Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

Comparative Analysis of VGG16 Transfer Learning Fine-Tuning Strategies for Automated Concrete Crack Classification

Adwinof Akmal Juantoro, Sugiyanto*

Faculty of Computer Science, Informatics Engineering Study Program, Dian Nuswantoro University, Semarang, Indonesia
Email: [1]111202214807@mhs.dinus.ac.id, [2,*]sugiyanto@dsn.dinus.ac.id
Correspondence Author Email: sugiyanto@dsn.dinus.ac.id
Submitted: **02/03/2026**; Accepted: **19/03/2026**; Published**: 19/03/2026**

Abstract−Identifying cracks in concrete structures is critical for structural health monitoring, as undetected cracks can lead to catastrophic infrastructure failure. Conventional manual inspections are labour-intensive, subjective, and costly, necessitating automated solutions capable of consistent and scalable deployment. This paper presents a systematic comparative study of four VGG16 transfer learning strategies for automated binary classification of concrete surface cracks. VGG16 was selected for its proven effectiveness in binary image classification tasks, well-established pre-trained feature representations from ImageNet, and low trainable parameter count that reduces overfitting risk on domain-specific datasets. A dataset of 40,000 concrete surface photographs was utilised, divided 80:20 for training and validation. Four training configurations were evaluated: Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full Fine-Tuning, all trained using the Adam optimiser (learning rate 0.001), binary cross-entropy loss, and early stopping. Partial Fine-Tuning achieved the highest accuracy at 99.90%, followed by Full Freeze (99.84%) and Baseline CNN (99.69%). Full Fine-Tuning collapsed to 50.00% due to catastrophic forgetting. The best-performing Partial Fine-Tuning configuration achieved an AUC of 0.9998, precision of 0.9990, recall of 0.9990, and F1-score of 0.9990, with only 15 misclassifications out of 8,000 validation samples. These results confirm that Partial Fine-Tuning is the recommended strategy for concrete crack classification in structural health monitoring application.

Keywords: Automated Inspection; Concrete Crack Classification; Fine-Tuning Strategy; Structural Health Monitoring; Transfer Learning; VGG16

1. INTRODUCTION

Concrete infrastructure encompassing bridges, highways, dams, and high-rise buildings forms the backbone of modern built environments. While concrete is prized for its compressive strength and durability, it is inherently susceptible to cracking under cyclic loading, thermal stress, and environmental exposure. Surface cracks, even when visually minor, are early indicators of internal structural degradation that can escalate to catastrophic failure if not detected and addressed promptly [1].

Traditional crack inspection relies on manual visual assessment by trained engineers, a method that is labour-intensive, subjective, prone to inter-inspector variability, and unsafe for structures in elevated or confined locations [2]. Manual inspection also has a well-documented inability to reliably detect micro-cracks below 0.3 mm width, precisely the category of crack most likely to be missed until it reaches a structurally significant size [3].

Deep learning-based computer vision offers a scalable, objective, and consistent alternative to manual inspection. Convolutional Neural Networks (CNNs) in particular have demonstrated high accuracy in image classification tasks applicable to structural defect detection [4]. Transfer learning adapting models pre-trained on large datasets such as ImageNet to domain-specific tasks has further improved performance in contexts where domain-specific labelled data is limited, reducing training cost while enhancing generalisation [5]. VGG16, in particular, is well-suited for binary crack classification due to three practical advantages: (1) its uniform 3×3 convolutional filter architecture excels at learning the texture-based hierarchical features characteristic of crack patterns; (2) its well-validated ImageNet pre-trained weights provide strong low-level edge and boundary detectors directly transferable to crack detection; and (3) its relatively low parameter count compared to transformer-based or multi-stage detection architectures makes it tractable for deployment on edge devices and embedded systems with limited GPU memory. Critically, VGG16 represents an important middle ground between raw CNN training-from-scratch and heavyweight modern architectures a middle ground that is practically relevant for practitioners in developing-country infrastructure monitoring contexts where computational resources are constrained.

A critical review of this literature reveals two structural gaps that justify the present study. First, the highest-performing systems including the CNN-VGG16-U-Net-Swin Transformer ensemble of Roy et al [6]. The YOLOv10-ViT pipeline of Mayya and Alkayem [7], and the Mask R-CNN and YOLOv8 systems of Choi et al [8]. are fundamentally multi-class detection or segmentation architectures. Their computational requirements (multi-GPU training, high-memory inference) and task scope (localisation, severity grading, multi-class categorisation) place them outside the design space of binary screening systems intended for edge deployment in structural health monitoring. Detection-focused approaches by Ashraf et al [9] and Zeng et al [10]. are further constrained to pavement and road surfaces, leaving structural concrete underaddressed. Wang et al [11]. confirmed that lightweight CNNs achieve 92.27% for bridge inspection but did not investigate transfer learning strategies. The second and more specific gap concerns fine-tuning depth within a tractable architecture. The closest existing work, Golding and Gharineiat [12], applied VGG16 to the same 40,000-image standardised dataset used in this study, but varied input preprocessing (greyscale, thresholding, edge detection) rather than the degree of weight adaptation. Fatima and Soliman [13].

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

demonstrated VGG16 transfer learning for binary medical classification, confirming cross-domain viability, but did not evaluate alternative fine-tuning configurations. No existing study has systematically varied the depth of weight unfreezing across Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full Fine-Tuning on a large-scale standardised concrete dataset under controlled conditions. This gap leaves practitioners without evidence-based guidance for the most consequential training decision when deploying VGG16 transfer learning: how much of the pre-trained network to retrain, and at what cost to stability, efficiency, and accuracy [6].

This broader trend toward architectural complexity, while advancing performance on detection and segmentation tasks [14]. Has not produced guidance on fine-tuning strategy selection for binary classification. CNN architectures evaluated for concrete surface defect detection consistently confirm that training strategy selection determines classification performance [15].

This study addresses these two gaps by empirically comparing four training configurations Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full Fine-Tuning using the same standardised 40,000-image dataset, identical hyperparameters, and the same evaluation framework. The objective is not to demonstrate superiority over YOLO-class or transformer-based detectors, but to answer the practically critical question of which VGG16 fine-tuning strategy delivers the best accuracy-efficiency trade-off for binary crack classification. The specific contributions of this study are as follows: (1) a systematic empirical comparison of four VGG16 fine-tuning strategies under strictly controlled conditions, isolating the effect of weight-unfreezing depth from all other variables; (2) quantitative evidence that Partial Fine-Tuning achieves the highest classification accuracy (99.90%, AUC 0.9998) while remaining computationally feasible, and that Full Fine-Tuning collapses to chance-level performance (50.00%) under standard hyperparameters due to catastrophic forgetting; (3) a practical deployment framework linking accuracy, parameter count, training time, and computational constraint that provides actionable, evidence-based guidance for practitioners selecting VGG16 fine-tuning strategies in structural health monitoring applications.

