ARID ZONE JOURNAL OF ENGINEERING, TECHNOLOGY & ENVIRONMENT
AZOJETE September 2025. Vol.21(3):820-836
Published by the Faculty of Engineering, University of Maiduguri, Maiduguri, Nigeria.
Print ISSN: 1596-2490, Electronic ISSN: 2545-5818
https://doi.org/10.63958/AZOJETE/2025/21/03/013
www.azojete.com.ng

ORIGINAL RESEARCH ARTICLE

CRACK AND SPALL DETECTION IN BUILDINGS USING YOLOV8 AND DETECTRON2–BASED WEB APPLICATION

W. A. Raheem, C. O. Folorunso, O. F. Odeyinka*, and F. F. Kamal

Department of Systems Engineering, University of Lagos, Akoka, Lagos, Nigeria
**Corresponding author's email: oodeyinka@unilag.edu.ng*

ARTICLE INFORMATION

Received: 4th July 2025
Revised: 14th September 2025
Accepted: 15th September 2025

Keywords:
Crack
Deep learning
Detectron2
YOLOv8
Spall
Structural health monitoring

ABSTRACT

Cracks and spalls in building structures pose serious risks to safety and durability. Conventional methods of detecting these defects are manual, time-consuming, and error-prone. Hence, this study develops a web-based system for automated defect detection using deep learning models. Two object detection models (YOLOv8 and Detectron2), and a CNN model (Resnet18), were trained on 1798 annotated images which consisted of benchmark datasets (METU, VCC) and with locally acquired images. Classification and object detection were done on both datasets acquired. YOLOv8 achieved a weighted average of 99.0% precision, 99.0% recall, and 99.0% accuracy, while Resnet18 reached 98.8.0% precision, 98.8% recall, and 98.8% accuracy weighted average for mild crack, severe crack and spall. For the object detection, YOLOv8 (mask mAP50 of 93.0%) achieved superior segmentation accuracy than Detectron2 (mask mAP50 of 87.5%). Both models demonstrated strong performance in detecting spalls, mild and severe cracks (mAP > 0.73). The Detectron2 model was deployed in a web-based application to enable real-time crack and spall identification. These results confirm the feasibility of AI-assisted structural health monitoring and highlight pathways for improving crack detection through balanced datasets, synthetic augmentation, and higher-resolution training in both Nigerian and global contexts.

1.0 Introduction

Building defects are unintended damages to the building structures which cause financial and environmental losses (Alomari, 2022). These defects can be a result of design faults, substandard materials, human errors, and natural factors (Czajkowska and Ingaldi, 2021). Cracks can appear in buildings during (or after) the construction process or when the building is exposed to additional load during its service life (Tzimas, 2023). Cracks can be classified based on their direction, width and depth (Scuro and Fusaro, 2022). A typical example is a recurring crack that reappears or continues to develop even after concrete has been repaired or strengthened. This type of fracture indicates ongoing structural stress or deterioration (Alomari, 2022). Hence, monitoring the state of buildings is essential to track their health over time, assess severity of defect (if any) and potential safety risks (Özgenel, 2019). This monitoring process aids early identification of damage, prevents financial loss and loss of life (Entezami, 2021). Some of the challenging issues affecting structural health monitoring include variation in environmental and operational conditions (Entezami, 2021). This involves a routine examination of structures with a view to ascertaining whether they are safe to use and strong enough to last longer (Psuj and Szymanik, 2023). Structural health monitoring focuses on the accuracy of measurements and reliability analysis to reduce false alarms and risks (Scuro and Fusaro, 2022).

However, conventional inspection of cracks and fractures is labor-intensive, subjective, and prone to human errors. Different lighting situations, material textures, and crack widths make it more difficult (Özgenel, 2019). Machine Learning (ML) techniques have been found to be effective in the field of structural health monitoring (Dong and Catbas, 2021). They can overcome some of the drawbacks associated with the conventional approaches in detecting structural damage and allow for real-time identification of such damage (Scuro and Fusaro, 2022; Xu *et al*., 2025). ML algorithms aid the prediction of critical failure at an early stage in different conditions (Singh *et al*., 2025). The use of advanced ML methods for crack detection has evolved with other artificial intelligence (AI) techniques been integrated in structural health monitoring (Naresh *et al*., 2023; Xu

et al., 2024). Figure 1 outlines the components of a system based on a deep neural network for structural health monitoring.

