Crack Detection in Building Through Deep Learning Feature Extraction and Machine Learning Approach

Afandi Nur Aziz Thohari [1]*, Aisyatul Karima [2]*, Kuwat Santoso [3], Roselina Rahmawati [4]*****
* Informatic Engineering, Politeknik Negeri Semarang
** Computer Engineering Technology, Politeknik Negeri Semarang
*** Road and Bridge Design, Politeknik Negeri Semarang
afandi@polines.ac.id [1], aisya.karima@polines.ac.id [2], kuwatsantoso@polines.ac.id [3], roselina.rahmawati@polines.ac.id [4]

Article Info

Article history:

Received 2024-03-29
Revised 2024-05-14
Accepted 2024-05-15

Keyword:

Crack Detection,
Deep Learning,
Machine Learning Algorithm,
MobileNetV2,
Deployment,
Raspberry Pi.

ABSTRACT

Buildings with cracks are extremely hazardous because they have the potential to cause destruction. Numerous occupants of structures such as houses and buildings are at risk when cracks appear. There are numerous techniques for identifying fractures in structures, including visual inspection, tool use, and expert inspection. The present study employed computer vision, a form of artificial intelligence, to detect cracks in buildings. This research aims to improve the accuracy of the crack detection model. The technique used is to perform a feature extraction process using a deep learning architecture, and then the features will be processed by a machine learning algorithm for image classification. Pre-processing was carried out using CLAHE to increase image sharpness in the crack area prior to the feature extraction process. In this study, the feature map produced by feature extraction was not processed using a fully connected layer. However, for the classification process, it is processed by a machine learning algorithm. This research employs MobileNetV2 as its deep learning architecture and K-NN, Naive Bayes, SVM, XGBoost, and Random Forest as its machine learning classifiers. Test results show that when dividing the 80:20 dataset, XGBoost algorithms can produce the highest accuracy, sensitivity, and specificity values of 99%. Tests in the real environment are performed by deploying Raspberry Pi. Test results show that the prototype can detect cracks in the structure surfaceat a distance of 10 meters in a bright environment. The crack detection process is carried out in real time at an average speed of 42fps.

I. Introduction

Building construction without knowing this poses a high risk. If these buildings are not maintained, the building's structural strength will decrease rapidly and the building will be damaged rapidly. Reduce or prevent damage to buildings should be done regularly. One damage often found in buildings is the crack on the surface of the walls [1]. Therefore, the cracks in building walls must be detected early so as not to cause greater damage.

The detection of structural cracks is very important to ensure the safety and reliability of structural structures. Cracks in buildings are signs of structural problems that require repair. There are several ways to detect structural cracks, such as regular visual inspections, crack measurement tools, and expert inspections [2][3]. The conventional methods of finding structural cracks require a lot of money and take a long time. Visually verifying structural fractures requires constant accuracy and attention. Meanwhile, average people only focus in the first 20 minutes [4]. Of course, this condition is less practical when the building is large and space-saving.

Another method that can be used to detect wall cracks is using a camera. A camera or smartphone takes a picture of the crack and then uses artificial intelligence technology to process it [5][6]. The model accuracy for crack detection was still relatively low in previous research [7]. Therefore, the aim of this research is to establish a classification system of structural defects that has high accuracy using computer vision technology. Computer vision technology has the advantage of being able to monitor in real time, detection processes are fast, and measurement results are accurate because they are trained from thousands of data [8].

Previously, there were several studies examining the construction of crack detection using artificial intelligence technology. One of them is the study of concrete structures to detect cracks using deep learning [9]. The algorithms used are the Convolutional Neural Networks (CNNs) using the VGG16 pre-selected model. There are three methods for processing images used in the preprocessing, namely grayscale, thresholding and edge detection. The test results showed that the best accuracy was 98% in a total of 20 periods.

The next research is to use deep learning to detect damage to buildings in real time [10]. The research used version 3 of You Only Look Once (YOLOv3) algorithm. The test data showed a precision value of 94.24% and a detection speed of 0.033 seconds. Then research into crack detection on concrete surfaces using U-Net and DeepLabV3+ [11]. Comparing U-Net and DeepLabV3+ architectures showed that U-Net's accuracy was higher than 96.47% when detecting cracks on concrete surfaces.

