Kurdistan Journal of Applied Research (KJAR)
Print-ISSN: 2411-7684 | Electronic-ISSN: 2411-7706

Original Article

Check for updates

Integrating Attention Modules with YOLOv8 for Enhanced Crack Detection and Segmentation

Samuel Owoeye [a *], **Folasade Durodola** [a], **Sikirulahi Abdulkareem** [a], **Olugbenga Omotainse** [b]

[a] Department of Mechatronics Engineering, Federal University of Agriculture, Abeokuta, Nigeria.
[b] Department of Agriculture and Bioresources Engineering, Federal University of Agriculture, Abeokuta, Nigeria.

Submitted: 27 September 2025
Revised: 4 February 2026
Accepted: 12 April 2026

***Corresponding Author**:
owoeyeso@funaab.edu.ng

Keywords: Attention mechanisms, Deep learning, Building inspection, Shuttle attention, YOLOv8.

How to cite this paper: S. Owoeye, F. Durodola, S. Abdulkareem, O. Omotainse," Integrating Attention Modules with YOLOv8 for Enhanced Crack Detection and Segmentation", KJAR, vol. 11, no. 1, pp: 121-142, Jun 2025, doi: 10.24017/science.2026.1.9

Abstract: Earlier cracks identification is very crucial in structural building maintenance as it is the main signifier of building deterioration. Manual inspection processes are slow, expensive and can be easily compromised by human error. Though the You Only Look Once version 8 (YOLOv8) has emerged as a powerful framework for automated crack detection, it faces limitations in accurately detecting small, irregularly shaped, and partially obscured cracks due to feature loss in deeper network layers and insufficient pixel-level precision. This study addresses these limitations by strategically integrating five attention mechanisms into YOLOv8's architecture: Convolutional Block Attention Module (CBAM), Efficient Channel Attention (ECA), Selective Kernel Attention (SKA), Shuffle Attention, and Global Attention Mechanism. The attention modules were placed at critical positions within the backbone and neck regions to enhance feature representation without compromising computational efficiency. Using a comprehensive dataset of 13,169 building crack images with 19,386 annotations, each attention-enhanced variant was trained and evaluated against the baseline YOLOv8 model. Results demonstrate consistent improvements across all attention mechanisms. CBAM achieved the highest segmentation accuracy with mask mean Average Precision (mAP) @0.5 of 0.820 (0.4% improvement), while ECA provided the most parameter-efficient enhancement, improving box precision by 3.5% with only 41 additional parameters. SKA excelled in recall performance, achieving 0.724 (1.0% improvement), valuable for comprehensive building crack detection. All variants maintained practical deployment feasibility, supporting real-time building inspection applications. The findings confirm that attention mechanism integration offers an effective approach to enhance YOLOv8 for building crack detection, providing empirical evidence for attention module selection based on specific deployment constraints and supporting the development of more reliable automated building inspection systems.

1. Introduction

Civil engineering infrastructures, such as bridges, buildings, pavements, and tunnels, are prone to structural cracks, which pose serious safety hazards. Early identification of these cracks is essential to avoid devastating failures and expensive maintenance [1]. As urban growth intensifies, structures are subjected to greater loading conditions and environmental factors, accelerating structural degradation [2]. A study by Li *et al.* [3] indicates that 60% of in-service buildings already suffer from varying forms of cracks, underscoring an urgent need for effective crack detection and segmentation methods.

Cracks represent the initial signs of structural degradation. Traditional crack inspection methods remain largely manual, time-consuming, and prone to human error. Conventional image processing techniques often struggle with complex backgrounds, varying illumination, hairline cracks, and noisy surface textures, particularly in hazardous environments [4]. Moreover, traditional non-destructive evaluation

methods can typically detect cracks only at the later stages of the degradation process, not in the early stages [5]. This limitation highlights the necessity for automated, early-stage detection approaches.

Recent advances in deep learning and computer vision have enabled automated crack detection and segmentation with high accuracy, significantly improving inspection efficiency [6-8]. Among these, You Only Look Once version 8 (YOLOv8) is an advanced object detection and segmentation framework known for its real-time performance and robust feature extraction capabilities [9]. However, while YOLOv8 excels at general-purpose segmentation [10], it struggles to achieve sufficiently high accuracy when dealing with small cracks, irregularly shaped cracks, and partially obscured cracks [11]. More specifically, three key limitations persist: firstly, even though YOLOv8 is quite effective in terms of real-time performance, efforts to enhance accuracy increase the cost of computation [12], secondly, at deeper layers of networks, fine features in the crack image are lost [13], lastly, YOLOv8 is trained with bounding-box and coarse mask supervision in mind; however, crack segmentation needs pixel-level accuracy, which the model does not have [14, 15].

This has led to the necessity to optimize the YOLOv8 model in crack detection by making changes to the architecture. Although these techniques demonstrate the potential of attention modules to enhance feature representation, their role in the recent YOLOv8 architecture is under-researched, particularly concerning segmentation.

The research question that this research aims to answer is to what extent can the incorporation of attention mechanisms in the segmentation head and backbone of YOLOv8 be used to both increase the accuracy and robustness of crack segmentation and to keep the speed of real-time inference constant? This question not only explores the architectural implications of integrating attention modules with YOLOv8 but also explores performance trade-offs between computational efficiency and segmentation quality in crack detection scenarios. The contributions of this work are mainly three parts. Firstly, the enhancement of crack segmentation performance of the YOLOv8 by integrating multi-scale attention modules. Secondly, benchmark crack segmentation datasets were evaluated thoroughly to observe the accuracy and the inference speed. Thirdly, evaluation of the impact of each attention mechanism, which would be useful in future studies on real-time detection of defects.

This research closed this gap between the current state of the art object detection structures and the requirements of the particular structure monitoring applications by improving the performance of the YOLOv8 on fine-grained segmentation tasks.

The remainder of this article is organized as follows. Section 2 reviews related work on crack detection methods, deep learning-based segmentation, and attention mechanisms. Section 3 describes the proposed YOLOv8 architecture with integrated attention modules. Section 4 presents the results and the evaluation metrics. Section 5 reports and discusses the results, including precision, viability, and integration. Section 6 concludes the paper with a summary of findings and directions for future research.

2. Literature Review

This review focuses on using the attention module to improve YOLOv8 for crack segmentation. The model is known for its speed and accuracy, but with small and obscured cracks, it struggles. In this paper, the YOLOv8 architecture was optimized with the use of Integrated Attention Modules to improve the accuracy of the YOLOv8 model, especially for little cracks, irregularly shaped, and obscured cracks.

Traditional techniques and conventional methods, including manual inspection, sensor-based observation and image processing techniques, have been mentioned as inefficient, expensive, subjective and weaker in a complicated background [16]. In contrast to that, deep learning models like YOLOv8 are already presented as providing novel, effective, precise, solid, and less biased models for crack detection and segmentation [17]. Cracks are often not symmetrical in their design and have different magnitudes, making it difficult to focus on and adequately analyse the important regions by models [18]. This implies that YOLOv8 performs well on more noticeable structural defects, but its efficiency and speed are lacking for finer details. Models struggle to distinguish actual cracks from pseudo-cracks like water stains, shadows, or surface textures [19].

High-precision models require substantial computational demand, resulting in large model sizes and slow detection speeds, hence constraining their utility in real-time scenarios. When tasked with detecting

irregular, fine, or small cracks in diverse and challenging environments, achieving high accuracy often necessitates a more complex model, which can lead to larger sizes and slower inference [20]. Even YOLOv8n, the smallest variant, which can be said to be very lightweight, uses numerous standard convolutions and cross stage partial – fast (C2f) modules. While these contribute to accuracy, they can reduce running speed and increase the number of parameters in the model [21].

The use of attention mechanisms is a welcome change from YOLOv8 due to the fact that the model will learn to focus on the most important features in an image, whilst ignoring the irrelevant noise and background interference. This "smarter" processing directly addresses the issues of large model size and slow detection speed by improving efficiency and accuracy without necessarily inflating the model's computational footprint proportionally. A study by Dong *et al*. [22] incorporated the Biformer attention mechanism to enhance the capacity of the network to process objects of different scales, successfully attaining both the global and local features. This enabled the model to have improved perception of cracks at various magnifications and concentrate on the areas of the pavement that are of concern, thereby increasing visual performance. By selectively emphasizing important features, it reduced the need for the model to process all information equally, making it more efficient. The similarity attention module (SimAM) mechanism is a significant enhancement because it's parameter-free. A study by Cao *et al.* [23] optimizes feature responses to human visual attention with the simulation of human visual attention to enable the model concentrates on important crack features (particularly thin and low-contrast) and to inhibit background noise without adding any learnable parameters. This benefits directly in terms of the complexity of the model and computational load, and enhances detection accuracy and robustness.

In a study by Zhang *et al.* [11], convolutional block attention module (CBAM) was added to make the model more sensitive to small cracks by giving greater attention to both channel and spatial representations. It automatically allocates weights to reject the critical features and suppress the redundant information. While CBAM involves learnable parameters that can add computational overhead, its inclusion is often paired with other lightweight strategies to achieve an overall balance. The large separable kernel attention (LSKA) module has been explicitly structured to find the presence of surface cracks and local features with emphasis on the shape of the cracks and adaptation to features such as brittle cracking and long shapes. More importantly, it does it by breaking large 2D convolution kernels into stacked 1D kernels, which reduces computation and memory spent relative to large kernel attention [12]. The triplet attention (TA) module is also added to the neck network to combine the context information and enhance the target representation and minimize the background interference, without compromising the computing efficiency [10].

The parameter-free mechanisms, such as SimAM, have a direct impact on lighter models and faster inference because they enhance feature quality without incurring overheads [11]. LSKA also directly reduces computation and memory. This facilitates deployment on resource-constrained devices or in real-time applications. In summary, attention mechanisms help YOLOv8 overcome its limitations regarding model size and speed by making the network more intelligent and selective in its feature processing. They enable the model to achieve higher accuracy and robustness, especially for challenging crack detection tasks, either by introducing little to no additional computational burden (e.g., SimAM, LSKA) or by ensuring that the extra complexity is justified by making the extracted features more useful and focused (e.g., Biformer, CBAM, Multi-Head Self-Attention, TA).

Finding methods of integrating attention mechanisms into deep learning architectures to detect cracks and structural defects has been a research focus within the scope of improved feature discrimination in deep, noisy backgrounds, where cracks are thin, irregular, and usually low-contrast. The studies below illustrate the application of various attention techniques in diverse crack detection structures and applications.

Cui *et al*. [24] proposed an Attention U-Network (Att-UNet), an improved fully convolutional network that incorporates an attention gate module into the standard UNet encoder-decoder architecture for end-to-end pixel-level crack segmentation. The attention gate module enabled the network to focus on critical crack regions, suppress irrelevant activations, and effectively extract multi-scale crack features. Evaluated on concrete crack images, Att-UNet outperformed both fully convolutional network and UNet across all test conditions in terms of accuracy, precision, and F1-score, while also demonstrating better generalization ability. This work established attention-gated skip connections as a powerful design choice

for crack segmentation tasks. Qu *et al.* [25] proposed a crack detection algorithm for concrete pavement that combines an encoder-decoder structure with Residual 2-stage Network (Res2Net) modules, attention mechanism integration, and cascade-parallel dilated convolutions. The attention mechanism in the encoder enabled fast focus on crack regions, while dilated convolutions enlarged the receptive field without reducing feature map resolution. The end prediction was a fusion of multi-scale side outputs of a feature pyramid decoder. Optimal dataset scale, optimal image scale, and average precision were tested and found to be worse than the method to be tested, using various datasets (public crack) of interest.