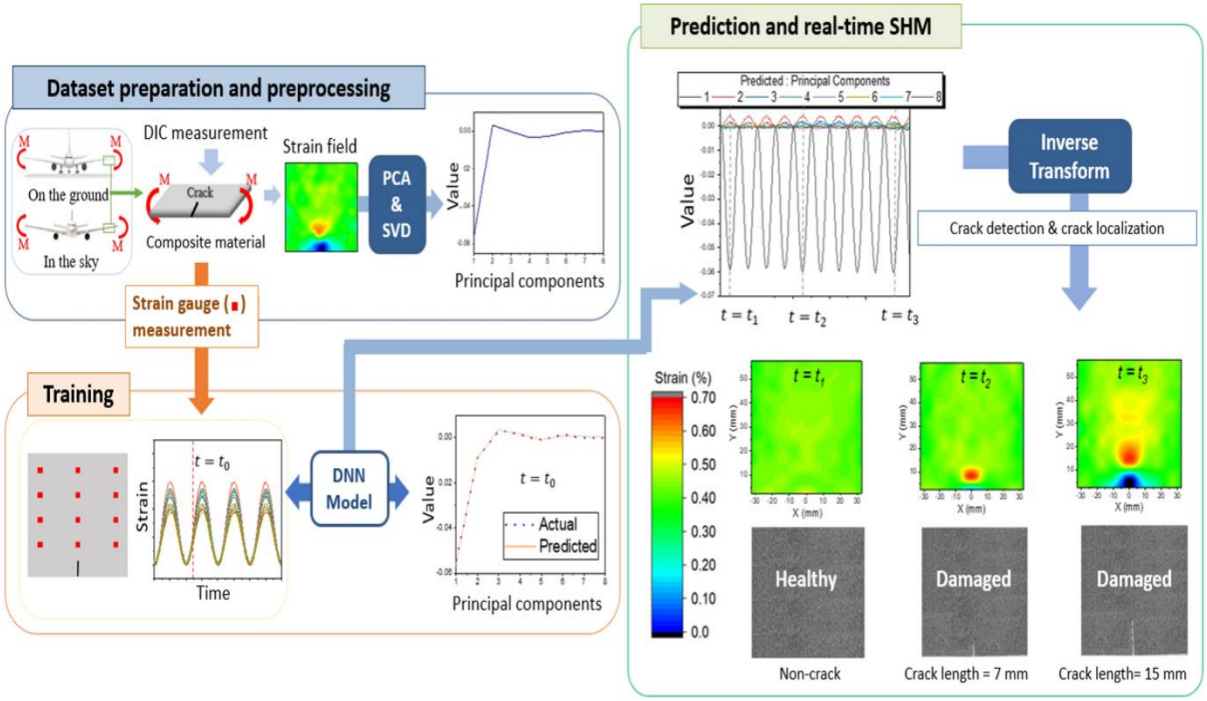

Figure 1: Deep Neural network-based Structural Health Monitoring

For example, Flah *et al.*, (2020) conducted an exhaustive study of meta-heuristic schemes and other applied ML techniques used for structural health monitoring. The shifting paradigm towards the use of machine learning for structural health monitoring was noted and it was recommended that the choice of appropriate algorithms and their training on enough databases must capture environmental and operating conditions. As a result, research interests towards using deep learning (DL) methods for analyzing the complexity of the building environment has deepened. In Rizia *et al.* (2025), deep learning algorithms was combined with other methods for identifying structural abnormalities and avoiding such scenarios. This is because DL techniques have robust frameworks, such as in computer vision and image processing algorithms, that make crack detection simpler. For example, Guo *et al.* (2019) presented a deep learning approach for comprehending damage using a methodology designed on solid feature extraction - a key step in the identification of damage. The deep learning algorithm used had the potential to understand feature representation from sensor data since the real-world damages vary for different buildings. In Hou *et al.*, (2021), a deep learning approach for safety in the Architecture, Engineering and Construction (AEC) industry was presented for use in structural health monitoring. Feature enhancement, improved robustness and reliability of systems in complex terrain were also identified as major advantages of deep learning (Singh *et al.*, 2025). Zhao *et al.* (2024) developed an instance segmentation method based on deep learning and computer vision algorithms to improve feature extraction and fine-crack detection for complex structures. Results showed a higher accuracy, increased gain and better balance between precision and efficiency. In Kiranyaz *et al.* (2021), the importance of 1-Dimensional Convolutional Neural Networks (CNNs) and their uses for crack detection was outlined. CNNs is one of the advanced deep learning architectures that are suitable for learning the features effortlessly from the sequential data such as the array of sensor measurements (Ahmadzadeh *et al.*, 2025). Knowledge in CNNs and its potential applications is useful in the development of improved structural health monitoring systems that incorporate deep learning for detection of damage under different circumstances.

While various methods for crack detection have been developed, there is still a lack of a simplified system for easy detection and early identification. With human effort remain time consuming and incur different costs, there is need for autonomous system that automate crack detection and also serves as a warning system (El-Din Hemdan and Al-Atroush, 2025). Autonomous systems are specialized computer programs capable of doing what knowledgeable human beings can do in a certain specialization and domain (Youwai *et al.*, 2024). Consequently, these systems can comprehend data originating from diverse chunks of the structure, for instance, strain gauges, vibration sensors, ultrasonic sensors, or thermal sensors for the purpose of detecting probable damage (or crack). In this context, these systems can process data, acquired from various sensors, including strain gauges, vibration sensors, ultrasonic even thermal ones, to identify the damages, especially cracks. The major advantage is in the flexibility that comes with the capacity to encode the expert knowledge

Corresponding author's email address: oodeyinka@unilag.edu.ng

and the reasoning methods in the form of a set of rules (Ahmadzadeh *et al*., 2025). These rules are then applied to help the system, analyze data received from sensors and come up with inferences and conclusions about the structural health based on some laid down principles, procedures and rules (Qiu *et al*., 2025). Hence, the aim of this work to develop a web application for automating crack and spalls detection in residential buildings using real-time object detection and techniques. This involves implementing feature extraction and segmentation techniques to accurately identify and classify cracks and spalls, and then integrate a detection system to work seamlessly for comprehensive structural assessment. To ensure accurate detection and localization of structural issues, this work also validates the performance of the system by testing on real-world scenarios. Section 2 describes the material and methods behind this work while section 3 presents the results of the analysis carried out. A suitable conclusion is provided in section 4.

2. Material and Methods

Two **high-performance object detection frameworks** were considered - YOLOv8 and Detectron2 in this study. Each detection algorithm consists of both image classification and image segmentation phases. Firstly, the image is analyzed using a deep convolutional neural network to detect the presence or absence of cracks and spalls. Once these defects are detected, an adaptive thresholding technique is applied to further process and refine the identified regions.

2.1 YOLOv8 Model Architecture

The YOLOv8 architecture was implemented in this work for object classification and detection. This architecture is divided into three parts, namely the backbone, the head and the output, which is intended to provide effective real-time object detection.

YOLOv8 Backbone

The process begins with a backbone, which creates feature maps at different scales (P1 through P5) by extracting hierarchical features from the input image. P5 is the highest-level feature in these feature maps, which show increasingly coarser spatial resolutions and deeper semantic information. At the backbone network, there exist the convolutional layers, batch normalization, ReLU activation functions, and sometimes the down-sampling function, such as max-pooling. The mathematical equation for a typical convolutional layer is given by in equation (1):

$$Conv(x, W, b) = \sigma(W * x + b) \quad 1$$

where: x is the input feature map, W is the convolutional filter, and b is the bias term, * denotes the convolution operation, and σ is the ReLU non-linear activation function.

Convolution can be defined as the sum of the product of an image patch and a filter in an element-wise manner. This reveals other patterns hidden in the image and is done by introducing the activation function that injects non-linearity into the network. The architecture of the Yolov8 is presented in Figure 2.

YOLOv8 Head

The multi-scale feature maps are processed by the head to produce predictions after feature extraction. To enhance multi-scale object identification capabilities, this section incorporates a Feature Pyramid Network (FPN) or a similar structure. Concat and Conv operations are used to merge and refine features from various backbone levels. To create a reliable feature representation for detection, up-sampling procedures are used to combine data from deeper layers with shallower ones. The detection head predicts bounding boxes of objects, the probability of each object's class, and the objectness score. For bounding box regression, the smooth L1 loss is commonly used and can be represented in Equation (2):

$$L_{loc} = \sum_{i=1}^{N} \sum_{j \in Pos} smooth\_L1(pred_{ij} - target_{ij}) \quad 2$$

where N is the number of bounding boxes, Pos is the set of positive samples, $pred_{ij}$ is the predicted bounding box, $target_{ij}$ is the ground truth bounding box, and $smooth\_L1$ describes the smooth L1 loss function.

Corresponding author's email address: oodeyinka@unilag.edu.ng

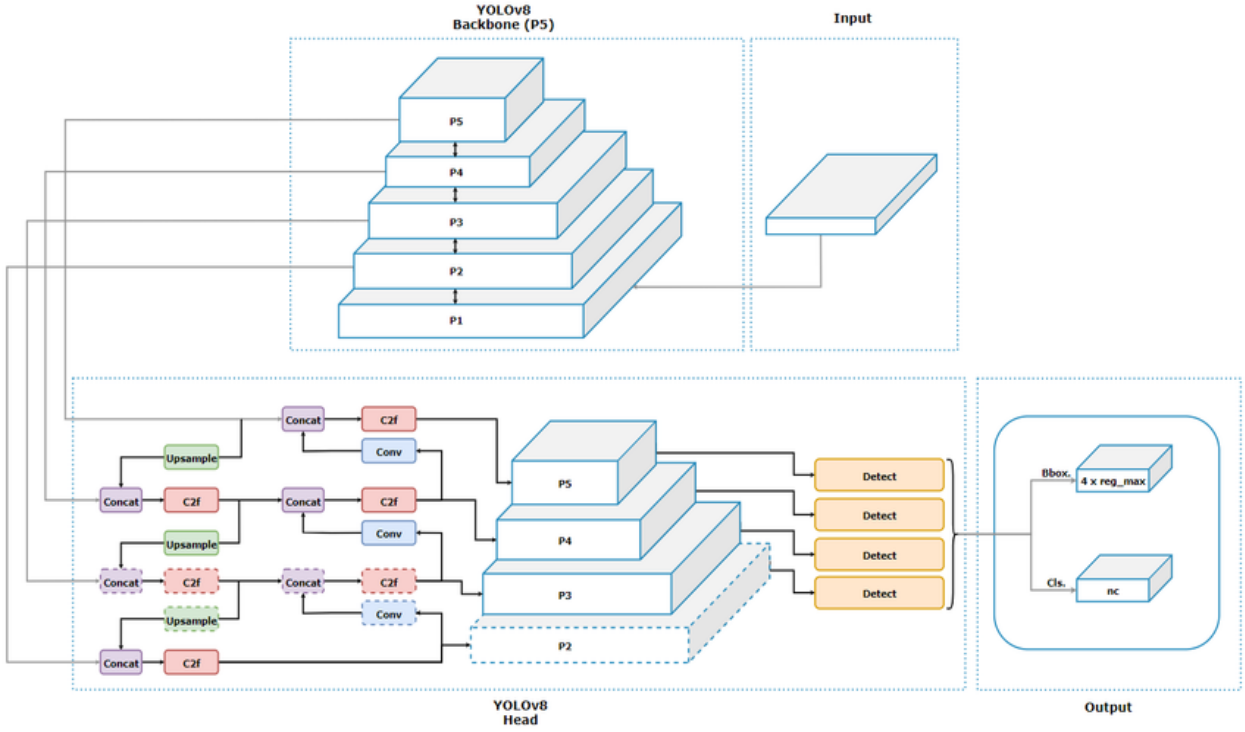

Figure 2: YOLOv8 architecture (Sharma *et al.*, 2024)

