buildings

MDPI

Article

Automated Surface Crack Identification of Reinforced Concrete Members Using an Improved YOLOv4-Tiny-Based Crack Detection Model

Sofía Rajesh [1], K. S. Jinesh Babu [1,*], M. Chengathir Selvi [2] and M. Chellapandian [1]

[1] Department of Civil Engineering, Mepco Schlenk Engineering College, Sivakasi 626005, India; sofiarajesh@mepcoeng.ac.in (S.R.); chellapandian@mepcoeng.ac.in (M.C.)
[2] Department of Computer Science Engineering, Mepco Schlenk Engineering College, Sivakasi 626005, India; chengathir@mepcoeng.ac.in
* Correspondence: ksjinesh@mepcoeng.ac.in

Abstract: In recent times, the deployment of advanced structural health monitoring techniques has increased due to the aging infrastructural elements. This paper employed an enhanced You Only Look Once (YOLO) v4-tiny algorithm, based on the Crack Detection Model (CDM), to accurately identify and classify crack types in reinforced concrete (RC) members. YOLOv4-tiny is faster and more efficient than its predecessors, offering real-time detection with reduced computational complexity. Despite its smaller size, it maintains competitive accuracy, making it ideal for applications requiring high-speed processing on resource-limited devices. First, an extensive experimental program was conducted by testing full-scale RC members under different shear span (a) to depth ratios to achieve flexural and shear dominant failure modes. The digital images captured from the failure of RC beams were analyzed using the CDM of the YOLOv4-tiny algorithm. Results reveal the accurate identification of cracks formed along the depth of the beam at different stages of loading. Moreover, the confidence score attained for all the test samples was more than 95%, which indicates the accuracy of the developed model in capturing the types of cracks in the RC beam. The outcomes of the proposed work encourage the use of a developed CDM algorithm in real-time crack detection analysis of critical infrastructural elements.

Keywords: crack detection; crack classification; crack count; object detection; YOLOv4-tiny

check for updates

Citation: Rajesh, S.; Jinesh Babu, K.S.; Chengathir Selvi, M.; Chellapandian, M. Automated Surface Crack Identification of Reinforced Concrete Members Using an Improved YOLOv4-Tiny-Based Crack Detection Model. *Buildings* **2024**, *14*, 3402. https://doi.org/10.3390/buildings14113402

Academic Editor: Hsuan-Teh Hu

Received: 13 September 2024
Revised: 12 October 2024
Accepted: 22 October 2024
Published: 26 October 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Due to the rapid boom in the infrastructure development of countries like India, the number of civil structural assets such as buildings, bridges, dams, tunnels, etc. has significantly risen. Reinforced concrete (RC), being the primary construction material, is subjected to different types of stresses during its lifespan mainly due to wind load, temperature variations, corrosion, etc. These stresses could result in severe damage to RC structures and even failures if left unnoticed over some time [1,2]. Hence, it is essential to periodically monitor the performance of existing structures so that the necessary interventions can be made at the correct time [3,4]. In general, direct periodical inspections are performed to monitor the progress of surface cracks in structural elements [5,6]. However, the use of manual inspection of cracks is arduous and subject to human error [7]. Also, it requires experience and specialist knowledge [8]. To tackle the issues in monitoring, automated monitoring and the and use of several sophisticated technologies have evolved in recent times [9,10]. Computer vision (CV) methods have appeared to be one of the potential solutions for the automated monitoring of cracks/distress in RC structures [3,9]. In CV, the digital images are processed to extract valuable information regarding the cracks [11]. CV can be broadly classified into two categories. One is the application of conventional image processing (IP) algorithms [11,12] and using a trained neural network (NN) to extract the features of the crack [13].

Conventional IP algorithms require extraction of the basic features of cracks like color, edges, and corners in an image [14–16]. These algorithms use features such as edge detection [15,16], segmentation [17], and filtering [14] to detect cracks in the captured digital images [18,19]. However, the use of conventional IP algorithms requires extensive human interventions for the selection of processing features in digital images. Therefore, the reliability and accuracy of the obtained results may vary based on the user's capability [1,20]. Machine learning (ML) algorithms have recently emerged for detecting the damages in concrete surfaces using better data extraction. Many conventional machine learning algorithms such as the decision tree [3], support vector machine (SVM) [1,3,21], and gradient boosting [22] have been used for the image data extraction. However, these ML algorithms are less efficient for real-time applications, as they mostly depend on the lower-level distress features present in the concrete surface [23]. Recently, the use of deep learning (DL) algorithms has evolved to be more effective in extracting the required details from the original image source. DL algorithms also make use of neural networks (NNs) for feature extraction. Still, they can analyze digital images or videos in a better way than the previously available ML algorithms do [12]. The deep learning algorithms possess the capability to obtain better results.

In DL-based CV methods, a convolutional neural network (CNN) that mimics the animal visual system is prominently used for analyzing digital images [23]. CNNs with different architectures have emerged, such as VGG [1], AlexNet [11,17,23], GoogLeNet [16], ResNet [16], and UNet [24–26]. All the aforementioned architectures consist of three basic layers: (a) convolution layer, (b) pooling layer, and (c) fully connected layers, as shown in Figure 1 [27]. The convolution layer is used for image processing utilizing filtration and de-ionization. Secondly, the pooling layer performs the threshold-setting process by morphological processing and binarization. In the last stage, the stacked inter-connected layers use classifiers for the detection of crack types and properties [28]. CNN does not rely on expert-set thresholds, as it can train itself adaptively through a feed-forward backpropagation algorithm. Moreover, the algorithm can effectively capture the grid topology of images and distinguish many object categories. It is highly robust to image variations for feature extraction [25,29]. As shown in Figure 2, CNN-based DL algorithms can be either a two-stage detector or a single-stage detector [30]. Both these detectors consist of the following important stages: namely, (a) object localization, (b) classification, and (c) detection. In the first form, the specific object in the form of a video (or) image is pinpointed to a location called the region of interest (ROI), which separates the specific object from the entire image. In the second stage, the labeling of specific objects present in the image is performed, which is termed classification. In the third stage, the first two steps (i.e., localization and classification) are combined to determine and locate the multiple numbers of similar objects present in the source file [17]. As seen in Figure 2a, a two-stage object detector takes two steps to localize and classify the object. It first generates the ROI, i.e., the object-containing area, and it predicts bounding boxes (BBs) that better fit and classify the object. The prominent two-stage detectors are region-based convolutional neural networks (R-CNNs) [17], Fast R-CNN [17] and Faster R-CNN [31,32]. Though the use of two-stage detectors is accurate in extracting the required information from the source image, their processing speed is slow [33].

In single-stage detectors, the ROI generation step is eliminated, i.e., both the object localization and classification stages are carried out in a single step called object detection. The single-stage detectors link the image pixels to bounding box (BB) coordinates and class probabilities in one step. Because of this, they are faster than two-stage detectors [31]. The prominent single-stage detectors are you only look once (YOLO) and single-shot multi-box detectors (SSD) [17,30,31]. Among the aforementioned single-stage detectors, YOLO performs better in various ways [33]. Du et al. [34] developed an algorithm for clarifying and detecting distress in pavements. They concluded that the proposed YOLO approach was able to automatically detect pavement distress in an accurate sense without any requirement for manual extraction of features from the source image. Similarly. Li

et al. [35] proposed the YOLO v3-lite algorithm, which is a lightweight crack detection network based on depth-wise separable convolutions. They assessed that the use of the proposed YOLO-v3 lite algorithm was able to reduce the computation time by more than 50% when compared to the conventional YOLO-v3 algorithm due to the use of pyramid network and BB regression. Similarly, several previous studies have used the YOLO algorithm for successfully detecting distress in structural elements [36–42].

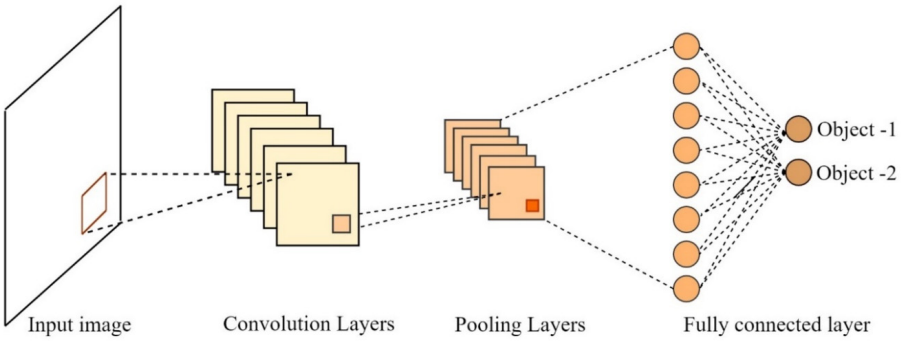


Figure 1. Layers in a convolution neural network.

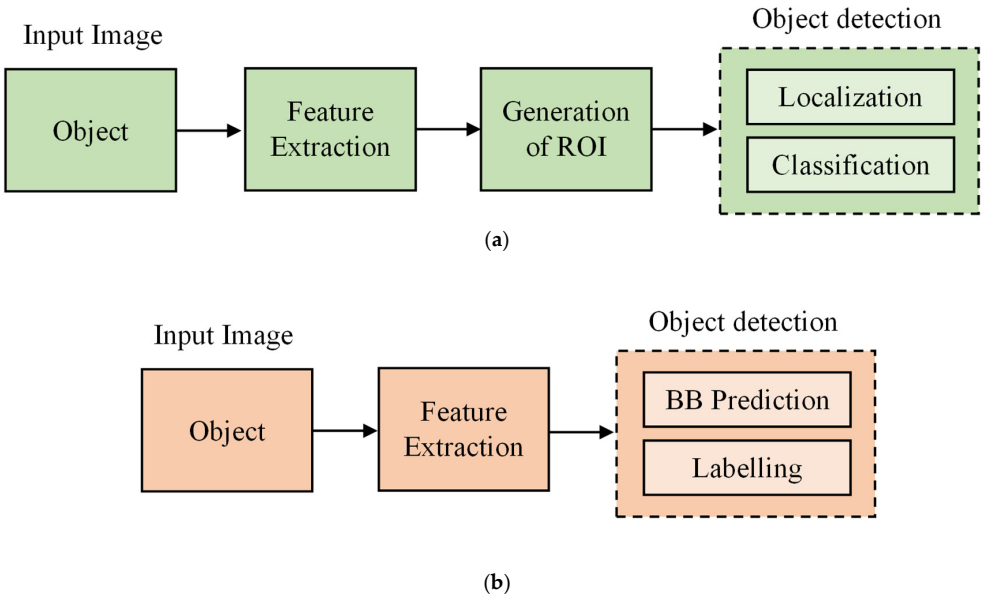


Figure 2. (**a**) Two-stage object detector. (**b**) Single-stage object detector.

However, adopting such algorithms for detecting the number of cracks in concrete structures and clarifying their type has not been explored so far. This paper proposes a novel crack detection model (CDM) using the YOLOv4-tiny algorithm, which uses a CNN-based single-stage detector, to detect the flexure and shear cracks present in the images of full-scale reinforced concrete (RC) beams. The proposed CDM not only detects the cracks but also provides the number of cracks and suitably classifies them. YOLOv4-tiny, one of the compact versions of the YOLO series, consumes less work space and processing time. For the training and testing of the CDM, a custom dataset that includes flexure and shear cracks was developed by conducting an experimental study on RC beam elements by varying the shear span to the effective depth ratio (a/d ratio). The outcomes of the proposed work will be highly beneficial in the use of the proposed YOLO model for practical structural health monitoring of artificial infrastructural assets, which will require immediate attention to function.

The following are the significant research contributions made through the present work:

(a) Automated classification and detection of flexure/shear cracks in RC beam elements using the modified YOLOv4-tiny model, overcoming the limitations seen in traditional methods.
(b) Quantifying the total flexure/shear cracks present in an image, which gives an option for further action on the structure.