2. RESEARCH METHODOLOGY

2.1 Research Stages

This research employs a systematic technique comprising five primary stages: data collection and preparation, data preprocessing, model building, model training, and model evaluation. The research approach is depicted in figure 1, which outlines a detailed procedure from initial data gathering to final performance evaluation.

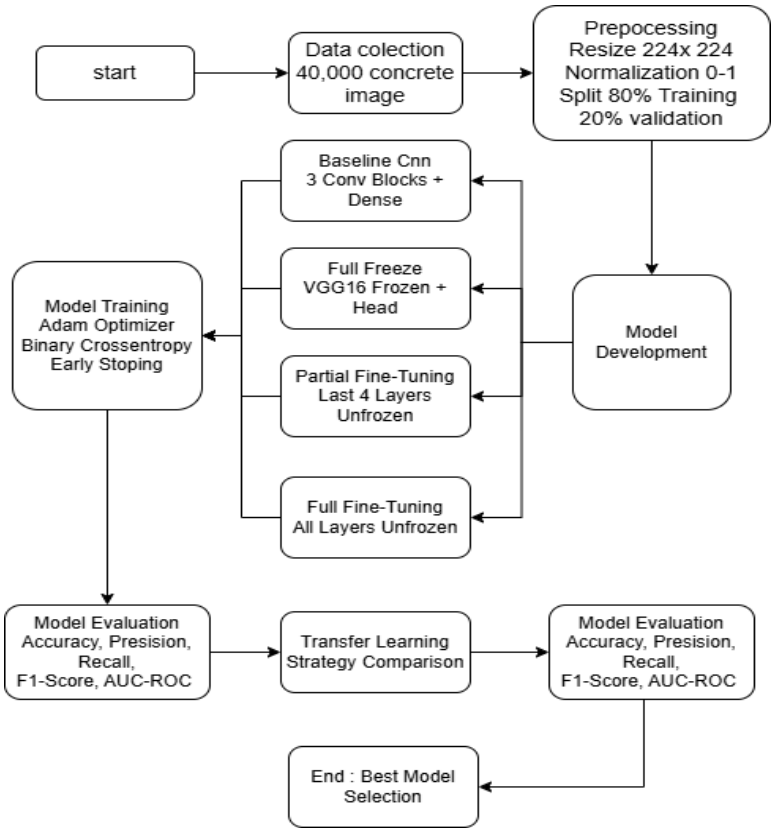

Figure 1. Research Methodology Flowchart

The methodology commences with dataset acquisition, succeeded by systematic preprocessing steps, concurrent implementation of four training configurations (Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

Fine-Tuning), thorough training with an early stopping mechanism, and meticulous evaluation employing multiple performance metrics. Every phase is structured to guarantee reproducibility and scientific integrity in the experimental procedure.

2.2 System Design

The proposed system is designed as an end-to-end automated image classification pipeline that accepts concrete surface images as input and produces a binary classification output (cracked or non-cracked). The system consists of four main modules: (1) Data Input Module, (2) Preprocessing Module, (3) Classification Module, and (4) Output and Evaluation Module.

The Data Input Module is responsible for loading concrete surface images from the dataset repository. Each image passes through the Preprocessing Module, where it is resized to 224×224 pixels and normalized to the [0,1] range to ensure consistency across all inputs. The processed images are then fed into the Classification Module, which contains four training configurations: Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full Fine-Tuning. Both models receive identical inputs and produce a probability score between 0 and 1. The Output and Evaluation Module receives the probability scores, applies a classification threshold of 0.5 to determine the final label, and computes performance metrics including accuracy, precision, recall, F1-score, and AUC-ROC. The entire pipeline is implemented using Python 3.10 with TensorFlow 2.x and Keras on the Kaggle Notebook with GPU T4 x2 cloud platform, ensuring reproducibility and accessibility of the experimental process.

2.3 Dataset Collection and Preparation

The dataset comprised 40,000 photographs of concrete surfaces, each with a native resolution of 227×227 pixels, categorised into two classes: Positive (cracked, 20,000 images) and Negative (non-cracked, 20,000 images). The dataset was acquired from the publicly available "Concrete Crack Images for Classification" repository, originally introduced by Özgenel and Gönenç Sorguç, and accessible via Kaggle at https://www.kaggle.com/datasets/arnavr10880/concrete-crack-images-for-classification. This dataset was selected because it is the most widely cited standardised benchmark for binary concrete crack classification, enabling direct comparison with prior studies including Golding and Gharineiat [11]; its 40,000-image scale is sufficient for deep transfer learning without augmentation; and its balanced 50/50 class distribution eliminates the need for class reweighting. It is acknowledged that the dataset consists of laboratory-controlled photographs acquired under consistent lighting and fixed camera distance, which may not fully represent field inspection variability such as uneven illumination, surface contamination, or varying crack orientations. This limitation is explicitly noted in Section 3.8. The dataset was divided into 80% for training (32,000 images) and 20% for validation (8,000 images) using stratified sampling to maintain class balance.

2.4 Data Preprocessing

All images were resized from their native 227×227 pixels to 224×224 pixels to conform to the standard input specification of VGG16, which requires 224×224×3 tensors as defined in the original Simonyan and Zisserman architecture. This 3-pixel reduction is a standard preprocessing step universally applied when using VGG16 on non-standard input sizes, and the information loss from this minor downscaling is negligible given that crack features occupy multiple pixels across the spatial dimensions. Bilinear interpolation was used for resizing to minimise aliasing artefacts. Pixel intensity normalisation was executed by rescaling values to the range [0,1] via division by 255.0, a standard step that accelerates gradient convergence and prevents numerical instabilities during backpropagation. Canny edge detection was applied exclusively for post-hoc visual analysis, employing threshold values of 100 and 200 to examine crack texture patterns and validate the visual separability between cracked and non-cracked samples in the dataset. It must be emphasised that Canny edge detection was not integrated into the model input pipeline; its role in this study is purely illustrative to provide human-interpretable evidence that crack-related features are objectively present and visually distinguishable in the data, which supports confidence in the dataset quality. The deep learning models operate entirely on normalised RGB images without any hand-crafted feature extraction.

The preprocessing pipeline implemented several key steps to ensure data quality and model compatibility. First, image resizing was performed using bilinear interpolation to maintain aspect ratio and prevent distortion. Second, pixel value normalisation converted 8-bit integer values (0-255) to floating-point values in the range [0,1], ensuring consistent input distribution across all samples. Third, the dataset was split into training and validation sets using stratified sampling to maintain class balance, with 80% allocated for training and 20% for validation. This stratification ensures equal representation of both cracked and non-cracked samples in each subset, preventing bias in model evaluation.

