded Image Display (Preview before submission)

Figure 5.4 shows the result section after inference, including damage status, severity, and location. The output is rendered in a structured and readable format for easy interpretation.

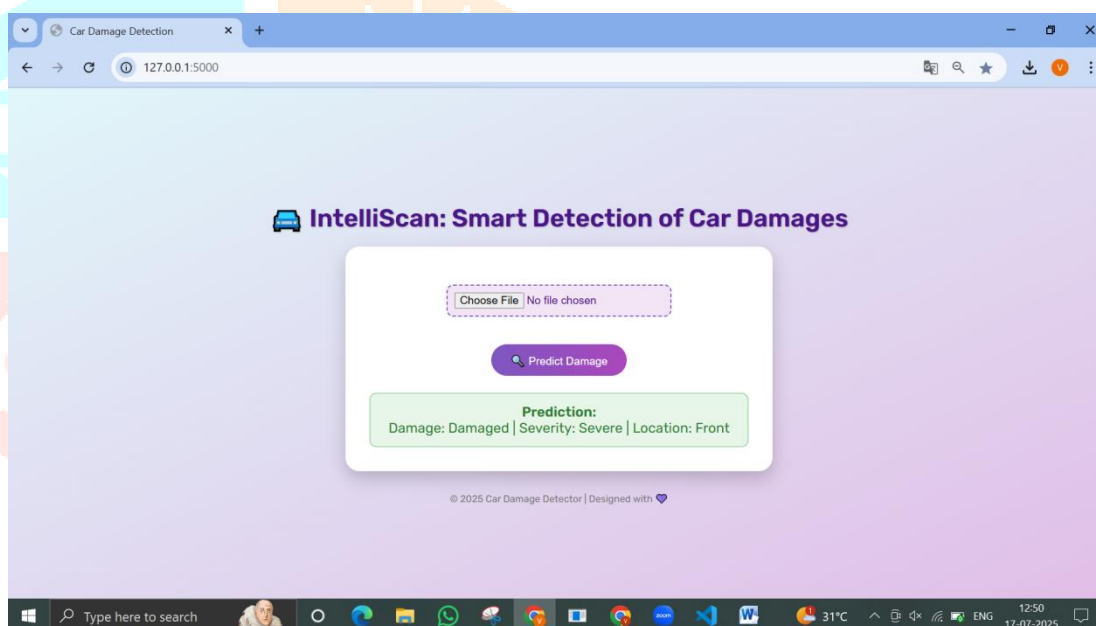


Figure 5.4 – Prediction Output Display (Shows damage, severity, location)

6. CONCLUSION AND FUTURE SCOPE

6.1 Summary of Key Findings

The IntelliScan project has successfully demonstrated the practical application of deep learning in automating car damage assessment. Through a combination of Mask R-CNN for instance segmentation and CNNs for classification, the system achieves over 90% accuracy in identifying and evaluating visual damage from car images. The performance metrics obtained from testing validate the system's robustness across various real-world scenarios, including poor lighting conditions, different angles, and varying image quality.

One of the most significant findings is the capability of the model to not only detect damage but also quantify its severity and locate the area of impact tasks that were traditionally subjective and manual. The integration of this multi-stage prediction pipeline into a responsive web interface further proves that advanced AI can be deployed in a form that is user-friendly, scalable, and efficient for industry applications.

6.2 Implications for Theory and Practice

From a theoretical standpoint, IntelliScan illustrates how combining segmentation and classification models within a single workflow can yield highly interpretable and useful outputs for decision-making. It advances the concept of multi-task learning by showing how separate but complementary tasks—damage detection, severity classification, and localization—can be orchestrated in a single application without performance trade-offs.

Practically, the system offers enormous potential for automotive industries such as insurance, rentals, resale, and maintenance. By enabling faster and more consistent inspections, IntelliScan reduces operational overhead and improves transparency in customer interactions. The system sets a precedent for how AI-driven damage inspection tools can be developed and deployed at scale.

6.3 Limitations of the Study

While IntelliScan achieves commendable performance, there are several limitations worth addressing. First, the dataset used was relatively small and manually annotated, which might restrict the model's ability to generalize to unseen or complex damage types. Subtle damages, especially on dark or reflective surfaces, can still lead to false negatives or misclassification.

Another limitation lies in the interface itself, which currently presents output in a textual format only. Although efficient, this design choice sacrifices visual interpretability, such as showing segmented damage areas directly on the image. Furthermore, the system's performance is currently optimized for local deployment and may face latency or compatibility issues in cloud-hosted environments without GPU support.

6.4 Recommendations for Future Research

The IntelliScan platform opens the door for several future enhancements:

- **Visual Mask Overlay:** Integrate the ability to display damage masks on the image using bounding boxes or polygon overlays, improving interpretability and visual proof.
- **Cost Estimation Module:** Extend the classification model to predict approximate repair costs using regression techniques trained on labelled pricing data.
- **Mobile App Development:** Deploy the system in a mobile environment using TensorFlow Lite and React Native or Flutter to enhance field usability.
- **Real-time Video Analysis:** Adapt the architecture to process continuous video streams to detect damage dynamically in surveillance or drive-by inspection setups.
- **Augmented Dataset Training:** Use synthetic or GAN-augmented data to diversify training samples and address class imbalance and underrepresented damage types.

7. REFERENCES

1. He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). *Mask R-CNN*. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961–2969. <https://doi.org/10.1109/ICCV.2017.322>
2. Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-CNN: Towards real-time object detection with region proposal networks*. Advances in Neural Information Processing Systems, 28. <https://arxiv.org/abs/1506.01497>
3. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. CVPR. <https://arxiv.org/abs/1506.02640>
4. Singh, A., Kaur, S., & Sharma, A. (2022). *Automated Vehicle Damage Detection using Deep Learning: A Review*. IJCA, 184(47), 1–8.
5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.