The research conducted by Xu *et al.* [26] dealt with the issue of the scarcity of training data in controlling dam crack detection by introducing the concept of community evolutionary generative adversarial network, which is set to enable the expansion of the crack image dataset by synthesizing features to improve its content, suggesting augmented feature-based RCNN (AF-RCNN), an extension of Faster-RCNN that adds an attention mechanism to the crack detection module to give proposal boxes containing crack targets adaptive weights and combine high-confidence candidates. AF-RCNN resulted in 81.07% mAP on the extended dam crack dataset, which is 8.39% higher than the original faster-RCNN baseline. Junhua *et al.* [27] introduced an automatic pavement crack detection algorithm that is based on YOLOv5, which includes attention modules and is aimed at detecting small pavement cracks in real-time. It tested several connection settings and discovered that YOLOv5-CoordAtt had the greatest accuracy of 95.27% with self-built data gathered in one of the Chinese cities called Linyi, and compared to the traditional image processing system, as well as other deep learning systems in different settings. This paper has a direct connection to the current project since it shows the efficiency of integrating attention modules into the YOLO architecture family in the detection of pavement cracks.

Jiang *et al*. [28] developed a two-step attention classification-and-segmentation network to detect the micro-crack anomaly in cells of photovoltaic modules. The classification network utilized transfer learning and deep supervision to forecast the probability of anomaly, and the segmentation network adopted the M-shaped encoder-decoder architecture with an internal attention unit to reduce background noise and increase micro-crack feature detection. The attention module was found to be very effective in increasing the accuracy of the classification and segmentation. The model was able to learn effectively on a small batch of labelled samples with an accuracy of 100.0% and an F1-score of 0.541 on real PV electroluminescence images. Jing *et al*. [29] created a model named attention residual U network, which is a road crack detector network that incorporates the use of CBAM in the encoder and decoder levels of UNet. The CBAM module that was used on both channel and spatial dimensions enabled the model to derive global and local detail information simultaneously. The basic block connections that were not necessary to replace the normal convolutional layers were done to prevent gradient vanishing as the depth increases, and the input-output CBAM feature connections lengthened the feature transmission path. The model was tested on the DeepCrack dataset, the crack forest dataset, and a custom road image dataset, where it showed better results with respect to the existing deep learning techniques and a higher level of integrity of the crack extraction.

Guo *et al*. [30] introduced a semantic segmentation network with a transformer to detect pavement cracks based on swin transformer as an encoder and UperNet with an attention module as a decoder. The hierarchical Swin Transformer architecture allowed the model to capture both the global and long-range semantic characteristics of pavement cracks due to the weakness of convolutional neural networks (CNNs), which lose information due to successive convolutions. Fine crack detail was retrieved by the attention module in the decoder; thus, fine and thin cracks could be accurately detected even when the conditions were noisy. The proposed model had the highest mF1 and mRecall scores at 0- pixel tolerance when compared with six semantic segmentation models of three public pavement crack datasets. A study by Liu *et al*. [31] proposed a Crack Transformer network, which is able to effectively utilize the transformer-style attention mechanisms to search for subtle crack patterns in order to perform fine-grained crack detection. Also, Jiang *et al.* [32] conducted a study that introduced attention mechanisms in every encoding skip-connection to attenuate the background noise and increase the clarity of the edges of the crack. On a similar note, Lin *et al.* [33] investigated the application of the attention mechanism to the YOLOv5 model, and this led to the detection of pavement cracks being improved.

Although there has been substantial progress in automated crack detection using the application of deep learning models like YOLOv8, there is a significant research gap in solving the difficulties that confront the detection of small and irregular shapes of cracks, as well as partially obscured cracks, on different structural scenarios. Conventional applications of YOLOv8 typically face challenges of preserving high pixel-level accuracy because features are lost in lower network layers, and this may result in a lower detection accuracy. In addition, research conducted so far has not paid much attention to the possibility of having attention mechanisms to make feature representation successful without putting too much of a load on the model in terms of computational load. Although certain studies have examined the problem of attention modules integration into the framework of different deep learning models, not a single study has conducted dedicated research to support their particular role in crack detection in the YOLO framework. This paper seeks to address this gap by thoroughly incorporating several attention modules into YOLOv8, examining their effect on crack detection, segmentation, and computational efficiency. Through this, we offer an in-depth discussion of how personalized attention systems may contribute to the enhancement of traditional detection systems and, eventually, lead to the creation of a more powerful automated system to monitor buildings and their safety conditions.

This paper has optimized the YOLOv8 architecture by using the integrated attention modules to enhance the accuracy of the YOLOv8 model, particularly on small cracks, irregular cracks and blurred cracks.

3. Materials and Methods

3.1. Data Collection

The dataset was created by combining three sources to cover various crack types and imaging conditions. The first source was custom data collected through web scraping from public repositories and structural inspection sites. The second and third sources were established Roboflow datasets: Crack-Detection-5 and Crack-4. Initial collection totaled 22,627 images with 32,824 annotations: Crack-Detection-5 (6,908 training, 1,924 validation, 987 test), Crack-4 (6,680 training, 1,845 validation, 923 test), and custom data (3,102 training, 129 validation, 129 test).

3.2. Data Preprocessing

The quality of the dataset was ensured through manual filtering. Pictures that contained distinct surfaces of concrete that had visible cracks were retained, and pictures with invalid tags, low resolution, or incorrect tags were removed. Dataset merging required careful manual selection to combine the three sources while removing duplicates and maintaining variety across different crack patterns and lighting conditions. All annotations were standardized to YOLOv8 format with consistent class mapping (crack = class 0). The image of the dataset used is shown in figure 1.

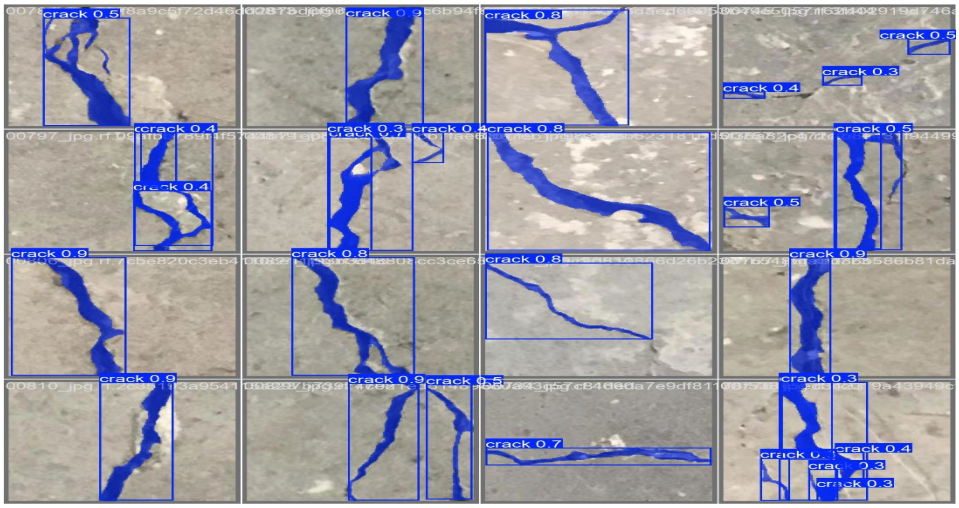


Figure 1: Sample of the dataset.

3.3. Data Augmentation

Data augmentation varied by source. The Crack-Detection-5 and Crack-4 datasets were obtained with pre-applied augmentations. For the custom scraped data, the Roboflow platform handled preprocessing, including auto-orientation correction and resizing to 640×640 pixels. Augmentation generated 3 outputs per training example using horizontal and vertical flips, rotations between -15° and +15°, brightness and contrast adjustments (factors 0.8 - 1.2), and grayscale conversion applied to 25% of images to simulate different lighting conditions.

3.4. Dataset Split and Validation

The final dataset contained 13,169 images with 19,386 crack annotations after filtering and manual curation. Data was split into 10,000 training images (14,640 annotations), 2,053 validation images (3,037 annotations), and 1,116 test images (1,709 annotations). Manual validation checks ensured no duplicates existed across splits, consistent class distribution, and proper image-annotation correspondence. All pre-processing and conversion scripts were documented for reproducibility.

3.5. Base YOLOv8 Architecture

3.5.1. YOLOv8 Segmentation Framework

YOLOv8 was selected as the base architecture for its unified framework supporting both object detection and instance segmentation tasks within a single model [34]. Its architecture is anchor-free, which directly regresses centres and sizes of objects, making it easier to detect than anchor-based approaches. The framework consists of three primary components: a CSPDarknet53 backbone for hierarchical feature extraction [35], a feature pyramid network (FPN) with path aggregation network (PAN) neck for multi-scale feature fusion [36], and dual prediction heads for simultaneous detection and segmentation outputs.

The backbone progressively extracts features at multiple scales through a series of convolutional blocks with residual connections, producing feature maps at 1/8, 1/16, and 1/32 of the input resolution. The neck component combines these multi-scale features through top-down and bottom-up pathways, enabling effective detection of cracks ranging from fine hairline patterns to larger structural damage. The detection head predicts bounding box coordinates, object- ness scores, and class probabilities, while the segmentation head generates pixel-level masks for precise crack boundary delineation [37].

The unified loss function, L_{total} combines three components as shown in equation 1 [38]:

$$L_{total} = L_{cls} + L_{box} + L_{seg} \quad (1)$$

where L_{cls} represents classification loss using binary cross-entropy for crack/non-crack discrimination, L_{box}denotes bounding box regression loss computed using complete intersection over union (CIoU), and L_{seg} represents segmentation loss calculated through binary cross-entropy at the pixel level.

3.6. Baseline Model Configuration

The YOLOv8n real-time object detection model series has several hyperparameters that can be optimised to enhance the performance of the model. The batch size is one of the more important parameters among them, with a default setting of 16 or 32, although it can be changed depending on the available memory in a graphics processing unit (GPU). It is possible to have larger batch sizes to increase the quality of gradient estimates, but this could come at the cost of increased memory usage and training time. The epochs determine the number of times that the entire training data will move through the model, and the default value typically runs to 100 or 300 epochs. Adaptive adjustments can be adjusted according to the rate of convergence of loss or validation accuracy, and early stopping can be used when the performance is not improving, hence promoting efficiency in the training.

Another critical hyperparameter is the learning rate, and its default value is usually 0.001, although 0.01 or 0.0001 can be used. Adaptive techniques, like the use of learning rate schedules, such as step decay, exponent decay or cyclical learning rates, can also boost model performance highly. The use of

warm-up methods of learning at the start of training and progressive decrease throughout the training process can further optimise results.

