

Automatic Crack Classification Model based on Convolutional Neural Network and Metaheuristic JAYA Algorithm

¹Pankaj Bansal, ²Ravi Gaba, ³Mohit Sharma

¹Research Scholar, Civil Department, BGIET, Sangrur, Punjab, India.

²AEE (OS), Punjab Pollution Control Board, Punjab, India.

³Assistant Professor, Civil Department, BGIET, Sangrur, Punjab, India.

Abstract: The main motive of this research is to build an automatic crack classification model in order to enhance the infrastructure system in civil engineering. In order to accomplish this goal, a convolutional neural network is taken into consideration, which takes the crack and non-crack image datasets as input and classifies them. Besides that, the input images are pre-processed using the enhancement and segmentation method in order to enhance the quality and find the region of interest in the images. Furthermore, the characteristics of the input images vary widely. Therefore, the JAYA algorithm is utilised to design the adaptive enhancement and segmentation method, which, according to the objective function, works on the image characteristics. The simulation evaluation is done on the standard dataset images of the crack. Further, various parameters are evaluated based on the confusion matrix in order to validate the performance of the proposed model. Finally, the comparative analysis shows that the proposed model outperforms the existing model.

Keywords: Crack, CNN, JAYA, Metaheuristic, Neural Network, Segmentation, Enhancement.

1. Introduction

Highways, bridges, pavement, tunnels, and other types of civil infrastructure may lose their physical and operational characteristics as a result of long-term operation and exposure to harsh situations related with internal faults, aggressive use, and constant load variations [1]. Thus, it is necessary to continuously and periodically examine the infrastructures to avoid unsafe working conditions, which necessitates the creation of precise and fast inspection techniques.

Cracking is one of the most common types of problems in civil infrastructure, and it develops quickly as a result of high traffic, material aging, and extreme changes in the environment [2]. Cracks often form and spread in three dimensions on the surface as well as within the structure, finally resulting in local damage or collapse. Cracking may speed the degradation process, therefore its frequency and seriousness are crucial signs of necessity for maintenance. The first stage of many diagnostic and inspection procedures is crack detection. Finding cracks helps with investment planning by managing the few available repair resources and predicting future circumstances. This way, maintenance efforts may be minimized while maintaining the infrastructure's capacity to satisfy service requirements.

Conventional techniques for inspecting infrastructure surfaces for cracks depend on highly qualified engineers with a wide range of knowledge to capture defects on structures at regular intervals or after catastrophes [3]. In order to make sure that infrastructures continue to satisfy safety and serviceability standards, they manually evaluate the functional and physical conditions of the systems. Even though manual inspection is often used, carrying out manual inspection and assessment procedures at set or regular intervals is expensive, labor-intensive, and time-consuming. Furthermore, since manual inspection alone rely on the expertise of the professionals, it is vulnerable to subjectivity and could compromise the reliability of the quantitative analysis. Vision-based

techniques for crack identification and structural health inspection are becoming more and more common as a result of advancements in visual sensing technology, such as digital cameras, which are inexpensive, dependable, high-resolution, and user-friendly. In the present scenario, automatic crack detection model is designed using the machine learning algorithm in order to enhance the accuracy and to reduce the cost factor.

The main aim of this research is to design an automatic crack classification model by utilising adaptive enhancement and segmentation methods along with a convolutional neural network. In the adaptive enhancement and segmentation methods, the quality of the image is enhanced and segmented according to its characteristics. In order to accomplish this goal, the metaheuristic JAYA algorithm is used. This algorithm searches for the best parameter value of the enhancement and segmentation methods based on the objective function. In the image enhancement, entropy is chosen, whereas in the segmentation, the Euclidean distance parameter is chosen as the objective function. Next, the CNN algorithm is trained and tested using the images obtained, and the simulation evaluation on the standard dataset of crack images is done after the segmentation method for crack classification purposes. Finally, the simulation evaluation on the standard dataset of crack images is done. From the results, we found that the proposed model achieves better accuracy over existing methods to classify crack and non-crack images.

The remaining structure of the paper is classified into six sections. Section 2 presents the related work. Section 3 defines the methods are taken into consideration for the proposed model. Section 4 explains the proposed crack classification model. The simulation evaluation is shown in Section 5. Finally, conclusion and future work is drawn for the paper in Section 6.

2. Related Work

In this section, we have studied and analysed the crack classification models are proposed in the literature. Xu et al. [4], developed an end-to-end approach to automatically identify bridge cracks using a convolutional neural network. They used simply images and captions as input, training a single CNN end-to-end using images to identify cracks. Without requiring pre-training or fine-tuning on additional datasets, the CNN model recorded an accuracy of 96.37% in identifying cracks. Laxman et al. [5], used the convolutional neural network for feature extraction and regression models (random forest and XGBoost) for crack classification and depth analysis. From the literature survey, we found that CNN is the most used method for crack classification. However, none of the researchers is done the pre-processing of the crack images in order to enhance the performance of the classification model. Therefore, in this research, this issue is overcome by pre-processing the dataset images using the enhancement and segmentation method by deploying the metaheuristic JAYA algorithm before classification using the CNN method.

