VOL. 23, NO. 1, 2024, 133-140
elektrika.utm.my
ISSN 0128-4428

ELEKTRIKA
Journal of Electrical Engineering

Crack Recognition in Masonry Structures: CNN Models with Limited Data Sets

Mazleenda Mazni[1,2*], **Abdul Rashid Husain**[1], **Mohd Ibrahim Shapiai**[3], **Izni Syahrizal Ibrahim**[4], **Devi Willieam Anggara**[5] and **Riyadh Zulkifli**[1]

[1]School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, Skudai, Johor, Malaysia.
[2]Faculty of Mechanical Engineering, Universiti Teknologi MARA Cawangan Johor, Kampus Pasir Gudang, Jalan Purnama, Masai, Malaysia.
[3]Centre for Artificial Intelligence and Robotics, Malaysia-Japan International Institute of Technology, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia.
[4]Forensic Engineering Centre, Institute for Smart Infrastructure and Innovative Construction, Faculty of Civil Engineering, Universiti Teknologi Malaysia, Johor, Malaysia.
[5]School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, Malaysia.

*Corresponding author: mazleenda@graduate.utm.my

Abstract: This research focuses on the task of Masonry Wall Crack Identification using limited data, employing state-of-the-art Convolutional Neural Network (CNN) models. The models investigated include VGG16, MobileNetV2, Xception, and DenseNet121. The dataset, consisting of 946 masonry wall images containing cracks, is used to evaluate the effectiveness of each model in this specific domain. The training set comprises 642 images, the validation set consists of 90 images, and the test set includes 214 images. The models are pretrained on large-scale datasets to extract robust features and are then fine-tuned on the masonry wall crack dataset. Among the models, DenseNet121 stands out, achieving a commendable accuracy of 85.98% in accurately identifying masonry wall cracks. This result underscores the efficacy of DenseNet121 for the challenging task of crack identification in masonry structures using limited data. This study not only contributes to the field of structural health monitoring but also emphasizes the practicality of employing CNN models for real-world applications, particularly in the critical domain of masonry crack identification.

Keywords: Crack identification, Crack Classification, Deep Learning, Convolution Neural Networks

Article History: received 3 January 2024; accepted 25 March 2024; published 28 April 2024.

1. INTRODUCTION

The integrity of masonry structures is crucial for the safety and stability of buildings. Masonry walls, being integral components of various constructions, are susceptible to cracks due to a variety of factors such as environmental conditions, material properties, and structural loads. Cracks are a prevalent occurrence in robust structures like building walls, roofs, bridges, and tunnels. Timely identification of these cracks holds significant importance as it serves as an early indicator, signaling potential issues. Conventional methods for manually detecting cracks are characterized by being time-intensive, requiring significant labor, posing potential dangers, and subject to subjective interpretations [1][2]. Furthermore, the assessment process's reliability is subjective, as it significantly depends on the inspector's skills and physical condition. Inexperienced or fatigued inspectors may inadvertently misreport damage, introducing a potential source of error. Manual inspections also pose safety challenges, especially in areas with limited access and difficult-to-reach structures.

To mitigate the limitations associated with manual inspection, there is a growing emphasis on employing vision-based methods for the assessment and monitoring of civil infrastructures [3][4]. Over time, researchers have shown significant interest in utilizing computer vision for the detection of cracks. Vision-based crack detection exemplifies a non-destructive assessment approach, proving particularly valuable for historical structures governed by stringent regulations [5]. In such cases, where even basic interventions like the placement of crack rulers are prohibited by conservation authorities, vision-based techniques offer an effective alternative.

In the realm of artificial intelligence, Deep Learning (DL), a subfield thereof, and its notable instrument, specifically Convolutional Neural Network (CNN), have demonstrated their effectiveness in the domain of object detection [6]. The allure of CNNs lies in their capacity to formulate predictive models without the need for predefined associations [7]. The characteristic of Deep Learning (DL) algorithms, coupled with the advancements in graphics processing units (GPUs) facilitating rapid computations, has significantly elevated their application across various domains. By employing these methods, there has been a substantial expansion in the capabilities and robustness of conventional approaches [8]. The rapid evolution of computer vision and machine learning (ML) technologies over recent years has given rise to numerous automated techniques as potent tools for addressing practical challenges in crack detection [9].

Furthermore, CNN models have exhibited superior performance in tackling crack identification problems, distinguishing themselves from conventional machine learning techniques by their ability to learn data representations without the imposition of handcrafted rules or prior knowledge [10][11]. Since the introduction of this technology, diverse applications have been explored in the pursuit of utilizing deep learning for structural crack identification. These applications encompass a range of contexts, such as buildings [12][13][14]; bridges [15][16][17], roads [18][19][20]; railway systems [21][22]; tunnels [23][24]; dam [25][26]; and monument [27].