Damage to building structures can be caused not only in buildings, but also in pavement. Research into the application of deep learning for detecting paving cracks has produced 82 per cent recall values and 83 per cent mFscore [12]. The method used in this study is to segment semantically by changing the U-Net architecture. In addition to pavement, crack detection is also useful for monitoring bridge age. The study on bridge crack detection gave 96% accuracy and precision [13]. This study uses a Single-shot Multibox Detector (SSD), which has additional parameters in the form of sliding windows.

In previous research, deep learning and computer vision have been used to detect structural cracks. However, the model built is only program code, and has not yet reached the stage of deployment in embedded devices. In previous research, the accuracy value was less than 90% and models were still overfit. The research aims to improve the accuracy of the construction crack detection by using deep learning and mechanical learning. Deep learning is used to extract features, while machine learning algorithms are used to classify.

The research uses a pre-trained MobileNetV2 model that is the most accurate to detect wall cracks [14]. In the meantime, five machine learning algorithms, K-NN, Nave Bayes, SVM, XGBoost, and Random Forest, are used. One of the improvements in this study is the deployment process. The most accurate model will be deployed on embedded devices for use in real environments.

II. METHODS

The methodology used in this research is shown in Figure 1. The novelty of other research is that the feature extraction process uses pre-trained MobileNetV2 models, while classification processes use machine learning algorithms. There are two scenarios used to achieve the best accuracy. In the first scenario, the data set is divided into 70% of training data and 30% of test data. The second scenario divides the data sets into 80% training data and 20% test data.

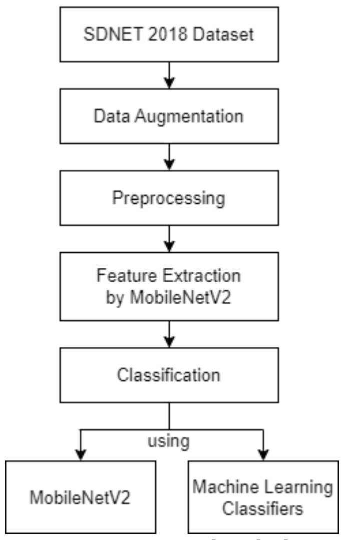


Figure 1. Proposed Method

A. Data Augmentation

Data used in this research is publicly available Structural Defect Network (SDNET) 2018 database [15]. It consists of 500 images of cracked and uncracked concrete walls. These images are 256x256 pixels standardized resolution. Figure 2 shows an example of the presentation of a set of data in the class of damaged walls.

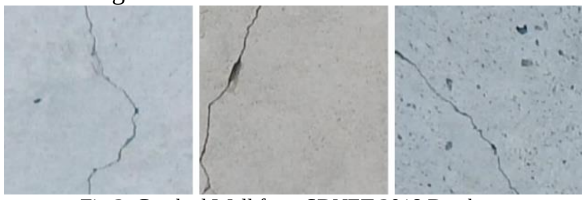


Fig.2. Cracked Wall from SDNET 2018 Database

In order to avoid overfitting in image acquisition, one method is to use data augmentation techniques to increase the number of datasets in order to improve the image recognition of the system [16][17]. In the structural defects images analysis task, convolutional networks proved very well with large datasets [18]. The techniques used in this research were rescale, shear, zoom and flip [19].

TABEL I
NUMBER OF DATASETS AFTER AUGMENTATION

Data	Before Augmentation	After Augmentation
Cracked	500	1.000
Non-Cracked	500	1.000
Total	1000	2.000

The augmentation process yielded a total of 2000 data samples consisting of 1000 samples classified as cracked and

1000 samples classified as non-cracked. Therefore, the quantity of data available to us is equitable. Table I presents information on the number of files after augmentation.

B. Preprocessing

SDNET 2018 Datasets include RGB image channels. In this study, RGB images are converted to grayscales to facilitate crack detection. Then, the gray scale image will be processed using Contrast Limited Adaptive Histogram Equalization Technology (CLAHE) [20]. The purpose of CLAHE is to increase contrast and clarity, allowing cracks to be easily detected. Figure 3 shows the implementation of a preprocessing for processing crack image data. The results of image processing using CLAHE show that crack texture is becoming increasingly visible [21].