2. Experimental Corroboration

The occurrence of cracks like flexure and shear in concrete beam elements is a defect that cannot be avoided but can be predicted at its onset so that catastrophic failures in important structures can be avoided. Studying the types of cracks and their locations is important in structural elements, considering their life safety. In general, flexural cracks develop in the middle span of the beam because of bending and are ductile. The flexure crack appears vertical (90°) and forms across the longitudinal section. On the other hand, shear cracks usually form near the supports at an angle of 45°, where shear stress is maximum. These cracks are brittle and are to be addressed with immediate effect. Hence, a detailed full-scale experimental study on RC beams is conducted to validate and demonstrate the feasibility of the image processing technique using the YOLOv4-tiny deep learning method. The digital images captured during the testing are used to create the dataset required for the training and testing of the proposed CDM. Out of the total images captured, 90% of the images are grouped for training the developed CDM, and the remaining ones are used for testing the accuracy of the model and validation. The methodology used for crack detection using the YOLOv4-tiny model is shown in Figure 3.

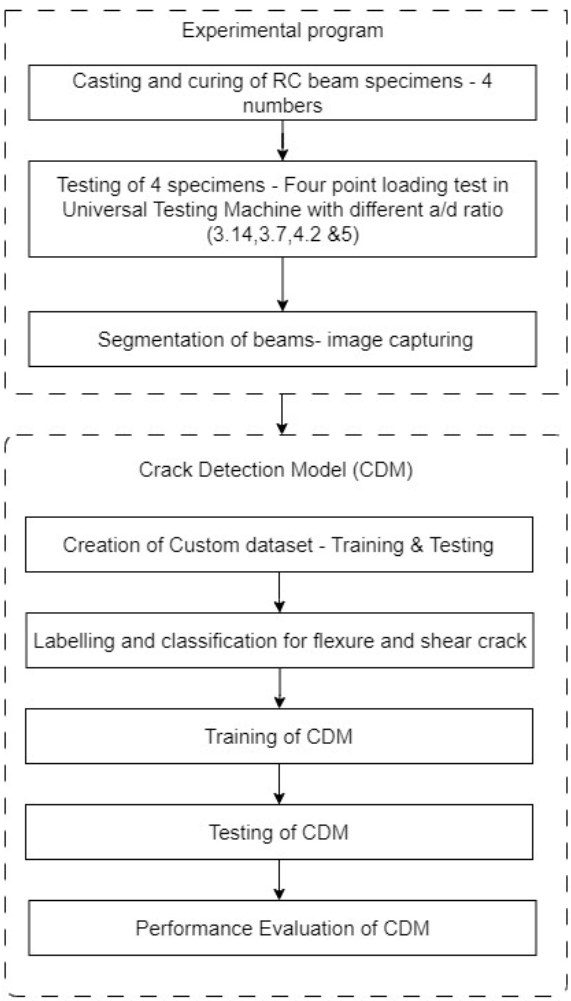


Figure 3. Methodology used for crack detection using the YOLOv4-tiny model.

Experimental Study and Dataset Preparation

The experimental part includes casting and testing of four full-scale reinforced concrete beams of length 2200 mm, as shown in Figure 4. The beams possess sectional dimensions of 100 mm (width) × 200 mm (depth). The concrete mix is designed to provide a 28-day compressive strength of 30 N/mm^2. The design mix used has a composition of 443 kg/m^3 of cement, 584 kg/m^3 of fine aggregate, 1158 kg/m^3 of coarse aggregate, and 186 kg/m^3 of water. The physical properties of the materials are tested, and the values are verified against the specifications given by the manufacturer. The specific gravity of cement, fine aggregate, and coarse aggregate were found to be 3.10, 2.75, and 2.57, respectively. The fineness modulus of fine aggregate and coarse aggregate were 2.5%, and 3.3%, respectively. Moreover, the water absorption of fine aggregate and coarse aggregates was 1.0% and 0.64%, respectively. The beams were reinforced with 3 numbers of 12 mm diameter in the tension reinforcement, which corresponds to a reinforcement ratio of 1.70%. Moreover, two bars of 10 mm diameter were provided on the compression side of the beam. To prevent the shear dominant failure in all the test specimens, 8 mm diameter two-legged vertical stirrups at 150 mm center-to-center spacing are provided. The tensile characteristics of steel reinforcement embedded in the RC beam elements, namely yield strength, elastic modulus, and Poisson's ratio, are 510 MPa, 205 GPa, and 0.28, respectively.

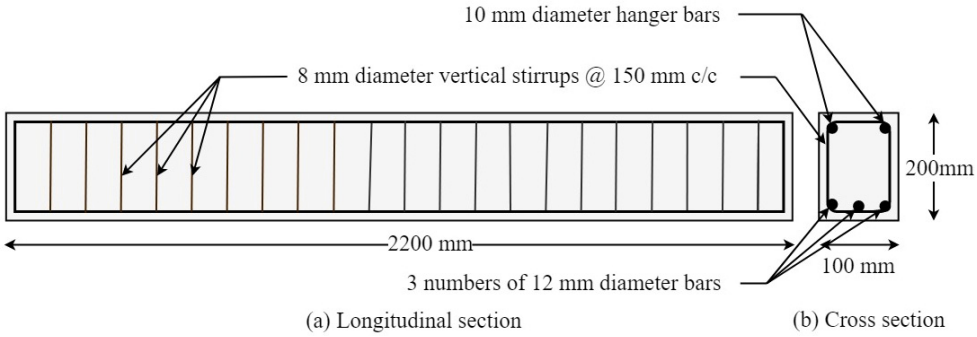


Figure 4. Sectional view of the test specimen used.

The concrete required for casting the full-scale beams was prepared in a machine mixer. After providing sufficient mixing time, the prepared concrete mix was poured into a beam mold in three layers, and each layer was compacted well using a needle vibrator. The beams are demolded after 24 h, and water cured for 28 days. In addition to the full-scale beams, auxiliary specimens such as cubes, cylinders, and prisms were cast to understand the mechanical characteristics of the developed concrete mix. The mean 28-day compressive strength of the concrete tested using cubes was found to be 40.4 MPa, with a standard deviation of 2.19 MPa. Similarly, the split tensile strength obtained by testing cylinders was 4.1 MPa, and the corresponding standard deviation was 0.22 MPa. As per the Indian standards, a minimum of three specimens were tested to find the average compressive strength and split tensile strength.

As shown in Table 1, the beam consists of four specimen series, B1, B2, B3, and B4, which correspond to different a/d ratios. The values of the a/d ratio are selected to understand both flexure and shear load combinations as reported by several previous studies [43–48]. Providing a longer a/d ratio in beam specimens will produce prominent flexure cracks due to the predominant bending failure [43,44]. On the other hand, beam specimens with a smaller a/d ratio will produce prominent shear cracks [45–48]. Hence, the present study aims to understand the effectiveness of the developed YOLOv4-tiny model by predicting the different types of cracks in RC beams by adjusting the a/d ratio. As a result, the flexure crack, shear crack, or a combination of flexure and shear cracks can be generated in the experiments. During the image capturing, the entire length of the beam of 2200 mm is divided into different segments where each segment represents portions such as CBZ, shear span, etc. For each segment, images were taken at different intervals of

loading, such as first crack, yielding of steel reinforcement, ultimate, etc. Hence, the image corresponding to crack initiation and propagation/branching can be easily identified.

Table 1. Experimental results of beam specimens.

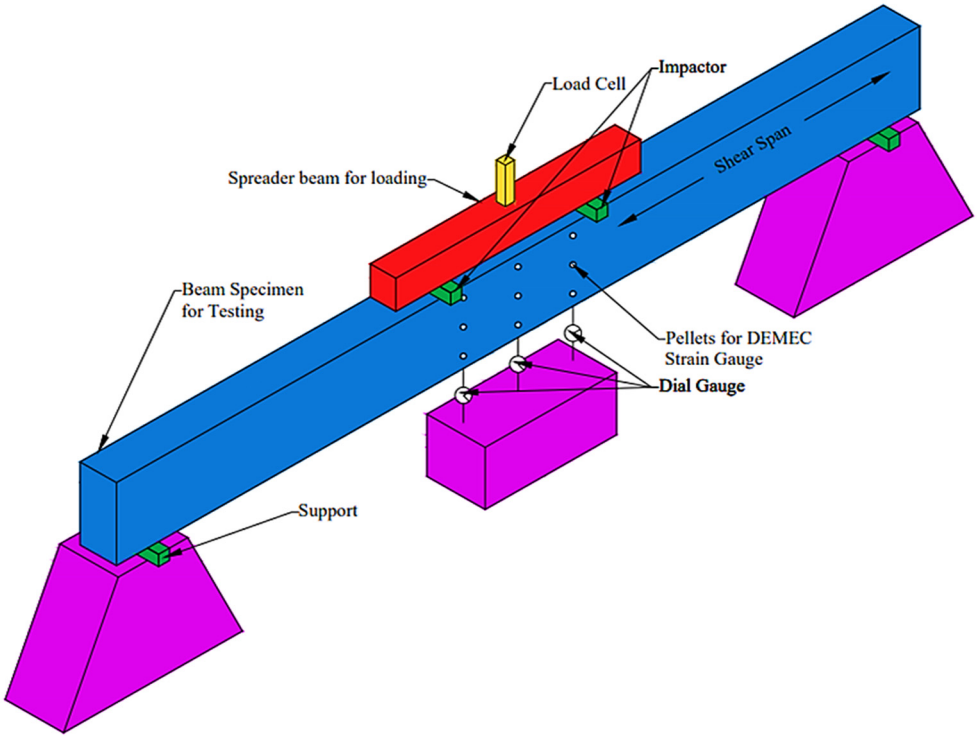
S. No.	Beam Specimen	a/d Ratio	First Crack			At Ultimate Load		Failure Modes
			Load (kN)	Displ. (mm)	Stiffness (kN/mm)	Load (kN)	Displ. (mm)	
1	B1	5.00	40.0	1.18	33.89	131.21	10.92	Flexure
2	B2	4.20	20.0	2.11	9.45	166.71	21.75	Flexure and Shear
3	B3	3.70	30.0	1.18	25.42	167.50	24.13	Flexure and Shear
4	B4	3.14	20.0	1.33	15.04	128.96	12.8	Shear

Figure 5a shows a schematic representation of the RC beams tested in a four-point bending configuration. Figure 5b shows the actual test setup used to determine the overall behavior of RC beams. The loading from the hydraulic jack is applied to the test specimen using a spreader beam setup and then further distributed using the two-point rollers. The beams are tested using the simply supported boundary condition. Figure 5a shows the presence of roller support at one end and the hinge support (roller welded to restrain lateral movement) at the other end. The beam is instrumented to measure the displacement and surface strains in addition to the digital image correlation analysis. Three dial gauges of 100 mm stroke were used at the locations such as mid-span and below each load point to measure the beam's vertical displacement. Moreover, the back side face of the beam is installed with pellets (Figure 5a) to measure the top, middle and bottom strains using demountable strain gauges. The loading is performed using the load-controlled procedure with a stepwise increment of 2 kN. The cracks were marked on the other face so that the initiation and propagation can be easily understood.