Finally, the dropout rate, which is defined between 0.0 and 0.5 depending on the model structure and task difficulty, is frequently 0.2. This parameter is also to be adjusted depending on the performance of the validation of the model. When there is a suspicion of overfitting, e.g. training and validation accuracy are high, one can increase the dropout rate, which can enhance the capacity of a model to generalise. All in all, proper hyperparameter management enables practitioners to improve the accuracy, robustness, and performance of the YOLOv8n model in a number of object detection tasks, thereby matching model training to the available computer resources and dataset.

The control experiments used YOLOv8n (nano variant) and 3,263,811 parameters and 11.5 giga floating point operations per second (GFLOPs) of computational complexity. This has been chosen as a lightweight implementation to trade off model capacity versus computational efficiency to allow a clear fault to consider the contributions of attention mechanisms without introducing undue complexity to the baseline. The model structure has 151 layers that are arranged into a backbone-neck-head architecture, where the depth of feature extraction of (64, 128, 256, 512) channels was deployed through the four primary stages. Standardisation of input preprocessing converts images to 640640 pixels with the use of letterboxing to preserve aspect ratios, and then normalises them in the [0, 1] range. The model initialisation uses common objects in context (COCO) pre-trained weights, where learned representations are used in identifying edges and texture, which can be transfer-learned to identify cracks. The last classification layer was changed to 80 COCO classes to a single-class crack detector (nc=1), and the dimensions of the output tensors were also changed accordingly.

3.7. Attention Module Integration

3.7.1. Selected Attention Mechanisms

Five attention mechanisms were selected for integration based on their proven effectiveness in computer vision tasks and computational feasibility for crack segmentation applications. The selection criteria prioritized mechanisms that address different aspects of feature enhancement: channel-wise recalibration, spatial feature refinement, and multi-scale feature integration.

a. **Convolutional Block Attention Module** which was presented in figure 2, was selected due to its dual-pathway design, which is a combination of channel and spatial attention methods. CBAM uses channel attention to decide what to attend to, and then spatial attention to decide where to attend to. It is especially appropriate in crack detection, where the importance of features and the localization of the position of the feature is of the essence. The lightweight nature of the module imposes very little computational burden but gives it a wide coverage in terms of attention [39].

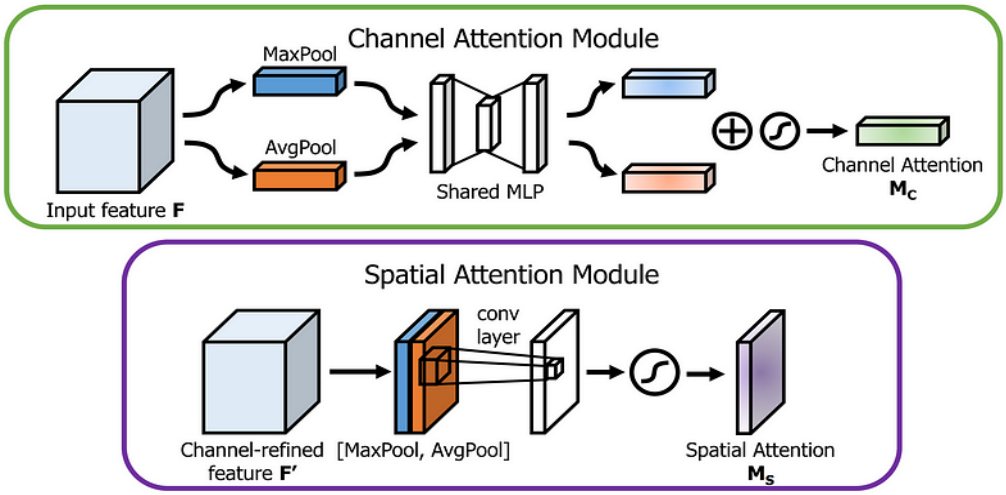


Figure 2: Architecture of the Convolutional Block Attention Module using channel attention and spatial attention to refine input feature maps [11].

b. ***Global Attention Mechanism*** reduces information loss during feature processing by preserving both spatial and channel information. Figure 3 shows the global attention mechanism (GAM) architecture; unlike traditional attention methods that compress spatial dimensions, GAM maintains the complete spatial structure while enhancing important features. It uses a three-dimensional attention map that considers spatial positions and channel relationships simultaneously. This comprehensive approach is valuable for crack segmentation as it preserves fine spatial details that are essential for detecting thin crack boundaries while maintaining global context awareness.

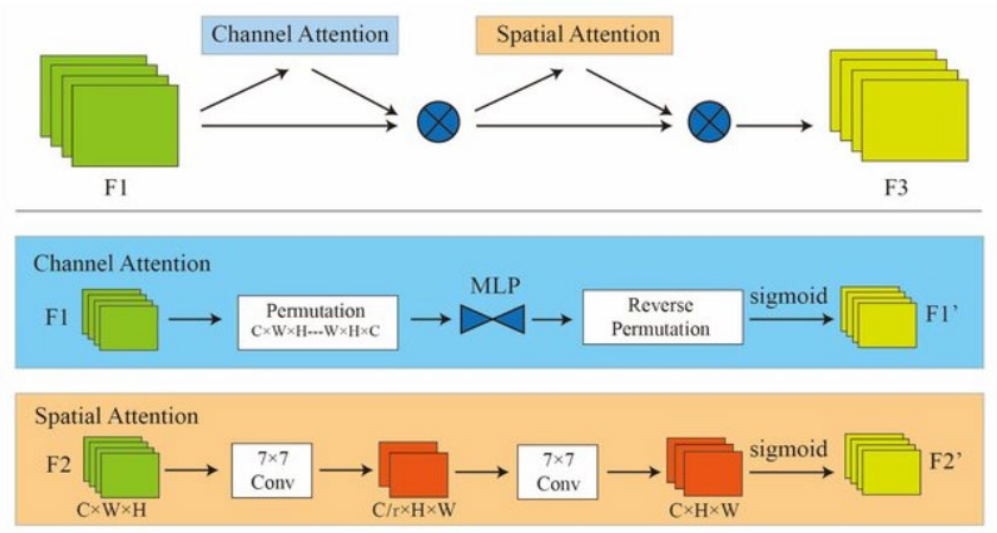


Figure 3: Architecture of the Global Attention Mechanism (GAM) that compress spatial dimensions and preserves complete spatial and channel information by applying a three-dimensional attention map [24].

c. ***Efficient Channel Attention*** illustrated in figure 4, offers a lightweight method for channel attention without dimensionality reduction. Efficient channel attention (ECA) produces channel attention weights with a one-dimensional convolution across channels, as opposed to the computational expense and possible loss of information of fully connected channel attention layers applied with other channel attention algorithms. It is efficient and effective because the mechanism utilises adaptive kernel sizes depending on the channel dimensions, making it useful in crack detection applications, where computational resources might be constrained [40].

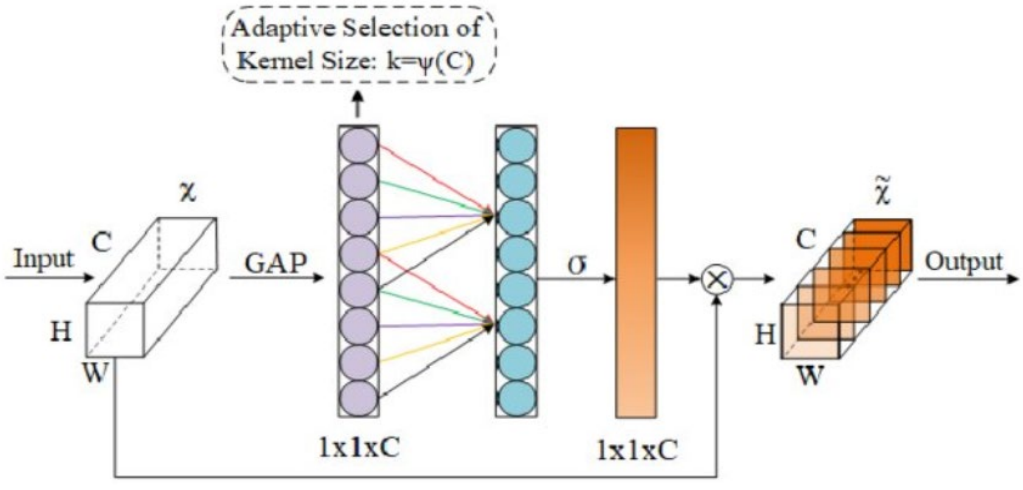


Figure 4: Architecture of the Efficient Channel Attention (ECA) module that generates channel attention weights using a fast one-dimensional convolution of size *k*, where *k* is adaptively determined by the channel dimension [40].

d. ***Selective Kernel Attention*** uses several convolutional kernels that have varying receptive fields and selectively pools the results, using attention weights learned. This multi-scale scheme allows the model to intelligently switch attention to the fine hairline cracks, reacquiring small receptivity fields to the large-scale structural destruction of large regions. The combination mechanism is selective and maximises feature extraction with different morphologies of cracks [41]. Figure 5 illustrates the SK architecture.

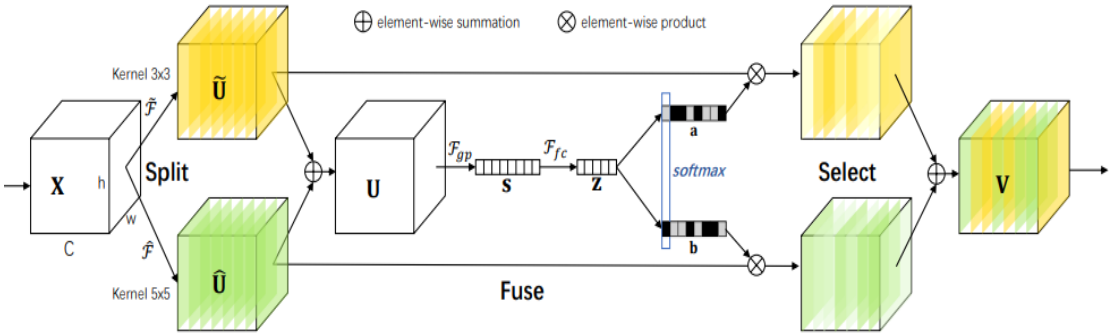


Figure 5: Architecture of the Selective Kernel Attention (SK) module that employs multiple parallel convolution branches with different kernel sizes to capture multi-scale receptive fields [41].

e. ***The Spatial Attention*** mechanism is entirely devoted to the enhancement of spatial features, as it generates spatial attention maps indicating the area of critical spatial features and inhibits the irrelevant background area [42]. The spatial attention (SA) architecture is represented in figure 6. In the case of crack segmentation tasks, this localized focus of attention assists the model to focus on the localization of cracks and disregard the complicated background texts that would otherwise lead to false positive identification.

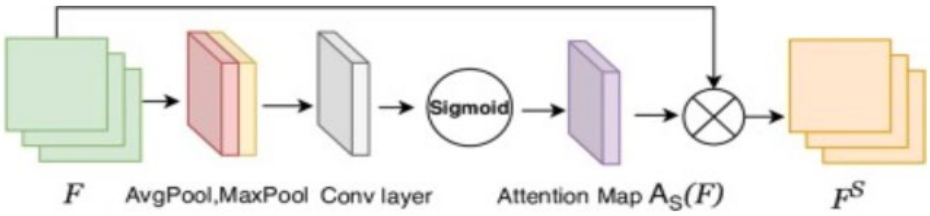


Figure 6: Architecture of the spatial attention (SA) module that generates a spatial attention map by applying average and max pooling operations along the channel axis, concatenating the resulting feature descriptors, and passing them through a convolutional layer followed by a sigmoid activation [30].