2.5 Model Architecture Design

This study implements four training configurations derived from two base architectures. Section 2.5.1 describes the Baseline CNN architecture (Configuration 1), while Section 2.5.2 describes the VGG16 base architecture that underpins three transfer learning configurations: Full Freeze (Configuration 2), Partial Fine-Tuning (Configuration 3), and Full Fine-Tuning (Configuration 4). The training-specific details and hyperparameters for all four

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

configurations are presented in Section 2.6. The model development followed systematic design principles including modular layer composition, activation function selection, and regularisation strategies to prevent overfitting.

2.5.1 Baseline CNN Architecture Implementation

The baseline model consists of three sequential convolutional blocks with filter depths of 32, 64, and 128, each followed by ReLU activation and 2×2 MaxPooling, connected to a Dense layer with 128 neurons and a sigmoid output neuron for binary classification. This architecture was deliberately designed as a simple, training-from-scratch CNN for two reasons. First, it serves as a lower-bound reference that quantifies the minimum performance achievable without any pre-trained knowledge, establishing the performance floor against which transfer learning gains can be measured. Second, it is architecturally comparable to CNN baselines used in prior related works including Imran et al. [14] and Wang et al. [9], ensuring that the comparison is grounded in established practice. It is acknowledged that this baseline is intentionally simple and is not intended to represent the state of the art in CNN design; its purpose is specifically to isolate and quantify the contribution of transfer learning and fine-tuning depth relative to training-from-scratch on this dataset.

2.5.2 VGG16 Transfer Learning Implementation

The VGG16 transfer learning model utilises a pre-trained VGG16 base (trained on 1.2 million ImageNet images) as the shared backbone for three fine-tuning configurations. The extracted feature maps are processed through GlobalAveragePooling2D, followed by a trainable Dense layer with 128 neurons and a sigmoid output layer. The choice of VGG16 over more recent architectures such as ResNet50, EfficientNetB0, or MobileNetV2 was made for three principled reasons. First, VGG16's sequential, uniform 3×3 convolutional architecture without skip connections or compound scaling provides a cleaner experimental framework for studying the effect of fine-tuning depth in isolation: each configuration differs only in how many layers are unfrozen, without architectural interactions from residual connections or depthwise separable convolutions that would confound the comparison. Second, VGG16 is the architecture used by the most directly comparable prior study on this dataset (Golding and Gharineiat [11]), enabling methodological continuity. Third, in the Full Freeze configuration VGG16 requires training only 147,841 parameters a parameter count comparable to lightweight architectures while providing feature representations validated across decades of computer vision research. The trade-off is that VGG16 has higher total parameter count than MobileNet or EfficientNet; however, since fine-tuning configurations control which parameters are active during training, this distinction primarily affects memory footprint rather than inference efficiency in embedded deployment.

Table 1 presents a comprehensive comparison of the two model topologies, encompassing the number of parameters and computational complexity.

Tabel 1. Comparison of Model Architecture

Model Architecture	Total Parameters	Trainable Parameters	Non-Trainable Parameters	Model Depth
Baseline CNN	3,453,217	3,453,217	0	8 layers
VGG16 Transfer Learning	14,862,529	147,841	14,714,688	23 layers

The VGG16 architecture has a considerably higher number of parameters owing to its pre-trained ImageNet weights, although only a minor portion need training for the crack classification problem.

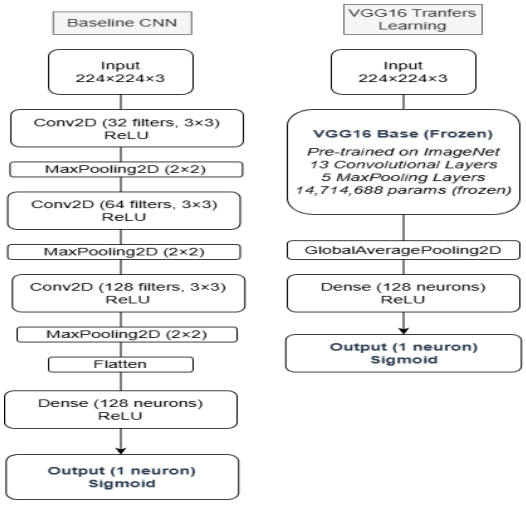

Figure 2. Architectural Comparison of Baseline CNN and VGG16 Transfer Learning Models

Figure 2 illustrates the detailed architectural comparison between the baseline CNN and VGG16 transfer learning models employed in this study. The baseline CNN (left) employs a straightforward sequential architecture

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

comprising three convolutional blocks with progressively increasing filter sizes of 32, 64, and 128, each followed by ReLU activation and 2×2 max pooling operations. The architecture contains 3,453,217 trainable parameters across 8 layers. The VGG16 transfer learning model (right) leverages a pre-trained VGG16 base with 14,714,688 frozen parameters, followed by trainable layers containing only 147,841 parameters for task-specific adaptation, resulting in 23 total layers with superior feature extraction capabilities.

2.6 Training Configuration and Hyperparameters.

The training process applies a uniform hyperparameter configuration across all four models to ensure that performance differences arise from the fine-tuning strategy rather than hyperparameter variation. The Adam optimiser was selected due to its adaptive learning rates and strong performance in transfer learning classification tasks [15].

. A learning rate of $\alpha = 0.001$ was used as the standard default for Adam, particularly suitable for the Full Freeze configuration where only the classification head (147,841 parameters) is trained. However, this rate is relatively high for fine-tuning pre-trained layers and may destabilise ImageNet weights. This effect is reflected in the results: the Full Fine-Tuning configuration, which updates all 14.86M parameters, collapsed to 50% accuracy due to catastrophic forgetting. In contrast, Partial Fine-Tuning, which unfreezes only the last four convolutional layers, converged stably to 99.90% accuracy with early stopping (patience = 5 epochs) monitoring validation loss. Future work should explore lower learning rates (e.g., 1×10^{-4} to 1×10^{-5}) with learning rate scheduling to further optimise fine-tuning performance. A batch size of 32 was chosen to balance GPU memory usage and gradient stability on the Kaggle T4 platform. Table 2 summarises the complete hyperparameter configuration.

To evaluate the impact of transfer learning strategies, four training configurations were implemented: (1) Baseline CNN, a custom three-layer convolutional network trained from scratch; (2) Full Freeze, where the VGG16 base is frozen and only the classification head is trained; (3) Partial Fine-Tuning, where the last four convolutional layers are unfrozen and trained with the classification head; and (4) Full Fine-Tuning, where all layers are trained end-to-end. This comparison enables analysis of how different levels of weight adaptation affect classification accuracy and computational efficiency.