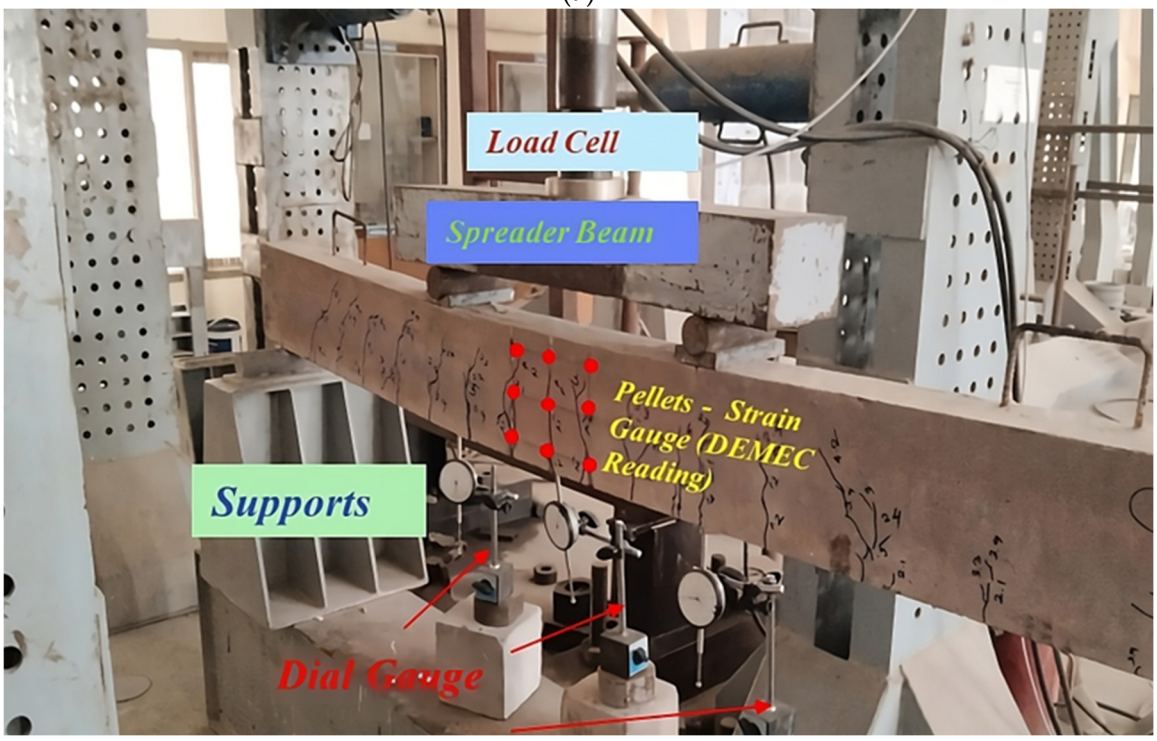
Table 1 depicts the test results for RC beams tested at varying shear span to effective depth ratios. Moreover, Figure 6 depicts the complete relationship between displacement and load of RC beams tested at different a/d ratios. For the beam B1 tested at an a/d ratio of 5.0, the first crack appeared at a load of 40 kN corresponding to a vertical displacement of 1.18 mm. The corresponding value of initial stiffness was calculated as 33.89 kN/mm. On further increase in load, the beam showed multiple formations of the flexure cracks and reached a peak corresponding to a load and vertical displacement of 131.21 kN and 10.92 mm. After this, the load started to reduce to the minimum value, and further data were not captured due to the load-controlled mode of testing. In the case of beam B2, the first crack appeared at a load and stiffness value of 20 kN and 9.45 kN/mm, respectively. Once the load reached closer to the peak value, some flexure cracks close to the support were converted to flexure-shear cracks, which started propagating towards the loading point and the support edge. The beam showed the peak load and corresponding displacement values of 166.7 kN and 21.8 mm, respectively. When compared to the beam at a high shear span ratio, beam B2 showed an increase in load by 21.2%, which can be attributed to the reduced bending effect under low a/d ratio. Similarly, the beams tested at an a/d ratio of 3.7 showed an increase in peak load by 27.7% when compared to beam B1 (i.e., a/d = 5.0). However, the beams tested at a low a/d ratio of 3.14 failed at a lower load of 128.96 kN due to the rapid initiation and propagation of the shear crack. This could also be due to the shorter distance between the support and load point at a/d of 3.14, as a result of which the inclined crack propagation was rapid, leading to a lower peak load.

In summary, only flexure cracks are formed in B1 due to the high a/d ratio value. On the contrary, only shear cracks are formed in B4, for which the a/d ratio is low. For beams B2 and B3, a combination of flexure and shear cracks are formed, and the a/d ratios are medium. These galleries of crack images captured using the digital camera are used as a custom dataset for training and testing the YOLOv4-tiny CBM. In Figure 6b,c, the flexure

and shear failure modes obtained for beams with high and low a/d ratios are presented. Since the beams tested under a/d of 4.2 and 3.14 are governed by flexure and shear failure modes, respectively, they are not presented for brevity.

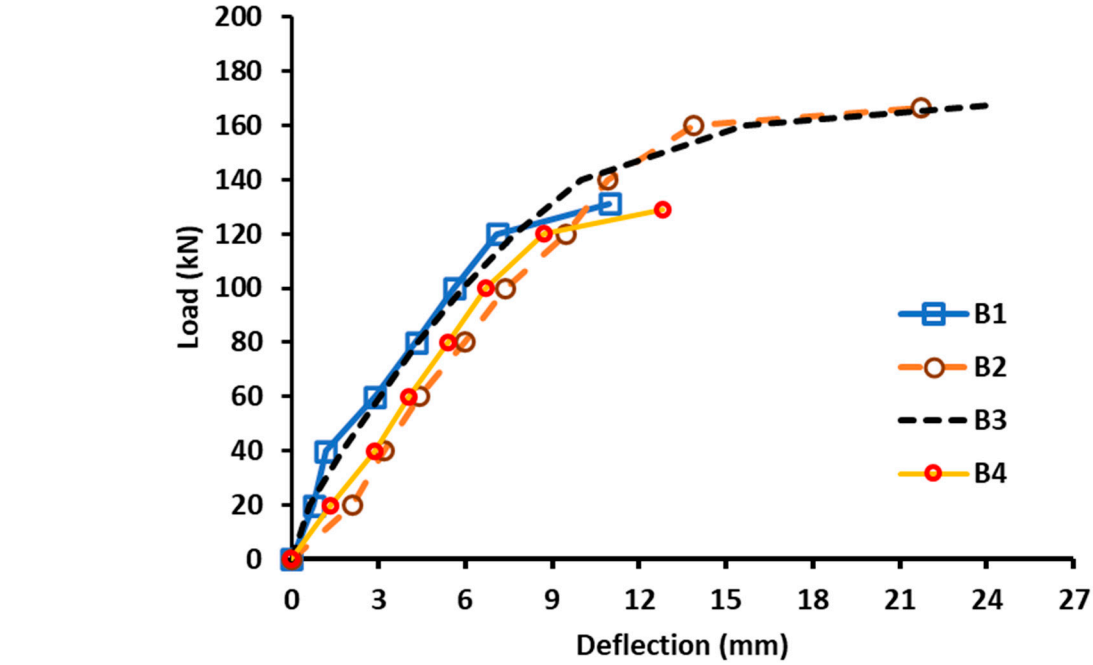


(**a**)

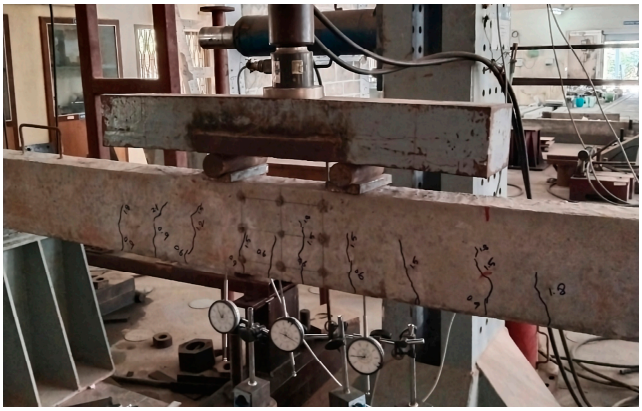


(**b**)

Figure 5. (**a**) Schematic representation of four-point loading test setup used. (**b**) Actual experimental setup for RC beams.



(**a**)



(**b**)



(**c**)

Figure 6. Overall behavior and failure modes of RC beams at different a/d ratios. (**a**) Load versus displacement behavior of RC beams. (**b**) Failure mode at a/d = 5.0. (**c**) Failure mode at a/d = 3.7.

3. Object Detection Using YOLO

The use of the YOLO algorithm for localization and classification of the objects simultaneously in the image has been successful in the past [36–42]. The entire image is divided into regions in the YOLO algorithm to make it suitable for training and testing. Being a single-stage object detector, the YOLO algorithm divides the entire image into regions, predicts the ROIs and computes class probabilities for each ROI in one step. Moreover, it identifies several ROIs and calculates the weighted class probability for each box. The first version of the YOLO architecture was introduced in the year 2016 [41], and then successively updated with two more versions, YOLOv2 and YOLOv3, in the years 2017 and 2018, respectively [42]. The next version, YOLOv4, was launched in the year 2020. YOLOv4-tiny is a lightweight and scaled-down version of the YOLOv4 architecture. Compared to the parent version, YOLOv4, which has more than a hundred layers, the tiny version has only 29 layers starting from zero. Also, the tiny architecture consists of only two YOLO heads for predicting the ROI instead of three as found in the parent version. This size reduction helps in bringing down the computational complexity and speeds up the training or testing processes. As a result of these reductions, YOLOv4-tiny significantly reduces the computational time but without much compromise in accuracy for several

real-time applications [42]. The YOLOv4-tiny algorithm shown in Figure 7 consists of steps such as backbone, neck and head for extracting the required outputs. The backbone of the YOLOv4-tiny algorithm is a DSP Darknet53, which performs the function of the extraction from the original images using a pre-trained CNN [42].

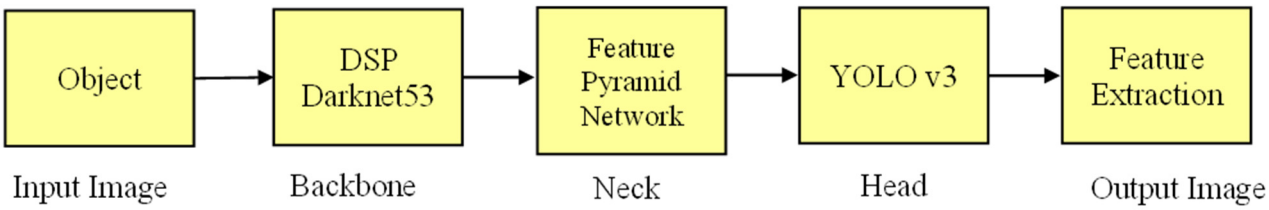


Figure 7. YOLOv4-tiny architecture.

3.1. YOLOv4-Tiny Algorithm

The YOLOv4-tiny network has a neck where feature aggregation takes place. At the neck, FPN performs the feature aggregation and produces two feature maps of sizes 26 × 26 and 13 × 13; they are forwarded to the YOLO heads for further processing. The YOLOv4-tiny network also has YOLOv3 as its head, like its parent version. Among the two heads, one receives the feature map of size 13 × 13 for the detection of large objects with minimal computational cost. The other one receives the feature map of size 26 × 26 for the precise detection of relatively small objects. The following are the assumptions made while developing the model.

- Clear visibility of cracks in the images selected for analysis.
- A dataset with flexure and shear crack of RC members with consistent image quality.
- The image will possess constant environmental factors, such as lighting.

These assumptions guide the model's development for effective detection under controlled conditions.

As shown in Figure 8, the backbone stage in the YOLOv4-tiny algorithm consists of numerous components that are directly and indirectly connected: an input layer, 18 convolution layers, nine routes, and three max-pooling layers. The input layer feeds the image of size 416 × 416 for training. The feature extraction of the convolution layers starts with that input image. The successive convolution layers extract a substantial number of learnable features from the input image. The convolution layers have a set of receptive filters that switch between 3 × 3 and 1 × 1 kernel sizes. Each convolution layer is connected with a batch normalization layer and a leaky rectified linear (Leaky-ReLU) activation layer to enhance computational efficiency. Routes are another part of the backbone stage that helps in recovering fine-grained characteristics of the image from the earlier convolution layers. In this way, it is possible to retain high-resolution quality even after multiple down samplings of an image. The backbone has two max pooling layers for down sampling the feature maps. The max-pool layer follows a convolution layer and uses the 2 × 2 kernel size and a stride of 2. This in turn allows the reduction of feature map size by 50%; it reduces the size of the feature map in half and cuts down the computational cost significantly.

3.2. Crack Detection Model

The YOLOv4-tiny algorithm is used to predict the types of cracks (i.e., flexure (or) shear) and the total number of cracks for the RC beams tested as a part of this test program. First, a labeled custom dataset was prepared to customize and prepare necessary files and folders in G-Drive. This step will be sequentially followed by building the darknet, grouping of datasets, and training the CDM. A piece of code is developed and added to the customized CDM of the YOLOv4-tiny algorithm to fetch and display the total number of cracks as well as the number of cracks in individual classes, i.e., flexure and shear, on screen and in a text file. Finally, the performance of the CDM during the training and testing phases is evaluated using appropriate evaluation metrics.