YOLOv8 Output

To locate and identify objects in the input image, the Output stage carries out the last detection operations after receiving the refined features from the Head. These procedures include bounding box regression (Bbox) and classification (Cls). The model learns by minimizing the loss between predicted bounding boxes and class probabilities (crack, spall) and the ground truth labels provided in the training data. Common loss functions include Intersection over Union (IoU) loss for bounding boxes and cross-entropy loss for classification. Equation (3) gives the loss function in terms of localization, confidence, and classification.

$$L = \lambda_{coord} L_{loc} + \lambda_{conf} L_{conf} + \lambda_{cls} L_{cls} \quad 3$$

where $\lambda_{coord,}$ $\lambda_{conf,}$ λ_{cls} are the coefficients for localization, confidence, and classification losses, respectively.

2.2 Detectron2 Model Architecture

For reliable object detection, the Generalized Region-Based Convolutional Neural Network (RCNN) architecture combines a Backbone Network, a Region Proposal Network (RPN), a Feature Pyramid Network (FPN), and Region of Interest (ROI) Heads (Box Head). Hierarchical features (res2, res3, res4, res5) are extracted from the input image by the Backbone Network, usually a ResNet. The FPN receives the lower-level features, while a P6 layer receives the higher-level features (res5). By up sampling and merging features from multiple backbone levels, the FPN creates a multi-scale feature pyramid (P2-P5) that offers rich contextual information at multiple sizes. The architecture of the Detectron2 is shown in Figure 3.

Based on the output of the FPN, the RPN predicts anchor deltas and objectness scores to provide a set of 1000 box proposals, which are further honed using Non-Maximum Suppression (NMS). These suggestions are then used by the ROI Heads, and the FastRCNN ConvFCHead (through a ROIPooler to extract fixed-size feature maps). The object detection pipeline is then completed by feeding these features into fully connected layers that predict class logits for 81 channels and bounding box predictions (80x4 channels). Following NMS, the final 100 detected boxes are obtained.

Corresponding author's email address: oodeyinka@unilag.edu.ng

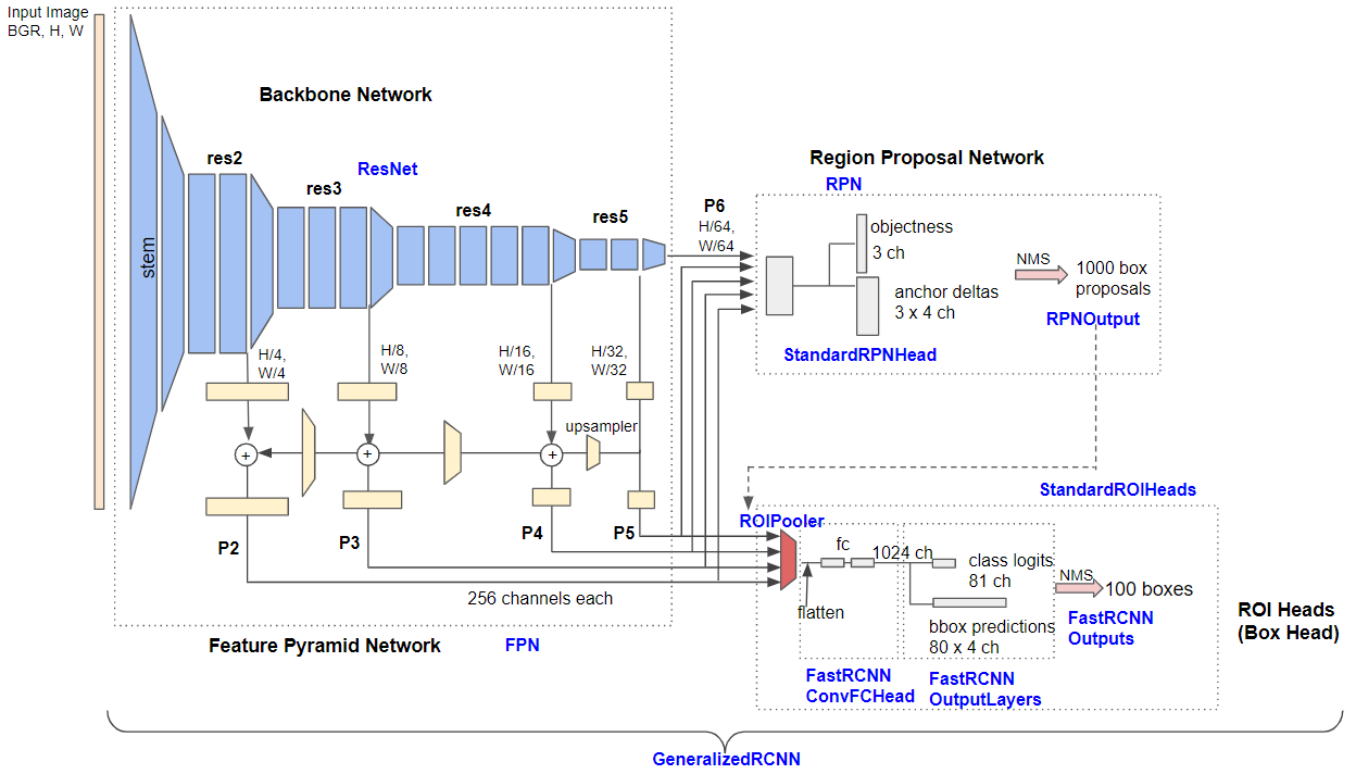

Figure 3: Architecture of Detectron2 (Devi and Naidu, 2023)

2.3 Resnet-18 Architecture

ResNet has varying numbers of layers; for example, ResNet-18 has 18 layers. Figures 4a and 4b shows the ResNet-18 and residual block architectures. In Figure 4a, the black curve arrow indicates the shortcut connections, and the two layers of the same color are residual blocks. While models get deeper, shortcut connections can effectively tackle the vanishing gradient issue. The residual block has two branches (as seen in Figure 4b) with the identity mapping indicated by the black curve arrow. The convolution function is represented by the f, while the input images are represented by the x. The sum of x and f(x) is the result of residual blocks. Later CNN models have largely inherited residual blocks.

