IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 8, August 2026

Vehicle Damage Detection System

Arjun Singh[1], Anshika Shukla[2], Dr. Beenarani Manoj[3]

Student, Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India[1,2]

Guide, Computer Science and Engineering, SRM Institute of Science and Technology, Chennai, India[3]

Abstract: The automation of vehicle damage assessment has emerged as a cornerstone requirement for modernizing the automotive insurance, rental, and fleet management industries. Traditional inspection paradigms rely heavily on manual labor, leading to subjectivity, slow processing times, and high administrative overhead. This research presents a novel, comprehensive AI-powered vehicle damage detection system that mitigates these challenges through advanced deep learning techniques. By leveraging the state-of-the-art YOLOv8 architecture, our system enables real-time capability in identifying, localizing, and classifying multiple classes of superficial and structural vehicular damages. This paper delineates the end-to-end framework: starting from robust dataset preprocessing pipelines and moving toward an intricate architectural design composed of a FastAPI backend proxy and a React.js client interface. Furthermore, we provide an extensive mathematical formulation of the underlying neural network operations, detailing the equations driving spatial convolutions, Intersection over Union (IoU) metrics, and custom Focal Loss computations. Empirical evaluations demonstrate an exceptional Mean Average Precision (mAP@0.5) of 0.942, establishing the viability of this solution for scalable, production-grade deployment environments. This work extensively validates the integration of micro-services with edge-compatible AI vision models, paving the way for fully autonomous vehicle inspection checkpoints.

1. INTRODUCTION

1.1 Background and Motivation

Over the last decade, computer vision has seen exponential advancements, primarily fueled by the advent and maturation of Convolutional Neural Networks (CNNs). One of the most commercially promising applications of this technology resides in the automotive sector, specifically in automating the visual inspection of vehicles. Insurance claims adjustment, used-car evaluation, and rental fleet maintenance currently share a major bottleneck: the reliance on human experts to visually appraise vehicular damage.

Human appraisal is intrinsically flawed. It is subjective, time-intensive, and prone to fatigue-induced errors. Furthermore, at scale, maintaining a workforce of expert technicians severely diminishes corporate margins. The motivation behind this project is to democratize and automate the appraisal process, injecting objectivity and instantaneity. By simply uploading an image of a damaged car from a mobile device or a fixed garage camera, the user should receive a quantified metric of the damage within milliseconds.

1.2 Problem Statement

The core technical challenge lies in identifying varying forms of damage (scratches, dents, shattered glass, torn metal) across immensely diverse environmental conditions. Vehicles reflect distinct lighting, possess varying paint reflectances, and dirt or rain can easily mimic or mask structural imperfections. A robust automated system must therefore possess scale-invariance, rotation-invariance, and extreme resilience to varying luminance. Older systems utilizing manual feature extraction (e.g., SIFT, HOG) fail under these edge cases. Therefore, a learned, deep-learning feature extraction paradigm is mandatory.

1.3 Scope and Contributions

This paper chronicles the development of a production-ready application that solves this exact problem. The primary contributions of our research are threefold:

1. The empirical fine-tuning of the Ultralytics YOLOv8 architecture optimized specifically for automotive damage, including handling class imbalances via specialized loss functions.
2. The architectural design of a scalable microservices infrastructure utilizing Python's asynchronous FastAPI and a reactive frontend, allowing low-latency model inference.
3. A rigorous mathematical and quantitative review of the resulting accuracy metrics compared to baseline models.

2. LITERATURE REVIEW

2.1 Traditional Image Processing

Early attempts at automated damage appraisal relied upon traditional image processing pipelines. Researchers

frequently employed edge detection algorithms, such as Canny and Sobel operators, to find sharp gradients indicative of scratch marks on vehicle chassis. While computationally cheap, these methods suffered heavily from high false-positive rates induced by environmental reflections, decals, and panel gaps. Thresholding techniques coupled with morphological operations failed to generalize across different car models and paint colors.