3.7.2. Integration Strategy and Placement

The attention module integration framework was systematically designed to optimize feature enhancement across the YOLOv8 architecture while preserving computational efficiency and architectural integrity. The strategic placement of attention mechanisms follows a multi-level integration approach that leverages the hierarchical nature of CNNs to maximize representational capacity at critical feature processing stages.

a. ***Hierarchical Integration Architecture:*** The integration strategy employs a three-tier placement methodology encompassing backbone, neck, and head regions. Within the backbone structure, attention modules are strategically positioned after key feature extraction blocks at scales P3, P4, and P5, corresponding to feature map resolutions of $\frac{H}{8} \times \frac{W}{8}$, $\frac{H}{16} \times \frac{W}{16}$and $\frac{H}{32} \times \frac{W}{32}$, where H and W are the input height and weight, respectively. This placement enables attention-guided feature learning during the fundamental representation extraction

phase, where attention mechanisms can identify and amplify discriminative features while suppressing irrelevant information. The mathematical formulation for backbone attention integration is expressed as in equation 2 [11]:

$$F_{backbone}^{(i)} = \mathcal{A}_i\left(C2f^{(i)}\left(F^{(i-1)}\right)\right) + C2f^{(i)}\left(F^{(i-1)}\right) \quad (2)$$

where A_i represents the attention function at layer i, $C2f^{(i)}$ denotes the C2f block operation, and the residual connection ensures gradient flow preservation, $F^{(i-1)}$ represents the input feature map from the previous layer

b. Multi-Scale Neck Enhancement: Neck-level integration targets the FPN and PAN components, where multi-scale feature fusion occurs. Attention modules are incorporated immediately following feature concatenation operations at resolutions corresponding to P3 and P4 levels. This placement strategy allows attention mechanisms to refine fused multi-scale representations, enhancing the semantic consistency across different resolution levels. The attention-enhanced feature fusion process is mathematically represented by equation 3 [27]:

$$F_{neck}^{fused} = \mathcal{A}_{neck}\left(\text{Concat}\left[F_{up}, F_{lateral}\right]\right) + \text{Concat}\left[F_{up}, F_{lateral}\right] \quad (3)$$

where F_{up} represents unsampled features, $F_{lateral}$ denotes lateral connections from the backbone, $\text{Concat}\left[F_{up}, F_{lateral}\right]$ combines the two features along channel dimension and $\mathcal{A}_{neck}$ is the feature refinment.

c. Computational Efficiency Considerations: The integration strategy incorporates computational efficiency as a primary constraint, with attention module placement optimised to balance representational enhancement against computational overhead. Early backbone placement provides maximal influence on downstream processing but incurs higher computational costs due to larger feature map dimensions. Conversely, neck-level placement offers targeted refinement with reduced computational burden while maintaining significant impact on final predictions. The total computational complexity introduced by the attention integration is quantified using equation 4 [28, 29]:

$$\text{FLOPs}_{total} = \sum_{i \in \{P3,P4,P5\}} (C_i \times H_i \times W_i \times \gamma_i) \quad (4)$$

where C_i, H_i, W_i represent channel, height, and width dimensions at scale i, and γi denotes the attention-specific computational coefficient

d. Skip Connection Preservation: All attention module integrations maintain the original architectural skip connections through additive residual pathways. This design choice ensures training stability and prevents gradient vanishing while allowing attention mechanisms to provide additive refinement to existing feature representations. The residual formulation enables the network to learn when attention mechanisms should be active versus when original features are sufficient. The comprehensive integration strategy results in a total of six strategically placed attention modules: three in the backbone (layers 5, 8, 11), two in the neck (layers 16, 20), creating a balanced distribution that maximises feature enhancement across the entire architectural hierarchy while maintaining computational feasibility for real-time applications.

3.7.3. Implementation Modifications

Implementation of attention mechanisms required systematic modifications to the Ultralytics YOLOv8 codebase while maintaining compatibility with existing training and inference pipelines [12]. The conv.py module was extended with custom attention mechanism class definitions following PyTorch's Module structure, with each attention module implemented as a standalone class featuring standardized forward pass interfaces for consistent integration across different network locations. Implementations included proper initialization methods, parameter counting, computational complexity calculations, and configurable parameters for hyperparameter tuning specific to crack segmentation tasks [43].

The tasks.py module was modified to register attention-enhanced architectures as new model variants, extending the existing model registry system to enable training using the standard. Ultralytics pipeline without requiring separate training scripts. Markup language configuration files were created for each attention mechanism variant, defining network architectures with specific attention module placements, types, locations, and hyperparameters in a structured format that integrates with YOLOv8's existing configuration system. This markup language-based approach enables easy experimentation with different attention configurations and systematic comparison of attention placement strategies [44].

Forward pass implementation carefully preserves feature map dimensions and tensor operations throughout the attention-enhanced network, with attention modules maintaining input-output dimension consistency to ensure seamless integration without requiring modifications to downstream components [45]. Special attention was given to batch dimension handling and GPU memory efficiency during attention computations. The implementation includes comprehensive error handling and validation checks to ensure attention modules function correctly across different input sizes and batch configurations, with integrated debug functionality enabling attention map visualisation and analysis to facilitate understanding of attention mechanism behaviour during crack detection tasks [46].

3.8. Experimental Design

The experimental framework was structured to provide a systematic evaluation of attention mechanism effectiveness for crack segmentation through controlled comparison across multiple attention variants. A baseline-first approach established performance benchmarks using the original YOLOv8 architecture, providing reference points for measuring attention mechanism improvements. Individual attention mechanisms were evaluated separately using identical training configurations, enabling identification of optimal attention types and their specific contributions to crack detection performance.

3.8.1.Training Configuration

Every variant of the models, including the baseline YOLOv8 and attention-enhanced models, was trained using the same hyperparameters to compare them fairly and isolate the effect of attentional mechanisms. The training used AdamW optimiser and an initial learning rate of 0.002, a final learning rate factor of 0.1, and a weight decay of 0.0005. The setup used 60 training epochs and 3 warmup epochs, and a cosine learning rate schedule to bring about a smooth convergence. AdamW optimiser is created to stabilise the learning rate as well as to reduce the loss rate.

Training was done in a batch size of 32, which was decided according to the memory limitation of the GPUs and the best convergence properties. Mosaic augmentation (turned off at epoch 15), random horizontal flips, small rotations, and photometric adjustments were also used as data augmentation. Each of the models was initialised with COCO pre-trained weights and had the last classification layer changed to a single-class detector of cracks. Fixed random seeds and consistent data loading procedures guaranteed deterministic training.

3.8.2. Evaluation Metrics

Model performance was evaluated using standard object detection (Box) and instance segmentation (Seg.) metrics. For detection performance, mean average precision (mAP) was calculated at IoU thresholds of 0.5 (mAP50), 0.75 (mAP75), and averaged across 0.5-0.95 (mAP50-95). Additional detection metrics included precision and recall to provide a comprehensive performance assessment.

The performance of segmentation was measured by mask-based adaptation of the same metrics, in which the computation of IoU was done on pixel-wise predictions as opposed to bounding boxes. The main evaluation measure was mAP50 of the detection and segmentation problems because it offers the most feasible evaluation of any crack identification database.

The complexity of models was used as a measure of computational efficiency based on the number of parameters, floating-point operations (FLOPs), and inference time per image. Convergence speed and final loss values were used to assess the usefulness of training. All measures were computed on the training set, on validation and at the end evaluation on the test set, which was held out.

3.8.3. *Hardware and Software Setup*

All experiments were conducted on Kaggle's cloud computing platform using dual Tesla T4 GPUs with 15GB of memory each. The computational environment included Python 3.11.13, PyTorch 2.6.0 with CUDA 12.4 support, and Ultralytics YOLOv8 framework version 8.3.127. Mixed precision training was enabled to optimize memory usage and training speed.

Attention mechanisms were included by systematic changes in the Ultralytics codebase: conv.py class definitions of custom attention modules, and integration of the training pipeline and YAML architecture specifications. The implementation had 5 strategically positioned attention modules: 3 in the backbone (layer 5, 8, 11) and two in the neck (layer 16, 20), with only slight computational overhead and architectural compatibility. The training made use of 8 dataloader workers who had fast access to images and automatic label verification. The dataset had 2053 validation images and 3,035 instances of cracks that offer strong coverage of evaluation. The deterministic training environments, the same data loading process and the systematic storage of checkpoints and results of the training ensured experimental reproducibility.

4. Results

4.1. Performance Improvement

The heatmap performance coverage of the performance is depicted in figure 7, which shows improvement of the performance as compared to the base model. It demonstrates the clear pictorial depiction of the role different attention modules play in improving the segmentation and detection measures. The ECA demonstrates the largest metric improvement, especially in segmentation and detection accuracy, where the improvement is found at 2.7 and 3.5 per cent, respectively. This is an indication that ECA has a great contribution to the object models' precision in categorizing and outlining in the segmentation task. As well, all the other modules, such as the SA, GAM and CBAM, show positive influences on the detection precision, whereas selective kernel attention (SKA) is negatively influential. Negative result was also recorded in the segmentation precision of GAM compared to the other attention modules, which had a positive result. The mAP50 of the detection and segmentation is showing a rather small increase in the case of SA, SKA and CBAM, but there was no increase in the mAP50 of GAM and ECA. This visualization shows the differences in the effectiveness of different attention mechanisms to construct crack segmentation.

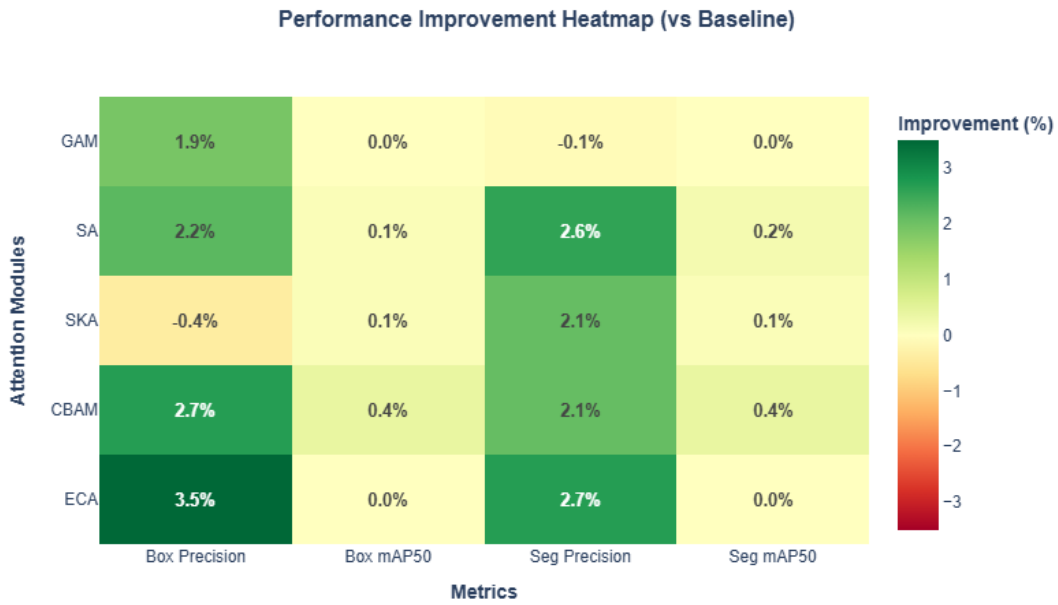


Figure 7: Performance improvement heatmap.