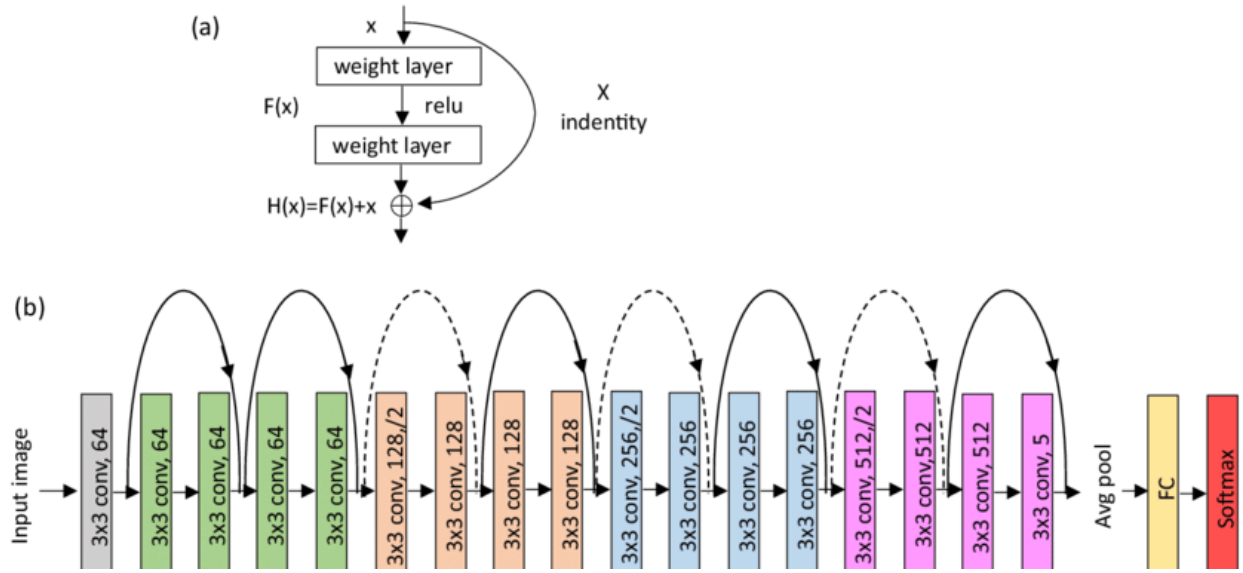

Figure 4: Resnet-18 architecture (Song, 2023)

2.4 Theoretical Frameworks

To enable precise and effective detection of cracks and spalls in building structures, this work employs YOLOv8 and Detectron2 within a web-based framework. Data collection and preprocessing, model training

and optimization, and web-based deployment are the three primary phases of the methodology as shown in Figure 5.

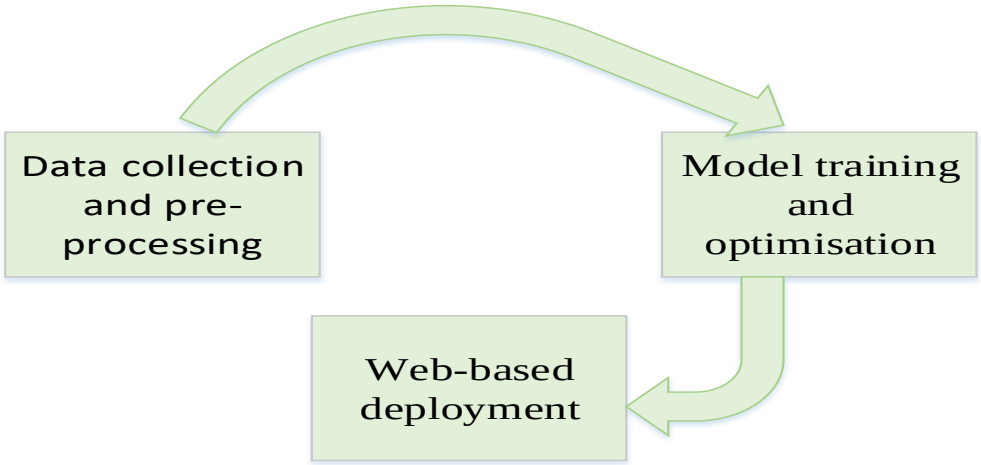

Figure 5: Methodology employed for the crack and spall detection

2.4.1. Data Acquisition and Pre-processing

High-resolution images of building surfaces, such as walls, slabs, and beams, were gathered from publicly accessible crack and spall datasets in addition to field-collected photos. The dataset used consisted of images was collected from the METU Campus Buildings dataset (Özgenel, 2019) and the VCC dataset available on Roboflow (Roboflow, 2023). These datasets were chosen because they are standardized, annotated, and widely recognized, making them excellent for initial model training and benchmarking. Though they do not fully reflect Nigerian-specific conditions, the locally captured images improve the relevance and transferability of the datasets for practical deployment in Nigeria. The images collected were from both private and public properties and are of different types of cracks and spalls under different lighting, angles, and building materials. Close-up images of buildings (such as in Figure 6a - c) and distant images (as in Figure 7a - c) were all captured. The resolution of the images in the METU Campus Buildings dataset was $227\ x\ 227$ pixels with RGB channels and they were generated from 458 high-resolution images originally sized at 4032 x 3024 pixels. These images were all standardised to 640 × 640 pixel in order to ensure consistent feature extraction. For the VCC dataset, the images were already in the standard $640x640$ pixels format. Depending on the extent of the damage, the cracks were categorized as mild, and severe. The mild crack had estimated crack width ranging from 1 to 15mm while the severe crack had estimated crack width measuring 15 to 25mm or more (usually above 25mm). Subsequently, the data sets were analyzed and processed for polysylabic labeling using LabelMe Software. This tool was used to classify the structural damages into 3 classes - mild crack, severe crack and spall. After labelling, a simple check was carried out in order to assess the reliability and usefulness of the annotated data.

The main detection framework for quickly and accurately identifying cracks and spalls is the YOLOv8 model. Rich multi-scale features are extracted from images via its backbone, a CSPDarknet variation improved with the C2f module and SPPF layer. These features are fused by the PAN-FPN neck, while bounding box regression and classification are carried out by the anchor-free decoupled head. To increase localization accuracy, the model is trained with adaptive IoU-based loss functions.

Similarly, Detectron2 is used for pixel-level segmentation in addition to YOLOv8 to provide accurate border delineation of identified cracks and spalls. The modular design of Detectron2, which is based on a ResNet-FPN backbone, makes it easier to apply Mask R-CNN for instance segmentation. By offering comprehensive masks, this phase improves YOLOv8's bounding box predictions and guarantees that fractures can be more accurately distinguished from background textures and other surface flaws.

Corresponding author's email address: oodeyinka@unilag.edu.ng

a: Mild Crack

b: Severe Crack

c: Spalling

Figure 6: Close-up images of the buildings

a: Mild Crack

b: Severe Crack

c: Spalling

Figure 7: Distant images of the buildings

The feature extraction and data cleaning techniques of YOLOv8 and Detectron2 reflect their deep learning architectures and detection goals. A convolutional backbone modeled after CSPDarknet is used for YOLOv8 feature extraction. Deeper layers encode high-level representations of cracks, spalls, and their spatial context, while early layers' capture low-level characteristics like edges, gradients, and textures. The methodology simplifies feature encoding for localization tasks by using an anchor-free detection mechanism that dynamically predicts object centres and dimensions. In order to improve robustness against environmental fluctuations, YOLOv8 data cleaning include deleting low-quality or incorrectly labelled images, normalizing image dimensions and colour distributions, and applying augmentations including random cropping, rotation, and contrast modifications. To handle both object detection and pixel-level segmentation, Detectron2's feature extraction uses a Feature Pyramid Network (FPN), which combines multi-scale features for fine-grained representation. Because of its sensitivity to label accuracy, Detectron2 requires more thorough data cleaning, which includes accurate mask annotations, the elimination of noisy boundaries, and pre-processing methods like resizing, normalization, and augmentation to maintain the structural and contextual integrity of defect regions. When combined, these strategies guarantee that both models receive high-quality, well-annotated inputs to maximize generalization and detection accuracy. Table 1 summaries the features employed by the two algorithms.