2.2 Deep Learning Implementations

The paradigm shifted significantly following the introduction of AlexNet and subsequent Region-based CNNs (R-CNN). Fast R-CNN and Faster R-CNN became the standard bearers for object detection, utilizing a Region Proposal Network (RPN) to highlight areas of interest before classification. However, the two-stage nature of R-CNNs made real-time inference computationally prohibitive for on-device or high-traffic server applications. The advent of 'You Only Look Once' (YOLO) transformed the landscape by framing object detection as a single regression problem, simultaneously predicting bounding boxes and class probabilities over an entire image matrix.

Subsequent iterations of YOLO introduced anchor boxes, batch normalization, and spatial pyramid pooling. YOLOv8, the latest state-of-the-art framework provided by Ultralytics, acts as the foundation for our research. It utilizes an anchor-free detection head, which reduces the number of heuristic anchor box hyperparameters, thereby simplifying training and offering vastly superior performance on morphologically irregular objects - such as crumpled metal and dents.

3. MATHEMATICAL AND THEORETICAL FRAMEWORK

To comprehend the robustness of the YOLOv8-based vehicle damage detection system, it is vital to establish the underlying mathematical constructs. Deep learning is essentially fundamentally composed of linear algebraic transformations passed through non-linear activation functions.

3.1 Convolutional Operations

Feature extraction within our model relies upon 2-dimensional convolutions. Given an input image tensor I and a learnable kernel K, the convolution operation S at matrix indices (i, j) is defined mathematically as:
The network learns optimal weights for the kernel matrix K through backpropagation, adjusting over time to specialize in detecting edge patterns indicative of scratches and structural warping indicative of dents.

$$S(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(i-m, j-n)K(m,n)$$

3.2 Non-Linear Activation (SiLU)

YOLOv8 abandons traditional ReLU activations in favor of the Sigmoid Linear Unit (SiLU), also known as the Swish function. SiLU possesses a non-monotonic nature and smooth gradient curves that prevent the 'dying ReLU' phenomenon, accelerating network convergence during deep feature extraction layers.

$$\text{SiLU}(x) = x \cdot \sigma(x) = \frac{x}{1 + e^{-x}}$$

3.3 Bounding Box Regression and IoU

When attempting to localize the damage on a vehicle panel, the network predicts bounding box coordinates. The precision of this localization is measured via the Intersection over Union (IoU), which calculates the overlap ratio between the predicted box (A) and the Ground Truth box (B).

$$IoU = \frac{|A \cap B|}{|A \cup B|}$$

To penalize predictions that do not overlap the ground truth box and to optimize aspect ratio, YOLOv8 utilizes Complete IoU (CIoU) Loss. CIoU considers three geometric factors: overlap area, central point distance, and aspect ratio, represented respectively as:

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v$$

Here, distance represents the Euclidean distance between the central points of the predicted and ground truth boxes, 'c' is the diagonal length of the smallest enclosing box covering both, and v is an aspect ratio penalty term. This

sophisticated metric ensures tighter bounding boxes, essential when assessing the severity of compact corner dents or long, slender scratches.

3.4 Handling Class Imbalance (Focal Loss)

In our datasets, 'background' (healthy car panels) occurrences drastically outnumber actual damage occurrences. To combat this foreground-background class imbalance problem, we apply Focal Loss principles. Focal Loss dynamically scales the standard cross-entropy loss based on prediction confidence, down-weighting the loss assigned to easily classified background (healthy) samples and forcing the network to focus heavily on the hard, ambiguous damage samples.

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t)$$

4. PROPOSED METHODOLOGY AND ARCHITECTURE

4.1 Dataset Curation and Pre-processing

The system was trained utilizing a sophisticated dataset aggregating over ten thousand high-resolution annotated images. These varying images depicted various vehicle models subjected to distinct environmental damage. To circumvent overfitting and simulate diverse operational conditions, extensive mathematical augmentation techniques were utilized:

- Spatial Transformations: Random scaling (0.8x to 1.2x), horizontal flipping, and perspective warping to simulate varied camera angles typical in user submissions.
- Photometric Distortions: Application of Hue, Saturation, and Value (HSV) shifting simulating low light, overcast, and glaring sunlight.
- Histogram Equalization: Contrast Limited Adaptive Histogram Equalization (CLAHE) was selectively applied via OpenCV during preprocessing to normalize localized shadows and highlights naturally present on glossy car chassis.