3. Methods

In this section, the methods are utilized for the proposed model is explained. The proposed model has two main stages, pre-processing stage, and classification stage. In the pre-processing stage, three methods are taken into consideration for accomplish two goals, namely, quality of the crack image and segmentation. The quality of the crack images is enhanced using the adaptive enhancement method whereas segmentation of the crack image is done using the adaptive k-mean clustering algorithm. In both methods, metaheuristic JAYA algorithm is utilized for determine the optimal parameters based on the objective function. On the other hand, in the classification stage, convolutional neural network is utilized for crack classification. Next, a detailed description of the methods is given which used in pre-processing and classification stages.

- **Crack Image Quality Enhancement using Image Enhancement Method:** In the pre-processing stage, the input crack images are enhanced using the enhancement method. In the proposed model, power law method is used to correct the gamma value of the image [6]. The gamma correction helps to control the brightness of the image. The power law method is determined using Eq. (1).

$$P = I_{Peak} \times \left[\frac{I}{I_{Peak}} \right]^{\gamma} \quad (1)$$

- **Crack Image Segmentation using k-mean Clustering Method:** Road image's crack information must be abstracted, so the image is easily computed and converted to binary [7]. Binary images are a particular kind of

grayscale image that are composed of two pixel types: white and black. The road image is separated into two groups: the backdrop region and the crack target area. Image processing depends on threshold selection for binary images. Initially, two pixel values are chosen as the first cluster center using the initial center technique. In this research, two suitable cluster center sites are chosen based on the crack image histogram. The closest distance will be used to divide each pixel point into two groups. The procedure is repeated iteratively until the clustering result converges. Next, as a clustering result, the most suitable spots are chosen. The K-means method is then applied to each and every pixel in the image. Equation (2) defines the difference between pixel values as the metric used to quantify the degree of a similarity among a group of comparable data samples.

$$J = \sum_{i=1}^K \sum_{x_j \in C_i} |x_j - m_i|^2 \quad (2)$$

where m_i is the mean of the points in cluster C_i , and x_j is a data point in the spatial of the road crack image. In this study, the best suitable pixel point is chosen as the cluster center based on the crack image's histogram. In the typical K-means method, the center point varies constantly, making the abstracted crack information's consequences even more unpredictable. The center point will shift as traversal times rise. Furthermore, the crack image has extremely large amounts of data, which increases resource usage and spatial complexity.

- **Metaheuristic JAYA Algorithm for Enhancement and k-mean Clustering Method:** The performance of the enhancement and k-mean clustering method depends on various parameters are used in it [8]. Therefore, in the proposed model, the optimal values of these parameters are determined using the JAYA algorithm. JAYA algorithm is an optimization algorithm is utilized in the various fields to find the optimal values of the parameter. Next, a detailed description of JAYA algorithm is given.

Rao recently introduced the JAYA algorithm, a population-based metaheuristic algorithm. The original purpose of the JAYA method was to solve optimization functions that were both limited and unconstrained. The name JAYA is Sanskrit in origin and signifies "victory". The characteristics of swarm-based intelligence and evolutionary algorithms are combined in this population-based metaheuristic algorithm. The "survival of the fittest" theory's natural behavior serves as its inspiration. This indicates that although JAYA solutions are ignoring the poorest solutions globally, they are drawn to the finest solutions worldwide. JAYA's search procedure strives to attain the worldwide best solutions and avoid failure by running out of the worst ideas. The JAYA algorithm provides a number of benefits over other population-based algorithms, including ease of implementation and the absence of dependency on algorithm-specific parameters (i.e., maximum number of iterations and population size).

The JAYA algorithm technique is divided into five stages: initializing the parameters, building the JAYA's initial population, evaluating the JAYA, updating the JAYA memory, and terminating rule. Figure 4.3 displays the JAYA algorithm's flowchart.

Step 1 Set the initial values for the JAYA algorithm and the optimization problem. At the beginning of the run, the JAYA Algorithm's parameters are set. There are interestingly no control settings in the JAYA algorithm. The population size N and the number of iterations T are the only two algorithmic parameters. The typical optimization constraint issue is:

$$\begin{aligned} &\min f(x) \\ &\text{S.t.} \\ &g_j(x) = c_j \quad \forall j = (1, 2, \dots, n) \\ &h_k(x) \leq d_k \quad \forall k = (1, 2, \dots, m) \end{aligned} \quad (3)$$

In the above equation, $f(x)$ is the objective function that is used to determine the solution's fitness value, $x = (x_1, x_2, \dots, x_D)$. The decision variable x_i is given a value inside the lower and upper limit range, ensuring that x_i falls within the interval $[X_{\min i}, X_{\max i}]$. The j th equality requirements are denoted by g_j , and the k inequality constraints by h_k . Typically, a benchmark dataset's dimensions, issue variables, and associated data are extracted.