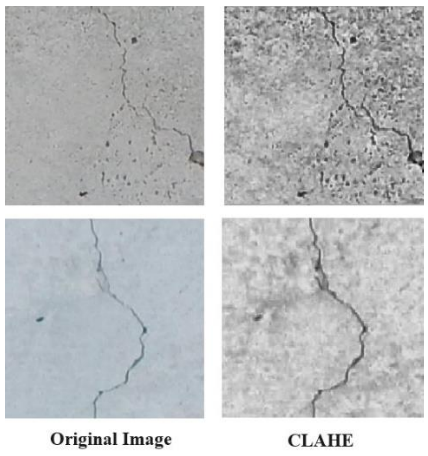


Figure 3. Sample of Preprocessing Implementations

C. Feature Extraction by MobileNetV2

MobileNetV2 is a Convolutional Neural Network (CNN) architecture that can be used for solving the problem of low computing devices such as cell phones, single board computers, etc [22]. MobileNetV2 is an improvement on the mobile network architecture. Mobile Network architecture and CNN architecture generally differ in the use of convolutionary layers [23].

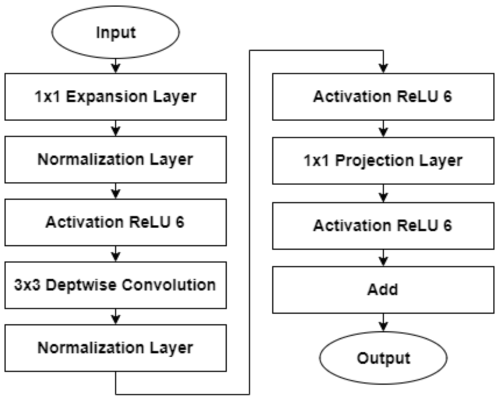


Figure 4. MobileNetV2 Bottleneck Layer

The MobileNetV2 convolution layer uses a filter thickness corresponding to the input image thickness. MobileNetV2 uses a deep-side convolution, a point-side convolution, a linear bottleneck, and a shortcut between bottlenecks. The MobileNetV2 architecture consists of several bottleneck layers. The operation at the bottleneck layer is shown in Figure 4.

The bottleneck layer consists of three operations, namely expansion 1x1, gradient 3x3, and point 1x1. There is another feature, the invention of a residual connection if the input size is the same as the output. MobileNetV2 has 53 convolutionary layers and 1 average pool. The mobilenetV2 architecture changes make the classification results more accurate and detect faster than the mobilenet and traditional CNN architectures [24].

D. Classification

The feature extraction conducted by MobileNetV2 produces 1024 features for each data. These features are inputs for fully connected layers and are activated with softmax as classification phase. MobileNetV2 has made significant progress in developing a lightweight model, as well as more complex and expensive computational models, especially for tasks such as image classification. Its architecture principles, such as inverting residuals and linear bottlenecks, influenced the design of later efficient models in the field of computer vision. The feature extraction and classification process in MobileNetV2 is shown in Figure 5.

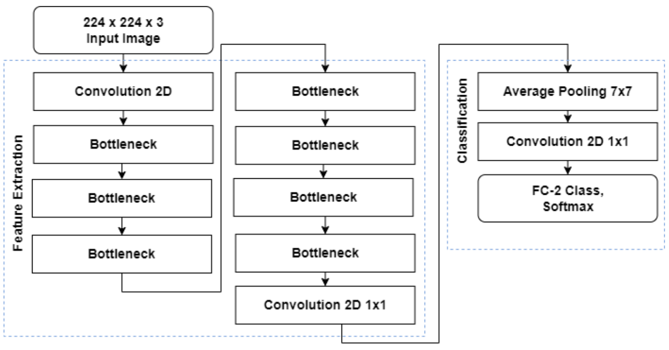


Figure 5. MobileNetV2 Architecture

In addition, machine learning classifiers can be used to classify the features extracted by MobileNetV2 using the features extraction algorithms. This research uses five supervised learning algorithms commonly used for classification: K-NN, Nave Bayes, SVM, XGBoost, and Random Forest [25], [26]. Grid Search technique is used to find optimal parameters for each algorithm [27]. The results of a K-NN algorithm grid search search for the best parameters show that the optimal neighbor number is 5. In the Bayes nave algorithm, the optimal alpha value is 0.01. The optimal hyperparameters of SVM algorithms use Radial Base Function (RBF) values of 2 and C. The number of random forests used in the classification algorithm is 100. Finally, for the XGBoost algorithm, the optimal learning rate is 0.01.