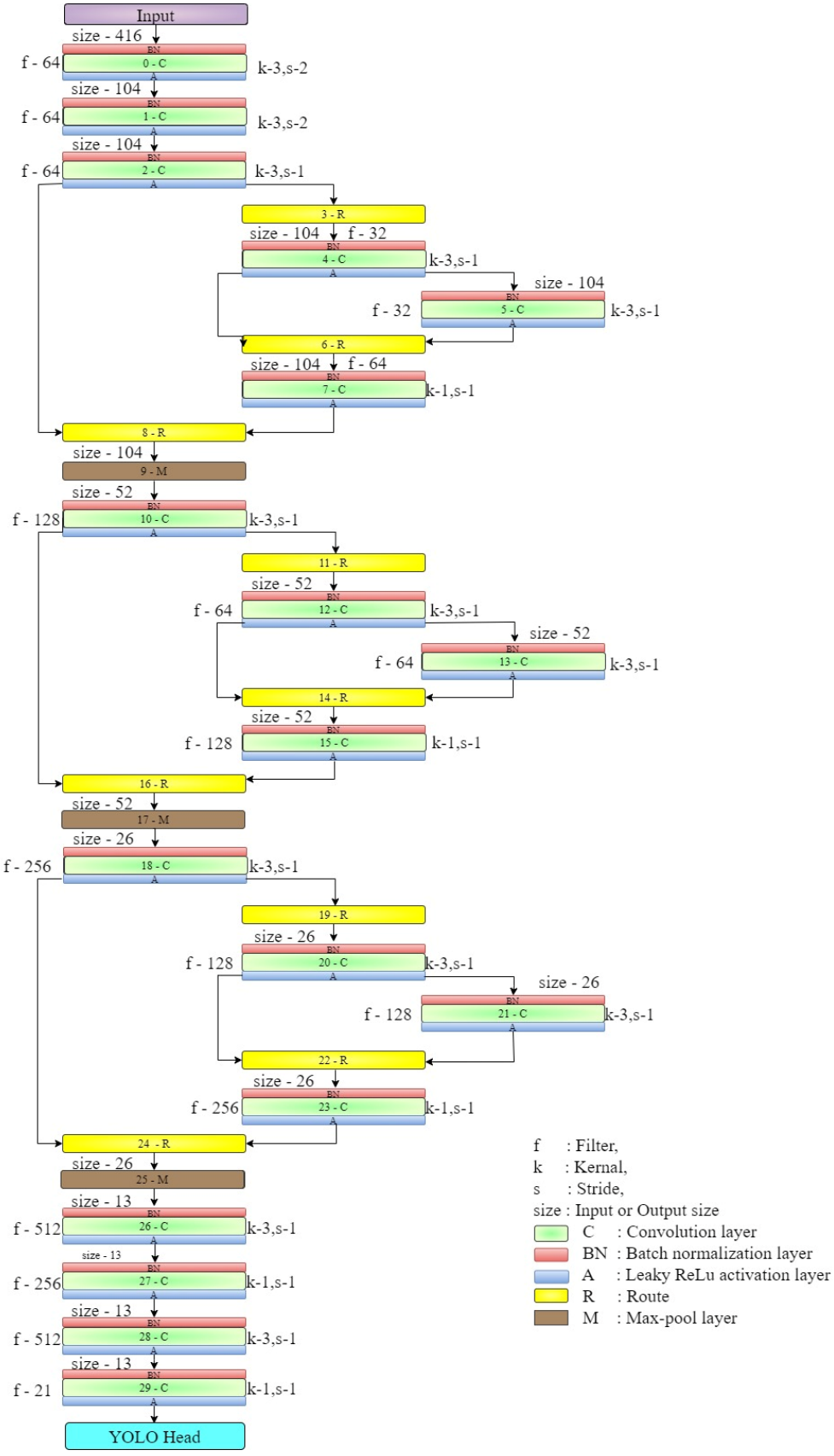


Figure 8. Convolution layers of YOLOv4-tiny.

3.2.1. Creation of Labeled Custom Dataset

A label creation tool, termed labellmg v1.8.2 [49], is used to prepare the custom dataset. The inputs for the label creation tool are the .jpg images, and the ground truth BBs are manually drawn for all the training images. At this step, each crack in an input image is labeled with its respective class, i.e., flexure or shear. The output from the label creation tool for each image is in the form of a text file that contains the coordinates of the BBs drawn, Figure 9. All the input images and their respective output files are placed in a compressed folder named obj.zip and saved in the local drive, Figure 10.

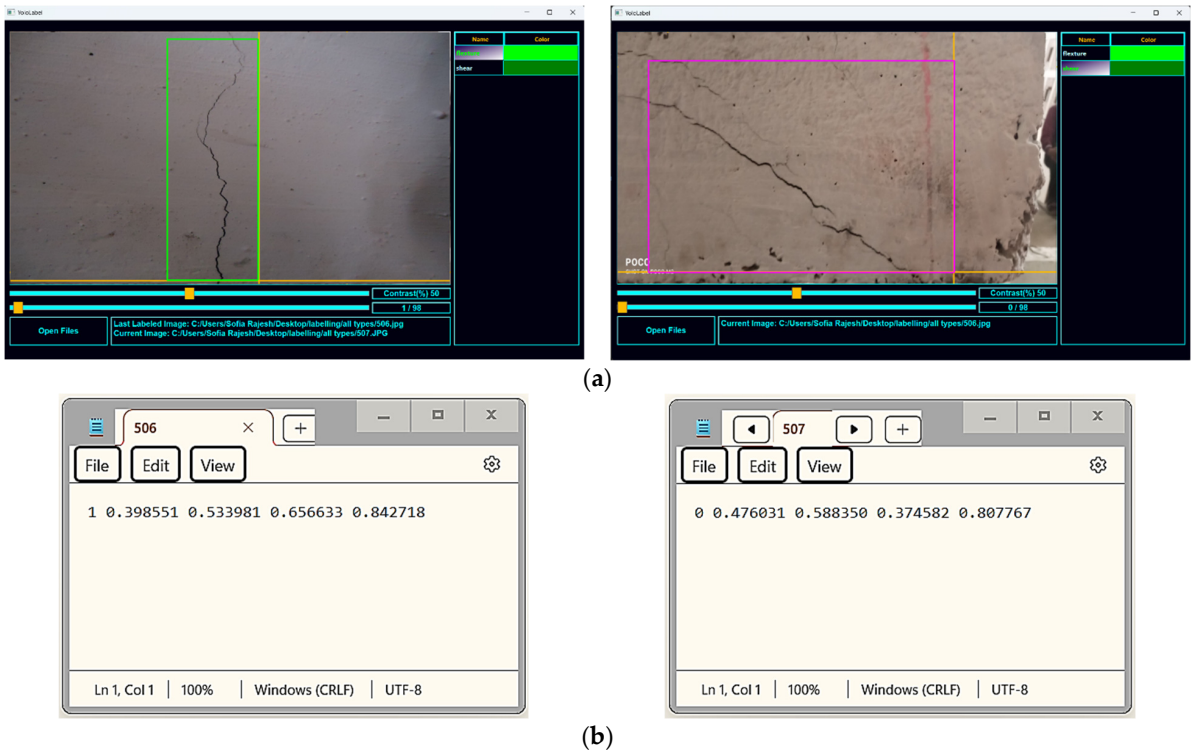


Figure 9. Creation of labeled custom dataset using labellmg. (**a**) Input images with ground truth BBs. (**b**) Output files with coordinates of the BBs.

3.2.2. Customization of the Configuration File

The configuration file of Yolov4-tiny, *yolov4-tiny-custom.cfg*, is downloaded from the GitHub repository and is customized, Figure 10. Several parameters, such as (a) the number of classes, (b) the number of iterations and (c) the step size, can be edited in the configuration file. The number of classes is set as 2 to represent the flexure and shear cracks. Moreover, the number of iterations is set as 6000 as per Equation (1). The steps are set as 4800, 5400 as per the thumb rule, i.e., steps are 80 to 90 percent of the maximum number of iterations.

$$\text{Number of iterations} = \max(6000, 2000 \times \text{number of classes}) \quad (1)$$

3.2.3. Preparation of Necessary Files and Folders in G-Drive

As shown in Figure 11, a folder named *yolov4-tiny-CDM* is created in the *MyDrive* of *G-Drive* with a subfolder named "*custom-weights*" and three files named "*group.py*", "*obj.names*" and "*obj.data*". The subfolder "*custom-weights*" is created to back up the weights that are generated during the training process of the CDM, Figures 10 and 11. A Python script, "*group.py*", is used for generating two files, namely "*train.txt*" and "*test.txt*", that are required for the training and testing processes, respectively. The file "*obj.names*" has the list of class names of the cracks, i.e., flexure and shear. The file "*obj.data*" holds the class

details and path details of folders and files that are necessary for the training and testing, as listed in Table 2. The compressed folder "*obj.zip*" and customized *yolov4-tiny-custom.cfg* configuration file, which are generated in the local drive, are also placed in the *yolov4-tiny-CDM* folder, Figures 10 and 11. After preparing and retrieving all the necessary folders and files in the *G Drive*, it is mounted on the Google-Colab virtual machine (VM) for training and testing, Figures 10 and 11. The *yolov4-tiny-CDM* folder will receive the "*crack_count.txt*" file that shows the total number of cracks as well as the number of flexure and shear cracks in an image after testing the image in Figures 10 and 11.

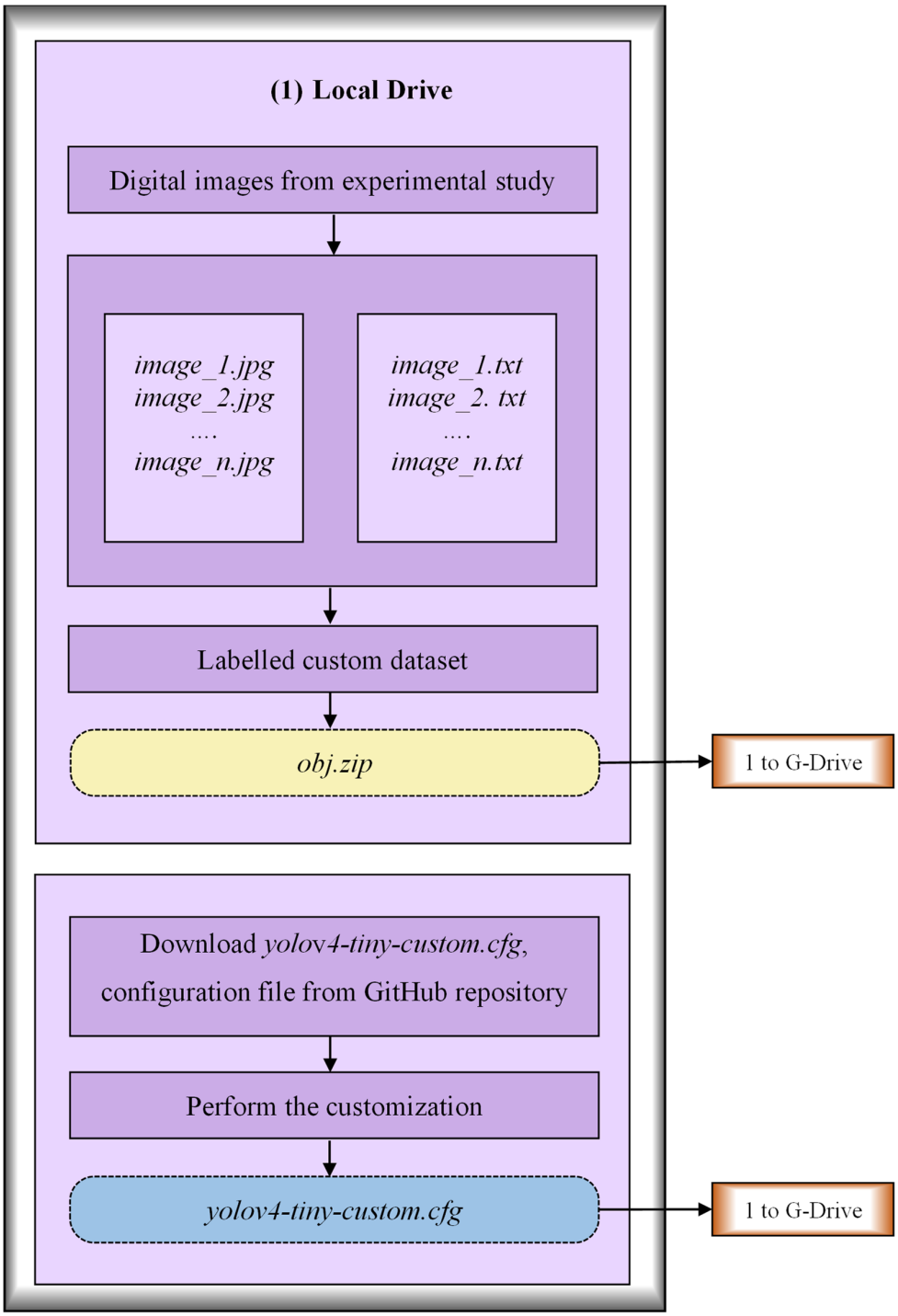


Figure 10. Customization of dataset and configuration file for Yolov4-tiny model.