Step 2: Creating the starting population for JAYA. The JAYA Memory (JM) is where the initial solutions, or population, of the JAYA algorithm are built and stored. As shown by Eq. 4, where N is the total number of solutions and D is the solution dimension, the JM is an augmented matrix of size $N \times D$. Traditionally, the solution is created at random using the formula $JM_{i,j} = X_{\min j} + (X_{\min j} - X_{\max j}) \times \text{rnd}$, where $\forall i \in (1, 2, \dots, N) \wedge \forall j \in (1, 2, \dots, D)$. The uniform function rnd creates random numbers between 0 and 1.

$$JM = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_D^1 \\ x_1^2 & x_2^2 & \dots & x_D^2 \\ \vdots & \vdots & \dots & \vdots \\ x_1^N & x_2^N & \dots & x_D^N \end{bmatrix} \begin{bmatrix} f(x^1) \\ f(x^2) \\ \vdots \\ f(x^N) \end{bmatrix} \quad (4)$$

Every solution's objective function $f(x_i)$ is similarly computed, and the JM solutions are arranged according to the values of their objective functions, which are sorted in ascending order. As a result, the best answer is x_1 , while the worst is x_N .

Step 3 The JAYA Evolution procedure. Repeat after repetition, The JAYA operator, which is derived from Eq. 5, modifies the decision variables of every solution in the JM.

$$x_j^{ij} = x_j^i + r_1 \times (x_j^1 - |x_j^i|) - r_2 \times (x_j^N - |x_j^i|) \quad (5)$$

Note that x_j^{ij} represents the new updated solution, whereas x_{ij} is the existing solution. The decision value x_{ij} has been modified to become x_j^{ij} . A random number in the interval $[0,1]$ is produced by the two uniform functions, r_1 and r_2 . The purpose of these produced random numbers is to find the ideal equilibrium between the processes of exploration and exploitation. It should be noted that the decision variable j in the best solution is x_1^j , whereas the decision variable j in the worst solution is x_N^j . JAYA algorithm diversity control depends on the distance between the best and worst solution choice variables. A closer distance suggests more exploitation, whereas a greater distance means more exploration.

Step4: Update the JM. There will always be updates to the JM solutions. The new solution's objective function value, $f(x_i)$, is computed. If $f(x_i) < f(x_i)$, the new solution x_i' will take the place of the existing solution x_i . The number of times this procedure is performed is N .

Step: 5 Stop rule. Steps 3 and 4 of the JAYA algorithm are repeated until the stopping rule is reached which can include the maximum number of repetitions T —is achieved. The flowchart of the JAYA algorithm is given in Figure 1.

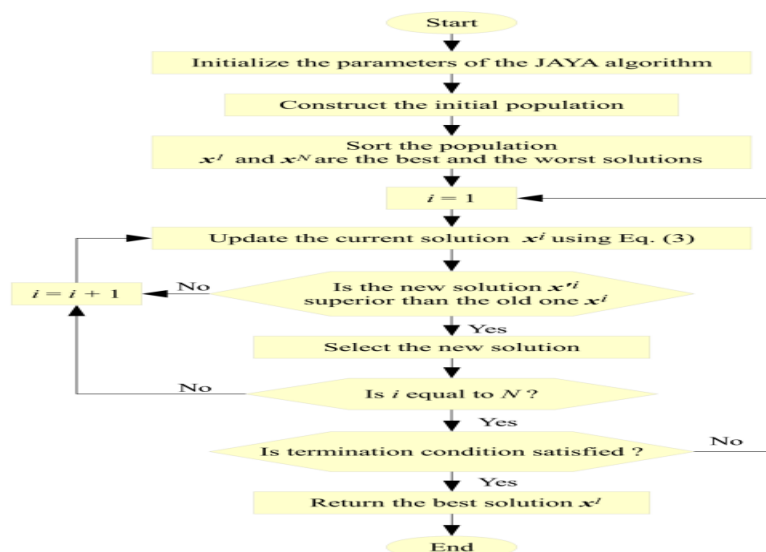


Figure 1 Flowchart of the JAYA Algorithm [8]

- **Convolutional Neural Network:** CNN is a type of deep neural design that is often used to sort images into groups. Convolutional layers are used to analyze the characteristics extracted from the input, whereas linked layers are used for classification. There is a pooling layer, an activation function, and a convolutional layer in each convolutional layer configuration. CNN uses a collection of linked kernel filters to capture the correlation between the input characteristics. The convolutional process is carried out by these filters, which make up the central component of the CNN, using the output from the layers before it. The CNN architecture's working concept is shown in Figure 2 [9].