E. Performance Analysis

The confusion matrix is a tool used in machine learning and statistics to evaluate the performance of a classification algorithm [28]. It is a square matrix that is often used to summarize the results of a classification problem. The confusion matrix provides a detailed breakdown of the correct and incorrect predictions made by the classifier. It is particularly useful for solving binary classification problems (two classes), but can be extended to multiclass classification.

The confusion matrix contains four terms that represent the result of the classification process. As shown in Table II, it includes True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). TP and TN provide the result of correct classification data, and FP and FN provide inaccurate classification details. The accuracy, sensitivity and specificity equations are found in (1), (2) and (3), respectively.

TABEL II
CONFUSION MATRIX

		Prediction	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

$$Accuracy = \frac{(TP + TN)}{(TN + FP + FN + TN)} \qquad (1)$$

$$Sensitivity = \frac{TP}{TP + FN} \qquad (2)$$

$$Specificity = \frac{TN}{TP + FN} \qquad (3)$$

III. RESULTS AND DISCUSSION

Wall crack detection uses SDNET 2018 datasets. The data used is a 2000 color image. The data is composed of 1000 cracked images and 1000 non-cracked images. Data addition is implemented to increase the number of images. Prior to the mobile network V2 feature extraction process, the preprocessing was done using CLAHE to improve the quality of image data.

TABEL III
THE EXPERIMENT RESULT USING 70:30 DATA COMPOSITION

No	Classifier	Accuracy	Sensitivity	Specificity
1	MobileNetV2	**97.83%**	**98.31%**	97.35%
2	K-NN	95.5%	92.78%	98.57%
3	Naïve Bayes	91.17%	87.76%	95.23%
4	SVM	97.67%	96.73%	98.63%
5	XGBoost	97.63%	97.04%	**98.64%**
6	Random Forest	96.17%	95.11%	97.26%

The metrics used to measure the success of the model are accuracy, sensitivity and specificity. Test scenarios use the dataset split ratio. In the first scenario, the data sets are divided into 70% training and 30% testing data. The second scenario is a ratio of 80% training data and 20% testing data. The results of the first scenario test are shown in Table III.

The dataset is divided into 70 % training data and 30 % testing data, resulting in measurement values of more than 90%. The MobileNetV2 classification has the highest accuracy and sensitivity values, 97.83% and 98.31%. Meanwhile, the XGBoost Classifier has a maximum accuracy of 98.64%. MobileNetV2 models with a database composition of 70:30 are the best performance. MobileNetV2's feature extraction and classification process is the result of 10 years of training. The number of periods determined using callback technology by determining a precision threshold of 98% [29]. When a threshold value is reached, the training will stop. The visualization of the training results of MobileNetV2 is shown in Figure 6.

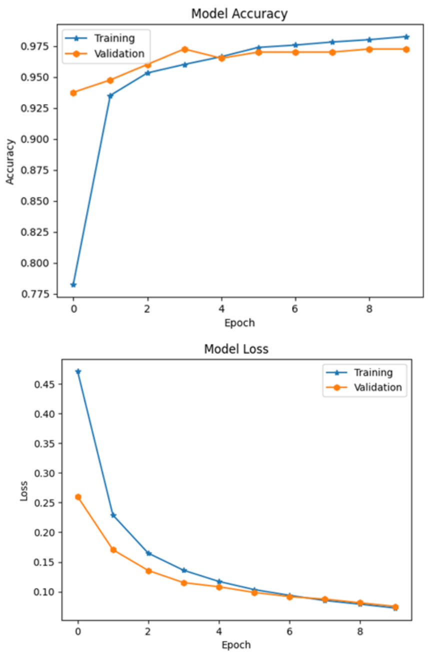


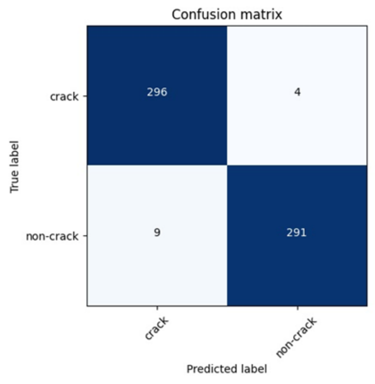
Figure 6. MobileNetv2's Accurate and Loss Values Graph