4.2. Computational Efficiency Analysis

Table 1 is a detailed comparison of the different model configurations of YOLOv8 in terms of complexity and efficiency of the model. As a starting point, the baseline YOLOv8 model, with 3,263,811 parameters, was used, and this serves as a point of reference when assessing the effects of other attention mechanisms. Increase of the YOLOv8 model with ECA has only achieved a small increase of only 41 parameters, with the training time of 2.730 hours and inference time of 3.1 milliseconds. This indicates that the performance of ECA increases by a large margin at the expense of only a small complexity increase. Conversely, the CBAM-augmented version of the model increases the number of parameters to 49446, resulting in a training time of 2.743 hours and an inference time of 3.2 milliseconds. Likewise, the SKA version adds the number of parameters by 24,416 parameters, and the training and inference time are also very similar to that of CBAM.

The SA model has a relatively small increase of 1,152 parameters with an inference time of 3.3 milliseconds and training time of 2.818 hours, which shows a possible complexity versus efficiency trade-off. Lastly, the GAM model has the most increased parameters of 66,580, which leads to the longest training time of 3.509 hours and the inference time of 4.2 milliseconds. GFLOPs is a measure of a computer's performance, specifically in tasks involving floating-point calculations. It is a measure of the number of billions of floating-point operations that the system can accomplish per second. The SKA and the ECA matched the baseline in terms of GFLOPs, whereas the other modules, CBAM, SA and GAM are 11.6, 12.1 and 12.4, respectively.

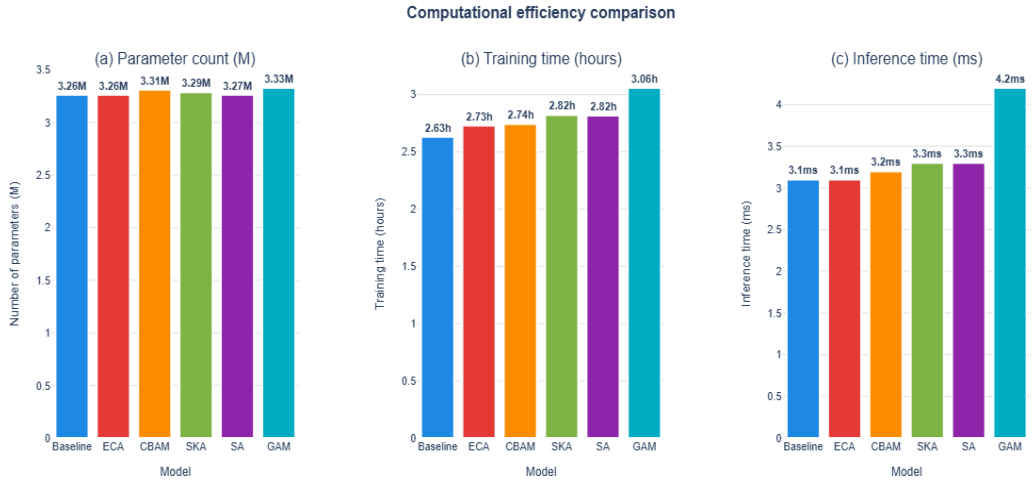
On balance, the table shows the trade-offs between increasing the complexity of models by using mechanisms of attention and the need to balance improvements in performance with computational efficiency. ECA is especially a beneficial option, where the complexity of GAM is too big a drawback to use it in situations where quick inference is needed.

Table 1: Model complexity and computational efficiency analysis.

Model	Inference Time (ms)	Parameters	Parameter Increase	GFLOPs	Training Time (h)
YOLOv8 (Baseline)	3,263,811	-	11.5	2.631	3.1
YOLOv8 + ECA	3,263,852	+41 (+0.001%)	11.5	2.730 (+3.8%)	3.1 (+0%)
YOLOv8 + CBAM	3,313,257	+49,446 (+1.52%)	11.6	2.743 (+4.3%)	3.2 (+3.2%)
YOLOv8 + SKA	3,288,227	+24,416 (+0.75%)	11.5	2.822 (+7.3%)	3.3 (+6.5%)
YOLOv8 + SA	3,264,963	+1,152 (+0.04%)	12.1	2.818 (+7.1%)	3.3 (+6.5%)
YOLOv8 + GAM	3,330,391	+66,580 (+2.04%)	12.4	3.059 (+16.3%)	4.2 (+35.5%)

Figure 8 provides a comparison of the computational efficiency of different models and offers three primary metrics: the number of parameters, training time and the inference time. The number of parameters is also relatively similar in all of the models, with the lowest number of parameters recorded in the Baseline and ECA models at about 3.26 million parameters and the highest number is registered in the GAM model at about 3.33 million parameters (Figure 8a). This minor upgrading of the parameters to the GAM model could be associated with its better performance, as the heatmap above had shown, and the trade-off between the complexity of the model and its performance could then be justified.

As shown in figure 8b, the models are also efficient in terms of training time, where the baseline and CBAM models have the lowest training time at about 2.7 hours, with the GAM model having the highest training time of about 3.0 hours. It means that GAM demonstrates better improvement in performance, though it is more expensive in terms of training time. Assessing the time of inference, the GAM model becomes the slowest at 4.2 milliseconds, as compared to the rest, which range between 3.1 and 3.3 milliseconds, as shown in figure 8c.



(a) (b) (c)

Figure 8: Comparative evaluation of attention mechanisms for computational efficiency. (a) Parameter count (in millions) for baseline model and five attention modules, (b) Training time (in hours) required for each model configuration, (c) Inference time (in milliseconds) per sample for each model.

4.3. Performance Metrics for Detection and Segmentation

Table 2 displays a detailed comparison of the object detection of the various YOLOv8 models augmented with various attention modules and the effects on the performance variables based on the object detection and object segmentation measures. The default YOLOv8 system demonstrates a medium level of performance in all metrics, with a precision of 0.846, a recall of 0.717, and a mAP of 0.836. The implementation of the ECA module leads to an increase of precision to 0.876, a slight decrease of recall to 0.709 and mAP50 to 0.836. The CBAM is also more effective than the baseline model, with a precision of 0.869, a recall of 0.713 and a mAP of 0.839, proving it to be successful in optimising attention on both channels and space.

SKA module gives a slightly worse precision of 0.843, but gives the best recall of 0.724, mAP50 of 0.837. The SA module attained a precision of 0.865, a recall of 0.709 and mAP50 of 0.837. The last significant improvement compared to the baseline and other attention modules was the precision of 0.862, recall of 0.711, and mAP50 of 0.8366 obtained by the GAM. The comparison reveals that although all the attention mechanisms improve the performance of YOLOv8, GAM is always the best, and therefore, its applicability to maximise the accuracy of detection and segmentation can be assessed in practical cases. All the attention modules posted a better mAP50-95, except for GAM, which has the same mAP50-95 as the baseline model.

Table 2: Performance metrics of YOLOv8 with attention modules (Detection).

Model	Precision	Recall	mAP50	mAP50-95
YOLOv8 (Baseline)	0.846	0.717	0.836	0.685
YOLOv8 + ECA	0.876	0.709	0.836	0.687
YOLOv8 + CBAM	0.869	0.713	0.839	0.688
YOLOv8 + SKA	0.843	0.724	0.837	0.685
YOLOv8 + SA	0.865	0.709	0.837	0.682
YOLOv8 + GAM	0.862	0.711	0.836	0.685
Best Performance	**0.876**	**0.724**	**0.839**	**0.688**
Improvement (%)	**+3.5**	**+1.0**	**+0.4**	**+0.4**

These segmentation results of the YOLOv8 models with several attention modules are significantly improved based on the major performance indicators, which are precision, recall, mAP50, and mAP50-95, as observed in table 3. The YOLOv8 base model has a precision of 0.849 and recall of 0.709, resulting in a mean average precision (mAP50) of 0.817 and a mAP50-95 of 0.459. The introduction of the ECA module results in marginal enhancements, with precision rising to 0.872 and recall to 0.704, but the

mAP50 remains unchanged. In contrast, the CBAM achieved a precision of 0.867 and a recall of 0.707, and an increase of mAP50 to 0.820.

The subsequent attention modules, SA and SKA, yielded a precision of 0.871 and 0.867, and a recall of 0.701 and 0.699, respectively, pushing their mAP50 to 0.818 and 0.819. However, the GAM achieves a slightly lower Precision than baseline YOLO at 0.848, recall at 0.711, and an mAP50 at 0.817. The mAP50-95 results showed an increase for all the attention modules, except GAM, which remains the same as the baseline YOLOv8. Overall, these results indicate a consistent trend of improvement across models, with the best performance showcasing a 2.7% increase in precision, a 0.3% increase in recall, a 0.4 % increase in mAP50 and a 1.1 % increase in mAP50-95 compared to the baseline model. This underscores the effectiveness of attention mechanisms in enhancing segmentation performance in YOLOv8.

Table 3: Performance metrics of YOLOv8 with attention modules (Segmentation).

Model	Precision	Recall	mAP50	mAP50-95
YOLOv8 (Baseline)	0.849	0.709	0.817	0.459
YOLOv8 + ECA	0.872	0.704	0.817	0.463
YOLOv8 + CBAM	0.867	0.707	0.820	0.464
YOLOv8 + SKA	0.867	0.699	0.818	0.462
YOLOv8 + SA	0.871	0.701	0.819	0.462
YOLOv8 + GAM	0.848	0.711	0.817	0.459
Best Performance	0.872	0.711	0.820	0.464
Improvement (%)	**+2.7**	**+0.3**	**+0.4**	**+1.1**

Table 4 demonstrates that all the models converged in the convergence points of 60 epochs, with divergences of 54-58 epochs showing that the optimization is not hindered by attention integration. The amount of GPU memory was also not an issue (6.12-6.76 GB of 15 GB used), and it was not close to the limit of the available training capabilities on the standard deep learning hardware that can usually be found in research facilities.

Table 4: Training performance summary across epochs.

Model	Final Epoch	Convergence Epoch	GPU Memory (GB)
YOLOv8 (Baseline)	60	54	6.12
YOLOv8 + ECA	60	57	6.22
YOLOv8 + CBAM	60	58	6.28
YOLOv8 + SKA	60	58	6.54
YOLOv8 + SA	60	57	6.53
YOLOv8 + GAM	60	58	6.76

4.4. Comparative Performance Analysis

Figure 9 provides a detailed comparison of various attention modules against a baseline in terms of performance metrics related to box detection and segmentation. The chart is divided into four quadrants, each representing different metrics namely box detection precision (Figure 9a), box mAP50 (Figure 9b), segmentation precision (Figure 9c), and segmentation mAP50 (Figure 9d). The ECA module stands out with the highest precision score of 0.876, significantly surpassing the baseline. CBAM follows closely, indicating that both modules are effective in enhancing box detection capabilities. Conversely, the SKA and GAM show lower performance, highlighting their limitations in this specific metric.