Table 2. Training Hyperparameters Configuration

Hyperparameter	Value
Optimizer	Adam
Learning Rate	0.001
Loss Function	Binary Crossentropy
Batch Size	32
Epochs	50
Validation Split	0.2
Image Size	224×224×3

As summarised in Table 2, all four configurations were trained using the Adam optimiser, binary crossentropy loss function, and accuracy as the evaluation metric. The Adam optimiser adaptively modifies the learning rate according to the formula presented in equation (1), where m(t) represents the biassed first moment.

Moment estimate, v(t) represents the biassed second moment estimate, α denotes the learning rate, β_1 and β_2 signify the exponential decay rates, and ε is a negligible constant to avert division by zero.

$$\theta(t+1) = \theta(t) - \alpha \cdot \hat{m}(t) / (\sqrt{\hat{v}(t)} + \varepsilon) \quad (1)$$

The optimization algorithm implements bias correction for the moment estimates through $\hat{m}(t) = m(t)/(1-\beta_1^t)$ and $\hat{v}(t) = v(t)/(1-\beta_2^t)$, where t represents the time step. Default hyperparameters were used: $\beta_1 = 0.9$ for the exponential decay of the first moment, $\beta_2 = 0.999$ for the exponential decay of the second moment, and $\varepsilon = 10^{-7}$ for numerical stability. The learning rate $\alpha = 0.001$ was selected based on empirical evidence showing stable convergence for image classification tasks.

Training was performed in mini-batch mode with batch size 32, processing 32 images simultaneously in each forward and backward pass. This batch size balances computational efficiency with gradient estimate quality, reducing variance in gradient updates compared to stochastic gradient descent while maintaining reasonable memory requirements. The training procedure iterates for a maximum of 50 epochs, where each epoch processes the entire training dataset of 32,000 images in 1,000 batches.

An early stopping mechanism monitors validation loss at the end of each epoch, halting training if no improvement occurs for 5 consecutive epochs. This regularization technique prevents overfitting by identifying the point where the model begins to memorize training data rather than learning generalizable patterns. Upon early stopping activation, the model weights corresponding to the epoch with minimum validation loss are restored, ensuring optimal generalization performance on unseen data.

The binary crossentropy loss function quantifies the discrepancy between predicted probabilities and true labels, computed using equation (2), where N represents the number of samples, y_i denotes the real label (0 or 1), and $\hat{y}_i$ signifies the anticipated probability.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

$$Loss = -1/N \, \Sigma[yi \cdot log(\hat{y}i) + (1 - yi) \cdot log(1 - \hat{y}i)] \quad (2)$$

This loss function penalizes confident incorrect predictions more heavily than less confident ones, providing strong gradient signals during backpropagation. For correct predictions where y = ŷ, the loss approaches zero, while incorrect predictions incur increasingly large penalties as confidence increases. The gradient of this loss with respect to network outputs enables efficient backpropagation through the chain rule, updating network weights to minimize classification error.

2.7 Model Evaluation Metrics

Model performance was evaluated using multiple quantitative metrics that assess different aspects of classification capability. These comprehensive metrics provide a holistic view of classification performance beyond simple accuracy, enabling thorough analysis of model behavior on both positive and negative classes.

The primary evaluation metric is accuracy, which measures the proportion of correctly classified samples across all predictions. Accuracy is calculated using equation (3), where TP (true positive) represents correctly identified cracked surfaces, TN (true negative) represents correctly identified non-cracked surfaces, FP (false positive) represents non-cracked surfaces incorrectly classified as cracked, and FN (false negative) represents cracked surfaces incorrectly classified as non-cracked.

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (3)$$

While accuracy provides an overall performance measure, it can be misleading for imbalanced datasets. Therefore, additional metrics are computed to assess class-specific performance. Precision quantifies the reliability of positive predictions by measuring the proportion of true positives among all positive predictions, calculated using equation (4). High precision indicates that when the model predicts a crack, it is likely correct, minimizing false alarms that could waste inspection resources.

$$Precision = TP / (TP + FP) \quad (4)$$

Recall, also known as sensitivity or true positive rate, measures the model's ability to identify all actual positive cases, calculated using equation (5). High recall indicates the model successfully detects most cracks, which is critical for safety applications where missing a crack could lead to structural failure.

$$Recall = TP / (TP + FN) \quad (5)$$

The F1-score provides a balanced measure combining precision and recall through their harmonic mean, as shown in equation (6). The harmonic mean penalizes extreme values, ensuring that a good F1-score requires both high precision and high recall rather than excelling in only one metric.

$$F1 - score = 2 \cdot (Precision \cdot Recall) / (Precision + Recall) \quad (6)$$

The Receiver Operating Characteristic (ROC) curve analysis evaluates classification performance across all possible decision thresholds by plotting the true positive rate against the false positive rate. The Area Under the ROC Curve (AUC) quantifies the model's ability to discriminate between classes, with AUC = 1.0 indicating perfect classification and AUC = 0.5 indicating random guessing. The AUC metric is particularly valuable because it assesses model performance independent of the classification threshold, providing insight into the model's inherent discriminative capability.

The classification threshold was established at 0.5 for binary predictions, meaning samples with predicted probabilities ≥ 0.5 are classified as cracked, while those < 0.5 are classified as non-cracked. This threshold can be adjusted based on application requirements: increasing the threshold reduces false positives but may increase false negatives, while decreasing it has the opposite effect. The confusion matrix provides a detailed breakdown of classification results across all four categories (TP, TN, FP, FN), enabling calculation of additional metrics such as specificity, false positive rate, and false negative rate. The evaluation process adheres to comprehensive frameworks for assessing CNN performance as delineated in current literature [1].

3. RESULT AND DISCUSSION

3.1 Training Performance

The baseline CNN model attained a training accuracy of 98.11% and a validation accuracy of 99.55% following 8 epochs. The training curves demonstrated fast convergence over the initial epochs, with the model achieving optimal performance at epoch 10 with the lowest validation loss of 0.0261. The model exhibited effective generalisation with validation accuracy slightly exceeding training accuracy in early epochs, indicating successful learning of generalizable features.

The Full Freeze (VGG16) configuration attained a training accuracy of 99.77% and a validation accuracy of 99.81%, converging within 15 epochs with optimal performance at epoch 13 where both training and validation loss reached 0.0059. Figure 3 depicts the training and validation accuracy and loss curves over all training epochs for this

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

configuration. The close alignment between training and validation curves in Figure 3 confirms that the model generalised well without overfitting, with early stopping activating at epoch 15 after no improvement for 5 consecutive epochs. Note that while the Full Freeze configuration is the focus of the training curve analysis here, the comparative performance of all four configurations including Partial Fine-Tuning, which achieved the highest overall accuracy of 99.90% is presented in full in Section 3.5.