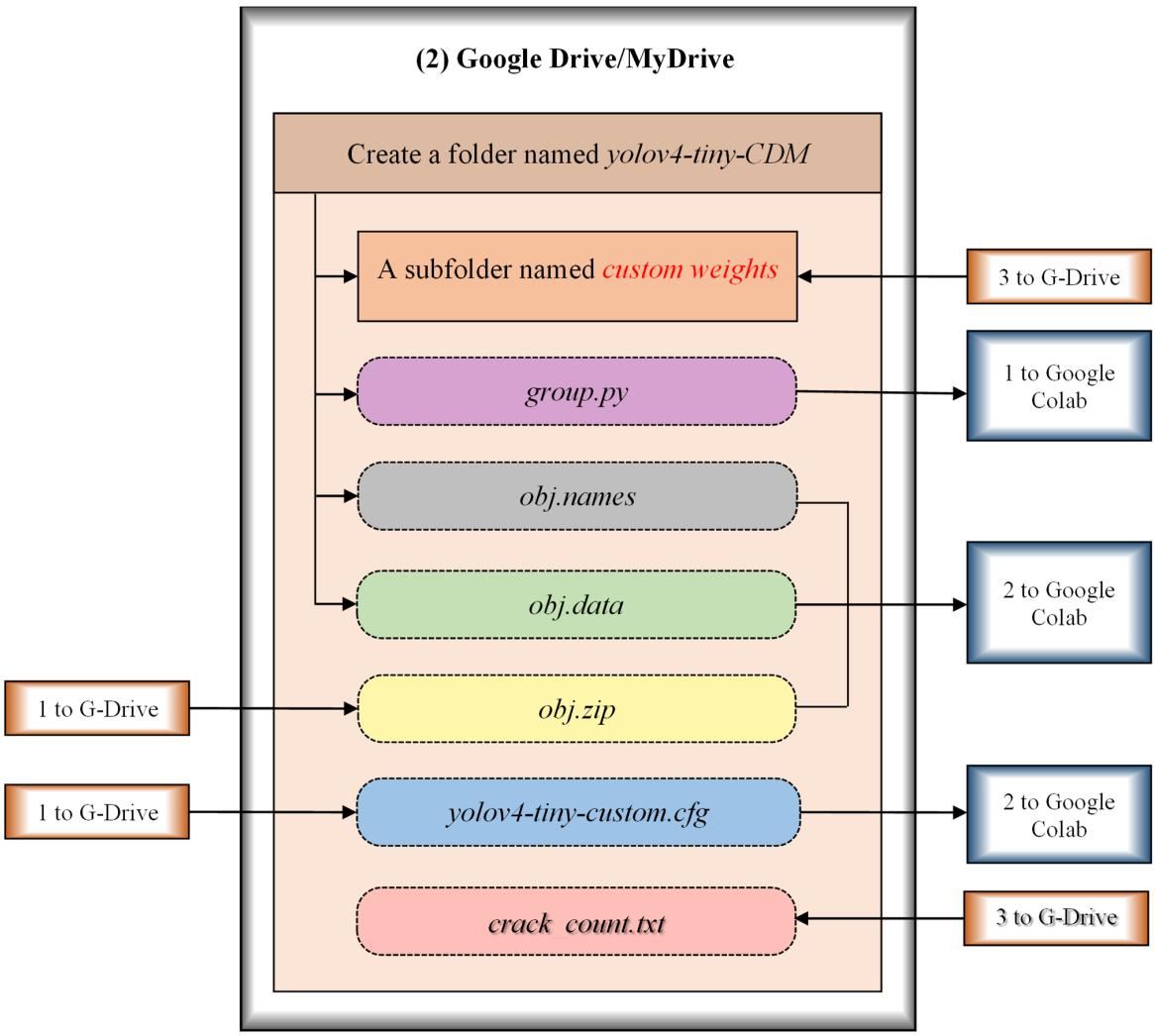


Figure 11. Creation of necessary folders and files in G-Drive.

Table 2. Details available in obj.data file.

Parameter	Value	Path
classes	2	-
train	-	data/train.txt
valid	-	data/test.txt
names	-	data/obj.names
backup	-	/mydrive/CDM/custom-weights

3.2.4. Building Darknet Directory and Grouping of Datasets

The Darknet GitHub repository is cloned on the Google-Colab VM, and the Darknet is built [50] (Figure 11). On Google-Colab VM, the Darknet is the current directory. It has several in-built folders, and among them, data and cfg are the folders that need modifications.

The Python script is transferred from the G-Drive to the data folder and executed (Figures 10 and 11). It generates two files, train.txt and test.txt, that contain the path of the training and testing images, respectively, and stores them in the data folder. The grouping of data for the training and testing is automized using group.py. However, this Python script can be avoided if the files train.txt and test.txt are prepared manually and uploaded in the data folder.

The files obj.names, obj.data and the folder obj.zip are also transferred from the G-Drive to the data folder. The customized configuration file, yolov4-tiny-custom.cfg, is transferred to the cfg folder (Figure 12).

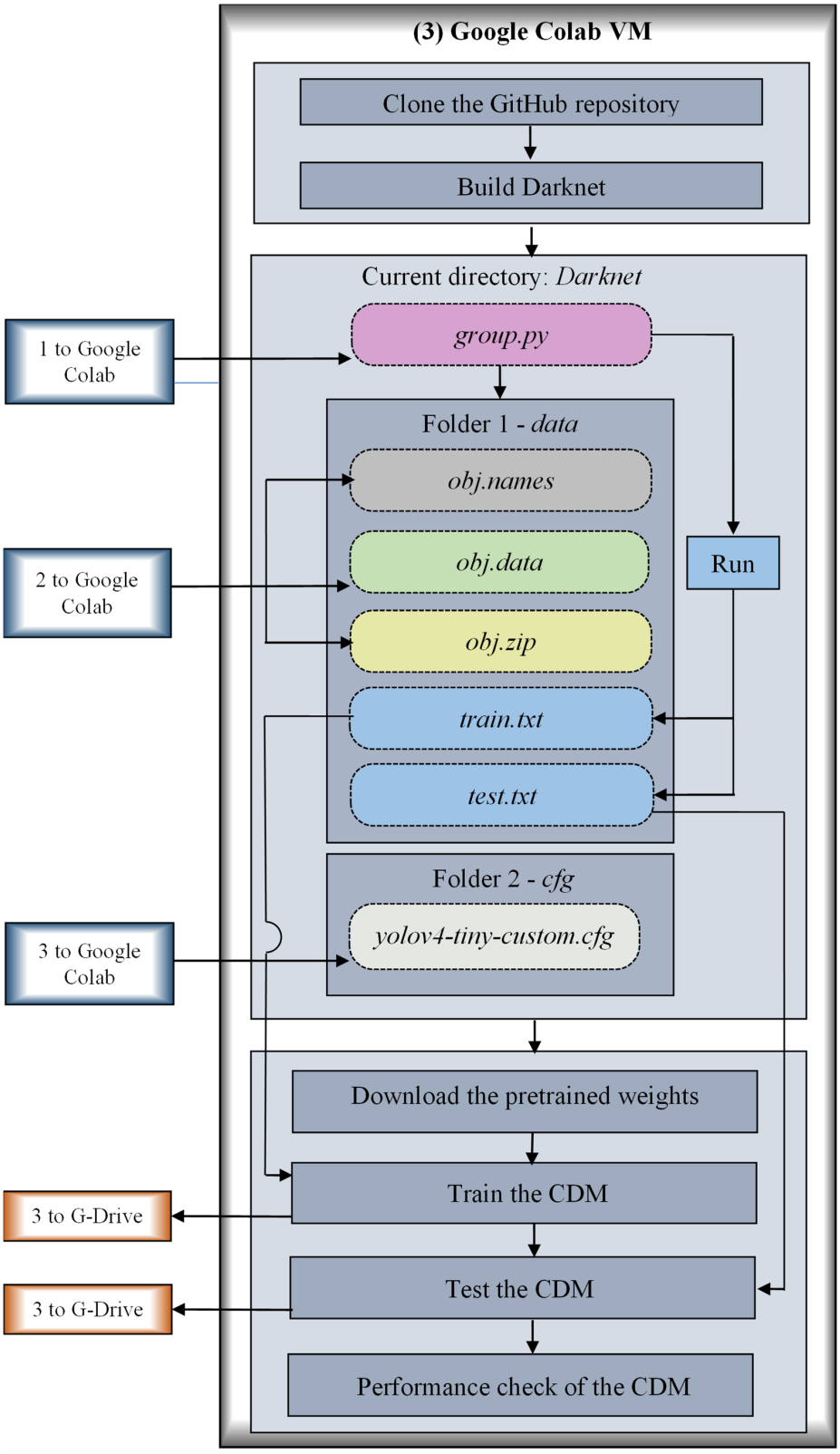


Figure 12. Training and testing of the CDM.

3.2.5. Training of the CDM

The backbone with the pre-trained weights of YOLOv4-tiny, "*yolov4-tiny.cov.29*", is downloaded from the GitHub repository to the *Darknet* directory. This backbone is used for the CDM and further training of the crack detection model by a transfer learning mechanism using the custom dataset. The transfer learning mechanism enhances the performance of YOLOv4-tiny in terms of accuracy, training efficiency, and generalization. It allows the model to learn the specific features of cracks in reinforced concrete while leveraging general visual features learned from large datasets. Moreover, this approach is particularly beneficial when data are limited or when rapid training is required for real-world crack detection tasks.

The training of CDM is continued using the transfer learning in YOLOv4-tiny until the maximum iteration limit is reached, i.e., 6000 iterations. During the training, weights are generated for every 1000 iterations, and they are saved in the sub-folder "*custom-weights*" created for the backup in G-Drive, Figures 10 and 11. The backup of weights in the G-Drive is useful if the training is interrupted in Google Colab VM before reaching the maximum iteration. Hence, the training continues from the last saved weights instead of starting from the beginning. It is worth noting that the number of iterations used for saving the weights need not be always 1000 and shall be customized to any value in the configuration file, "*yolov4-tiny-custom.cfg*".

4. Results and Discussions

The training and testing of the CDM are conducted on Google Colab VM. A typical statistical benchmark for evaluating the speed of the computer is obtained by measuring the billion floating-point operations per second (FLOPS), and it is found to be 6.789 FLOPS. The workspace allocation in the VM for training and testing is 26.22 MB and 72.42 MB, respectively.

4.1. Training of the CDM

The batch size, subdivision, learning rate, decay, and momentum are the hyper-parameters that can influence the training process. However, optimization to obtain the values for these parameters is not carried out exclusively, and the default values in the configuration file, *yolov4-tiny-custom.cfg*, are retained. The CDM is trained with a batch size of 64 and a subdivision of 16. The values for the learning rate, decay, and momentum are taken as 0.00261, 0.0005, and 0.9, respectively. The lower learning rate and decay and higher momentum help to escape from the local optima and can yield a near-optimal or even global optimal set of weights. The average loss curve can be considered as an indicator to ensure that YOLOv4-tiny has learned progressively and ended up with no overfitting issues. The average loss function is used to penalize the classification and bounding box errors [27]. For the YOLOv4-tiny-based CDM, the average loss decreases from 0.49 to 0.06. Initially, the loss rate is very steep because the classification and bounding box errors are more, Figure 13. With the further increase in the training module, the error in predicting the ROI is gradually reduced, and thereby, the slope of the curve becomes stable. When the CDM is completely trained for the custom dataset without overfitting, there is no significant scope for further improvement, and the loss curve is flattened.

4.2. Testing of the CDM

The trained CDM with the optimal set of weights is tested using the images obtained from full-scale tests. For each test image, the CDM detects the crack by predicting a BB around it and labeling the class of the crack as flexure or shear, as shown in Figure 14. After testing an image, the result can be viewed both on screen and in the *crack_count.txt* file, Figure 15. If an image has multiple cracks, each crack is enclosed by a BB with a label showing the class of the crack. From the results, it is inferred that the CDM has detected all the cracks in the test images and labeled them as flexure or shear precisely. Even though the training takes much time, the time required for testing is much less. The mean, median, and standard deviation of the detection time for each of the test images is found to be only

5.04 milliseconds, 4.95 milliseconds, and 0.27 milliseconds, respectively. The reduction in computational layers from YOLOv4 to YOLOv4-tiny results in a trade-off between speed and accuracy. YOLOv4-tiny is much faster and suitable for real-time applications but has lower bounding box precision and recall, especially for small, fine, or complex cracks. Strategies like transfer learning, data augmentation, and anchor box optimization can help mitigate some of these losses, making YOLOv4-tiny a viable option for efficient, real-time crack detection despite its reduced accuracy compared to YOLOv4.