Table 1: Summary of the features used by the two algorithms

Feature Category	YOLOv8	Detectron2
Visual Features	Edges, contours, gradients, textures, and geometric patterns for object localization.	Pixel-level details, fine textures, boundary shapes for precise segmentation.
Structural Features	Bounding box centre points and sizes are predicted dynamically (anchor-free).	Instance-level and semantic-level segmentation with multi-scale feature representation.
Contextual Features	Spatial relationships and environmental variations (e.g., lighting, occlusion).	Hierarchical scene understanding to refine pixel masks and object boundaries.
Processing Focus	Real-time detection with emphasis on speed and robust localization.	High-accuracy segmentation with fine-grained spatial detail.

2.4.2 Model training and Optimization using Detection Based on YOLOv8 and Detectron2

For the experiments, Nvidia Tesla T4 GPUs were used to train the aforementioned deep neural networks of the proposed system. The training process was done using PyTorch and Google Colab Platforms. The training set was comprised of 1798 images in COCO format while the validation set comprised of 515 images which

Corresponding author's email address: oodeyinka@unilag.edu.ng

was kept constant between the two models. A test set of 255 images was used to evaluate both models. The distribution of the different classes in the model is shown in Table 2. The distribution (599 milds, 600 severe, 599 spall in training) shows that the dataset used was balanced with an equal set of data from each category.

Table 2. Distribution of instances among all 3 categories in the train and validation datasets

	Classes/Categories	Training Instances	Validation Instances	Test Instances	Total
1	Mild Crack	599	172	85	856
2	Severe Crack	600	171	85	856
3	Spall	599	172	85	856
	Total	**1798**	**515**	**255**	**2568**

The dataset was augmented via geometric and photometric transformations. Geometric transformation (including rotation, flipping, cropping, and scaling) and photometric adjustments (such as brightness, contrast shifts, noise, blur) were carried out on the dataset. The hyperparameters (the number of epochs and the learning rate) were adjusted. Also, an optimizer was chosen to make adjustments to the weights of the model with regards to the calculated loss, after each training cycle. In this experiment, Adam optimizer was employed, which can commonly be used in deep learning and provides good results in multiple tasks.

The Adam optimizer updates the weights (θ) of a neural network based on the gradients (g) calculated during backpropagation as shown in Equation (4):

$$\theta_{t+1} = \theta_t - \propto \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad 4$$

where θ_t is the current weight value at iteration t, θ_{t+1} is the updated weight value at iteration t+1, $\propto$ (learning rate) is the hyperparameter that controls the step size of the update, ϵ is a small constant value (usually 10^{-8}) added to the denominator to prevent division by zero and $\hat{m}_t$ is the bias-corrected first moment estimate of the gradient as shown in Equation (5), which is an exponentially decaying average of past gradients. Equation (6) represents the average direction of recent gradient changes,

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad 5$$

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \quad 6$$

where $\beta_1$is the hyperparameter is the decay rate in the moving average of the gradients and its typical value is almost one.

For example, 0.9, g_t is the derivative of the loss function of the weights at the current iteration. It is the 'second moment estimate of the gradient, $\hat{v}_t$ is an exponentially weighted moving average of the squared past gradients as shown in Eq. (7). Equation (8) expresses the change of gradients within a recent period of time.

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad 7$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \quad 8$$

β_2 is the hyperparameter that controls the decay rate of the moving average of the squared gradients (usually set to a value even closer to 1, for instance, 0. 999). g_t^2 is the gradient of the loss function with respect to the weight at the current iteration.

2.4.4 Integrating Detectron2 Model into a Web Application

A web application allows users to access the crack detection model from any device with a web browser, eliminating the need for local software installation. This provides a real-time inference as users can upload images directly to the web application, enabling real-time crack detection and visualization of results. The

Corresponding author's email address: oodeyinka@unilag.edu.ng

classification result showed better than the detection, however, since the aim of the study is to detect the various types of defects, the detection was implemented on the webpage. The Detectron2 architecture was integrated into the web application, a user-friendly tool for real-time building defect detection was created, making it accessible to a wider audience and facilitating efficient defect identification in various applications. The architecture of the web application used for the study is shown in Figure 8.

Web application allows for scalability because it can handle multiple users requests simultaneously, making them suitable for large-scale deployments. The web application is deployed to a cloud platform, Amazon Web Services, AWS, to make it accessible to users over the internet. The integration process of the web application used in this study is described as follows:

1. **Flask App Setup:** Flask is a web framework; used to build the backend logic for the web application. It offers functionalities for handling user input, file uploads and model inference.
2. **React App setup:** The react framework is used to handle the frontend logic for the web application for user friendly interface that allows users to easily upload images, view results and understand the defect detection outcome. It offers functionalities for validating the user input, image and file uploads, inference results and visualisation.
3. **Model Loading and Pre-processing:** The saved weights and configuration files from the Detcetron2 training were loaded into the web application through the Flask app implementation. Image pre-processing methods to resize, normalise were also created to match the input format of the model.

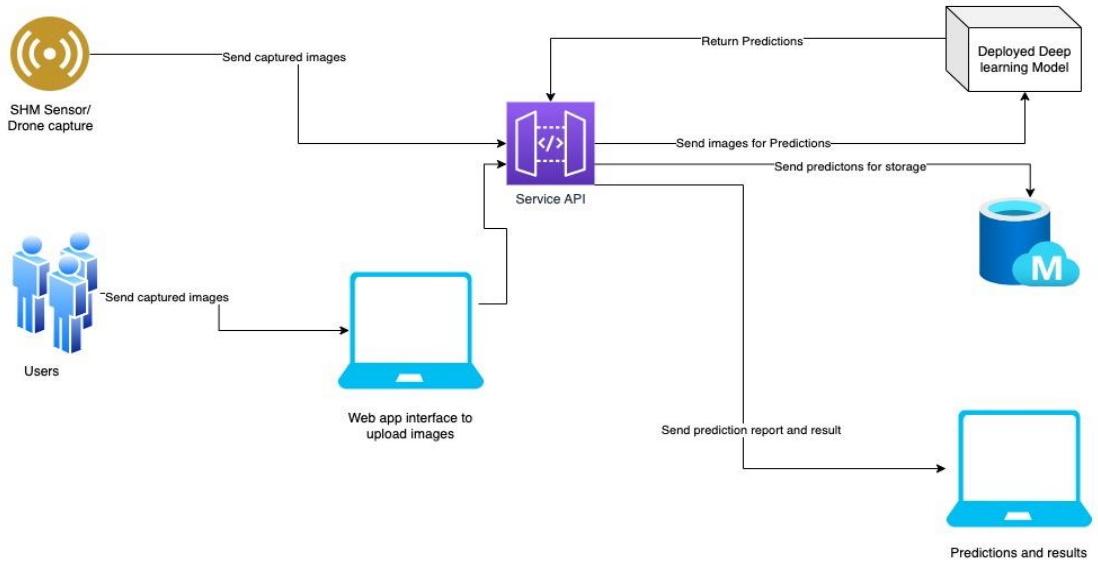

Figure 8: Web Application architecture design for Detectron2

The following steps are employed to use the web Application:

1. Once image has been uploaded the loaded Detectron2 model is used to perform inference and generate predictions in the form of bounding boxes and masks for detected defect.
2. The OpenCV and Matplotlib library are used to visualise the original image with the predicted bounding boxes and segmentation masks overlaid on top, highlighting the detected defect.

2.5 Evaluation Metrics

To quantify the accuracy and performance of the proposed detection models, the key metrics used include: Precision, Recall, accuracy, mAP@0.5, and mAP@0.5:0.95**, for both** bounding box (B) **and** mask (M) **p**redictions. These metrics give a complete assessment of how efficiently an algorithm can separate the target features in the actual images, using the classifiers. Intersection over Union (IoU) is the area where the ground-truth box/mask and the anticipated bounding box/mask overlap. The values obtained stem from the calculated IoU and have a connection with the threshold identified for the IoU as shown in Equation (9).