2. MATERIAL AND METHOD

In the first stage, we collect images that showcase concrete cracks from sourced from the work of Hallee Mitchell J. et al. in [28], is a valuable asset for crack detection research. In the subsequent phase, we engage in preprocessing, where we discard unnecessary data and remove undesirable elements like noise and shadows. The third step involves labeling the data, classifying images into two categories: those with cracks and those without. Moving forward, the fourth step encompasses training the model, employing labeled datasets from the prior step and open-source repositories. Finally, in the fifth step, we assess the effectiveness of the trained model by testing it with images sourced from campus buildings.

2.1 Data acquisition

The dataset utilized in this investigation comprises images of concrete surfaces, classified into two categories: 'negative,' denoting surfaces devoid of cracks, and 'positive,' signifying surfaces exhibiting cracks. In total, the dataset comprises 946 files, with 642 allocated to the training set, 90 to validation and 214 data for-testing. The dataset demonstrates a near-balanced distribution between the two classes, with 55% of samples classified as 'crack' and 45% as 'non-crack', ensuring a well-rounded representation of both categories. It is noteworthy that no data augmentation methods, such as random rotation or flipping, were applied during dataset preparation. The images have undergone consistent resizing to dimensions of 224 × 224 pixels. The dataset undergoes a strategic partitioning process, wherein it is divided into training, validation, and testing sets, each fulfilling unique and specific roles in the experimental design. The training set is instrumental in facilitating the learning process for the model. Through exposure to a multitude of examples, the model refines its parameters and gains the ability to make accurate predictions. The validation set plays a crucial role in the fine-tuning phase of model development. It serves as a diagnostic tool, allowing for adjustments to be made to enhance the model's performance based on its evaluation on previously unseen data. The testing set serves as the ultimate benchmark, evaluating the model's proficiency on entirely new and unseen data. This phase gauges the model's generalization ability and provides insights into its real-world applicability and effectiveness.

2.2 CNN Classifier Model Configuration

A plethora of existing convolutional neural networks is available for classification tasks, with an illustrative comparison presenting the manifold architectural alternatives in terms of their accuracy, prediction speed, and model sizes. VGG16 [29], MobileNetV2 [30][31], Xception [32], and DenseNet121 have been chosen based on their reputation for achieving a balance between superior accuracy and relatively compact model sizes.

For instance, research by Simonyan and Zisserman in [33] showcased the effectiveness of VGG16 in achieving high accuracy on image classification tasks. Similarly, MobileNetV2, proposed by Sandler et al. 2018 [34], has been widely acclaimed for its exceptional performance in terms of accuracy and efficiency, particularly on mobile and embedded devices. Xception, introduced by Chollet et. al in [35], employs depth wise separable convolutions to achieve impressive results with reduced computational complexity compared to traditional architectures. DenseNet121, proposed by Huang et al. [36], has also demonstrated remarkable accuracy while maintaining relatively compact model size through dense connections between layers. Their suitability for various applications, particularly in scenarios with limited computational power or memory constraints, makes them preferred choices in many cases. Table 1 offers an overview of the principal characteristics inherent to these pre-existing model instances. The inclusion of a masonry image further enhances the comprehensiveness of the analysis, providing a practical context for the assessment of these neural networks in real-world scenarios.

Table 1. Features of the chosen networks

Network	Size (MB)	Parameter (Million)
VGG16	528	138.4
MobileNetV2	14	3.5
Xception	88	22.9
DenseNet121	33	8.1

In the configuration of these parameters, the batch size and the number of iteration is established for the process of training the model. Specifically, the batch size is set at 16, and the model is trained for 100 epochs. Subsequently, a pre-trained convolutional neural network (CNN) model is loaded from the ImageNet dataset, excluding the top classification layer. The subsequent step involves the immobilization of the layers in the pre-trained model. Following this, a global average pooling layer is introduced to flatten the spatial representation of the preceding model's output. To mitigate overfitting, dropout is implemented as a regularization technique. Specifically, dropout layers with a dropout rate of 0.5 were strategically inserted into the network. Following a certain layer or set of layers, the first dropout layer was applied. Subsequently, after incorporating a fully connected layer with 1024 units and Rectified Linear Unit (ReLU) activation, another dropout layer with the same dropout rate of 0.5 was added. This approach aims to enhance the generalization ability of the model by randomly dropping 50% of the units during training, thereby reducing the likelihood of overfitting to the training data. ReLU is a linear function that produces zero for negative inputs, effectively deactivating the neuron as depicted in equation 1. This strategy provides computational benefits as not all neurons are activated concurrently. Efficient updating of model parameters relies on calculating derivatives for both actual and expected values. The computational process involves the utilization of a loss function, is specified as binary cross entropy. The binary cross-entropy loss function evaluates the discrepancy between predicted and actual binary outcomes, facilitating the iterative refinement of model parameters during the training process. This selection underscores the model's objective of minimizing the disparity between predicted and ground truth labels, ultimately enhancing its performance in binary classification tasks. The loss function assumes a crucial role in iteratively updating model variables during the training procedure.

$$f(x) = max\ (0, x) \tag{1}$$

The model is then compiled utilizing the Adam optimizer, incorporating specified parameters such as the learning rate, beta values, and epsilon. Throughout the training process, this optimizer will be employed to optimize the model.