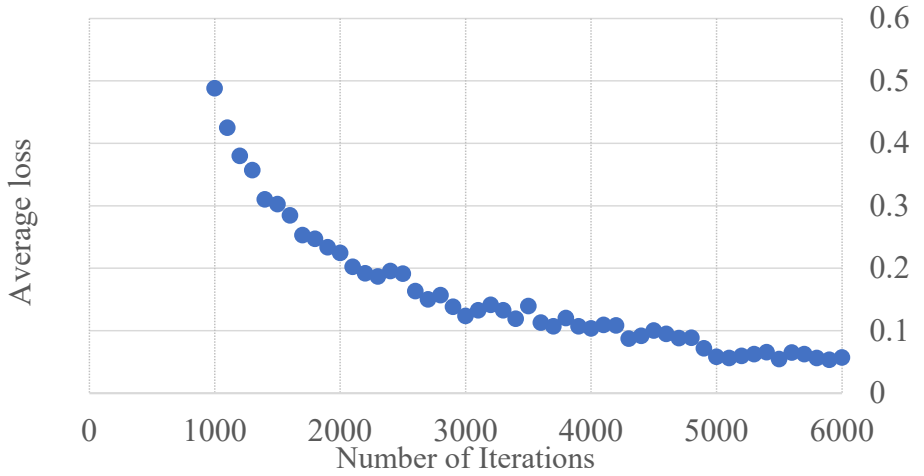


Figure 13. Relationship between average loss and the number of iterations.

To make the images suitable for training and testing, the entire length of the beam (L = 2200 mm) is divided into several segments such as constant moment zone (CMZ), shear span (a), etc. Hence, each beam contains separate segments which may contain few or many or no cracks. Similarly, the number of cracks also depends on the load value (i.e., first crack load or peak load) during which it was captured. In the constant moment zone, the beam had the formation of one flexure crack at the load when crack initiation just occurred. Similarly, at a higher load level, as shown in Test images 10–11 of Figure 14, the beam had formation of three flexure cracks in the constant moment zone.

4.3. Performance Evaluation of the CDM

For the performance analysis of the CDM during training, intersection over union (IOU), F1 score, precision (P), recall (R) and mean average precision (mAP) are considered. For the testing, a confidence score (CS) is considered.

4.3.1. Performance During Training

IOU is determined by the overlap between the ground truth and the predicted BBs, Equation (2). The threshold for IOU is taken as 0.5. The average IOU for the training images is found to be 82.93% for the set threshold, Figure 16.

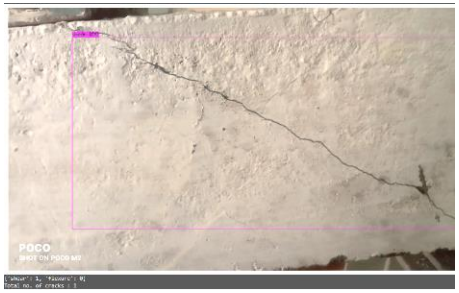
$$\text{IOU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} \tag{2}$$



Test Image-1



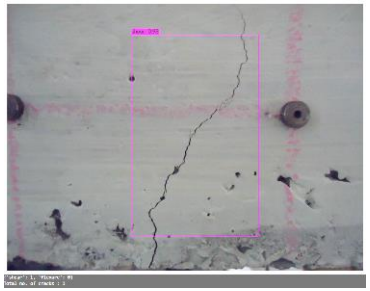
Test Image-2



Test Image-3



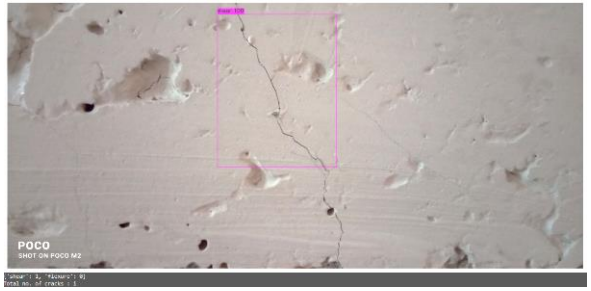
Test Image-4



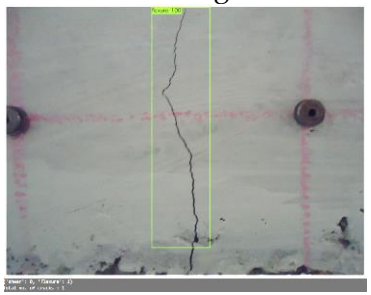
Test Image-5



Test Image-6



Test Image-7



Test Image-8



Test Image-9

Figure 14. *Cont.*

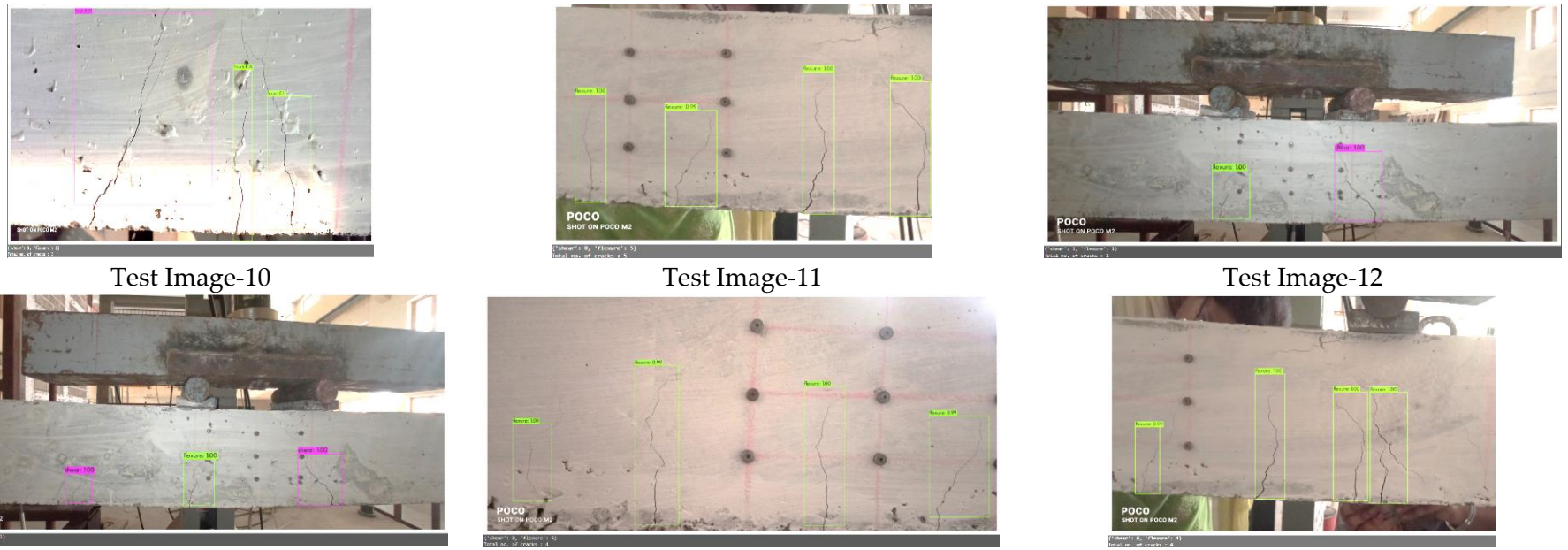
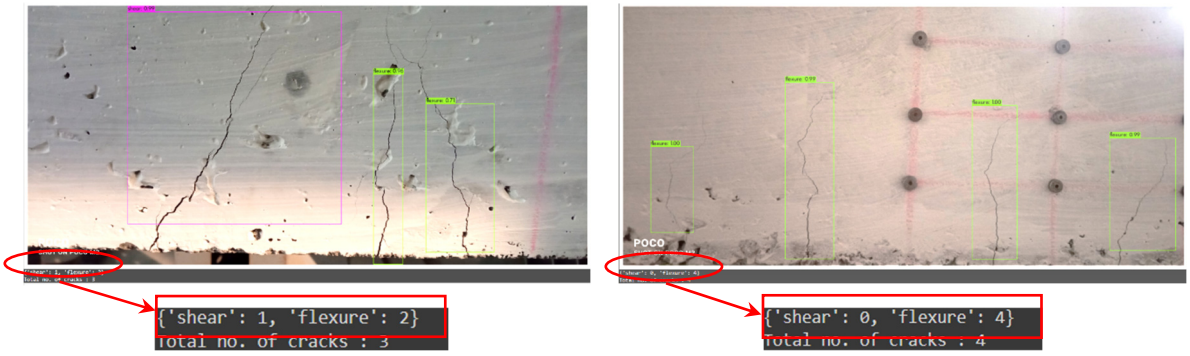
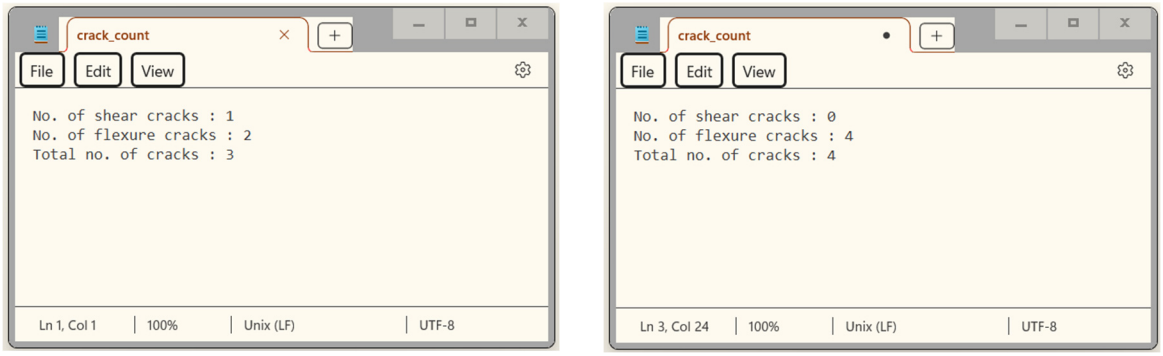


Figure 14. Crack detection results of crack detection model.



(**a**)



(**b**)

Figure 15. Crack count results. (**a**) Outputs on screen. (**b**) Outputs as text file.

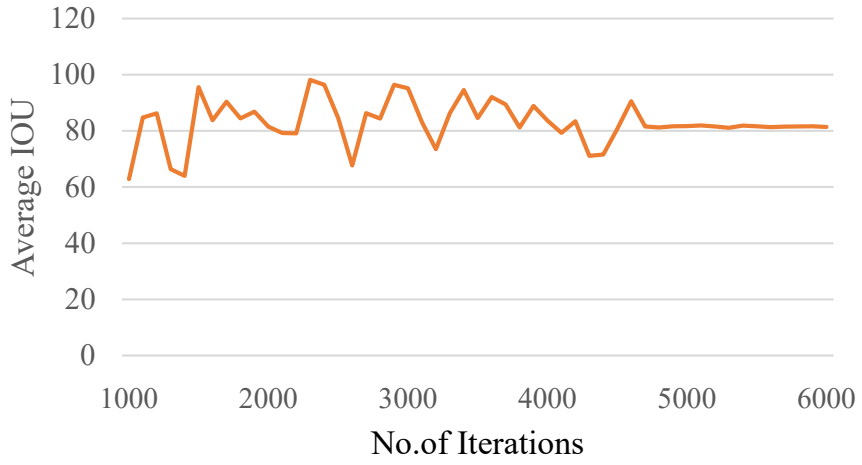


Figure 16. Average IOU.