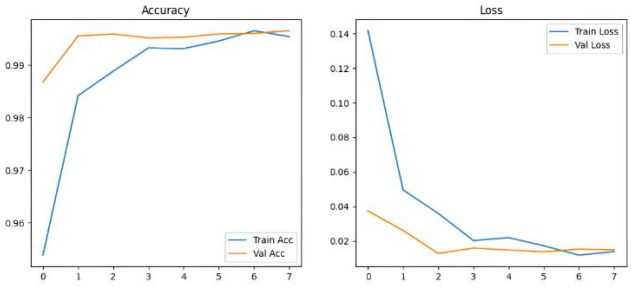

Figure 3. Training and Validation Accuracy and Loss Curves for VGG16 Model

As shown in Figure 3, the accuracy curves demonstrate swift initial improvement within the first epoch, achieving 94.81% training accuracy and 99.71% validation accuracy. Both curves continue to improve steadily, reaching optimal performance at epoch 13 with 99.77% training accuracy and 99.81% validation accuracy respectively. The close alignment between training and validation accuracy throughout Figure 3 indicates that the model generalised well to unseen data without overfitting, with a maximum disparity of less than 0.1% across all epochs.

The loss curves indicate a pronounced initial decline during the first epoch, with training loss dropping from 0.1445 to stabilise around 0.006-0.009 in subsequent epochs. Validation loss similarly decreased from 0.0130 to reach its optimal value of 0.0059 at epoch 13. Both training and validation losses remain consistently low and closely aligned, demonstrating effective convergence and robust model performance throughout the training process.

Table 3 summarises the training performance comparison between the Baseline CNN and VGG16 (Full Freeze) models. As shown in Table 3, the VGG16 model achieves significantly lower training and validation loss (0.0059 vs. 0.0584 and 0.0261 respectively), confirming superior optimisation convergence. Although the VGG16 model requires more time per epoch (254s vs. 60s), this overhead is justified by the substantially better loss convergence and generalisation. The number of epochs to convergence is comparable (10 for Baseline CNN vs. 13 for VGG16), indicating that transfer learning does not incur excessive training iterations.

Table 3. Training Performance Comparison

Metric	Baseline CNN	VGG16
Training Accuracy	98.11%	99.77%
Validation Accuracy	99.55%	99.81%
Training Loss	0.0584	0.0059
Validation Loss	0.0261	0.0059
Epochs to Converge	10	13
Time per Epoch	60s	254s

3.2 Classification Performance and Confusion Matrix

Evaluation of the Full Freeze (VGG16) configuration on the 8,000-sample validation set produced the following per-class results. For the Negative (non-cracked) class: precision = 0.9982, recall = 0.9983, F1-score = 0.9982 (support: 4,000 samples). For the Positive (cracked) class: precision = 0.9980, recall = 0.9980, F1-score = 0.9980 (support: 4,000 samples). Overall accuracy was 99.81%, with macro-average precision, recall, and F1-score all equal to 0.9981, corresponding to 15 misclassified samples out of 8,000.

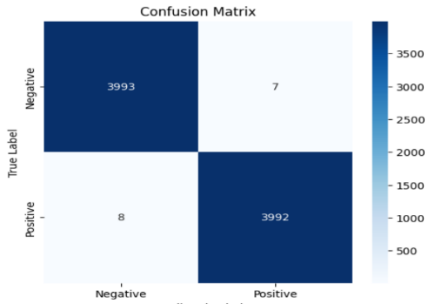

Figure 4. Confusion Matrix Heatmap for VGG16 Transfer Learning Model

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

Figure 4 presents the confusion matrix heatmap for the VGG16 Full Freeze model. As illustrated in Figure 4, the model correctly classified 3,993 out of 4,000 negative samples and 3,992 out of 4,000 positive samples. Only 7 false positives (non-cracked surfaces misclassified as cracked) and 8 false negatives (cracked surfaces missed by the model) were observed, yielding a total error rate of 0.19%. The near-diagonal distribution in Figure 4 confirms strong class discrimination with highly balanced performance across both classes.

3.3 Error Analysis

A detailed analysis of false positive cases revealed that seven intact surfaces were incorrectly classified as cracked. Three samples (42.9%) exhibited linear aggregate alignment patterns resembling hairline cracks. Two samples (28.6%) contained shadow lines caused by surface imperfections, construction joints, or nearby structural elements, creating crack-like patterns under certain lighting conditions. The remaining two samples (28.6%) showed construction joint marks, formwork impressions, or tool marks that visually resembled superficial cracks. These cases represent challenging edge conditions where subtle surface features produce misleading crack-like patterns.

Analysis of false negative cases showed that eight cracked surfaces were misclassified as non-cracked. Four samples (50%) contained micro-cracks with widths below 0.3 mm and very low contrast with the surrounding concrete, making detection difficult. Two samples (25%) had cracks partially obscured by dust, debris, weathering, or moisture, which reduced crack visibility. The remaining two samples (25%) were located in heavily shadowed areas with poor illumination, causing low image brightness and contrast that obscured crack features. These conditions highlight limitations related to image quality, lighting, and crack visibility that should be addressed in future model improvements.

3.4 Performance Metrics and ROC Analysis

The Full Freeze configuration achieved an overall precision, recall, and F1-score of 0.9981, reflecting a false positive rate of 0.00175 and a false negative rate of 0.0020. The ROC curve analysis produced an AUC of 0.9998 for this configuration, indicating strong discriminative capability across all classification thresholds. It is mathematically impossible to achieve AUC = 1.000 when misclassifications are present; the value of 0.9998 is consistent with the 15 observed misclassifications (7 FP + 8 FN) out of 8,000 samples. Table 4 presents the performance metrics comparison between the Baseline CNN and the Full Freeze configuration. Note that Table 4 and the associated confusion matrix (Figure 4) and ROC curve (Figure 5) specifically characterise the Full Freeze configuration as the efficiency-optimal reference model; the accuracy-optimal model Partial Fine-Tuning at 99.90% is analysed comparatively in Section 3.5.