2.3 System specification

The study utilized Python within the Google Colab environment as the programming language, leveraging TensorFlow and Keras as the primary libraries. The hardware configuration encompassed an Intel(R) Core (TM) i5-3337U CPU operating at a clock speed of 1.80GHz, 4.00 GB of RAM, and the Windows 10 Pro operating system.

3. RESULT AND DISCUSSION

The assessment of crack classification outcomes involved the utilization of a confusion matrix, a widely employed tool in classification scenarios for evaluating classifier performance as depicted in Figure 1. The accuracy of the model reflects the percentage of images accurately classified based on their crack type. In contrast, the recall and precision scores assess the model's ability to accurately identify concrete cracks among all images identified as containing cracks and among all images overall, respectively, regardless of crack presence. The F1 score, resulting from the combination of precision and recall, provides a comprehensive assessment of the model's efficacy. The precise formulas utilized for these assessments are delineated in equations (2) to (5).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3}$$

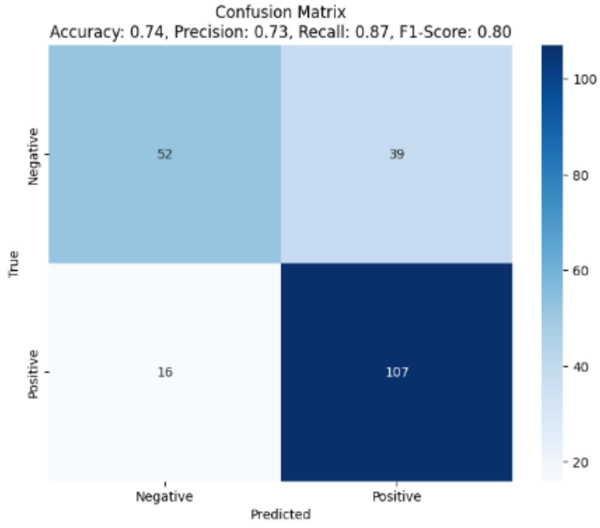
$$\text{Precision} = \frac{TP}{TP+FP} \tag{4}$$

$$\text{F1}_{score} = 2 \times \frac{Precision \times Recall}{Precision+Recall} \tag{5}$$

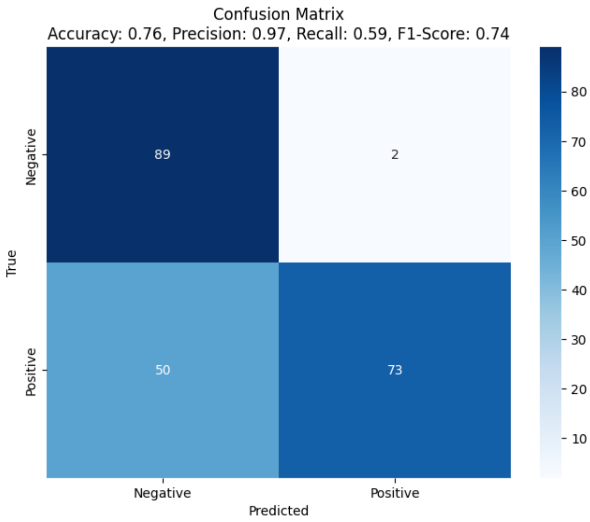
In the context of this research, True Positive (TP) signifies the number of crack images that have been accurately classified, whereas True Negative (TN) represents the correct classification of images without cracks. False Positive (FP) refers to the misclassification of images without cracks as having cracks, and False Negative (FN) indicates the misclassification of crack images as lacking cracks. The outcomes, encompassing TP, TN, FP, and FN, are illustrated in the binary confusion matrix. Given the binary nature of the classification task involving only two classes, namely "crack" and "non-crack," a binary confusion matrix is employed. These metrics play a crucial role in evaluating the precision and dependability of the model. As result, accuracy, precision, recall, and f1-scores value for all the four models are summarized in Table 2.