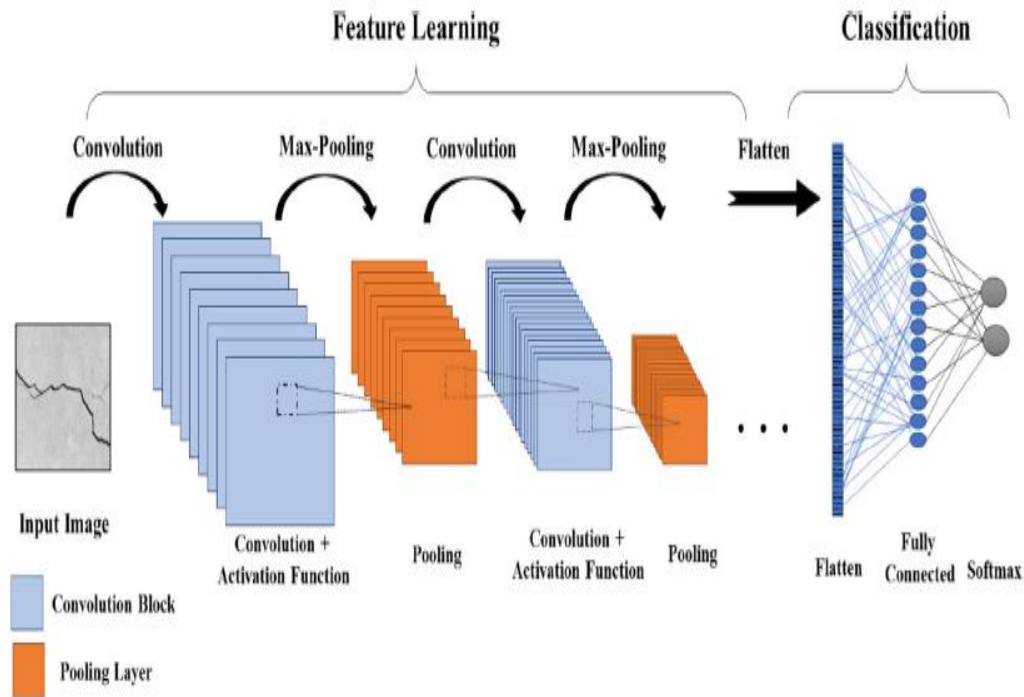


Figure 2: Architecture of CNN [9]

The following is the mathematical expression for the convolution of the output y at every location of P_y :

$$y(P_y) = \sum w(P_G).x(P_y + P_G) \quad (6)$$

where the variables x , w , G , and P_G represent the input, filter, field, and positions in the convolutional layer, respectively. The model's nonlinearity is improved by the activation function. CNN often uses sigmoid, ReLU, and Tanh as some of its activation functions. Depending on how the images are represented, the inputs of a 2D CNN layer may be shown as many 2D matrices with various channels. Multiple filters are enabled in the convolutional layer, which may scan inputs and generate mappings for the outputs. A minimum of $M \times N$ filters is required to execute the convolutional operations when there are M inputs and N outputs.

4. Proposed Crack Classification Model

This section explains how the proposed crack classification model is designed using a convolutional neural network. Figure 3 shows the flowchart of the proposed model. Initially, the standard dataset is read. The dataset contains crack and non-crack images. After that, pre-processing of the proposed model is done using the adaptive image enhancement and segmentation method. In the adaptive approach, the optimal parameter values of the image enhancement and segmentation methods are determined using the optimisation JAYA algorithm. This algorithm searches for the best value based on the objective function. Further, the convolutional neural network is trained and tested using the segmented images. Finally, the performance evaluation of the proposed model is done using the various performance metrics.