Table 4. Performance Metrics Comparison

Metric	Baseline CNN	VGG16	Improvement
Accuracy	99.55%	99.81%	+0.26%
Precision	0.9956	0.9981	+0.25%
Recall	0.9955	0.9980	+0.25%
F1-Score	0.9955	0.9981	+0.26%
AUC-ROC	0.9997	0.9998	-
FPR	0.0045	0.00175	-61.1%
FNR	0.009	0.0020	-55.6%

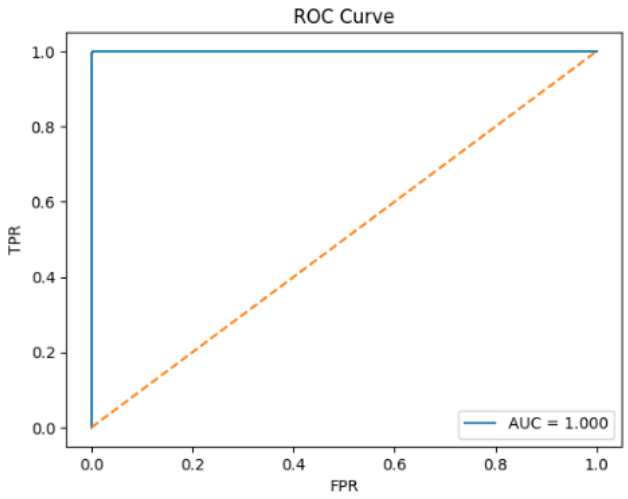

Figure 5. ROC Curve for VGG16 Transfer Learning Model (AUC = 1.000)

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

Table 4 demonstrates that the VGG16 Full Freeze configuration outperforms the Baseline CNN across all evaluation metrics. The most notable improvements are in error reduction: the false positive rate (FPR) decreased by 61.1% (from 0.0045 to 0.00175) and the false negative rate (FNR) decreased by 55.6% (from 0.009 to 0.0020), which are critical gains for safety-sensitive infrastructure inspection where missed cracks carry high structural risk. Figure 5 presents the ROC curve for the VGG16 Full Freeze model. As depicted in Figure 5, the curve hugs the top-left corner of the plot, achieving an AUC of 0.9998, which confirms near-perfect class separation across all possible classification thresholds. This result indicates that the model maintains high discriminative power regardless of whether the threshold is adjusted for higher sensitivity or higher specificity in practical deployment scenarios

3.5 Transfer Learning Strategy Comparison

To evaluate the impact of different transfer learning strategies on classification performance and computational efficiency, four training configurations were systematically compared. Table 5 presents the final accuracy and total training time for each configuration.

Table 5. Comparison of Transfer Learning Strategies

Training Configuration	Accuracy	Training Time
Baseline CNN	99.69%	804.38s
Full Freeze (VGG16)	99.84%	3266.03s
Partial Fine-Tuning (VGG16)	99.90%	3449.71s
Full Fine-Tuning (VGG16)	50.00%	6654.73s

The results reveal a clear pattern in the trade-off between accuracy and computational cost. Partial Fine-Tuning achieved the highest accuracy at 99.90%, marginally outperforming Full Freeze (99.84%) and Baseline CNN (99.69%), while requiring moderate training time of 3449.71 seconds. Full Freeze demonstrated the best balance between accuracy and efficiency with only 147,841 trainable parameters and competitive accuracy of 99.84%. Notably, Full Fine-Tuning collapsed to 50.00% accuracy equivalent to random guessing despite consuming the most training time at 6654.73 seconds. This phenomenon is attributed to catastrophic forgetting, where unrestricted gradient updates across all 14.86 million parameters destabilized the pre-trained feature representations of VGG16. These findings confirm that Partial Fine-Tuning is the optimal strategy for this task, while Full Fine-Tuning should be avoided without careful learning rate scheduling.

3.6 Visual Analysis and Edge Detection

The visualisation of model predictions on validation photos shown good accuracy. Of the 8 example predictions presented, the model accurately classified 6 photos, achieving a 75% accuracy rate, with both true positives and true negatives correctly identified. The two misclassified samples were instances with subtle or unclear crack patterns that could provide challenges even for human inspectors. Figure 6 presents representative examples from the dataset, featuring four negative (non-cracked) photos in the upper row and four positive (cracked) images in the lower row. The negative samples display homogenous concrete surfaces with natural texture variations from aggregate particles, but the positive samples distinctly exhibit visible crack patterns with varied widths and orientations, ranging from hairline cracks to more significant structural fractures.

Figure 6. Sample Images of Negative and Positive Concrete Cracks from Dataset

Figure 7 illustrates the actual prediction results from the VGG16 model on validation samples, showcasing both accurate classifications and errors. The visualisation displays eight concrete surface photos organised in two rows, each annotated with the true class (green for "Positive" or black for "Negative") and the anticipated class. In the top row, all four samples exhibit accurate positive predictions with distinctly visible cracks, illustrating the model's proficiency in identifying diverse crack forms, including diagonal, curved, and branching cracks. The bottom row

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

illustrates the model's performance on more complex cases: the first image depicts a correct positive prediction despite a minor crack, the second and third images exemplify two misclassification instances where "True: Negative" surfaces were erroneously classified as "Pred: Positive" due to ambiguous surface textures resembling hairline cracks, and the fourth image presents another accurate positive prediction featuring a discernible thin crack. These results confirm the model's excellent accuracy while emphasising edge circumstances that necessitate more focus in future model enhancements.

Figure 7. Prediction Results Showing True Labels and Model Predictions on Validation Set

Canny edge detection visualisation demonstrated clear distinctions between cracked and uncracked concrete. Negative samples exhibited little edge responses, predominantly identifying surface texture imperfections such as aggregate particles. Positive samples had distinct linear or branching edge patterns associated with crack structures, confirming the visual attributes that the deep learning model is trained to recognise. Figure 8 presents a 4×4 grid comparison between original concrete surface photographs and their corresponding Canny edge detection results, with negative samples and their edge maps in the first two rows and positive (cracked) samples in the final two rows. As shown in Figure 8, fracture patterns yield continuous linear features in the edge maps, while non-cracked surfaces produce scattered, discontinuous edge responses associated with surface roughness. This qualitative analysis although Canny edge detection was not directly integrated into the CNN pipeline validates that crack-related visual patterns are objectively present in the data and confirms the ability of deep learning models to autonomously learn these hierarchical features more effectively than hand-crafted edge operators.