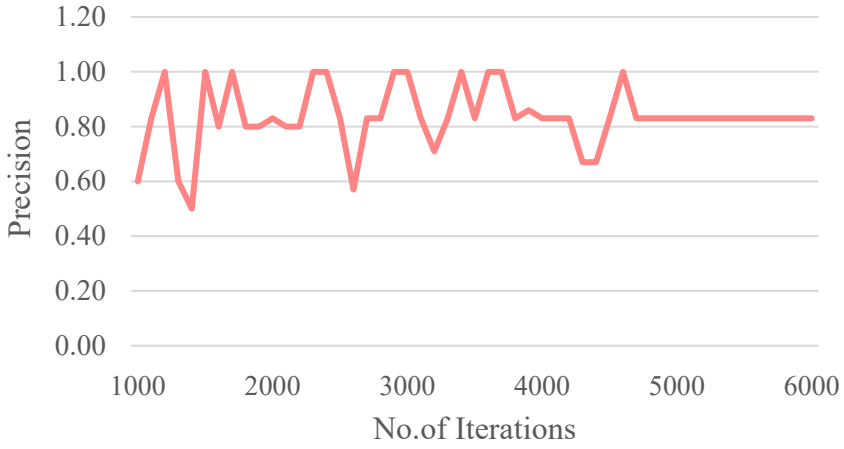
The F1 score is a measure of a model's accuracy on a dataset, and it is a function of P and R given in Equation (3). P is the percentage of true positives predicted in Equation (4), and R is the capacity to detect all the true positives, as in Equation (5). TP, FP and FN refer to true positive, false positive and false negative, respectively.

$$F1 = 2 \times \frac{P \times R}{P + R} \times 100 \tag{3}$$

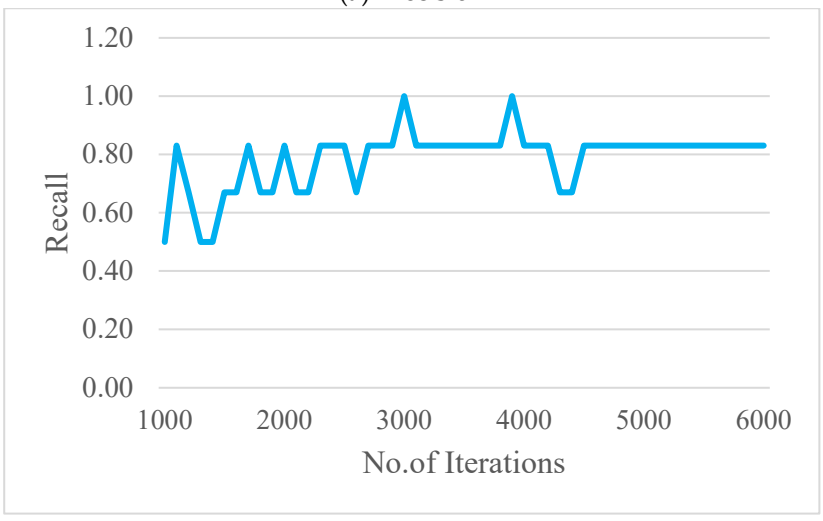
$$P = \frac{TP}{TP + FP} \times 100 \tag{4}$$

$$R = \frac{TP}{TP + FN} \times 100 \quad (5)$$

Initially, the values of P and R are found to be fluctuating, and thereby, the F1 score is also fluctuating. However, once the CDM is trained, all these values have been stabilized, as shown in Figure 17. It takes 4700 iterations to attain this stabilized state during the training. For the trained CDM, the values of P, R and F1 score are found to be 0.83. The graphs of average IOU in Figure 16 and Precision (P), Recall (R), F1 score, and mean Average Precision (mAP) in Figure 18 offer a comprehensive view of the model's performance in detecting cracks. The average IOU assesses how well the predicted bounding boxes overlap with the actual crack areas, indicating localization accuracy. Precision measures how many of the detected cracks are correctly identified, while Recall reflects the proportion of actual cracks the model successfully detects. The F1 score provides a balanced measure of both precision and recall. The AP value is the area under the P–R curve. Like P, R and F1 score, the mAP is also fluctuating initially and becomes stabilized after 4900 iterations during the training, Figure 1.

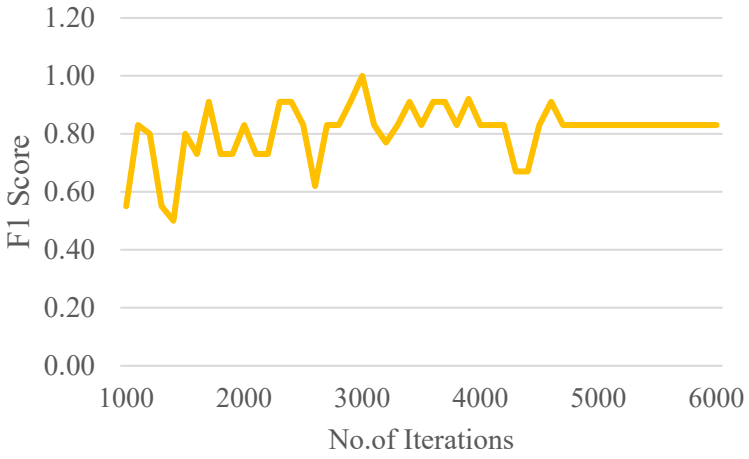


(**a**) Precision



(**b**) Recall

Figure 17. *Cont.*



(**c**) F1 score

Figure 17. P, R and F1 score curves.

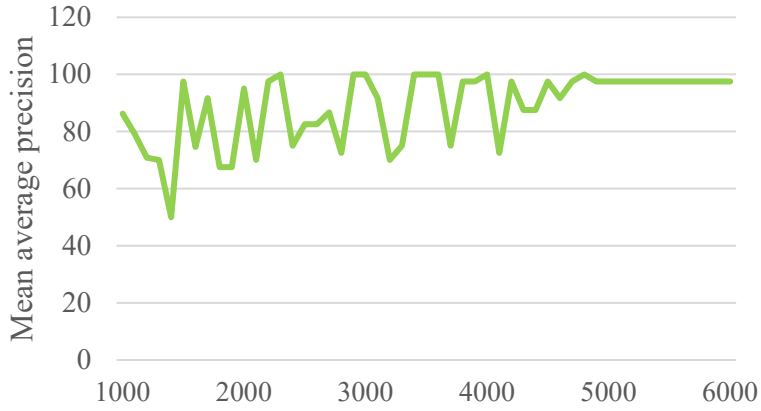


Figure 18. Mean average precision.

The mean average precision (mAP) score calculated using Equation (6) and shown in Figure 18 summarizes the model's overall detection accuracy across different confidence levels. It is the weighted average of average precision (AP) values of all the classes.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (6)$$

In general, both the IoU and mAP scores can be used to represent the accuracy in predictions when compared to the benchmark. Moreover, these metrics demonstrate the CDM's effectiveness in identifying cracks with high precision and consistency. From the present work, the average intersection over union (IoU) score was 83%, which depicts a reasonable accuracy of predictions along with a reduced computational time. The mAP score of the trained CDM is 87.0% which is slightly lower when compared to the values reported in the literature. In the case of predictions obtained from Wu et al. [36], the improved YOLOv4 model was able to predict the cracks with a mean average precision (mAP) range of more than 90%. However, the above-mentioned study does not correlate the findings in terms of intersection over union (IoU) score. The reduced processing time of YOLOv4-tiny makes it highly suitable for real-time, large-scale infrastructure monitoring, where speed, scalability, and energy efficiency are critical.

4.3.2. Performance During Testing

The CS shows the level of certainty or likelihood of whether a BB has a crack inside it and the class to which it belongs, as in Equation (7).

$$CS = Probability(object) * IOU * Probability(class_i \ / object) \quad (7)$$

where $Probability(object)$ and $Probability(class_i \ / object)$ are the probability that the BB contains a crack in it and the probability that the crack belongs to $class_i$ if a crack is present, respectively. Here, $i = 1$ if a class is flexure; $i = 2$ if a class is shear. The maximum, minimum, and mean CS for the test images are found to be 1.00, 0.94 and 0.99, respectively, as in Table 3. The CS between 0.94 and 1.00 indicates a high level of certainty that the detected object is accurately classified and localized by the CDM. This ensures that the algorithm is highly confident that the identified region is a true crack and not a false positive. High confidence scores are crucial for minimizing misclassifications and ensuring reliable results, especially in safety-critical applications like structural health monitoring, where incorrect identification can lead to inaccurate assessments or overlooked damage.

Table 3. Confidence score for the selected images used for testing.

Image	No. of Cracks Present	Crack Type	CS
Test Image-1	1	Shear	1.00
Test Image-2	1	Flexure	0.97
Test Image-3	1	Shear	0.99
Test Image-4	1	Flexure	0.98
Test Image-5	1	Shear	0.96
Test Image-6	1	Flexure	0.97
Test Image-7	1	Shear	0.96
Test Image-8	1	Flexure	1.00
Test Image-9	1	Shear	1.00
Test Image-10	3	Shear	0.99
		Flexure	1.00
Test Image-11	3	Flexure	1.00
		Flexure	0.99
		Flexure	0.99
Test Image-12	2	Flexure	1.00
		Shear	1.00
Test Image-13	2	Shear	0.98
		Flexure	1.00
Test Image-14	2	Flexure	1.00
		Flexure	1.00
Test Image-15	4	Flexure	0.94
		Flexure	0.99
		Flexure	1.00
		Flexure	0.99
		Mean	0.99
		Maximum	1.00
		Minimum	0.94

It is worth adding that the confidence scores of YOLOv4-tiny in older or more deteriorated structures could be lower compared to newer structures, due to the increased irregularity, noise, and complexity in crack patterns. In such cases, to further improve the accuracy in crack detection, additional training with specific datasets focused on diagonal crack patterns or adjustments to the anchor box configuration could help the model better handle these more complex crack types. However, considering these important aspects is the scope for further work.

5. Conclusions

In this study, the adaptability of the YOLOv4-tiny-based crack detection model for reinforced members subjected to combined flexure and shear loads is discussed. The objective of the work was to extract the details of defects present in RC members (i.e., cracks) and to quantify them in terms of number and crack type. To validate the developed YOLO model, extensive full-scale testing consisting of RC beams subjected to different shear span-to-depth ratios was carried out. The digital images of the cracks developed during the load application were captured and placed in a custom dataset that had flexure, shear, or a combination of flexure and shear cracks. YOLOv4-tiny consists of only 29 layers, which helps in reducing the computational time and speeding up the object detection process. The following major conclusions are drawn:

- The use of the YOLOv4-tiny-based CDM for RC members is highly accurate in detecting the total number of cracks as well as the types of cracks.
- The performance of the CDM was evaluated through standard evaluation metrics: intersection over union, F1 score, precision, recall, mean average precision, and confidence score. The average IOU score was found to be 82.93% for the set threshold of 0.5. However, it is very important to draw the bounding boxes precisely for the custom dataset images since the performance of the CDM highly depends on the ground truth images. The values of precision and recall and the F1 score for the CDM were found to be 0.83. The mean average precision was found to be 87.5%.
- The P, R, F1 score and mAP are stabilized within 5000 iterations. Therefore, 6000 iterations are sufficient to train the CDM. However, if the number of classes is greater than three, then the thumb rule 2* number of classes can be tried.
- For the gallery of test images, the mean confidence score obtained for the detection of flexure and shear cracks is 0.98 and 0.99, respectively.
- This work mainly focused on the cracks developed in beam elements exposed to four-point flexure, which creates flexure and shear cracks while bending. However, other types of cracks, like compression cracks, torsional cracks, corrosion cracks, etc., are also possible in structural elements. In future studies, the YOLOv4-tiny-based CDM can be extended to detect the above-mentioned cracks in images after training with the appropriate dataset.
- Recommendations for further work: The YOLOv4-tiny algorithm used in the present work can be adapted for detecting any type of crack, such as compression or torsional cracks. This can be accomplished through the expansion of the dataset with examples of new crack types. Moreover, the model needs to apply advanced data augmentation for crack diversity and modify training labels to differentiate between crack types. This can also be achieved by fine-tuning the model through transfer learning. However, care should be taken to optimize the anchor boxes and loss functions for irregular crack patterns. By incorporating these strategies, the YOLOv4-tiny algorithm can be effectively adapted to detect and classify a broader range of structural cracks. Considering the importance, the detection of cracks other than flexure-shear can be considered as the scope for further work.
- Additionally, the model will be refined for use in real-time structural assessment, allowing for continuous monitoring of critical infrastructure. By integrating these advanced features, the system will be more adaptable to diverse structural conditions and will offer more actionable insights for preventive maintenance and timely interventions, improving the overall safety and durability of concrete structures.