The results reveal notable performance variations among the models in different metrics. DenseNet121 emerges as the top performer in terms of accuracy, achieving the highest value of 0.860. When it comes to precision, MobilenetV2 outshines the other models with a remarkable score of 0.973. Xception exhibits the highest recall among the models, reaching 0.894. Lastly, in terms of F1 score, DenseNet121 leads with the highest value of 0.877. Subsequently, as depicted in Table 3, various examples of crack identification utilizing the VGG16, MobilenetV2, Xception, and DenseNet121 classifier are showcased. VGG16 and DenseNet121 display strong capabilities, achieving perfect True Positive (TP) rates by correctly identifying all wall crack images. MobileNetV2 and Xception also perform well with slightly lower TP rates, still commendable at 0.9 and 0.88 respectively, indicating their ability to recognize most cracks. However, significant differences emerge in False Positive (FP) classifications. Xception exhibits a relatively high FP rate of 0.47, suggesting a tendency to misclassify crack images as non-cracks. Conversely, Xception's elevated False Negative (FN) rate of 0.6 indicates difficulties in correctly identifying wall cracks, possibly due to limitations in feature extraction. Meanwhile, DenseNet121 demonstrates a slightly higher True Negative (TN) rate, excelling in recognizing non-crack images, while VGG16 and MobileNetV2 display higher FN rates, indicating less accurate detection of non-cracks wall.

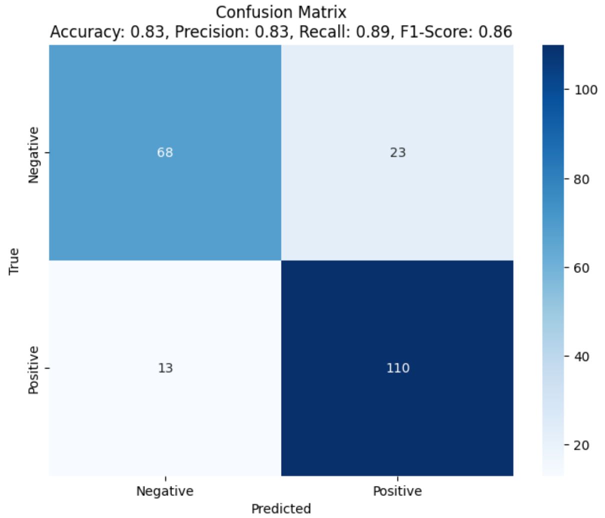
Each instance furnishes particulars such as the anticipated class and the associated prediction outcome for the imperfection. It illustrates the crack classification outcomes using the pre-trained models VGG16, MobilenetV2, Xception, and DenseNet121. The confidence percentage is a probability value obtained from the model's prediction, which uses a specified threshold to categorize images as positive or negative. The results are visualized in a grid of subplots for each tested image. The sample images indicate that the confidence levels for images 1 and 2 are highest for VGG16 and DenseNet121, while for MobileNetV2 and Xception, they are deemed satisfactory.