According to Figure 6, the resulting model does not exhibit overfitting. This is proved by the tendency to increase accuracy in the training process. Meanwhile, the loss value tends to continue to decrease or decrease. The accuracy verification value is quite high and tends to continue to rise. The validation loss is also relatively small and tends to continue to decline.

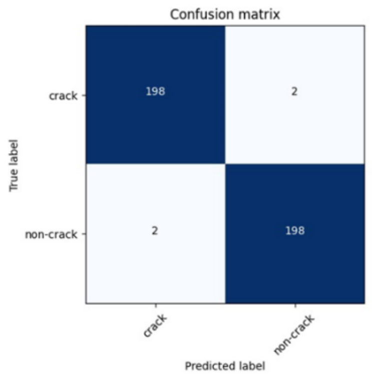
TABEL IV
THE EXPERIMENT RESULT USING 80:20 DATA COMPOSITION

No	Classifier	Accuracy	Sensitivity	Specificity
1	MobileNetV2	97.25%	97.96%	96.55%
2	K-NN	96.0%	93.39%	98.93%
3	Naïve Bayes	91.50%	88.78%	94.62%
4	SVM	97.0%	96.07%	97.95%
5	XGBoost	**99.0%**	**99.0%**	**99.0%**
6	Random Forest	97.0%	95.63%	98.45%

Meanwhile, the results of the test of the model in the dataset composition 80:20 are shown in Table IV. The experimental results show that the accuracy value of the 80:20 dataset division is higher than the 70:30 dataset division. The XGBoost Classifier produces the highest accuracy, sensitivity, and specificity, at 99%. The accuracy, sensitivity and specificity values are obtained using the calculation confusion matrix shown in Figure 7.



(a)



(b)

Figure 7. Confusion Matrix
(a) MobileNetV2 Classifier with 70:30 Data Composition
(b) XGBoost Classifier with 80:20 Data Composition

Based on the results of the Confusion matrix calculation, XGBoost classifiers have a lower prediction error rate than MobileNetV2 classifiers. MobileNetV2 has 13 prediction errors and XGBoost only 4 errors. XGBoost produces the lowest error rate because of a gradient boosting technique, where each machine learning model is built takes the residual error from the previous model as the target variable. This way, each model built will focus on the reduction of errors in the previous model. The 80:20 composition of training and test data produces a more accurate model with a low prediction error rate. Thus, the 80:20 dataset-based XGBoost machine learning model is chosen for single-board computers, namely Raspberry Pi.

The results of model testing in real environment are very high in terms of model accuracy as shown in table IV. In the meantime, the accuracy values are relatively close to those in the test environment. The model can detect wall cracks in real time. The test results show that the camera displays a bounding box and a condition of crack when it reaches a damaged wall or building. The average speed of FPS (frame per second) for crack detection is 42 fps. The results of crack detection on the wall can be seen in Figure 8.

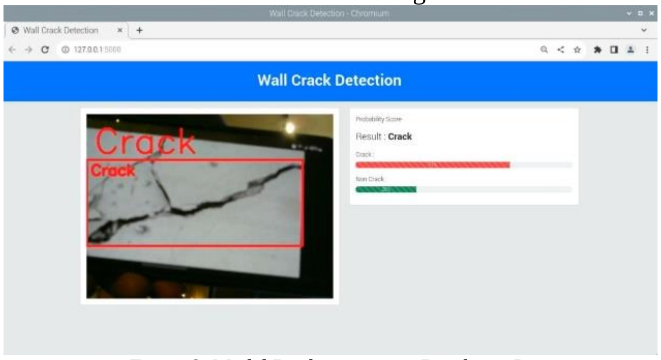


Figure 8. Model Deployment in a Raspberry Pi

Based on prototype test results, a threshold value for camera distance and intensity can be obtained. The distance from the object tested is 1 to 10 meters, and the light intensity is classified as bright, dark, and dark depending on the standards of The Iluminating Engineering Society (IES) [30]. According to IES standards, the bright category value is more than 120 lux, dim is 40–119 lux, and dark is less than 40 lux.