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \quad 9$$

The Precision is the percentage of all positive instances (true positives + false positives) that were accurately predicted as positive cases (true positives):

$$Precision = \frac{t_p}{t_p + f_p} \quad 10$$

Recall indicates how many ground-truth positives the model can identify:

$$Recall = \frac{t_p}{t_p + f_n} \quad 11$$

Accuracy is the frequency with which a model, method, or procedure yields accurate results (true positive + true negative) in relation to the total number of instances assessed (true positive + true negative + false negative + false positive):

$$Accuracy = \frac{t_p + t_n}{t_p + t_n + f_n + f_p} \quad 12$$

The Mean Average Precision (mAP) is the mean of AP across multiple IoU thresholds and/or classes, it is the most widely used object detection metric. It evaluates a model's ability to predict class labels and bounding boxes. The **mAP@0.5 shows the** AP is computed at IoU ≥ 0.5. While the mAP@0.5:0.95 shows that the AP is averaged over IoU thresholds from 0.5 to 0.95 in steps of 0.05, it is stricter and more robust.

3. Results and Discussion

The performance of each model was evaluated on the validation and testing set using various metrics - Mean Average Precision (mAP), Precision (P), Recall (R), and Accuracy. The result for classification using Resnet-18 and YOLOv8 is shown in Table 3.

Table 3: Classification result for model training

Class	YOLOv8			Resnet-18			Support
	Precision	Recall	F1-score	Precision	Recall	F1-score	
Mild-crack	0.97	1.00	0.98	0.98	1.00	0.99	85
Severe-crack	1.00	1.00	1.00	0.99	1.00	0.99	85
spalls	1.00	0.96	0.98	1.00	0.96	0.98	85
accuracy	0.99	0.99	0.99	0.99	0.99	0.99	255
Macro avg.	0.99	0.99	0.99	0.99	0.99	0.99	255
Weighted avg.	0.99	0.99	0.99	0.99	0.99	0.99	255

For automated damage identification in concrete structures, a comparison of the YOLOv8 and ResNet-18 models showed good classification performance across all target classes. For example, ResNet-18 performed better for minor cracks (Precision = 0.98, Recall = 1.00, F1-score = 0.99) than YOLOv8 (Precision = 0.97, Recall = 1.00, F1-score = 0.98). For significant cracks, YOLOv8 demonstrated perfect detection (Precision = 1.00, Recall = 1.00, F1-score = 1.00) while ResNet-18 had lower precision and F1scores (Precision = 0.99, Recall = 1.00, F1-score = 0.99). Both models produced similar results for the detection of spalls, with the same F1-scores (0.98), precision (1.00), and recall (0.96). The overall accuracy of both models was 0.99, which is corroborated by the weighted and macro-average scores of 0.99 for all classes. This demonstrates a balanced performance regardless of the class distribution. These findings demonstrate the resilience of both architectures, with ResNet-18 performing better for detecting minor cracks and YOLOv8 better at identifying serious severe cracks.

Table 4 outlines the results of training of the YOLOv8 model. Overall, its precision (0.854), recall (0.847), and excellent accuracy in both detection (BoxmAP50 = 0.930) and segmentation (MaskmAP50 = 0.920) indicated a very strong performance.

Corresponding author's email address: oodeyinka@unilag.edu.ng

Table 4: YOLOv8 model training results

Class	Images	Instances	BoxP	BoxR	BoxmAP50	MaskmAP50
all	179	179	0.854	0.847	0.930	0.920
mildcrack	52	52	0.872	0.654	0.863	0.841
severecrack	62	62	0.802	0.887	0.934	0.931
spall	65	65	0.889	1.000	0.994	0.989

Spalls had the best results across individual classes with near-perfect mAP values (BoxmAP50 = 0.994; MaskmAP50 = 0.989) and perfect recall (1.000). Severe cracks also demonstrated strong performance with balanced precision (0.802), recall (0.887), and high mAP scores. While precision (0.872) and mAP values remained strong, mild cracks displayed a relatively lower recall (0.654), indicating a little lesser sensitivity to tiny faults.

The high discriminative power of the YOLO v8 model was further validated by the ROC analysis. With a micro-average AUC of 1.000, the area under the curve (AUC) values were 0.999 for moderate cracks, 1.000 for severe cracks, and 0.999 for spalls as shown in Figure 9. The ROC curves for all classes were located close to the plot's upper-left corner, showing almost flawless true positive rates and low false positive rates. The ability of the model to detect safety-critical structural faults is demonstrated by its flawless AUC score for severe cracks. Generally, these findings confirm the ability of the model to identify different types of concrete.

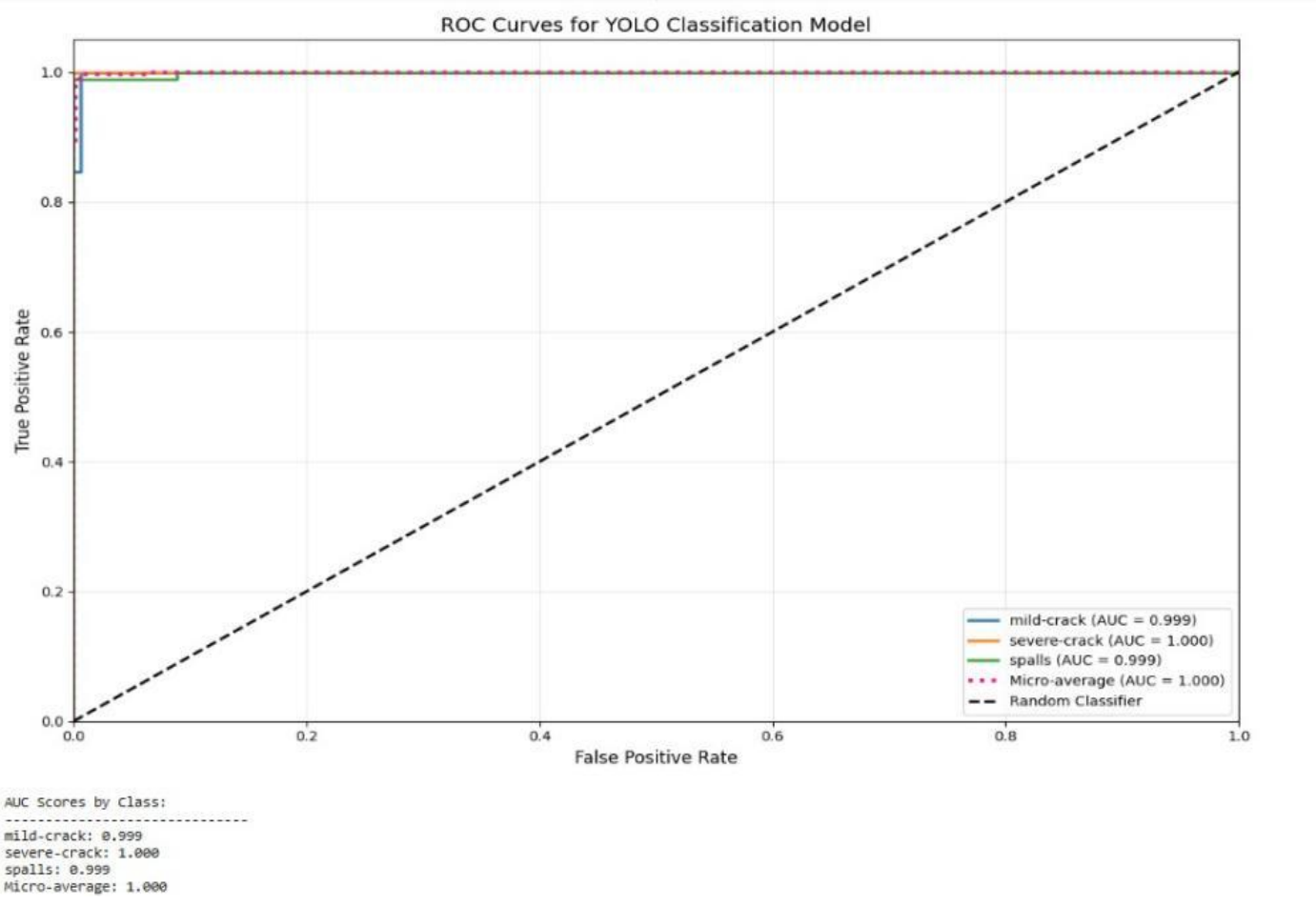

Figure 9: ROC Curve for YOLOv8 model for classification

Table 5: Average Precision for Detectron2 Models at IoU > 0.5

Class	AP	Class	AP	Class	AP	COCO mAP@50	COCO mAP@50-95
Mild-crack	74.58%	Severe-crack	86.5%	Spall	97.51%	87.5%	86.2%