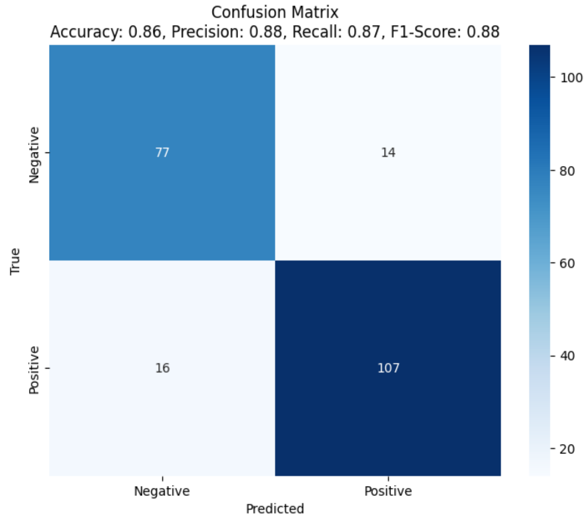
VGG16



MobileNetV2



Xception



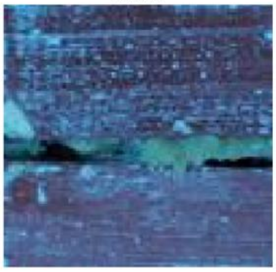
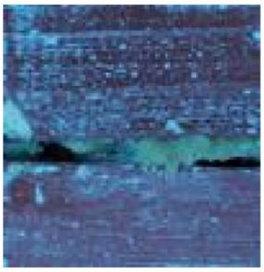
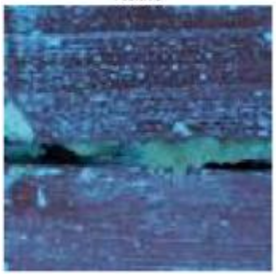
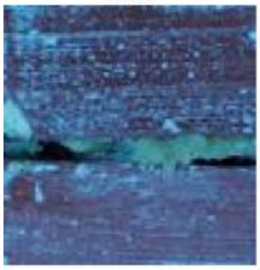
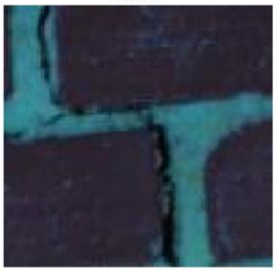
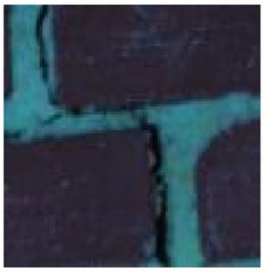
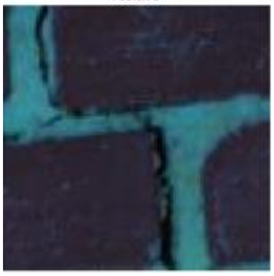
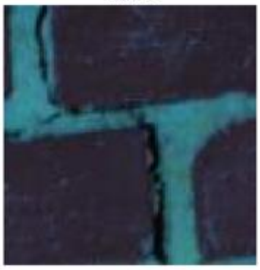
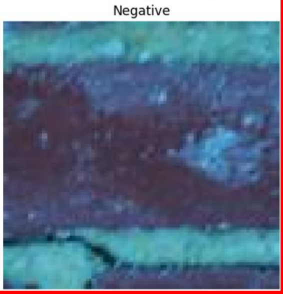
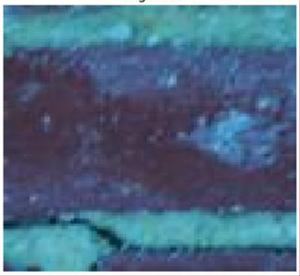
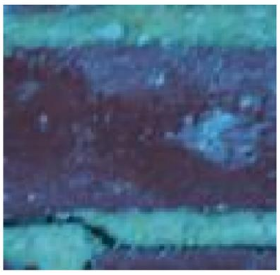
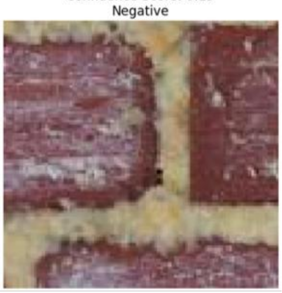
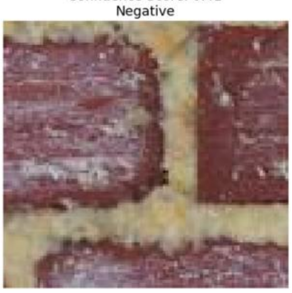
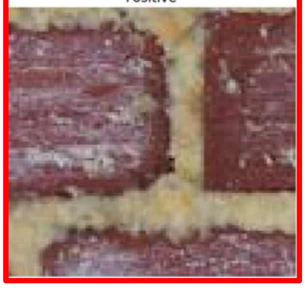
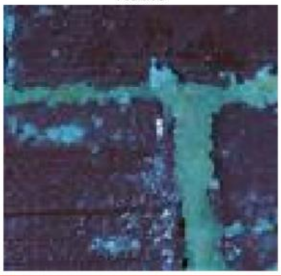
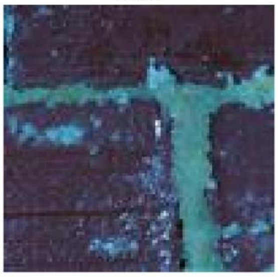
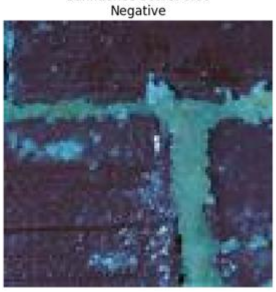
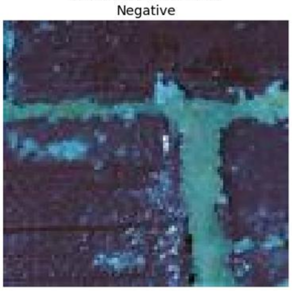
DenseNet121

Figure 1. Confusion matrix for each model

Table 2. Evaluating Performance Metrics for Different Pre-trained CNN Models

Parameters	VGG16	MobilenetV2	Xception	DenseNet121
Accuracy	0.742991	0.757009	0.831776	**0.859813**
Precision	0.732877	**0.973333**	0.827068	0.884298
Recall	0.869919	0.593496	**0.894309**	0.869919
F1 Score	0.795539	0.737374	0.859375	**0.877049**

Table 3. Sample images for crack prediction using Different Pre-trained CNN Models