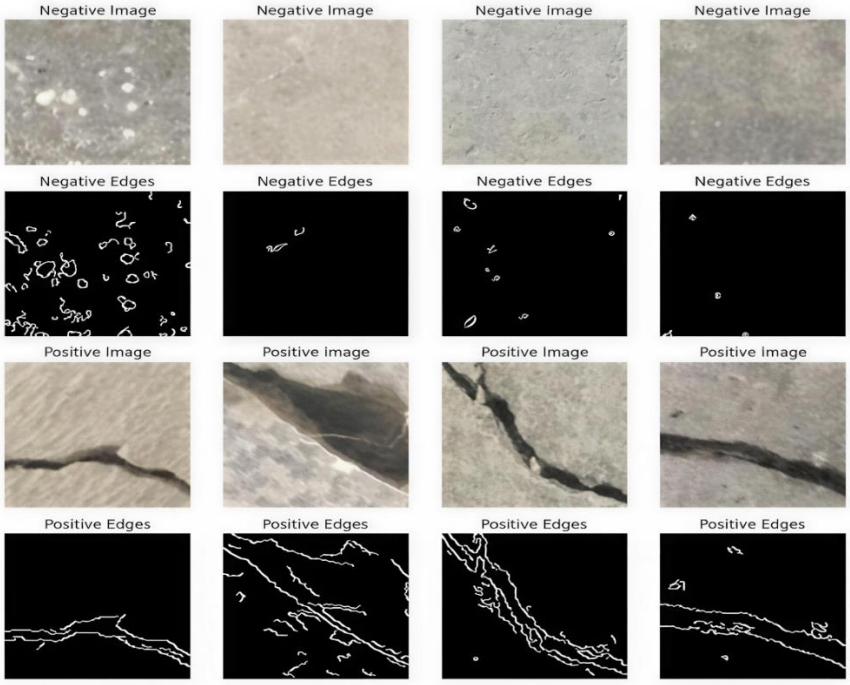

Figure 8. Visualization of Original Images and Canny Edge Detection Results for Negative and Positive Samples

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

The visualisation is structured in a four-by-four grid, displaying negative photos alongside their edge maps in the first two rows, and positive images with their edge maps in the final two rows. The edge detection outcomes reveal that fracture patterns yield continuous linear features in the edge maps, while non-cracked surfaces produce scattered, discontinuous edge points associated with surface roughness. This preprocessing technique, although not directly employed in the CNN pipeline, offers significant insights into the distinguishing characteristics between cracked and unbroken concrete surfaces, hence validating the superior performance of deep learning models in autonomously learning these hierarchical features.

3.7 Discussion

The Full Freeze (VGG16) configuration achieved a validation accuracy of 99.81% compared to 99.55% for the Baseline CNN an absolute improvement of 0.26%, corresponding to a reduction in misclassifications from 36 to 15 out of 8,000 samples. This improvement is attributable to the hierarchical feature representations pre-trained on ImageNet, which provide low-level edge and texture detectors directly applicable to crack detection without requiring these features to be learned from scratch. This finding aligns with transfer learning studies in related domains [6], [7], [13]. The performance gain can be quantified using the relative improvement formula shown in equation (7):

$$Relative\ Improvement\ = [(P(VGG16)\ -\ P(baseline))\ /\ P(baseline)]\ \times\ 100\% \quad (7)$$

Applying this formula, the relative accuracy improvement is 0.26%. More importantly, the false positive rate decreased by 61.1% and false negative rate decreased by 55.6%, which are critical improvements for safety-critical infrastructure inspection applications, representing a 57.8% reduction in overall error rate from 0.45% to 0.19% [2].

An AUC of 0.9998 indicates that the Full Freeze configuration maintains strong class discrimination across all possible classification thresholds a result consistent with the observed 15 misclassifications out of 8,000 samples, since a mathematically perfect AUC of 1.000 is unattainable when any misclassifications are present. The high AUC value is practically significant because it confirms that the model's discriminative power is robust to threshold adjustment, allowing practitioners to shift the decision boundary toward higher sensitivity (lower threshold) for safety-critical inspection contexts without substantially degrading precision [2].

Our results align with and exceed the performance reported in recent literature. Roy et al. reported 98.88% precision using hybrid CNN-VGG16-U-Net architecture [6], while Bukaita et al. achieved 94.7% accuracy with custom CNN for multi-class severity classification [16]. Ahmad and Shaban demonstrated the effectiveness of combining YOLOv10 and ViT for multi-class crack type classification [7]. Our VGG16 transfer learning model achieved 99.81% validation accuracy for binary classification, demonstrating competitive performance that can be attributed to the early stopping mechanism and optimal hyperparameter configuration [15].

The computational efficiency of transfer learning was demonstrated, as the VGG16 model converged with optimal performance at epoch 13 within 15 total epochs, whereas the baseline CNN required 8 epochs with optimal performance at epoch 10. Although the VGG16 model requires 254 seconds per epoch compared to baseline's 60 seconds, this computational overhead is justified by substantial performance improvements and superior loss convergence. This enhanced training characteristics translates to tangible advantages in model quality and generalization capability, corroborating the findings related to efficient model deployment for field applications [11].

Although Canny edge detection facilitated effective visualisation for human analysis of crack patterns, it did not directly enhance the deep learning workflow. This observation substantiates the assertion that contemporary CNNs can autonomously acquire hierarchical features that exceed those derived from manual feature extraction techniques. The edge detection analysis functioned mainly as a validation mechanism to verify the presence of visible crack patterns in the data.

3.8 Limitations and Future Applications

Several limitations should be acknowledged. First, the dataset consists of controlled laboratory images with consistent lighting and resolution, which may not fully represent field conditions with varying environmental factors [3], [12], [16]. Second, the binary classification approach does not assess crack severity or provide localization information, which would be valuable for maintenance prioritization [7], [10]. Third, the model was evaluated on static images rather than real-time video streams, which may introduce additional challenges in practical deployment [17].

Potential applications include integration with unmanned aerial vehicles (UAVs) for bridge inspection, robotic systems for tunnel assessment, and handheld devices for building surveys. Recent studies have successfully demonstrated the effectiveness of deep learning approaches for crack classification and detection in various applications [8], [18], [19], validating the feasibility of automated inspection systems. The model's efficiency makes it suitable for edge computing deployment on resource-constrained devices, as demonstrated by similar lightweight and deployable architectures [11], [20].

4. CONCLUSION

This study presents a systematic comparison of four VGG16 transfer learning strategies Baseline CNN, Full Freeze, Partial Fine-Tuning, and Full Fine-Tuning for automated binary concrete crack classification on a 40,000-image

Building of Informatics, Technology and Science (BITS)
Volume 7, No 4, March 2026 Page: 2597−2608
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i4.9468

dataset. Partial Fine-Tuning achieved the highest accuracy at 99.90%, followed by Full Freeze (99.84%) and Baseline CNN (99.69%), while Full Fine-Tuning collapsed to 50.00% due to catastrophic forgetting caused by unrestricted gradient updates across all 14.86 million VGG16 parameters. The Full Freeze configuration delivered the best efficiency-accuracy trade-off with only 147,841 trainable parameters and an AUC of 0.9998, achieving an error rate of 0.19% (15 misclassifications out of 8,000 samples). These findings provide concrete guidance for applied informatics practitioners: Partial Fine-Tuning is recommended when maximum accuracy is the priority, Full Freeze when computational efficiency is critical, and Full Fine-Tuning should be avoided without careful learning rate scheduling. Future research should extend the system toward multi-class crack severity classification, crack localisation, field condition validation, and edge deployment optimisation for UAV-based structural health monitoring.