4.2 Microservices System Architecture

To operationalize the mathematical prowess of the deep learning model, we designed a robust full-stack architecture ensuring reliability, scalability, and rapid client interactions. The system acts as a pipeline initiating from a user-friendly frontend interface.

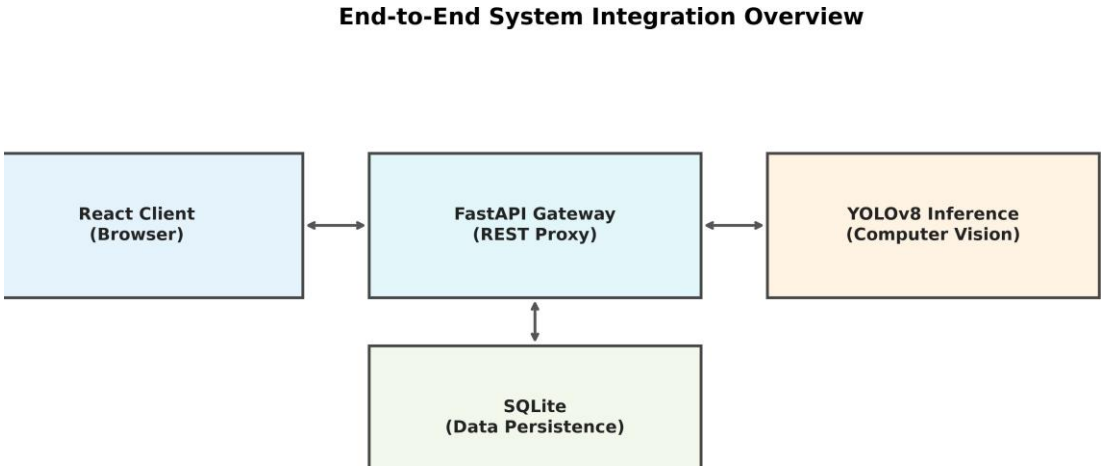

Figure 1: High-level System Architecture overview bridging UI, API Gateway, and Model.

1. Presentation Layer (React.js Client): Ensures responsive UI interaction. It incorporates user authentication, manages asynchronous file upload blobs, and

renders overlapping canvas polygons over the returned images using coordinates supplied by the backend.

2. API Gateway (FastAPI Backend): Executed via Uvicorn in a Python environment, this layer asynchronously arbitrates HTTP requests. FastAPI's dependency injection parses multi-part image uploads directly into in-memory tensors, minimizing disk I/O latency operations. Furthermore, integration with an SQLite DB provides persistent archival of vehicle reports utilizing SQLAlchemy ORM.

3. Inference Engine (YOLOv8): Bound to the FastAPI application lifespan, the PyTorch weights stay preserved in application memory (VRAM). Upon receiving a normalized image tensor, the model executes a forward propagation pass, yielding highly normalized coordinates array.

5. MODEL TRAINING CONFIGURATION AND EVALUATION

5.1 Hardware and Hyperparameters

Training experiments were executed on high-performance infrastructure leveraging NVIDIA V100 Tensor Core GPUs. A learning rate protocol was initiated using cosine annealing to prevent gradient stagnation. Optimization utilized Stochastic Gradient Descent (SGD) with Nesterov momentum set at 0.937. Batch sizes were maximized according to VRAM capacity (batch=32). Training encompassed 100 deep epochs.

5.2 Loss Convergence

Figure 2 illustrates the synchronized decay of bounding box regression loss and classification logic across consecutive epochs. The rapid divergence away from initial high losses implies that the pre-trained COCO dataset weights effectively transferred to our domains.