TABEL V
THE EXPERIMENT RESULT USING 80:20 DATA COMPOSITION

Range	light	dim	dark
	> 120 lux	40 – 199 lux	< 40 lux
1 meter	detected	detected	detected
2 meter	detected	detected	detected
3 meter	detected	detected	Not detected
4 meter	detected	detected	Not detected
5 meter	detected	detected	Not detected
6 meter	detected	detected	Not detected
7 meter	detected	detected	Not detected
8 meter	detected	Not detected	Not detected
9 meter	detected	Not detected	Not detected
10 meter	detected	Not detected	Not detected

Table V shows the results of the distance and light intensity testing of the prototype. Based on Table V, the prototype is known to be unable to identify objects (crack) at a distance of more than 7 meters under dark conditions. At the same time, in the dark, the prototype cannot identify objects more than 2 meters away. This is due to the camera's limited ability to capture objects in the dark environment. According to the results of the Table V, the ideal distance for crack detection is 2 meters with a bright light intensity.

IV. CONCLUSION

Previous research [7] reported that crack detection using the CNN algorithm produced a model with an accuracy of 89%. Meanwhile, the accuracy of the resulting model is 94% by using transfer learning CNN. In this research, a crack detection model was built by combining deep learning architecture with machine learning algorithms. Based on the test results, a model was obtained with an accuracy of 99%. The process of achieving high accuracy is by dividing the dataset with a composition of 80:20. Next, feature extraction process was performed using MobileNetV2 and classification process was performed using XGBoost. In addition to the composition of the data sets, the use of CLAHE in the preprocessing process also affects the accuracy of the model. Model testing is not limited to simulation environments, but also to real environments. Testing in real environments is done by deploying the model to the Raspberry Pi. The test results show that the model can detect building surface cracks in real time at 42 fps detection speed. Further research will be carried out to detect cracks in the pavement, bridge and road, as well as in buildings.

ACKNOWLEDGMENT

We gratefully acknowledge the generous support provided by Politeknik Negeri Semarang for funding this research.

REFERENCES

[1] A. Rohmat, "Analisis Kerusakan Struktur Dan Arsitektur Pada Bangunan Gedung," vol. 2, no. 2, pp. 134–140, 2020.

[2] Y. Ren *et al.*, "Image-based concrete crack detection in tunnels using deep fully convolutional networks," *Constr. Build. Mater.*, vol. 234, p. 117367, 2020, doi: 10.1016/j.conbuildmat.2019.117367.

[3] K. Mantani and M. Fauzan, "Desain dan Analisis Struktur Bangunan Adat Sumatera Barat Terhadap Ketahanan Gempa," *J. Tek. Sipil dan Lingkung.*, vol. 4, no. 1, pp. 25–36, 2019, doi: 10.29244/jsil.4.1.25-36.

[4] D. Cornish and D. Dukette, *The Essential 20: Twenty Components of an Excellent Health Care Team*, First. Pittsburgh: RoseDog Books, 2009.

[5] T. Ni, R. Zhou, C. Gu, and Y. Yang, "Measurement of concrete crack feature with android smartphone APP based on digital image processing techniques," *Measurement*, vol. 150, p. 107093, 2020, doi: https://doi.org/10.1016/j.measurement.2019.107093.

[6] H. Perez and J. H. M. Tah, "Deep learning smartphone application for real-time detection of defects in buildings," *Struct. Control Heal. Monit.*, vol. 28, no. 7, pp. 1–15, 2021, doi: 10.1002/stc.2751.

[7] Y. Chen, Z. Zhu, Z. Lin, and Y. Zhou, "Building Surface Crack Detection Using Deep Learning Technology," *Buildings*, vol. 13, no. 7, 2023, doi: 10.3390/buildings13071814.

[8] N. O'Mahony *et al.*, "Deep Learning vs. Traditional Computer Vision BT - Advances in Computer Vision," 2020, pp. 128–144.