Author Contributions: Conceptualization, S.R. and K.S.J.B.; methodology, S.R. and K.S.J.B.; software, M.C.S.; validation, S.R., M.C.S. and M.C.; formal analysis, S.R.; investigation, S.R.; resources, K.S.J.B. and M.C.; data curation, S.R.; writing—original draft preparation, S.R.; writing—review and editing, K.S.J.B., M.C.S. and M.C.; visualization, S.R.; supervision, K.S.J.B.; project administration, K.S.J.B.; funding acquisition, M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data will be made available by the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Munawar, H.S.; Hammad, A.W.A.; Haddad, A.; Soares, C.A.P.; Waller, S.T. Image-Based Crack Detection Methods: A Review. *Infrastructures* **2021**, *6*, 115. [CrossRef]
2. Rana, I.A.; Khaled, S.; Jamshed, A.; Nawaz, A. Social protection in disaster risk reduction and climate change adaptation: A bibliometric and thematic review. *J. Integr. Environ. Sci.* **2022**, *19*, 65–83. [CrossRef]
3. Palevičius, P.; Pal, M.; Landauskas, M.; Orinait, U.; Timofejeva, I.; Ragulskis, M. Automatic Detection of Cracks on Concrete Surfaces in the Presence of Shadows. *Sensors* **2022**, *22*, 3662. [CrossRef] [PubMed]
4. Kim, J.J.; Kim, A.-R.; Lee, S.-W. Artificial Neural Network-Based Automated Crack Detection and Analysis for the Inspection of Concrete Structures. *Appl. Sci.* **2020**, *10*, 8105. [CrossRef]
5. Valença, J.; Puente, I.; Júlio, E.; González-Jorge, H.; Arias-Sánchez, P. Assessment of cracks on concrete bridges using image processing supported by laser scanning survey. *Constr. Build. Mater.* **2017**, *146*, 668–678. [CrossRef]
6. Adhikari, R.S.; Moselhi, O.; Bagchi, A. Image-based retrieval of concrete crack properties for bridge inspection. *Autom. Constr.* **2014**, *39*, 180–194. [CrossRef]
7. Park, S.E.; Eem, S.-H.; Jeon, H. Concrete crack detection and quantification using deep learning and structured light. *Constr. Build. Mater.* **2020**, *252*, 119096. [CrossRef]
8. Valença, J.; Dias-da-Costa, D.; Júlio, E.; Araújo, H.; Costa, H. Automatic crack monitoring using photogrammetry and image processing. *Measurement* **2013**, *46*, 433–441. [CrossRef]
9. Fujita, Y.; Hamamoto, Y. A robust automatic crack detection method from noisy concrete surfaces. *Mach. Vis. Appl.* **2011**, *22*, 245–254. [CrossRef]
10. Mohan, A.; Poobal, S. Crack detection using image processing: A critical review and analysis. *Alex. Eng. J.* **2018**, *57*, 787–798. [CrossRef]
11. Spencer, B.F., Jr.; Hoskere, V.; Narazaki, Y. Advances in Computer Vision-Based Civil Infrastructure Inspection and Monitoring. *Engineering* **2019**, *5*, 199–222. [CrossRef]
12. Xu, Y.; Cai, Y.; Li, D.; Zhang, T. Building crack monitoring based on digital image processing. *Frat. Integrità Strutt.* **2020**, *52*, 1–8. [CrossRef]
13. Rao, A.S.; Nguyen, T.; Palaniswami, M.; Ngo, T. Vision-based automated crack detection using convolutional neural networks for condition assessment of infrastructure. *Struct. Health Monit.* **2020**, *20*, 2124–2142. [CrossRef]
14. Ali, R.; Chuah, J.H.; Talip, M.S.A.; Mokhtar, N.; Shoaib, M.A. Structural crack detection using deep convolutional neural networks. *Autom. Constr.* **2022**, *133*, 103989. [CrossRef]
15. Wang, Y.; Zhang, J.Y.; Liu, J.X.; Zhang, Y.; Chen, Z.P.; Li, C.G.; He, K.; Yan, R.B. Research on Crack Detection Algorithm of the Concrete bridge based on image processing. *Procedia Comput. Sci.* **2019**, *154*, 610–616. [CrossRef]
16. Abdel-Qader, I.; Abudayyeh, O.; Kelly, M.E. Analysis of Edge-Detection Techniques for Crack Identification in Bridges. *J. Comput. Civ. Eng.* **2003**, *17*, 255–263. [CrossRef]
17. Rajadurai, R.-S.; Kang, S.-T. Automated Vision-Based Crack Detection on Concrete Surfaces Using Deep Learning. *Appl. Sci.* **2021**, *11*, 5229. [CrossRef]
18. Miao, P.; Srimahachota, T. Cost-effective system for detection and quantification of concrete surface cracks by combination of convolutional neural network and image processing techniques. *Constr. Build. Mater.* **2021**, *293*, 123549. [CrossRef]
19. Wenga, A.X.; Huanga, Y.; Wang, W. Assessment of the feasibility of detecting concrete cracks in images acquired by unmanned aerial vehicles. *Autom. Constr.* **2019**, *105*, 102819. [CrossRef]
20. Aravind, N.; Nagajothi, S.; Elavenil, S. Machine learning model for predicting the crack detection and pattern recognition of geopolymer concrete beams. *Constr. Build. Mater.* **2021**, *297*, 123785. [CrossRef]
21. Zhang, A.; Wang, K.C.P.; Li, B.; Yang, E.; Dai, X.; Peng, Y.; Fei, Y.; Liu, Y.; Li, J.Q.; Chen, C. Automated Pixel-Level Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network. *Comput.-Aided Civ. Infrastruct. Eng.* **2017**, *32*, 805–819. [CrossRef]
22. Chun, P.-J.; Izumi, S.; Yamane, T. Automatic detection method of cracks from concrete surface imagery using two-step light gradient boosting machine. *Comput.-Aided Civ. Infrastruct. Eng.* **2020**, *36*, 61–72. [CrossRef]
23. Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *Comput. Sci.* **2021**, *2*, 420. [CrossRef] [PubMed]
24. Yao, Y.H. Feature Extraction and Image Recognition with Convolutional Neural Networks. *IOP Conf. Ser. J. Phys. Conf. Ser.* **2018**, *1087*, 062032. [CrossRef]
25. Liu, F.; Wang, L. Net-based model for crack detection integrating visual explanations. *Constr. Build. Mater.* **2022**, *322*, 126265. [CrossRef]

26. Hacıefendioğlu, K.; Başağa, H.B.; Kahya, V.; Özgan, K.; Altunışık, A.C. Automatic Detection of Collapsed Buildings after the 6 February 2023 Türkiye Earthquakes Using Post-Disaster Satellite Images with Deep Learning-Based Semantic Segmentation Models. *Buildings* **2024**, *14*, 582. [CrossRef]
27. Ye, W.; Deng, S.; Ren, J.; Xu, X.; Zhang, K.; Du, W. Deep learning-based fast detection of apparent concrete crack in slab tracks with dilated convolution. *Constr. Build. Mater.* **2022**, *329*, 127157. [CrossRef]
28. Su, C.; Wang, W. Concrete Cracks Detection Using Convolutional Neural Network Based on Transfer Learning. *Hindawi Math. Probl. Eng.* **2020**, *2020*, 7240129. [CrossRef]
29. Ali, L.; Alnajjar, F.; Al Jassmi, H.; Gocho, M.; Khan, W.; Serhani, M.A. Performance Evaluation of Deep CNN-Based Crack Detection and Localization Techniques for Concrete Structures. *Sensors* **2021**, *21*, 1688. [CrossRef]
30. Yao, G.; Sun, Y.; Wong, M.; Lv, X. A Real-Time Detection Method for Concrete Surface Cracks Based on Improved YOLOv4. *Symmetry* **2021**, *13*, 1716. [CrossRef]
31. Guo, C.; Lv, X.L.; Zhang, Y.; Zhang, M.L. Improved YOLOv4-tiny network for real-time electronic component detection. *Sci. Rep.* **2021**, *11*, 22744. [CrossRef] [PubMed]
32. de Menezes, R.S.T.; Magalhaes, R.M.; Maia, H. Object Recognition Using Convolutional Neural Networks. In *Recent Trends in Artificial Neural Networks—From Training to Prediction*; IntechOpen: London, UK, 2019. [CrossRef]
33. Zhou, X.; Koltun, V.; Krahenbühl, P. Probabilistic two-stage detection. *arXiv* **2013**, arXiv:2103.07461v1.
34. Du, Y.; Pan, N.; Xu, Z.; Deng, F.; Shen, Y.; Kang, H. Pavement distress detection and classification based on YOLO network. *Int. J. Pavement Eng.* **2020**, *22*, 1659–1672. [CrossRef]
35. Li, Y.; Han, Z.; Xu, H.; Liu, L.; Li, X.; Zhang, K. YOLOv3-Lite: A Lightweight Crack Detection Network for Aircraft Structure Based on Depthwise Separable Convolutions. *Appl. Sci.* **2019**, *9*, 3781. [CrossRef]
36. Wu, P.; Liu, A.; Fu, J.; Ye, X.; Zhao, Y. Autonomous surface crack identification of concrete structures based on an improved one-stage object detection algorithm. *Eng. Struct.* **2022**, *272*, 114962. [CrossRef]
37. Jiang, Y.; Pang, D.; Li, C. A deep learning approach for fast detection and classification of concrete damage. *Autom. Constr.* **2021**, *128*, 103785. [CrossRef]
38. Tran, T.V.; Nguyen-Xuan, H.; Zhuang, X. Investigation of crack segmentation and fast evaluation of crack propagation, based on deep learning. *Front. Struct. Civ. Eng.* **2024**, *18*, 516–535. [CrossRef]
39. Dong, X.; Liu, Y.; Dai, J. Concrete Surface Crack Detection Algorithm Based on Improved YOLOv8. *Sensors* **2024**, *24*, 5252. [CrossRef]
40. Kheradmandi, N.; Mehranfar, V. A critical review and comparative study on image segmentation-based techniques for pavement crack detection. *Constr. Build. Mater.* **2022**, *321*, 126162. [CrossRef]
41. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016. [CrossRef]
42. Montalbo, F.J.P. A Computer-Aided Diagnosis of Brain Tumors Using a Fine-Tuned YOLO-based Model with Transfer Learning. *KSII Trans. Internet Inf. Syst.* **2020**, *14*, 4816–4834. [CrossRef]
43. Brindha, U.; Maheswaran, J.; Chellapandian, M.; Arunachelam, N. Quantitative Assessment of Strengthening Strategies and Design Recommendations for the Repair of Corrosion-Damaged Reinforced Concrete Members. *Buildings* **2023**, *13*, 1080. [CrossRef]
44. Maheswaran, J.; Chellapandian, M.; Arunachelam, N. Retrofitting of severely damaged RC members using fiber reinforced polymer composites: A comprehensive review. *Structures* **2022**, *38*, 1257–1276. [CrossRef]
45. Chellapandian, M.; Maheswaran, J.; Arunachelam, N.; Prem Kumar, N. Shear Behavior of Low-Cost and Sustainable Bio-Fiber based Engineered Cementitious Composite Beams –Experimental and Theoretical Studies. *J. Build. Eng.* **2024**, *84*, 108497. [CrossRef]
46. Maheswaran, J.; Chellapandian, M.; Arunachelam, N.; Hari, M.N.T. Thermal and Durability Characteristics of Optimized Green Concrete Developed using Slag Powder and Pond Ash. *Mater. Res. Express* **2023**, *10*, 095503. [CrossRef]
47. Maheswaran, J.; Chellapandian, M.; Subramanian, M.V.R.; Murali, G.; Vatin, N.I. Experimental and Numerical Investigation of Shear Behavior of Engineered Cementitious Composite Beams Comprising Fibers. *Materials* **2022**, *15*, 5059. [CrossRef]
48. Arulanandam, P.M.; Subramanian, M.V.R.; Chellapandian, M.; Murali, G.; Vatin, N.I. Analytical and Numerical Investigation of the Behavior of Engineered Cementitious Composite Members under Shear Loads. *Materials* **2022**, *15*, 4640. [CrossRef]
49. Available online: https://github.com/heartexlabs/labelImg#labelimg (accessed on 1 July 2022).
50. Available online: https://github.com/AlexeyAB/darknet (accessed on 1 July 2022).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.