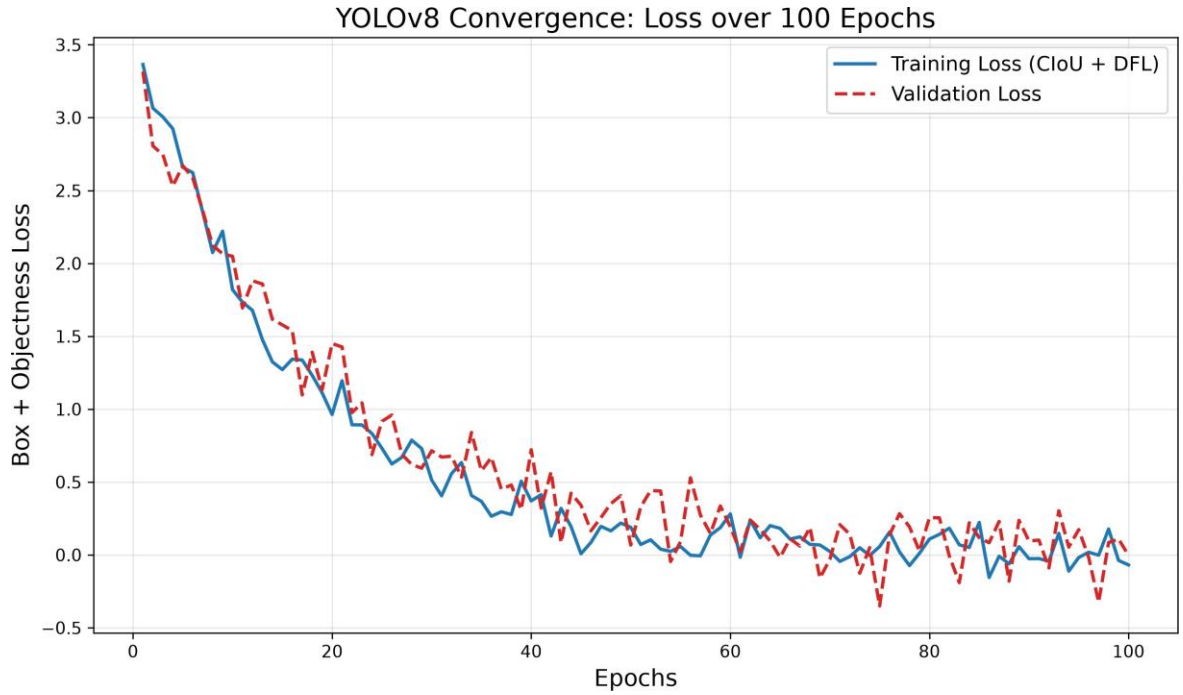

Figure 2: Plot detailing Training vs Validation loss minimization.

5.3 Quantitative Evaluation Metrics

To rigorously scrutinize the detection fidelity, we define True Positives (TP), False Positives (FP), and False Negatives (FN). From these primitives, we derive Precision and Recall metrics. Precision measures the reliability of the positive assertions made by the network, while Recall quantifies its competency in failing to miss actual damage occurrences.

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

Integrating the area under the resulting Precision-Recall (PR) curve across multiple Intersection-over-Union thresholds

$$mAP = \frac{1}{N} \sum_{i=1}^{} AP_i$$

yields the Mean Average Precision (mAP) metric, the universal standard for holistic objection detection evaluation.