The Box mAP50 quadrant reveals a similar trend, with ECA achieving the highest score of 0.839. The segmentation metrics also reflect this pattern, where ECA and CBAM perform well, while GAM and SKA lag behind. Notably, the segmentation precision and segmentation mAP50 metrics show that ECA consistently outperforms other modules, reinforcing its effectiveness across multiple evaluation criteria. Table 1 presents comprehensive performance metrics across all variants. This comprehensive comparison illustrates the strengths and weaknesses of each attention module, providing valuable insights for optimizing model architectures in tasks related to object detection and segmentation.

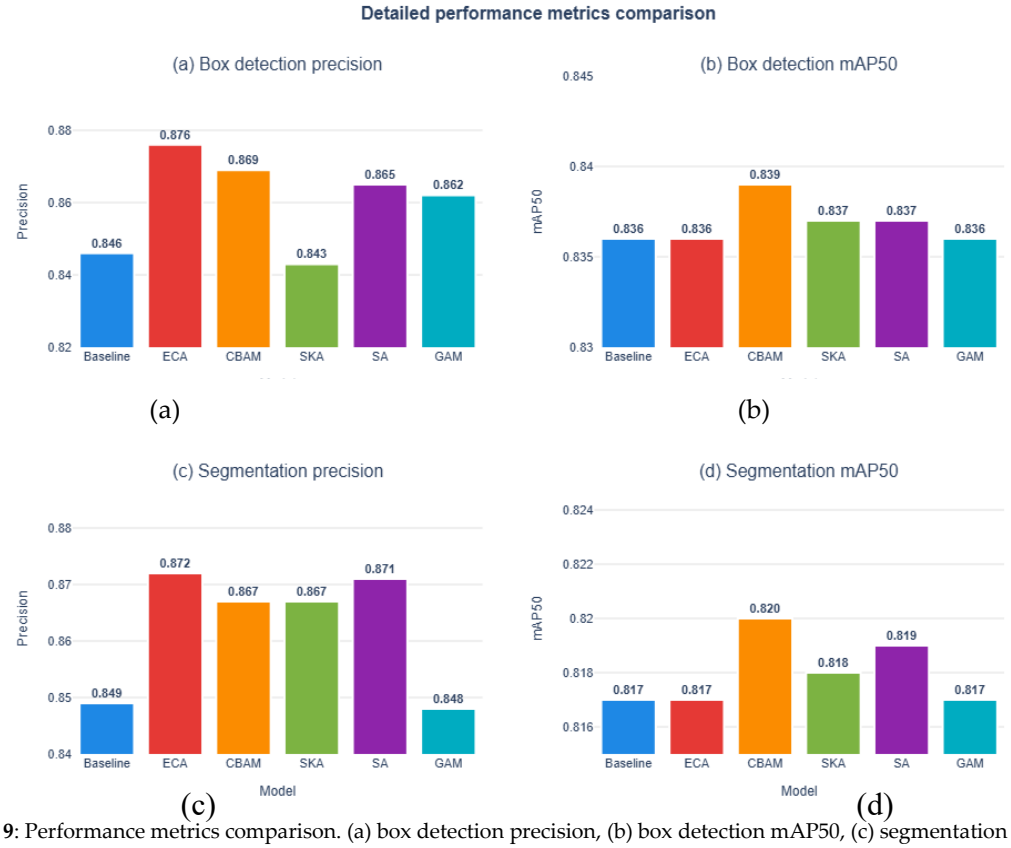


Figure 9: Performance metrics comparison. (a) box detection precision, (b) box detection mAP50, (c) segmentation precision, and (d) segmentation mAP50.

A more detailed performance comparison across a variety of metrics in a radar chart format has been provided in figure 10, and it enables a visual comparison of the performance of the various models, namely baseline, ECA, CBAM, SKA, SA and GAM with respect to various criteria: box precision, box mAP50, segregated precision, segmentation mAP50, and training speed. The performance of each model is determined by a specific shaded area that shows its strengths and weaknesses. An example is that CBAM also seems to be strong in detection mAP50, segmentation precision and segmentation mAP50, which means that it is strong at detecting and segmenting objects accurately; in addition, its training is also competitive.

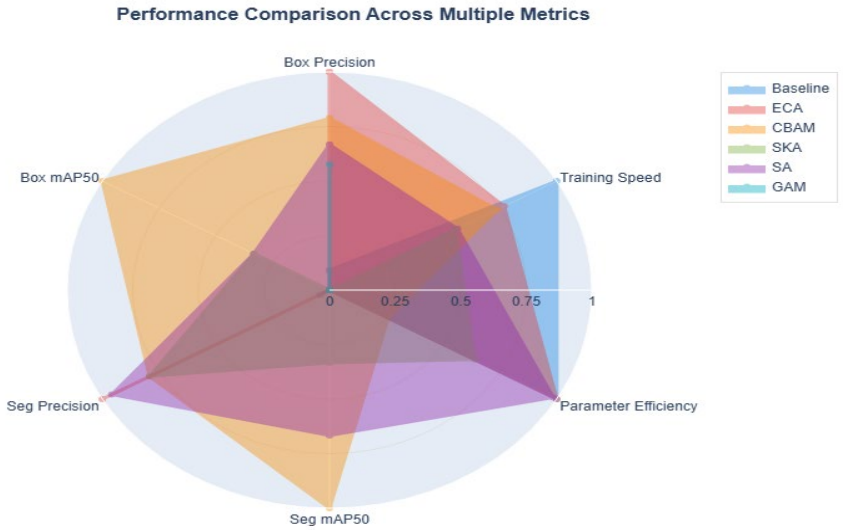


Figure 10: Multi-metric performance profile of baseline YOLOv8 and its attention-enhanced variants (ECA, CBAM, SKA, SA, and GAM).

4.5. Statistical Analysis of Results

Statistical analysis, as shown in table 5, reveals consistent improvements across all attention variants, for the segmentation mAP50, the individual gains ranging from 0.000 (ECA, GAM) to 0.003 (CBAM). While seemingly modest, these improvements are meaningful given the strong baseline performance and challenging pixel-level segmentation task for building cracks with varying widths, orientations, and surface contexts. More substantial gains emerged in precision metrics, with segmentation precision improvements ranging from -0.1% (GAM) to +2.7% (ECA), demonstrating attention mechanisms' effectiveness in reducing false positives crucial for building maintenance decision-making. The low variance among mechanisms ($\sigma^2 \approx 0.000002$) indicates robust improvements regardless of specific attention type, validating the integration methodology. The overall percentage score is between 6.2 % for ECA and 1.8 for GAM.

Table 5: Performance improvement analysis over baseline.

Model	Box-P	Box-mAP50	Seg-P	Seg-mAP50	Overall Score (%)
YOLOv8+ ECA	+0.030 (+3.5%)	+0.000 (0.0%)	+0.023 (+2.7%)	+0.000 (0.0%)	6.2
YOLOv8+ CBAM	+0.023 (+2.7%)	+0.003 (+0.4%)	+0.018 (+2.1%)	+0.003 (+0.4%)	5.6
YOLOv8 + SKA	-0.003 (-0.4%)	+0.001 (+0.1%)	+0.018 (+2.1%)	+0.001 (+0.1%)	1.9
YOLOv8 + SA	+0.019 (+2.2%)	+0.001 (+0.1%)	+0.022 (+2.6%)	+0.002 (+0.2%)	5.1
YOLOv8+GAM	+0.0016(+1.9%)	+0.000 (0.0%)	-0.001 (-0.1%)	+0.000 (0.0%)	1.8

The training convergence for the attention modules at various epochs for both the detection and segmentation mAP50 is shown in figure 11. All the models used in the detection mAP50 training curves, i.e., baseline, ECA, CBAM, SKA, SA, and GAM, have an upward trend in performance during training as a sign of successful learning, as shown in figure 11a. Likewise, the segmentation mAP50 training curves demonstrate that all the models have an upward trend with the epochs (see figure 11b). GAM is again the best with the highest mAP50 score, followed by ECA and CBAM, with the baseline model being the poor. These curves are smooth, and it means that the models are learning strongly without apparent overfitting.

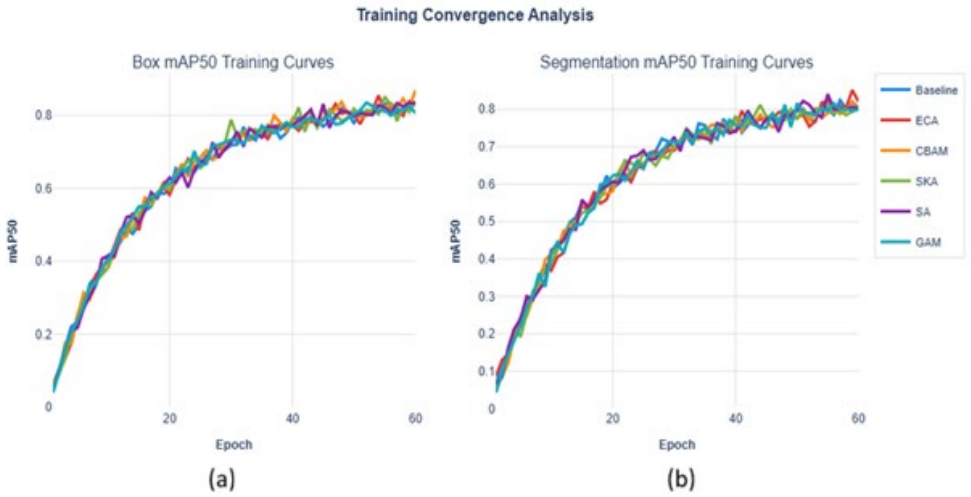


Figure 11: Training curves against epoch. (a) box mAP50 curves (b) segmentation mAP50 training curves.

5. Discussion

This combination of attention in YOLOv8 to construct crack segmentation has proven that it can increase the detection and segmentation performance, and it confirms the hypothesis that the strategic placement of attention will improve feature representation to construct damage assessment. The investigation offers empirical data on attention-based architectural adjustments in purveying health-inspecting applications in buildings, in addition to delivering some vital insights into performance-efficiency trade-offs and deployment realities in the case of building inspections.

A striking finding is that ECA achieved the highest precision improvements (3.5% for detection, 2.7% for segmentation) yet showed zero improvement in mAP metrics. This apparent paradox reveals how different evaluation metrics capture distinct performance aspects. ECA's substantial precision gains (0.846 to 0.876) indicate successful suppression of false crack detections through adaptive channel attention that emphasizes discriminative feature channels while suppressing noise-prone channels that respond to non-crack surface features like water stains, paint irregularities, or surface textures common on building walls. mAP evaluates performance across multiple IoU thresholds, incorporating both precision and recall through the precision-recall curve. A model can improve precision without improving mAP, if recall decreases proportionally or if improvements occur only at specific confidence thresholds that don't substantially shift the overall curve [47, 48]. ECA's precision gains came with stable recall (0.717 to 0.709), suggesting a precision-recall trade-off where reduced false positives were balanced by slightly fewer true detections.