Corresponding author's email address: oodeyinka@unilag.edu.ng

The trained Detectron2 model showed a strong detection performance across all defect categories. It had better detection for severe faults and spalling than mild cracks, as seen by its average precision (AP) values of 74.58% for mild cracks, 86.5% for severe cracks, and 97.51% for spalls. With an overall COCO metrics of 87.45% and a tighter mAP@50–95 of 86.20%, the model's robustness is further supported. The ability of the model to maintain precise localization under different intersection-over-union thresholds is demonstrated by the closeness of these two values, and suggests consistent generalization performance across defect kinds.

Plate 1a: First YOLOv8 predictions

Plate 1b: Second YOLOv8 predictions

Plate 1c: Third YOLOv8 predictions

In Plates 1a, 1b and 1c, the performances of YOLOv8 in identifying and classifying structural flaws under different circumstances are shown. Bounding boxes and overlays emphasize the precise localization of cracks and spalls of various sizes and orientations in the YOLOv8 predictions. The model successfully captures small discontinuities and has significant robustness under various texturing and lighting conditions. Minor partial detections point to the advantages of including multi-scale feature improvement or higher-resolution inputs for better boundary delineation.

The prediction abilities of the Detectron2 explains its dual capabilities in object detection and instance segmentation (as shown in Plates 2 a -c). The coloured overlays and bounding boxes to depict detected flaws and locations of interest.

Plate 2a: First Detectron2 predictions

Corresponding author's email address: oodeyinka@unilag.edu.ng

Plate 2b: Second Detectron2 predictions

Plate 2c: Third Detectron2 predictions

When compared to YOLOv8, Detectron2 provides more detailed spatial context, especially in terms of shape and area coverage, and effectively distinguishes cracks, spalls, and other possible structural anomalies. Green, red, orange, and purple highlighted masks visibly represent the segmentation fidelity and localization accuracy in a variety of architectural situations. Together, these visual findings validate the ability of both models to provide automated structural evaluation. Though YOLOv8 provides faster, high-confidence detections, while Detectron2 gives deeper spatial segmentation - together confirming their complementary roles in improving deep learning-based structural health monitoring systems. Based on these image inferences from the model, it is clear that the detectron2 model performs better at detecting structural defects compared to the YOLOv8 model.

Plate 3: Image of the web application prediction on a mild crack

Plate 3 shows that the supplied image was successfully analyzed, exposing the existence and features of structural flaws. With a "num_of_defects" of 3 and a "image_quality" rating of "Good," the analysis specifically indicated that the input image was appropriate for precise analysis. In this instance, "mildcrack" with a count of three indicates three instances of mild cracks found on the building. The "present_classes" object describes the sorts of faults found. Visual localization of these flaws inside the image is made possible by the "coordinate" array, which gives the exact pixel coordinates for each identified "mildcrack" instance. The present null values

Corresponding author's email address: oodeyinka@unilag.edu.ng

in the fields "structural_reinforcement_needed" and "types_of_defects" suggest that the system either did not identify the need for reinforcement or did not further classify defect types in light of this particular analysis. Lastly, for easier visualization and validation of the detection results, "masked_image_url" indicates the location of a processed image where the identified flaws are visually emphasized, most likely using masking or bounding boxes. An automated inspection of a structure is shown in Plate 4, where any structural flaws found are immediately labelled on the picture. The subsequent output, which displays num_of_defects: 2; image_quality: Good, confirms that the scene has overlays indicating two moderate cracks that have been found.

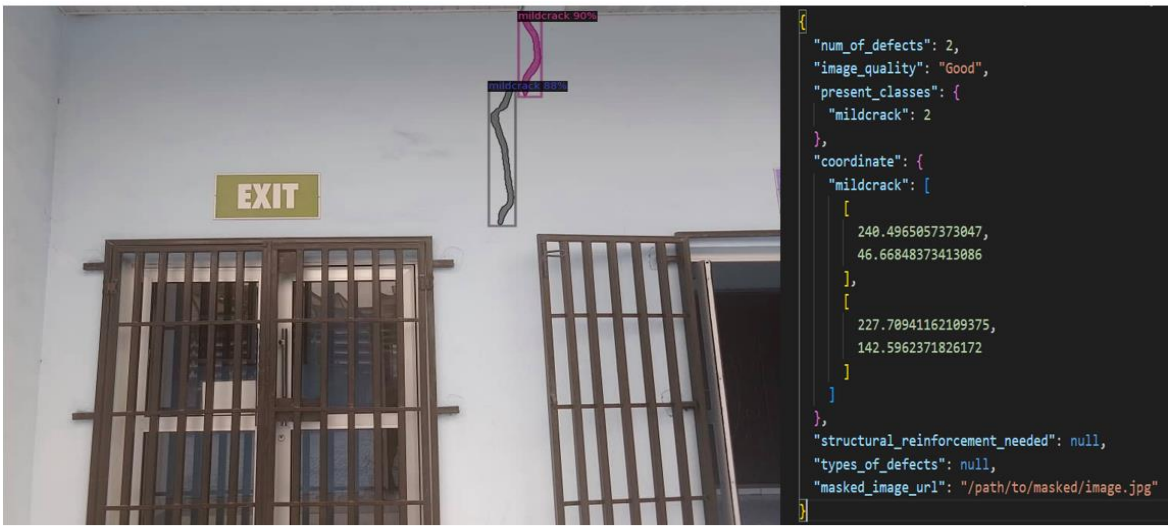

Plate 4: Image of the web application prediction on a mild crack

Each flaw is given precise bounding coordinates, proving the system's capacity for spatially precise localization. Interestingly, structural_reinforcement_needed and types_of_defects are both reported as null, suggesting that the cracks found are small and do not currently require repair. By combining quantitative metadata with visual annotation, this example demonstrates the potential of the system for real-time, automated structural health assessment, minimizing the need for manual inspections and promoting evidence-based maintenance decisions. An automated flaw detection and analysis system applied to structural elements - specifically, detecting spalling in a concrete beam is demonstrated in Plate 5. Three flaws were successfully identified by the system, which also labelled the image quality as "Good" and recognised "spall" as a major defect class.

Plate 5: Image of the web application prediction on a mild spall

The discovered spall's exact location within the structure is indicated by detailed coordinate data. This methodology facilitates proactive maintenance and ensures structural integrity by providing a strong method for non-destructive evaluation and condition assessment of infrastructure.

Corresponding author's email address: oodeyinka@unilag.edu.ng

Plate 6: Image of the web application prediction on a mild spall

Two separate flaws that were expressly classified as "spall," or localised surface damage or degeneration, were found in the concrete infrastructure analysis shown in Plate 6. This assessment's supporting picture quality was judged "Good," guaranteeing the accuracy of the defect diagnosis. These spall flaws' exact coordinates were [51.931008367919922, 294.204912109375] and [50.09276300810547, 350.27086961090625], which provided accurate geographic references for more research or rehabilitation.