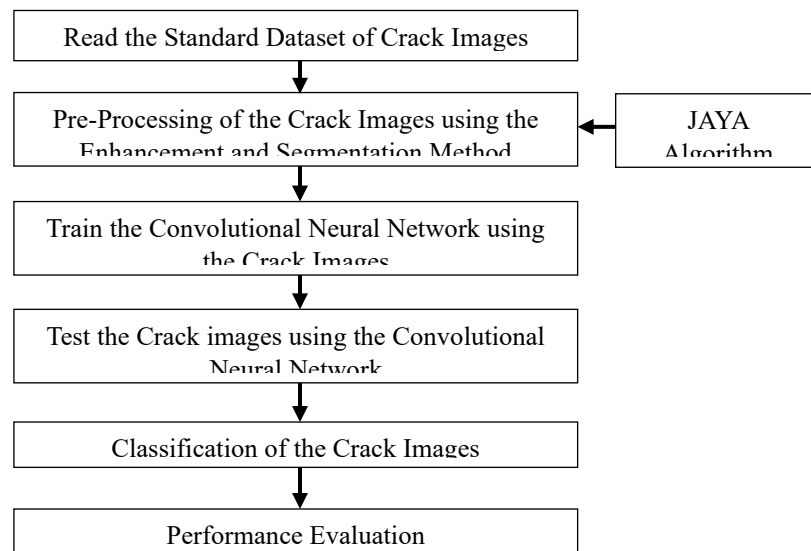


Figure 3 Flowchart of the Proposed Crack Classification Model

5. Simulation Evaluation

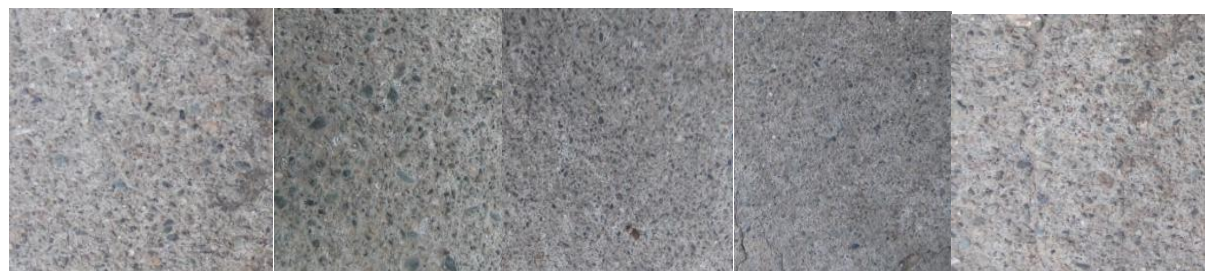
In this section, the simulation evaluation of the proposed crack classification model is done to evaluate its performance over the existing models. The proposed model was simulated in MATLAB 2018a software.

5.1 Standard Dataset

To identify the characteristics of images with and without cracks, a binary-class CNN model was created for crack identification. The model was trained, validated, and tested using a publicly available online dataset that included images of surfaces of concrete with and without cracks. The Middle East Technical University campus buildings were the locations of these images, which were taken from Kaggle [10]. There are 40,000 227x227 pixel images with RGB channels. Two sets of these images were created: “Undamaged” images are those without cracks, while “Cracked” images are those with cracks. 20,000 images make up each collection. Fig. 4 displays a selection of the common images—both with and without cracks—that have been used in the research.



(a) Crack Images



(b) Non-Crack Images

Figure 4: Crack and Non-Crack Image Database

5.2 Simulation Setup Configuration

This section shows the simulation setup configuration is defined for the proposed model to classify the crack images. In the proposed model, JAYA algorithm is utilized in the pre-processing step which searches the best parameter values of the enhancement and segmentation method. Therefore, different parameters of JAYA algorithm need to initialise, as shown in Table 1.

Table 1 Simulation Setup Configuration

Parameter	Values
Population	10
Iteration	15
Objective Function	Entropy, pdist2

5.3 Simulation Results

In this section, the performance metrics are explained that used for evaluate the proposed model and compare with the existing models. A number of assessment variables often used in classification tasks are employed to objectively assess the model's performance. Below are some basic concepts [11]:

1. TP (True Positives): The count of classes that are expected to be positive and are positive. The term “TP” in the algorithm denotes the quantity of cracks that are accurately identified as genuine.
2. TN (True Negatives): The total number of classes that are expected to be negatively assessed. The number of backgrounds that are accurately identified as backgrounds in the model is denoted by the symbol TN.
3. FP (False Positives): The total number of classes that are expected to be positive but are actually negative. The term “FP” in our approach denotes the number of backgrounds that are inaccurately classified as cracks.
4. FN (False Negatives): The quantity of positive classes that are forecasted to be negative classes. The number of cracks that are mistakenly recognized as the background in the model is denoted by FN.

- Accuracy: The ratio of effectively categorized cases to the total number of instances, which indicates the classifier's overall accuracy. In our approach, accuracy is defined as the proportion of properly recognized backgrounds and cracks. Equation (1) displays its computing formula:

$$A = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

- Precision: What proportion of the predicted positive cases are real positive instances. The accuracy in our model is defined as the percentage of actual cracks among all the cases that are categorized as cracks. Equation (2) provides its computation formula.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

- Sensitivity (also called Recall): The efficacy of a classifier is represented by the proportion of positive cases that are properly classified out of all the positive occurrences. In our methodology, the percentage of actual cracks that are identified as cracks is referred to as sensitivity. Equation (3) displays its formula:

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

- F-Score: The F-Measure measures the correlation between the positive label assigned to the data and the label assigned by the classifier. It provides a thorough examination of both precision and sensitivity. Equation (4) provides its calculation formula:

$$F - Score = \frac{2PR}{P+R} \quad (4)$$

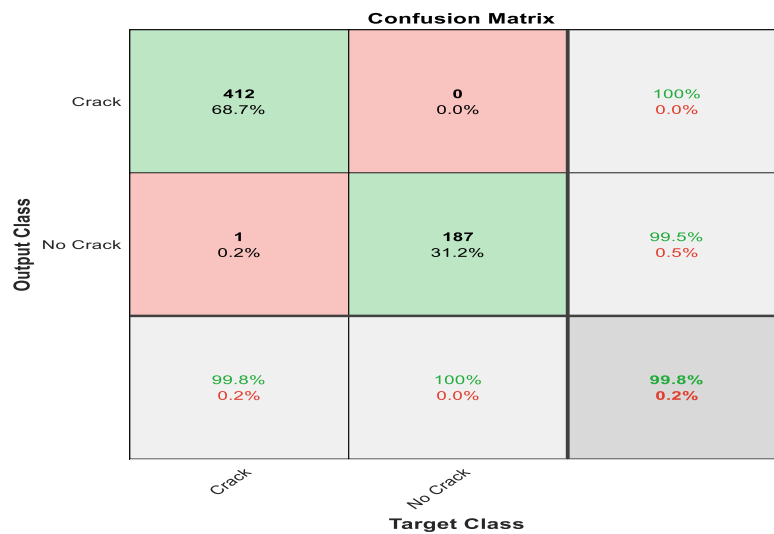


Figure 5 Confusion Matrix for the Proposed Model

Figure 5 shows the confusion matrix is evaluated for the proposed model. The confusion matrix shows the total number of true positive, true negative, false positive, and false negative cases is determined using the proposed model to classify the crack images. Based on this matrix, other parameters are determined for the proposed model, as shown in Table 2.

Table 2 Performance Evaluation

N-Classes	Accuracy	Recall	Precision	F-Score
1	0.99833	0.99758	1	0.99879
2	0.99833	1	0.99468	0.99733

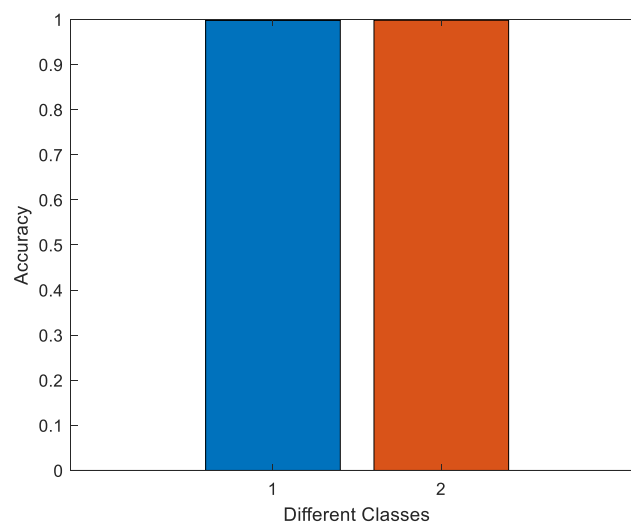


Figure 6 Accuracy for Different Classes of Crack Images

Figure 6 shows the accuracy comparison of the proposed model for the different classes (crack and non-crack images). The result shows that the proposed model is effectively classify the different classes with high accuracy value of 0.99833.