CBAM obtained the best segmentation performance, making it the best to use in pixel -level building crack delineation. This advantage can be attributed to the fact that CBAM is a dual-pathway sequential architecture that combats what to attend (channel attention) and where to attend (spatial attention) [49]. The initial step in channel attention is recalibration between the features, which models inter-channel dependencies and prioritizes crack-discriminative channels, as well as avoiding background texture on concrete surfaces, painted walls or tile patterns. These features are further processed by spatial attention, which focuses on informative spatial areas to emphasize the areas that contain cracks whilst ignoring non-important background details.

The positive change in mAP at 0.5:0.95 by 1.1% shows higher performance at higher requirements of an accurate boundary prediction (Tighter thresholds). The boundary precision in CBAM is directly solved by attention mechanisms of the spatial attention that produces pixel-level attention weights, emphasizing the edges of cracks, resulting in a more accurate segmentation mask, which is important in crack width measuring and extent estimations, which are key parameters in building structural safety measurements and repair cost estimations. The middle-grade calculation cost is acceptable for an accuracy-critical building assessment. SKA got the best recall, which means it is better placed to detect different building crack instances with its multi-scale receptive field technique. Simultaneous fine-grained local patterns and larger contextual information are captured using parallel processing paths, each with different kernel sizes, and attention-weighted fusion varies receptive fields according to the characteristics of the crack [36]. This is the reason why SKA has an advantage over recalls, which can identify the scene that fixed-receptive-field methods fail. This can detect cracks that the fixed-receptive-field methods do not detect because of scale mismatch, which is actually useful when dealing with buildings where hairline surface cracks exist on facades, and larger structural damage occurs in load-bearing walls and columns. The recall enhancement is accompanied by a minor reduction in precision, which signifies a coverage-over-selectivity trade-off that would be useful in safety-critical building inspection, where structural defects would be missed, and this could endanger occupant safety.

The range of parameter efficiency of the tested attention mechanisms is 1,600-fold; GAM parameters are 66,580 compared to ECA 41 parameters. The outstanding efficiency of ECA is based on its parameter-free adaptive channel attention by the one-dimensional convolution with adaptive kernel blocks, without connected layers and dimensionality reduction [50]. The result shows that 41 parameters yield a 3.5 per cent increase in precision, suggesting that assumptions of parameter budgets needed to get mean-significant gains are invalid, with much deeper implications for the usage of portable building inspection cameras with limited memory and computing capabilities, including handheld inspection tablets or permanent monitoring cameras on buildings.

On the other hand, the number of 66,580 parameters generated by GAM did not bring any significant gains, which may indicate too large a model capacity to build the complexity of the crack detection task. This brings in a principle of sufficient capacity: attention mechanisms must have just enough representational power to represent task-relevant discriminations, and more power is of diminishing value. In the case of crack detection, channel-level discriminations (which convolutional filters are sensitive to crack patterns and not to wall textures) seem to have low capacity (channel-level discriminations of ECA prove successful), and GAM might have too many parameters to spare for a task-specific crack detection attention pattern, and instead offer global attention across all dimensions.

Analysis of inference speed demonstrates that all variants are viable at real time, and processing times of 3.1-4.2ms per image will translate to 238-323 frames per second, which is much more than the 30 fps needed to build inspection applications. ECA has a zero inference overhead due to simple one-dimensional convolution operations, whereas CBAM has 3.2% inference overhead due to efficient dual attention computation. Even the 35.5% overhead of GAM has real-time capability at 7.9x of 30 fps requirements, with 7.9x speed margin to preprocessing and post-processing operations common in building inspection processes.

The overhead in terms of one-time training costs is acceptable since the percentage of training time is up by 3.8-16.3. All attention-enhanced variants converged at similar convergence points within the assigned 60 epochs, which means that attention integration does not hamper the optimization. This confirms the residual integration approach, which simultaneously conserves the original feature pathways but enables attention mechanisms to offer additional refinements [51]. The amount of memory used by GPUs was not near the limits of consumer-grade GPUs (6.12-6.76 GB out of 15 GB) and therefore could be trained on consumer-grade GPUs typically found in research buildings and structural engineering departments in most cases without the need to have specialized high-performance computing support [52].

Attention mechanism characteristics have a wide range of values and thus are selected based on the needs of deployment, depending on the context [53, 9]. When deploying to mobile/edge, with resource constraints, ECA gives the best parameter efficiency as well as no inference overhead with a significant precision improvement. Its low computational footprint allows it to be deployed on embedded platforms, mobile robots, and drone platforms, with very little memory and processing bandwidth, and precision increases the rate of false alarms and the unnecessary site inspections.

CBAM achieves peak segmentation performance when the precision of analysis plays a major role in the performance of offline analysis, and the processing time is not crucial to the detection accuracy. The intermediate computational burden is acceptable in personal processing of the collected inspection images in batch processing. The ability of CBAM to provide high boundary accuracy makes it one of the most reliable in the measurement of crack extents to aid the judgment of the damage and repair strategies. In balanced general-purpose applications, SA provides an efficient tradeoff between good precision at low parameters, giving a middle-ground solution appropriate in general infrastructure monitoring, where neither extreme efficiency nor maximum accuracy is the main limiting factor.

The strength of this research is its ability to provide a systematic and comparative appraisal of the five attention mechanisms (CBAM, ECA, SKA, SA, GAM) that are incorporated into YOLOv8. Through evaluation of the various attention designs under an identical baseline architecture and quantification of the various aspects of performance, the study offers subtle insights into trade-offs, with CBAM being the most effective in terms of enhancing mask accuracy, whereas ECA is the least parameter-efficient. The generalizability of the study to the YOLOv8 setting and selecting modules according to the particular deployment limitations (accuracy vs. compute vs. recall) is improved using this comparative approach. Nonetheless, the research has its limitations, which define the future work opportunities. Others are small and some negative, which might be close to experimental variance and so need extensive validation; and improvements on building-crack data may not be extrapolated to other defect types or imaging conditions without cross-domain testing, but issues of inference latency and parameter numbers have been reported, details of long-term field trials are scarce and practical control issues (such as in adverse lighting, heavy occlusion or using an UAV) are important next steps.

6. Conclusions

This study demonstrates that integrating attention mechanisms into the YOLOv8 architecture meaningfully enhances the model's ability to detect and segment building cracks. By systematically evaluating several attention modules within the same framework, the work shows that attention can strengthen feature representation, mitigate information loss in deeper layers, and improve detection of small, irregular, and partially obscured defects—without undermining real-time deployment feasibility. The comparative results offer clear guidance for practitioners: different attention designs present distinct trade-offs, enabling informed selection based on priorities such as segmentation fidelity, detection completeness, or parameter efficiency.

While the findings validate the utility of attention modules for crack inspection, they also highlight the need for further validation and refinement. Broader testing across diverse datasets, detector backbones, and imaging conditions will help confirm robustness and generalizability. Additional ablation studies on module placement and hyperparameter tuning could refine integration strategies and uncover further performance gains. Practical deployment factors—such as memory and energy use on edge devices, training resource demands, and behavior under adverse environmental conditions—warrant deeper investigation, as do long-term field trials to assess operational reliability and maintenance impacts.

The research supports attention-enhanced YOLO models as a promising direction for automated structural-health monitoring. With continued optimization, cross-domain validation, and deployment-focused evaluation, these approaches have the potential to improve inspection accuracy, reduce manual effort, and contribute to more efficient and reliable infrastructure maintenance practices. The study advocates attention-enhanced YOLO models as a good future of automated structural-health monitoring. As further optimization, cross-domain validation, and deployment-oriented testing are introduced, these strategies can lead to an increase in accuracy of the inspections, less manual work, and better, more efficient, and effective infrastructure maintenance practices may be adopted.

Author contributions: Samuel Owoeye: Conceptualization, Investigation, Project administration. **Folasade Durodola:** Writing – original draft, Writing – review & editing. **Sikirulahi Abdulkareem:** Formal Analysis, Methodology, Validation, Software. **Olugbenga Omotainse:** Investigation, Editing, Resources.

Data availability: Data will be available upon reasonable request by the authors.

Conflicts of interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding: The authors did not receive support from any organization for the conduct of the study.

References

[1] M.Z. Khan, M. Shahzadi, A. Khan, U. Ali, M.A.S. Hassan, and M. Hussain, "Review on crack detection in civil infrastructure using structural health monitoring and machine learning techniques," *Innovative Infrastructure Solutions,* vol. 10, no. 8, pp. 348 , 2025, doi: 10.1007/s41062-025-02147-y.

[2] M. K. Askar, R.F. Al-Kamaki, and K.M. Hisham, "Cracks in concrete structures causes and treatments: A review," *Journal of Duhok University,* vol. 26, no. 2, pp. 148-165, 2023, [Online]: Available: https://journal.uod.ac/index.php/uodjournal/article/view/3294. [Accessed Sep, 2. 2025].

[3] X. Li, X. Langxing, W. Mengpu, Z. Lixiao, and Z. Chen, "An underwater crack detection method based on improved YOLOv8," *Ocean Engineering,* vol. no. 3, 313, pp. 119508, 2024, doi: 10.1016/j.oceaneng.2024.119508.

[4] H. S.Munawar, A. W. Hammad, A. Haddad, C. A. Soares, S. T. Waller, ' Image-based crack detection methods: A review," *Infrastructures*, vol. 6, no. 8, pp. 115, 2021, doi: 10.3390/infrastructures6080115.

[5] N. Meyendorf, "Early detection of materials degradation," *In AIP Conference Proceedings*, vol. 36, pp. 020002-1-020002-10, 2017, doi: 10.1063/1.4974543.

[6] T. Yamane, and P. Chun, "Crack detection from a concrete surface image based on semantic segmentation using deep learning." *Journal of Advanced Concrete Technology*, vol. 18, no. 9, pp. 493-504, 2020., doi: 10.3151/jact.18.493.

[7] J. Zhang, S. Qian, and C. Tan, "Automated bridge surface crack detection and segmentation using computer vision-based deep learning model," *Engineering Applications of Artificial Intelligence,* vol. 115, pp. 105225, 2022, doi: 10.1016/j.engappai.2022.105225.

[8] A. Mohan, and P. Sumathi, "Crack detection using image processing: A critical review and analysis." *Alexandria engineering journal,* vol. 57, no. 2, pp. 787-798, 2018, doi: 10.1016/j.aej.2017.01.020.

[9] R. Varghese, and M. Sambath, "YOLOv8: A novel object detection algorithm with enhanced performance and robustness." In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems* (ADICS), pp. 1-6, 2024, doi: 10.1109/ADICS58448.2024.10533619.

[10] Z.F. Elsharkawy, H. Kasban, and M.Y. Abbass, "Efficient surface crack segmentation for industrial and civil applications based on an enhanced YOLOv8 model." *Journal of Big Data,* vol. 12, no. 1, pp. 16, 2025, doi: 10.1186/s40537-025-01065-1.

[11] Z. Zhang, H. Zhang, and T. Zhang, "Enhanced YOLOv8-based pavement crack detection: A high-precision approach." *PLoS ONE,* vol. 20, no. 5, pp. e0324512, 2025, doi: 10.1371/journal.pone.0324512.