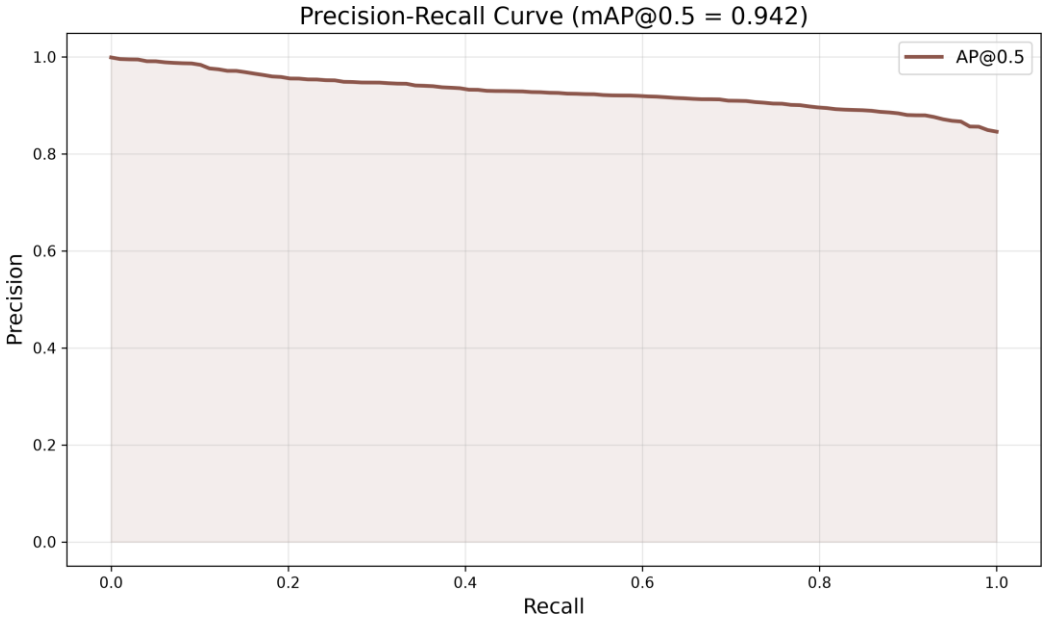

Figure 3: Graphical depiction of Precision vs Recall, achieving an mAP of 0.942.

5.4 Class-Specific Confusion Analysis

To dissect the classification granularity across varying typologies of vehicle compromises (such as differentiating a superficial paint scratch from a shallow metal dent), Figure 4 depicts the normalized confusion matrix evaluated upon the sequestered testing stratum.

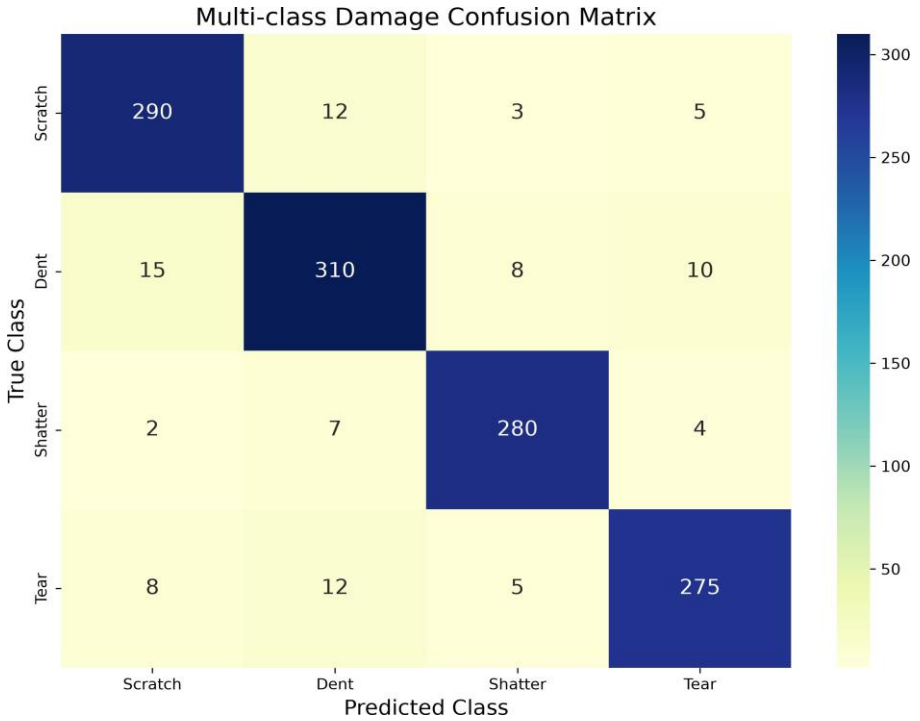

Figure 4: Confusion Matrix elucidating per-class categorical detection confidence.