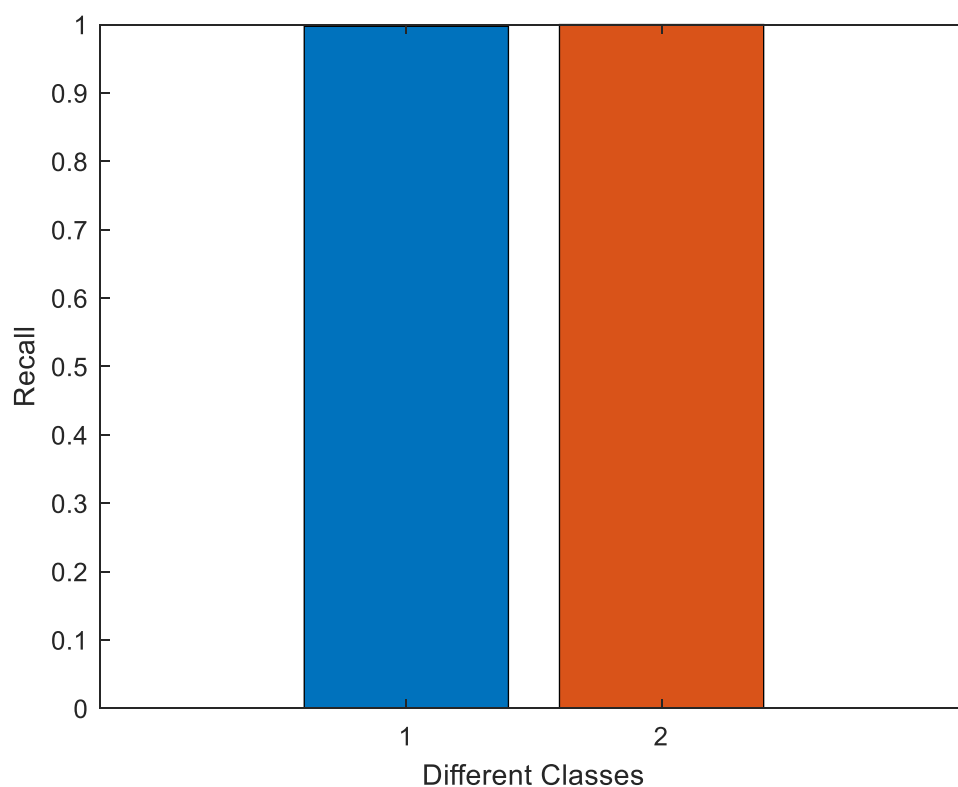


Figure 7 Recall for Different Classes of Crack Images

Figure 7 shows the recall parameter comparison of the proposed model for the different classes (crack and non-crack images). The result shows that the proposed model is effectively classify the different classes with high recall value of 0.99758 for class 1 and 1 value for class 2.

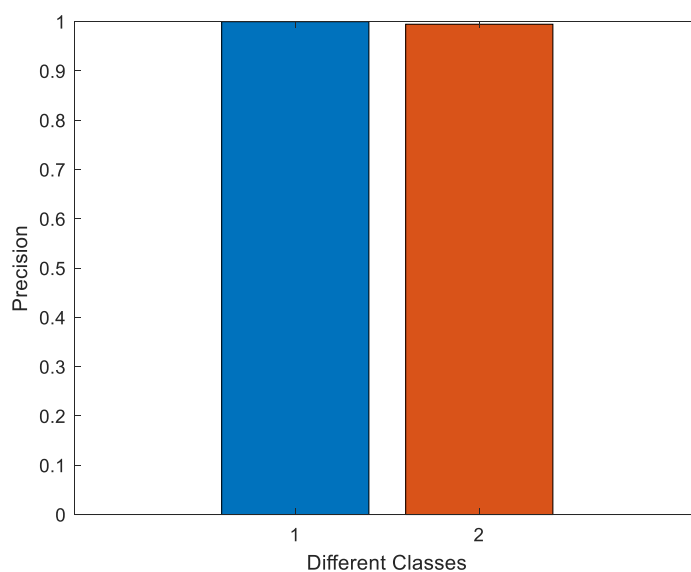


Figure 8 Precision for Different Classes of Crack Images

Figure 8 shows the precision parameter comparison of the proposed model for the different classes (crack and non-crack images). The result shows that the proposed model is effectively classify the different classes with high recall value of 1 for class 1 and 0.99468 value for class 2.

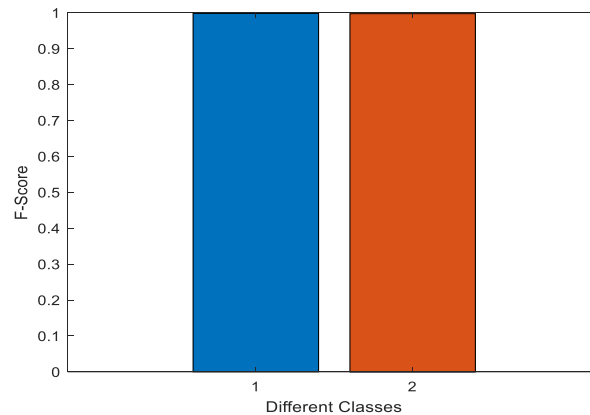


Figure 9 F-Score for Different Classes

Figure 9 shows the recall parameter comparison of the proposed model for the different classes (crack and non-crack images). The result shows that the proposed model is effectively classify the different classes with high recall value of 0.99879 for class 1 and 0.99733 value for class 2.

5.4 Comparative Analysis

In this section, we have performed the comparative analysis of the proposed model with the existing crack classification model based on CNN based on accuracy parameter. The result shows that the proposed model outperforms over the existing model.

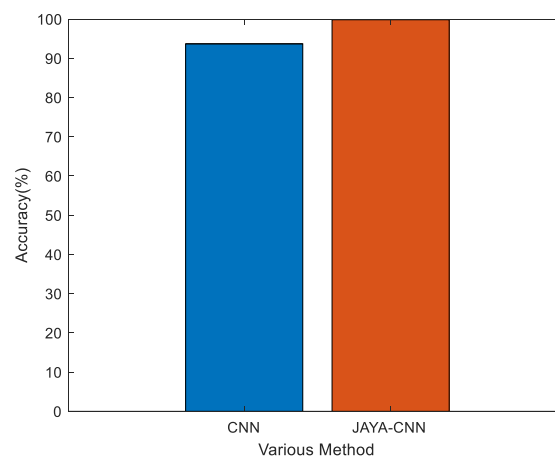


Figure 5.6 Comparative Analysis based on Accuracy Parameter

6. Conclusion and Future Work

In this research, we have developed an automatic crack classification model using a convolutional neural network. The performance of the proposed model is enhanced by pre-processing the crack images using adaptive enhancement and segmentation methods. In these methods, the optimal parameter values are determined using the JAYA algorithm. JAYA is a metaheuristic algorithm that comes under the optimisation class and determines the optimal parameter value based on the defined objective function. After pre-processing the crack images, the CNN model is trained and tested for crack classification. The simulation evaluation of the proposed model is done using various performance metrics. The result shows that the proposed model achieves high accuracy (0.99833)