[9] V. P. Golding, Z. Gharineiat, H. S. Munawar, and F. Ullah, "Crack Detection in Concrete Structures Using Deep Learning," *Sustain.*, vol. 14, no. 13, 2022, doi: 10.3390/su14138117.

[10] P. Kumar, S. Batchu, N. Swamy 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.

[11] P. N. Hadinata, D. Simanta, L. Eddy, and K. Nagai, "Crack Detection on Concrete Surfaces Using Deep Encoder-Decoder Convolutional Neural Network: A Comparison Study Between U-Net and DeepLabV3+," *J. Civ. Eng. Forum*, vol. 7, no. 3, p. 323, 2021, doi: 10.22146/jcef.65288.

[12] Y. Zhang, X. Gao, and H. Zhang, "Deep Learning-Based Semantic Segmentation Methods for Pavement Cracks," *Inf.*, vol. 14, no. 3, 2023, doi: 10.3390/info14030182.

[13] C. Wan *et al.*, "Crack detection for concrete bridges with imaged based deep learning," *Sci. Prog.*, vol. 105, no. 4, pp. 1–21, 2022, doi: 10.1177/00368504221128487.

[14] S. B. Ali, R. Wate, S. Kujur, A. Singh, and S. Kumar, "Wall Crack Detection Using Transfer Learning-based CNN Models," 2020, doi: 10.1109/INDICON49873.2020.9342392.

[15] M. Maguire, "Structural Defects Network (SDNET) 2018." https://www.deeplearningbook.org/ (accessed Oct. 18, 2023).

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT press Cambridge, 2016.

[17] F. Chollet, *Deep Learning with Python*. Manning, 2017.

[18] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *J. Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0197-0.

[19] A. Sangam, "Image-Augmentation-Using-OpenCV-and-Python." https://github.com/AISangam/Image-Augmentation-Using-OpenCV-and-Python (accessed Oct. 18, 2023).

[20] F. M. Hana and I. D. Maulida, "Analysis of contrast limited adaptive histogram equalization (CLAHE) parameters on finger knuckle print identification," *J. Phys. Conf. Ser.*, vol. 1764, no. 1, pp. 0–6, 2021, doi: 10.1088/1742-6596/1764/1/012049.

[21] M. Harichandana, V. Sowmya, V. V. Sajithvariyar, and R. Sivanpillai, "Comparison of Image Enhancement Techniques for Rapid Processing of Post Flood Images," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. XLIV-M-2–2, no. June, pp. 45–50, 2020, doi: 10.5194/isprs-archives-xliv-m-2-2020-45-2020.

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

[23] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017. doi: https://doi.org/10.48550/arXiv.1704.04861.

[24] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "MobileNetV2 Model for Image Classification," in *2020 2nd International Conference on Information Technology and Computer Application, ITCA 2020*, 2020, pp. 476–480, doi: 10.1109/ITCA52113.2020.00106.

[25] A. N. A. Thohari and R. D. Ramadhani, "Performance Comparison Supervised Machine Learning Models to Predict Customer Transaction Through Social Media Ads," *J. Comput. Networks, Archit. High Perform. Comput.*, vol. 4, no. 2, pp. 116–126, 2022, doi: 10.47709/cnahpc.v4i2.1488.

[26] L. Savitri and R. Nursalim, "Klasifikasi Kualitas Air Minum menggunakan Penerapan Algoritma Machine Learning dengan Pendekatan Supervised Learning," *Diophantine J. Math. Its Appl.*, vol. 2, no. 01, pp. 30–36, 2023, doi: 10.33369/diophantine.v2i01.28260.

[27] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning: Methods, Systems, Challenges*, 1st ed. Springer Publishing Company, Incorporated., 2019.

[28] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.

[29] R. Cai, "Automating bird species classification: A deep learning approach with CNNs," *J. Phys. Conf. Ser.*, vol. 2664, no. 1, 2023, doi: 10.1088/1742-6596/2664/1/012007.

[30] D. L. DiLaura, K. W. Houser, R. G. Mlstrick, and G. R. Steffy, *The Lighting Handbook 10th Edition*, 10th ed. New York, USA: IES, 2011.