	VGG16	MobilenetV2	Xception	DenseNet121
Positive	Confidence Score: 1.00 Positive 	Confidence Score: 0.90 Positive 	Confidence Score: 0.88 Positive 	Confidence Score: 1.00 Positive 
	Confidence Score: 1.00 Positive 	Confidence Score: 1.00 Positive 	Confidence Score: 0.80 Positive 	Confidence Score: 1.00 Positive 
	Confidence Score: 0.02 Negative 	Confidence Score: 0.07 Negative 	Confidence Score: 0.47 Negative 	Confidence Score: 0.56 Positive 
Negative	Confidence Score: 0.15 Negative 	Confidence Score: 0.41 Negative 	Confidence Score: 0.60 Positive 	Confidence Score: 0.44 Negative 
	Confidence Score: 0.87 Positive 	Confidence Score: 1.00 Positive 	Confidence Score: 0.00 Negative 	Confidence Score: 0.39 Negative 

4. LIMITATION

While our study contributes valuable insights into masonry crack identification using Convolutional Neural Network (CNN) models, it is essential to acknowledge several limitations that may impact the interpretation and generalizability of our findings:

a. **Limited dataset size**: The dataset used for masonry wall crack identification is relatively small, containing fewer than 1000 images. This limited dataset size may affect the generalizability of the findings and the robustness of the trained models, potentially leading to overfitting on the available data.
b. **Dependency on pre-trained models**: The study heavily relies on pre-trained models such as VGG16, MobileNetV2, Xception, and DenseNet121 for feature extraction and fine-tuning. While pre-trained models offer advantages in terms of efficiency and effectiveness, their performance may influenced by the characteristics of the pre-training datasets and may not fully generalize to the masonry crack identification task.
c. **Sensitivity to image quality**: The performance of the CNN models may be sensitive to variations in image quality, such as lighting conditions, resolution, and image artifacts. In real-world applications, variations in image quality can significantly impact the accuracy and reliability of crack detection algorithms.
d. **Limited evaluation metrics**: The evaluation of model performance primarily focuses on accuracy, precision, recall, and F1 score. While these metrics provide insights into the overall performance of the models, they may not capture specific nuances or trade-offs in crack detection tasks, such as the detection of small or subtle cracks versus larger, more prominent cracks.
e. **Generalization to diverse masonry structures**: The study's findings may be specific to the characteristics of the dataset and may not fully generalize to diverse masonry structures with varying materials, construction methods, and crack patterns. Evaluating model performance across a broader range of masonry structures could provide a more comprehensive understanding of their effectiveness in real-world applications.

5. CONCLUSION

In conclusion, this research addresses the critical task of identifying masonry wall cracks with limited data, leveraging advanced Convolutional Neural Network (CNN) models, including VGG16, MobileNetV2, Xception, and DenseNet121. The comprehensive evaluation, conducted on a dataset comprising 856 masonry wall images, reveals the superior performance of DenseNet121. With a notable accuracy of 85.98%, DenseNet121 excels in accurately identifying masonry wall cracks, emphasizing its effectiveness for this challenging task with limited data. Furthermore, our models achieved commendable performance metrics, including precision of 97.33% by MobileNetV2, recall of 89.43% by Xception, and F1 score of 87.70% by DenseNet121, providing additional validation of the efficacy in masonry crack identification. This study not only advances the field of structural health monitoring but also underscores the practical applicability of CNN models in real-world scenarios, particularly in the crucial domain of masonry crack identification. Our forthcoming research endeavors will revolve around the integration of the proposed models into practical applications within the realm of detecting cracks in brickwork masonry. These applications will be designed to operate on camera-equipped handheld devices, including commercial Unmanned Aerial Vehicle (UAVs), and robotic platforms. By implementing these models in real-world scenarios, such as building inspections, we aim to assess their effectiveness, reliability, and adaptability. It is imperative to acknowledge that the current capabilities of the proposed methods are confined to the detection of cracks in brickwork masonry without providing information on their severity. Consequently, we intend to expand our dataset and delve into deep learning methodologies to gauge the extent of identified cracks. This strategic enhancement aims to elevate the accuracy and applicability of our models in real-world settings.

ACKNOWLEDGMENT

The authors extend their sincere appreciation for the financial backing received for this study. The UTM HIR (Q.J13000.245108G87) and MOHE FRGS (R.J130000.78085F400) research grants provided crucial support to this project. Furthermore, the assistance rendered by Universiti Teknologi MARA Cawangan Johor (UiTM Johor) through Grant (600-TNCPI 5/3/DDN (01) (001/2021) significantly contributed to the successful culmination of this research endeavor.

REFERENCES

[1] Y. Li, M. Yu, D. Wu, R. Li, K. Xu, and L. Cheng, “Automatic pixel-level detection method for concrete crack with channel-spatial attention convolution neural network,” *Structural Health Monitoring*, vol. 22, no. 2, pp. 1460–1477, 2023, doi: 10.1177/14759217221109496.