over the existing CNN algorithm. Further, future work is defined to enhance the proposed crack classification model.

- In the proposed model, the same dataset is used for training and testing purposes. In the future, data augmentation will be performed to evaluate the robustness of the model.
- In the future, real-time datasets will be tested in place of standard datasets for validation purposes.
- In the pre-processing stage, a single objective function is used to determine the optimal parameter values of the enhancement and k-mean clustering methods. In the future, multi-objective functions will be designed.

References

- [1] D. Ai, G. Jiang, S.-K. Lam, P. He, and C. Li, "Computer vision framework for crack detection of civil infrastructure—A review," *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105478, Jan. 2023, doi: 10.1016/j.engappai.2022.105478. [Online]. Available: <http://dx.doi.org/10.1016/j.engappai.2022.105478>
- [2] 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, p. 103989, Jan. 2022, doi: 10.1016/j.autcon.2021.103989. [Online]. Available: <http://dx.doi.org/10.1016/j.autcon.2021.103989>
- [3] O'Brien, Darragh, John Andrew Osborne, Eliseo Perez-Duenas, Roddy Cunningham, and Zili Li., "Automated crack classification for the CERN underground tunnel infrastructure using deep learning," *Tunnelling and Underground Space Technology*, vol. 131, pp. 104668, 2023.
- [4] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, and X. Chen, "Automatic Bridge Crack Detection Using a Convolutional Neural Network," *Applied Sciences*, vol. 9, no. 14, p. 2867, Jul. 2019, doi: 10.3390/app9142867. [Online]. Available: <http://dx.doi.org/10.3390/app9142867>
- [5] K. C. Laxman, N. Tabassum, L. Ai, C. Cole, and P. Ziehl, "Automated crack detection and crack depth prediction for reinforced concrete structures using deep learning," *Construction and Building Materials*, vol. 370, p. 130709, Mar. 2023, doi: 10.1016/j.conbuildmat.2023.130709. [Online]. Available: <http://dx.doi.org/10.1016/j.conbuildmat.2023.130709>
- [6] Sukh Sehaj Singh, Rohit Sachdeva, and Rajeev Sharma, "An integrated technique for underwater image enhancement: Color Correction and Dehazing," *Advances in Mathematics: Scientific Journal*, vol. 9, no.6, pp. 3865–3877, 2020.
- [7] F. Cui, Z. Li, and L. Yao, "Images Crack Detection Technology based on Improved K-means Algorithm," *Journal of Multimedia*, vol. 9, no. 6, Jun. 2014, doi: 10.4304/jmm.9.6.822-828. [Online]. Available: <http://dx.doi.org/10.4304/jmm.9.6.822-828>
- [8] R. A. Zitar, M. A. Al-Betar, M. A. Awadallah, I. A. Doush, and K. Assaleh, "An Intensive and Comprehensive Overview of JAYA Algorithm, its Versions and Applications," *Archives of Computational Methods in Engineering*, vol. 29, no. 2, pp. 763–792, May 2021, doi: 10.1007/s11831-021-09585-8. [Online]. Available: <http://dx.doi.org/10.1007/s11831-021-09585-8>
- [9] B. Kim, N. Yuvaraj, K. R. Sri Preethaa, and R. Arun Pandian, "Surface crack detection using deep learning with shallow CNN architecture for enhanced computation," *Neural Computing and Applications*, vol. 33, no. 15, pp. 9289–9305, Jan. 2021, doi: 10.1007/s00521-021-05690-8. [Online]. Available: <http://dx.doi.org/10.1007/s00521-021-05690-8>
- [10] "Concrete Crack Images for Classification," [www.kaggle.com](https://www.kaggle.com/datasets/arnavr10880/concrete-crack-images-for-classification). <https://www.kaggle.com/datasets/arnavr10880/concrete-crack-images-for-classification>
- [11] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, and X. Chen, "Automatic Bridge Crack Detection Using a Convolutional Neural Network," *Applied Sciences*, vol. 9, no. 14, p. 2867, Jul. 2019, doi: 10.3390/app9142867. [Online]. Available: <http://dx.doi.org/10.3390/app9142867>