ACKNOWLEDGMENT

The authors acknowledge the use of publicly available Concrete Crack Images for Classification dataset and computational resources provided through the Kaggle platform for conducting this research.

REFERENCES

[1] P. Kumar, S. Batchu, N. S. S, and S. R. Kota, "Real-Time Concrete Damage Detection Using Deep Learning for High Rise Structures," *IEEE Access*, vol. 9, pp. 112312–112331, 2021, doi: 10.1109/ACCESS.2021.3102647.
[2] H. S. Munawar, F. Ullah, D. Shahzad, A. Heravi, S. Qayyum, and J. Akram, "Civil Infrastructure Damage and Corrosion Detection: An Application of Machine Learning," *Buildings*, vol. 12, no. 2, 2022, doi: 10.3390/buildings12020156.
[3] M. Panta, M. T. Hoque, M. Abdelguerfi, and M. C. Flanagin, "IterLUNet: Deep Learning Architecture for Pixel-Wise Crack Detection in Levee Systems," *IEEE Access*, vol. 11, no. January, pp. 12249–12262, 2023, doi: 10.1109/ACCESS.2023.3241877.
[4] Z. X. Zhang, H. L. Zhang, and T. J. Zhang, "Enhanced YOLOv8-based pavement crack detection: A high-precision approach," *PLoS One*, vol. 20, no. 5 MAY, pp. 1–18, 2025, doi: 10.1371/journal.pone.0324512.
[5] S. Katsigiannis, S. Seyedzadeh, A. Agapiou, and N. Ramzan, "Deep learning for crack detection on masonry façades using limited data and transfer learning," *J. Build. Eng.*, vol. 76, no. March, p. 107105, 2023, doi: 10.1016/j.jobe.2023.107105.
[6] S. Roy, B. Yogi, R. Majumdar, P. Ghosh, and S. K. Das, "Deep learning-based crack detection and prediction for structural health monitoring," *Discov. Appl. Sci.*, vol. 7, no. 7, 2025, doi: 10.1007/s42452-025-07272-y.
[7] A. M. Mayya and N. F. Alkayem, "Enhance the Concrete Crack Classification Based on a Novel Multi-Stage YOLOV10-ViT Framework," *Sensors*, vol. 24, no. 24, 2024, doi: 10.3390/s24248095.
[8] Y. Choi, "Application of Mask R-CNN and YOLOv8 Algorithms for Concrete Crack Detection," *IEEE Access*, vol. 12, no. November, pp. 165314–165321, 2024, doi: 10.1109/ACCESS.2024.3469951.
[9] A. Ashraf, A. Sophian, A. A. Shafie, T. S. Gunawan, N. N. Ismail, and A. A. Bawono, "Efficient Pavement Crack Detection and Classification Using Custom YOLOv7 Model," *Indones. J. Electr. Eng. Informatics*, vol. 11, no. 1, pp. 119–132, 2023, doi: 10.52549/ijeei.v11i1.4362.
[10] K. Zeng, R. Fan, and X. Tang, "Efficient and accurate road crack detection technology based on YOLOv8-ES," *Auton. Intell. Syst.*, vol. 3, 2025, doi: 10.1007/s43684-025-00091-3.
[11] X. Wang, F. Zhang, and X. Zou, "Efficient Lightweight CNN and 2D Visualization for Concrete Crack Detection in Bridges," *Buildings*, vol. 15, no. 18, p. 3423, Sep. 2025, doi: 10.3390/buildings15183423.
[12] V. P. Golding, Z. Gharineiat, H. S. Munawar, and F. Ullah, "Crack Detection in Concrete Structures Using Deep Learning," *Sustainability*, vol. 14, no. 13, p. 8117, Jul. 2022, doi: 10.3390/su14138117.
[13] T. Fatima and H. Soliman, "Application of VGG16 Transfer Learning for Breast Cancer Detection," *Inf.*, vol. 16, no. 3, 2025, doi: 10.3390/info16030227.
[14] M. M. Islam, M. B. Hossain, M. N. Akhtar, M. A. Moni, and K. F. Hasan, "CNN Based on Transfer Learning Models Using Data Augmentation and Transformation for Detection of Concrete Crack," *Algorithms*, vol. 15, no. 8, p. 287, Aug. 2022, doi: 10.3390/a15080287.
[15] M. Shahin, F. F. Chen, M. Maghanaki, A. Hosseinzadeh, N. Zand, and H. Khodadadi Koodiani, "Improving the Concrete Crack Detection Process via a Hybrid Visual Transformer Algorithm," *Sensors*, vol. 24, no. 10, p. 3247, May 2024, doi: 10.3390/s24103247.
[16] W. Bukaita, K. Vankudothu, and J. Khan, "Automated Multi-Class Concrete Crack Detection and Severity Classification Using CNN-Based Deep Learning," *Am. J. Civ. Eng.*, vol. 13, no. 4, pp. 197–210, Jul. 2025, doi: 10.11648/j.ajce.20251304.12.
[17] L. Ali, F. Alnajjar, H. Al Jassmi, M. Gocho, W. Khan, and M. A. Serhani, "Performance Evaluation of Deep CNN-Based Crack Detection and Localization Techniques for Concrete Structures," 2021.
[18] P. Jing, H. Yu, Z. Hua, S. Xie, and C. Song, "Road Crack Detection Using Deep Neural Network Based on Attention Mechanism and Residual Structure," *IEEE Access*, vol. 11, pp. 919–929, 2023, doi: 10.1109/ACCESS.2022.3233072.
[19] E. Wahyudi, B. Imran, A. Subki, Z. Zaeniah, L. D. Samsumar, and S. Salman, "Crack Detection of Concrete Surfaces with A Combination of Feature Extraction and Image-Based Backpropagation Artificial Neural Networks," *Ilk. J. Ilm.*, vol. 16, no. 3, pp. 228–235, 2024, doi: 10.33096/ilkom.v16i3.2249.228-235.
[20] A. Altaf, A. Mehmood, M. L. Filograno, S. Alharbi, and J. Iqbal, "Deployable Deep Learning Models for Crack Detection: Efficiency, Interpretability, and Severity Estimation," *Buildings*, vol. 15, no. 18, pp. 1–21, 2025, doi: 10.3390/buildings15183362.