4. Conclusion

This study presented the development of a web-based system for detecting structural cracks and spalls in buildings using deep learning–based computer vision. This work demonstrated the practical integration of the Detectron2 model into a web-based application, enabling real-time crack and spall detection accessible through any browser. This deployment illustrates the feasibility of scalable, automated structural health monitoring systems for residential and institutional buildings. The developed system represents a significant step toward automating structural defect detection in the Nigerian context and beyond. By combining benchmark datasets with local data, it provides a framework that balances reproducibility with contextual relevance. Future work can focus on expanding the dataset to better capture region-specific crack patterns, integrating IoT-enabled monitoring for continuous assessment, and refining the models to improve accuracy on fine cracks. Such advancements will move closer to achieving reliable, real-time, and cost-effective solutions for ensuring the safety and longevity of built infrastructure.

REFERENCES

Ahmadzadeh, M., Zahrai, SM. and Bitaraf, M. 2025. An integrated deep neural network model combining 1D CNN and LSTM for structural health monitoring utilizing multisensor time-series data. *Structural Health Monitoring*, *24*(1): 447–465. https://doi.org/10.1177/14759217241239041

Alomari, O. 2022. Identification and Categorization of Building Defects. *Civil Engineering and Architecture*, *10*: 438–446. https://doi.org/10.13189/cea.2022.100204

Czajkowska, A. and Ingaldi, M. 2021. Structural Failures Risk Analysis as a Tool Supporting Corporate Responsibility. *Journal of Risk and Financial Management*, *14*(4): Article 4. https://doi.org/10.3390/jrfm14040187

Devi, D. and Naidu, SV. 2023. Leveraging Deep Learning Models for Weld Defect Detection. *Industrial Engineering Journal*, *52*(12).

Dong, CZ. and Catbas, FN. 2021. A review of computer vision–based structural health monitoring at local and global levels. *Structural Health Monitoring*, *20*(2): 692–743. https://doi.org/10.1177/1475921720935585

Corresponding author's email address: oodeyinka@unilag.edu.ng

El-Din Hemdan, E., & Al-Atroush, ME. 2025. A Review Study of Intelligent Road Crack Detection: Algorithms and Systems. *International Journal of Pavement Research and Technology*, 1–31. https://doi.org/10.1007/s42947-025-00556-x

Entezami, A. 2021. *Structural Health Monitoring by Time Series Analysis and Statistical Distance Measures*. Springer International Publishing. https://doi.org/10.1007/978-3-030-66259-2

Flah, M., Nunez, I., Ben-Chaabene, W., and Nehdi, ML. 2020. Machine Learning Algorithms in Civil Structural Health Monitoring: A Systematic Review. *Archives of Computational Methods in Engineering*, *28*(4): Article 4. https://doi.org/10.1007/s11831-020-09471-9

Guo, T., Wu, L., Wang, C., and Xu, Z. 2019. Damage detection in a novel deep-learning framework: A robust method for feature extraction. *Sage Journals*. https://doi.org/10.1177/1475921719846051

Hou, L., Chen, H., Zhang, GK, and Wang, X. 2021. Deep Learning-Based Applications for Safety Management in the AEC Industry: A Review. *Applied Sciences*, *11*(2): Article 2. https://doi.org/10.3390/app11020821

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., and Inman, DJ. 2021. 1D convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, *151*: 107398. https://doi.org/10.1016/j.ymssp.2020.107398

Naresh, M., Sikdar, S. and Pal, J. 2023. Vibration data-driven machine learning architecture for structural health monitoring of steel frame structures. *Strain*, *59*(5): e12439. https://doi.org/10.1111/str.12439

Özgenel, ÇF. 2019. *Concrete Crack Images for Classification* [Dataset]. Mendeley. https://doi.org/10.17632/5Y9WDSG2ZT.2

Psuj, G. and Szymanik, B. 2023. Structural Health Monitoring: Latest Applications and Data Analysis. *Applied Sciences*, *13*(13): 7617. https://doi.org/10.3390/app13137617

Qiu, S., Zhang, P., Tang, X., Zeng, Z., Li, S. and Hu, B. 2025. Algorithm for crack detection of terracotta warriors based on fusion of multi-network hyperspectral images. *Journal of the Franklin Institute*, *362*(12): 107860. https://doi.org/10.1016/j.jfranklin.2025.107860

Rizia, Mst. M., Reyes-Munoz, J. A., Ortega, A. G., Choudhuri, A. and Flores-Abad, A. 2025. Intelligent Crack Detection in Infrastructure Using Computer Vision at the Edge. *Expert Systems*, *42*(2): e13784. https://doi.org/10.1111/exsy.13784

Roboflow 2023. *Crack and spall oke Dataset Overview*. Crack and Spall Oke Computer Vision Dataset. https://universe.roboflow.com/vcc-zmks5/crack-and-spall-oke

Scuro, C., and Fusaro, PA. 2022. Structural Health Monitoring Systems: An Overview. *2022 IEEE International Workshop on Metrology for Living Environment (MetroLivEn)*: 232–236. https://doi.org/10.1109/MetroLivEnv54405.2022.9826933

Sharma, S., Sisir D. and Mansi B. 2024. *Transfer Learning for Wildlife Classification: Evaluating YOLOv8 against DenseNet, ResNet, and VGGNet on a Custom Dataset.* arXiv preprint arXiv:2408.00002

Singh, P., Wijethunga, R., Sadhu, A. and Samarabandu, J. 2025. Expert evaluation system for pothole defect detection. *Expert Systems with Applications*, *277*: 127280. https://doi.org/10.1016/j.eswa.2025.127280

Song, H. 2023. A Consistent Mistake in Remote Sensing Images' Classification Literature. *Intelligent Automation & Soft Computing*, *37*(2): 1381–1398. https://doi.org/10.32604/iasc.2023.039315

Tzimas, M. 2023. Structural Health Monitoring Using Machine Learning and Synthetic Data. *Graduate Theses, Dissertations, and Problem Reports*. https://doi.org/10.33915/etd.11833

Corresponding author's email address: oodeyinka@unilag.edu.ng

Xu, J., Luo, L., Saw, J., Wang, C.-C., Sinha, S. K., Wolfe, R., Soga, K., Wu, Y., and DeJong, M. 2024. Structural health monitoring of offshore wind turbines using distributed acoustic sensing (DAS). *Journal of Civil Structural Health Monitoring*, *15*(2): Article 2. https://doi.org/10.1007/s13349-024-00883-w

Xu, J., Wang, S., Han, R., Wu, X., Zhao, D., Zeng, X., Yin, R., Han, Z., Liu, Y. and Shu, S. 2025. Crack segmentation and quantification in concrete structures using a lightweight YOLO model based on pruning and knowledge distillation. *Expert Systems with Applications*, *283*: 127834. https://doi.org/10.1016/j.eswa.2025.127834

Youwai, S., Chaiyaphat, A. and Chaipetch, P. 2024. A Fused Deep Learning Expert System for Road Damage Detection and Size Analysis Using YOLO9tr and Depth Estimation. *2024 International Conference on Intelligent Computing and Next Generation Networks (ICNGN)*, 1–5. https://doi.org/10.1109/ICNGN63705.2024.10871286

Zhao, M., Xu, X., Bao, X., Chen, X. and Yang, H. 2024. An Automated Instance Segmentation Method for Crack Detection Integrated with CrackMover Data Augmentation. *Sensors*, *24*(2): 446. https://doi.org/10.3390/s24020446

Corresponding author's email address: oodeyinka@unilag.edu.ng