[2] M. Mazni, A. R. Husain, M. I. Shapiai, I. S. Ibrahim, D. W. Anggara, and R. Zulkifli, “An investigation into real-time surface crack classification and measurement for structural health monitoring using transfer learning convolutional neural networks and Otsu method,” *Alexandria Engineering Journal*, vol. 92, no. October 2023, pp. 310–320, 2024, doi: 10.1016/j.aej.2024.02.052.

[3] P. Savino and F. Tondolo, “Civil infrastructure defect assessment using pixel-wise segmentation based on deep learning,” *Journal of Civil Structural Health Monitoring*, vol. 13, no. 1, pp. 35–48, 2023, doi: 10.1007/s13349-022-00618-9.

[4] B. F. Spencer, V. Hoskere, and Y. Narazaki, “Advances in Computer Vision-Based Civil Infrastructure Inspection and Monitoring,” *Engineering*, vol. 5, no. 2, pp. 199–222, 2019, doi: https://doi.org/10.1016/j.eng.2018.11.030.

[5] C. Yeum and S. Dyke, “Vision-Based Automated Crack Detection for Bridge Inspection,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 30, 2015, doi: 10.1111/mice.12141.

[6] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, “Object Detection with Deep Learning: A Review,” *IEEE Transactions on Neural Networks and Learning*

Systems, vol. 30, no. 11, pp. 3212–3232, 2019, doi: 10.1109/TNNLS.2018.2876865.

[7] D. Liang, X. F. Zhou, S. Wang, and C. J. Liu, "Research on Concrete Cracks Recognition based on Dual Convolutional Neural Network," *KSCE Journal of Civil Engineering*, vol. 23, no. 7, pp. 3066–3074, 2019, doi: 10.1007/s12205-019-2030-x.

[8] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Computer Science*, vol. 2, no. 6, pp. 1–20, 2021, doi: 10.1007/s42979-021-00815-1.

[9] Y. Yu, M. Rashidi, B. Samali, M. Mohammadi, T. N. Nguyen, and X. Zhou, "Crack detection of concrete structures using deep convolutional neural networks optimized by enhanced chicken swarm algorithm," *Structural Health Monitoring*, vol. 21, no. 5, pp. 2244–2263, Jan. 2022, doi: 10.1177/14759217211053546.

[10] L. Ali, F. Alnajjar, H. Al Jassmi, M. Gochoo, W. Khan, and M. A. Serhani, "Performance evaluation of deep CNN-based crack detection and localization techniques for concrete structures," *Sensors*, vol. 21, no. 5, pp. 1–22, 2021, doi: 10.3390/s21051688.

[11] R. Ali, J. H. Chuah, M. S. A. Talip, N. Mokhtar, and M. A. Shoaib, "Structural crack detection using deep convolutional neural networks," *Automation in Construction*, vol. 133, no. November 2020, p. 103989, 2022, doi: 10.1016/j.autcon.2021.103989.

[12] D. Kang, S. S. Benipal, D. L. Gopal, and Y. J. Cha, "Hybrid pixel-level concrete crack segmentation and quantification across complex backgrounds using deep learning," *Automation in Construction*, vol. 118, no. May, p. 103291, 2020, doi: 10.1016/j.autcon.2020.103291.

[13] Z. Wang, G. Xu, Y. Ding, B. Wu, and G. Lu, "A vision-based active learning convolutional neural network model for concrete surface crack detection," *Advances in Structural Engineering*, vol. 23, no. 13, pp. 2952–2964, 2020, doi: 10.1177/1369433220924792.

[14] X. Wu and X. Liu, "Building crack identification and total quality management method based on deep learning," *Pattern Recognition Letters*, vol. 145, pp. 225–231, 2021, doi: 10.1016/j.patrec.2021.01.034.

[15] S. Li and L. Sun, "Detectability of Bridge-Structural Damage Based on Fiber-Optic Sensing through Deep-Convolutional Neural Networks," *Journal of Bridge Engineering*, vol. 25, no. 4, p. 04020012, 2020, doi: 10.1061/(asce)be.1943-5592.0001531.

[16] E. McLaughlin, N. Charron, and S. Narasimhan, "Automated Defect Quantification in Concrete Bridges Using Robotics and Deep Learning," *Journal of Computing in Civil Engineering*, vol. 34, no. 5, p. 04020029, 2020, doi: 10.1061/(asce)cp.1943-5487.0000915.

[17] K. Jang, Y. K. An, B. Kim, and S. Cho, "Automated crack evaluation of a high-rise bridge pier using a ring-type climbing robot," *Computer-Aided Civil and Infrastructure Engineering*, vol. 36, no. 1, pp. 14–29, 2021, doi: 10.1111/mice.12550.

[18] T. S. Tran, V. P. Tran, H. J. Lee, J. M. Flores, and V. P. Le, "A two-step sequential automated crack detection and severity classification process for asphalt pavements," *International Journal of Pavement Engineering*, vol. 0, no. 0, pp. 1–15, 2020, doi: 10.1080/10298436.2020.1836561.