[12] D. Lin, X. Tian, J. Duan, D. Zhou, D. Zhao, D. Cao, "DA-RDD: toward domain adaptive road damage detection across different countries." *IEEE Transacations on Intelligent Transportation System,* vol. 24, no. 3, pp. 3091–3103, 2023, doi: 10.1109/TITS.2022.3221067.

[13] H. Liu, C. Jia, F. Shi, X. Cheng, M. Wang,and S. Chen, "Staircase cascaded fusion of lightweight local pattern recognition and long-range dependencies for structural crack segmentation." arXiv preprint arXiv:2408.12815, 2024, doi: 10.48550/arXiv.2408.12815.

[14] H. Kim, E. Ahn, M. Shin, and S. Sim, "Crack and non-crack classification from concrete surface images using machine learning." *Structural Health Monitoring*, vol. 18, no. 3, pp. 725- 738, 2019, doi: 10.1177/1475921718768747.

[15] M. Yaseen, "What is YOLOv8: An in-depth exploration of the internal features of the next-generation object fetector". *arXiv preprint arXiv*:2408.15857, 2024., doi: 10.48550/arXiv.2408.15857.

[16] H. Yao, et al., "A detection method for pavement cracks combining object detection and attention mechanism." *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 22179-22189, 2022, doi: 10.1109/TITS.2022.3177210.

[17] P. Gupta, and M. Dixit, "Image-based crack detection approaches: A comprehensive survey." *Multimedia Tools and Applications*, vol. 81, no. 28, pp. 40181-40229, 2022, doi: 10.1007/s11042-022-13152-z.

[18] G. Boesch, "YOLOv8: A Complete Guide." *Viso.ai Blog*. 2024. Accessed Sep., 22, pp. 24-25, 2025, https://viso.ai/deep-learning/yolov8-guide

[19] J. Zhang, Z. V. Beliaeva, and Y. Huang, "Accuracy–efficiency trade-off: Optimising YOLOv8 for structural crack detection," *Sensors*, vol. 25, no. 13, pp. 3873, 2025, doi: 10.3390/s25133873.

[20] F. Yu, G. Ye, Q. Jiang, K. Yuen, X. Chong, and Q. Jin, "Imaging-based instance segmentation of pavement cracks using an improved YOLOv8 network," *Structural Control and Health Monitoring*, vol. 1660649, pp.1-22, 2025, doi: 10.1155/stc/1660649.

[21] Z. Liua, H. Yao, X. Zhong, and Z. Deng, "A real-time pavement crack detection method based on an improved lightweight YOLOv8 model," *International Journal of Applied Mathematics in Control Engineering*, 7, pp. 171-176, 2024. [Online]: Available: http://www.ijamce.com/Papers/IJAMCE20241208.pdf. [Accessed Sep, 2. 2025].

[22] X. Dong, Y. Liu, and J. Dai, "Concrete surface crack detection algorithm based on improved YOLOv8," *Sensors*, vol. 24, no. 16, pp. 5252, 2024. doi: 10.3390/s24165252.

[23] T. Cao, W. Li, H. Sun, P. Wang, and Z. Gong, "YOLOv8-PCD: A pavement crack detection method based on enhanced feature fusion," *In SPIE Conference Proceedings*, Dalian, China, 2024, pp. 13421, doi: 10.1117/12.3054712.

[24] X. Cui, W. Qicai, D. Jinpeng, X. Yanjin, and D. Yun, "Intelligent crack detection based on attention mechanism in convolution neural network," *Advances in Structural Engineering*, vol. 24, no. 9, pp. 1859-1868, 2021, doi: 10.1177/1369433220986638.

[25] Z. Qu, C. Wen, W. Shi-Yan, Y. Tu-Ming, and L. Ling, "A crack detection algorithm for concrete pavement based on attention mechanism and multi-features fusion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11710–11719, 2022, doi: 10.1109/tits.2021.3106647.

[26] G. Xu, H. Xu, Z. Yuwei, and W. Chunyan "Dam crack image detection model on feature enhancement and attention mechanism," *Water*, vol. 15, no. 1, pp. 64, 2022, doi: 10.3390/w15010064.

[27] R. Junhua, Z. Guowu, M. Yadong, Z. De, L. Tao, and Y. Jun, "Automatic pavement crack detection fusing attention mechanism," *Electronics*, vol. 11, no. 21, pp. 3622, 2022, doi: 10.3390/electronics11213622.

[28] Y. Jiang, and Z. chunhui, "Attention classification-and-segmentation network for micro-crack anomaly detection of photovoltaic module cells," *Solar Energy*, vol. 238, pp. 291–304, 2022, doi: 10.1016/j.solener.2022.04.012.

[29] P. Jing, Y. Haiyang, H. Zhihua, X. Saifei, and S. Caoyuan, "Road crack detection using deep neural network based on attention mechanism and residual structure," *IEEE Access*, vol. 11, pp. 919–929, 2022, doi: 10.1109/ACCESS.2022.3233072.

[30] F. Guo, L. Jian, L. Chengshun, and Y. Huayang, "A novel transformer-based network with attention mechanism for automatic pavement crack detection," *Construction and Building Materials*, vol. 391, pp. 131852–131862. 2023, doi: 10.1016/j.conbuildmat.2023.131852.

[31] H. Liu, X. Miao,C. Mertz, C. Xu, and H. Kong, "CrackFormer: Transformer network for fine-grained crack detection," *In Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3763-3772, 2021, doi: 10.1109/ICCV48922.2021.00376.

[32] X. Jiang, L. Jiang,A, Wang, K. Zhu, and Y. Gao, "CrackSegDiff: Diffusion probability model-based multi-modal crack segmentation," *arXiv preprint arXiv*:2410.08,2024. doi: 10.48550/arXiv.2410.08100.

[33] F. Lin, J. Yang, J. Shu, and R. J. Scherer, "Crack semantic segmentation using the U-Net with full attention strategy," *arXiv preprint arXiv*:2104.14586, 2021. doi: 10.48550/arXiv.2104.14586.

[34] L. Yang, R. Zhang, L. Li, and X. Xie, X. "SimAM: A simple, parameter- free attention module for convolutional neural networks," *In Proceedings of the 38th International Conference on Machine Learning*, vol. 139, pp. 11863-11874, 2021. . [Online]: Available: https://proceedings.mlr.press/v139/yang21o/yang21o.pdf. [Accessed Sep, 2. 2025].

[35] T. Diwan, G. Anirudh, and J.V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 9243-9275, 2023, doi: 10.1007/s11042-022-13644-y.

[36] C.-Y. Wang, H. -Y. Mark Liao, Y. -H. Wu, P. -Y. Chen, J. -W. Hsieh and I. -H. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA, USA, 2020, pp. 1571-1580, doi: 10.1109/CVPRW50498.2020.00203.

[37] T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 936-944, doi: 10.1109/CVPR.2017.106.

[38] L.T. Ramos, and A. D. Sappa, "A decade of you only look once (YOLO) for object detection: A review," *IEEE Access*, vol. 13, pp. 192747-192794, 2025, doi: 10.1109/ACCESS.2025.3630988.

[39] X. Tong, L. Zhihong and L. Fangrong, "Succulent plant image classification based on lightweight GoogLeNet with CBAM attention mechanism," *Applied Sciences*, vol.15, no. 7, pp. 3730, 2025, doi: 10.3390/app15073730.

[40] L. Mengxuan, Z. Luo, and M. Jiang, "Intelligent modulation recognition of frequency-hopping communications: theory, methods, and challenges," *Big Data and Cognitive Computing*, vol. 9, no. 12, pp 318, 2025, doi: 10.3390/bdcc9120318.

[41] Y. Chen, Y. Tao, D. Shuai, W. Like, P. Bida, and W. Yunlong, "Enhancing crack segmentation network with multiple selective fusion mechanisms," *Buildings*, vol. 15, no. 7, pp. 1088, 2025, doi: 10.3390/buildings15071088.

[42] Z. Cai, D. Yuming, Z. Jianwei, and F. Yuan, "SA-ResNet: An intrusion detection method based on spatial attention mechanism and residual neural network fusion," *Computers, Materials and Continua*, vol. 83, no. 2, 2025, doi: 10.32604/cmc.2025.061206.

[43] A.L.C. Ottoni, A.M Souza, and M.S. Novo, "Automated hyperparameter tuning for crack image classification with deep learning," *Soft Computing*, vol. 27, no 23, pp. 18383–18402, 2023, doi: 10.1007/s00500-023-09103-x.

[44] F. Gerz, M. G. Schneider, and M. Jelali, "Integration of vision transformer networks in yolov8 for object detection: comparative study on plant disease detection," *IEEE Access*, vol. 14, pp. 27303-27338, 2026, doi: 10.1109/ACCESS.2026.3665969.

[45] Q. Zaheer, Q. Shi, S. M. A. Hassan Shah, C. Ai, and J. Wang, "Intelligent multitasking framework for boundary-preserving semantic segmentation, width estimation, and propagation modeling of concrete cracks," *Journal of Infrastructure Systems*, vol. 31, no. 3, pp. 04025009, 2025, doi: 10.1061/JITSE4.ISENG-2574.

[46] D. Nguyen, V. -D. Hoang and V. -T. -L. Le, "A lightweight multi-scale attention model for small object detection in UAV imagery," *IEEE Access*, vol. 14, pp. 12579-12593, 2026, doi: 10.1109/ACCESS.2026.3656179.

[47] C. Liu, X. Zeng, R. Lin, X. Liang, Z. Freyberg, E. Xing, and M. Xu, "Deep learning-based supervised semantic segmentation of electron cryo-subtomograms," *In 2018, the 25th IEEE International Conference on Image Processing*, pp. 1578-1582, 2018, doi: 10.1109/ICIP.2018.8451386.

[48] O. E. Olorunshola, E.I. Martins, and E. E. Abraham, "A comparative study of YOLOv5 and YOLOv7 object detection algorithms," *Journal of Computing and Social Informatics,* vol. 2, no. 1, pp. 1-12, 2023, doi: 10.33736/jcsi.5070.2023.

[49] L. Fadia, S. Vatsal, H. Mohammad, W. Jonathan, and A. Majid, "A novel multi-modal dual pathway network with hierarchical channel-spatial attention and adaptive feature fusion for viral genomic variant classification," *Network Modelling Analysis in Health Informatics and Bioinformatics,* vol. 14, no. 1, pp. 75, 2025, doi: 10.1007/s13721-025-00576-4.

[50] A. Haboub, B. Hamza, and B. Abdesselam, "DCT-based channel attention for multivariate time series classification," *IEEE Open Journal of the Computer Society*, vol. 6, pp. 1110-1120, 2025, doi: 10.1109/OJCS.2025.3586682.

[51] S. Li and T. Ma, "Pathway representation learning for interpretable graph neural networks," *IEEE Access*, vol. 14, pp. 41979-41997, 2026, doi: 10.1109/ACCESS.2026.3673117.

[52] Y. Song, Mi, H. Xie, and H. Chen, "November. Powerinfer: Fast large language model serving with a consumer-grade GPU," *In Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, pp. 590-606, 2024, doi: 10.1145/3694715.3695964

[53] D. Soydaner, "Attention mechanism in neural networks: where it comes and where it goes." *Neural Computing and Applications*," vol. 34, no. 16, pp. 13371-13385, 2022, doi: 10.1007/s00521-022-07366-3.