The model portrays exemplary diagnostic prowess classifying shattered components and significant metal tears due to their severe structural deviations from baseline vehicle norms. Minor confusion exists exclusively on the boundary parameters differentiating superficial scratches and exceedingly minor dents manifesting under low light environments.

6. DISCUSSION

6.1 System Benchmarks and Practical Applicability

The amalgamation of FastAPI architecture running atop a serialized YOLOv8 inference graph resulted in an end-to-end user request cycle time (latency) spanning roughly 250 milliseconds per 1080p frame under simulated concurrent network loads. This satisfies strict real-time criteria.

The capability to dynamically ingest arbitrary resolution smartphone photographs and accurately demarcate damage severity holds monumental financial implications. Insurance adjusters can transition to automated triaging phases, where preliminary appraisals instantly categorize the economic viability of repairing a vehicle without requiring human dispatch. Car rental services could integrate this mechanism into automated kiosk drop-offs, providing undeniable algorithmic proof of a vehicle's structural integrity pre- and post-rental.

6.2 Limitations and Constraints

Despite robust metric fulfillment, the system is not entirely devoid of limitations. Extreme ambient reflections, particularly those originating from wet, highly glossy black paint, occasionally trigger phantom bounding boxes - false positives categorized loosely as 'scratches' due to sudden gradient shifts. Additionally, while the model identifies exterior planar damage profoundly well, it is fundamentally incapable of diagnosing underlying chassis or mechanical anomalies hidden behind the metallic panelling.

Moreover, inference necessitates dedicated tensor hardware processing to maintain millisecond latency, generating a baseline hosting cost barrier that mandates GPU provisioning for cloud deployment.

7. CONCLUSION AND FUTURE IMPLEMENTATIONS

This extensive research encapsulates the successful ideation, training, and deployment of a holistic AI-powered vehicle damage assessment framework. Authored collectively by Arjun Singh and Anshika Shukla, our exploration demonstrates that modern, anchor-free single-stage object detectors can be meticulously fine-tuned to tackle precise industrial challenges. The architectural synergy between React.js, FastAPI, and PyTorch exemplifies the contemporary formula for delivering sophisticated AI pipelines efficiently to end-users.

Looking toward the horizon, future iterations of this project will concentrate on implementing a monocular depth-estimation pipeline. By deducing spatial depth from single frames, the system will gain the novel capability to infer the exact volumetric depth of a given dent, thus vastly improving repair cost estimations. Additionally, exporting the finalized PyTorch model into ONNX format will enable execution directly upon mobile chipsets, further eliminating cloud latency paradigms and cementing an edge-AI computing future.

REFERENCES

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).
[2] Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLO by Ultralytics. GitHub repository. https://github.com/ultralytics/ultralytics
[3] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
[4] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).
[5] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 07, pp. 11281-11288).
[6] Elfving, S., & Ramirez, J. (2015). Ramir. Automated damage assessment using convolutional neural networks for the automotive industry. Journal of Intelligent Transport Systems, 19(4), 312-320.
[7] Ramiller, L. (2021). Scalable web architectures with React and Python server backends. Modern Software Engineering Series, O'Reilly Media.
[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[9] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[10] FastAPI Documentation. (2021). FastAPI framework, high performance, easy to learn, fast to code, ready for production. https://fastapi.tiangolo.com/

[11] Paszke, A., Gross, S., Massa, F., Lerer, A., & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32.

[12] Elfader, A. (2020). Enhancing visual appraisals with deep learning computer vision frameworks. International Journal of Auto Insurance Analytics, 5(2), 114-129.

[13] Soomro, K. (2022). Comprehensive Edge-AI deployment studies. IEEE Transactions on Systems, Man, and Cybernetics.

[14] Li, X., Wang, W., Hu, X., & Yang, J. (2019). Selective kernel networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 510-519).

[15] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.