[19] R. Kalfarisi, Z. Y. Wu, and K. Soh, "Crack Detection and Segmentation Using Deep Learning with 3D Reality Mesh Model for Quantitative Assessment and Integrated Visualization," *Journal of Computing in Civil Engineering*, vol. 34, no. 3, p. 04020010, 2020, doi: 10.1061/(asce)cp.1943-5487.0000890.

[20] Q. Mei and M. Gül, "A cost effective solution for pavement crack inspection using cameras and deep neural networks," *Construction and Building Materials*, vol. 256, p. 119397, 2020, doi: 10.1016/j.conbuildmat.2020.119397.

[21] W. Wang *et al.*, "Automated crack severity level detection and classification for ballastless track slab using deep convolutional neural network," *Automation in Construction*, vol. 124, no. January, p. 103484, 2021, doi: 10.1016/j.autcon.2020.103484.

[22] B. Xia, J. Cao, X. Zhang, and Y. Peng, "Automatic concrete sleeper crack detection using a one-stage detector," *International Journal of Intelligent Robotics and Applications*, vol. 4, no. 3, pp. 319–327, 2020, doi: 10.1007/s41315-020-00141-4.

[23] J. S. Lee, S. H. Hwang, I. Y. Choi, and Y. Choi, "Estimation of crack width based on shape-sensitive kernels and semantic segmentation," *Structural Control and Health Monitoring*, vol. 27, no. 4, pp. 1–21, 2020, doi: 10.1002/stc.2504.

[24] Y. Ren *et al.*, "Image-based concrete crack detection in tunnels using deep fully convolutional networks," *Construction and Building Materials*, vol. 234, p. 117367, 2020, doi: 10.1016/j.conbuildmat.2019.117367.

[25] Y. Li *et al.*, "A deep residual neural network framework with transfer learning for concrete dams patch-level crack classification and weakly-supervised localization," *Measurement*, vol. 188, no. August 2021, p. 110641, 2021, doi: 10.1016/j.measurement.2021.110641.

[26] M. Flah, A. R. Suleiman, and M. L. Nehdi, "Classification and quantification of cracks in concrete structures using deep learning image-based techniques," *Cement and Concrete Composites*, vol. 114, no. May, p. 103781, 2020, doi: 10.1016/j.cemconcomp.2020.103781.

[27] N. Wang, X. Zhao, L. Wang, and Z. Zou, "Novel System for Rapid Investigation and Damage Detection in Cultural Heritage Conservation Based on Deep Learning," *Journal of Infrastructure Systems*, vol. 25, no. 3, p. 04019020, 2019, doi: 10.1061/(asce)is.1943-555x.0000499.

[28] M. J. Hallee, R. K. Napolitano, W. F. Reinhart, and B. Glisic, "Dataset for Crack Detection in Images of Masonry Using CNNs." Zenodo, Jul. 2021. doi: 10.5281/zenodo.5108846.

[29] C. V. Dung and L. D. Anh, "Autonomous concrete crack detection using deep fully convolutional neural network," *Automation in Construction*, vol. 99, no. July 2018, pp. 52–58, 2019, doi: 10.1016/j.autcon.2018.11.028.

[30] M. Mazni, A. R. Husain, M. I. Shapiai, I. S. Ibrahim, R. Zulkifli, and D. W. Anggara, "Real-Time Crack Classification with Wall-Climbing Robot Using

MobileNetV2," in *Methods and Applications for Modeling and Simulation of Complex Systems*, F. Hassan, N. Sunar, M. A. Mohd Basri, M. S. A. Mahmud, M. H. I. Ishak, and M. S. Mohamed Ali, Eds., Singapore: Springer Nature Singapore, 2024, pp. 319–328.

[31] W. Qayyum, A. Ahmad, N. Chairman, and A. Aljuhni, "Evaluation of GoogLenet, Mobilenetv2, and Inceptionv3, pre-trained convolutional neural networks for detection and classification of concrete crack images," *1st International Conference on Advances in Civil & Environmental Engineering, University of Engineering & Technology Taxila, Pakistan*, no. March, pp. 2–3, 2022, [Online]. Available: https://www.researchgate.net/publication/359615441

[32] R. E. Philip, A. D. Andrushia, A. Nammalvar, B. G. A. Gurupatham, and K. Roy, "A Comparative Study on Crack Detection in Concrete Walls Using Transfer Learning Techniques," *Journal of Composites Science*, vol. 7, no. 4, 2023, doi: 10.3390/jcs7040169.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–14, 2015.

[34] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018, doi: 10.1109/CVPR.2018.00474.

[35] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1800–1807, 2017, doi: 10.1109/CVPR.2017.195.

[36] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, no. April, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.