**Soil Classification based USCS utilizing ADABoost and the Convolutional Neural Network in Nsukka.**

**Abstract**

In this novel study, the use of image preprocessing techniques with the ADABoost Algorithm as well as the Convolutional Neural Network (CNN) was used in the prediction of the grade of a soil sample directly from the image of the soil. The preprocessing techniques employed primarily involved the use of the canny edge detection algorithm as well as a technique termed naïve-particle-size together with statistical summaries obtained from the naïve-particle-size to train the ADABoost algorithm while the raw images were passed directly into the CNNs.

After training and testing on both models, we obtained an F1\_macro score of 0.8 on the ADABoost algorithm and 0.6 on the CNN. This makes sense as due to the issue of distance primarily, the CNN made some mistakes in classification. This is a step towards automated Soil classification using machine learning.

**Keywords**: Convolutional neural network; Soil Classification; Damage detection; Deep learning; Image processing; Machine learning

1. **Introduction**

Cracks development are common occurrence in various concrete structures like pavements, beams, columns, walls, and bridges, etc. The presence of cracks on structures can sometimes result in the deterioration of the steel bars used for the construction of the structures when exposed to moisture leading to the integrity of the structure itself being undermined. Hence, the utilization of crack detection techniques to analyze the presence of cracks on a structure is not only germane in ensuring the longevity of the structure but also in maintaining its public safety.Traditionally, most crack detection techniques often rely on manual field surveys conducted by humans which often suffer from several limitations such as poor reproducibility, poor repeatability, excessive time and labor requirement, and high risk to surveyors. For instance, the Golden Gate Bridge, which is located at San Francisco, CA, USA, as well as other high-rise buildings and large structures constructed in several places across the world requires constant inspection which is not only time-consuming but also labor-intensive.

To address these limitations, significant research efforts have been devoted to the development of automated crack survey techniques. And most of these crack survey techniques are often geared towards overcoming the limitations of manual surveys by leveraging advanced technologies and algorithms to automatically detect and evaluate cracks. The automated crack detection techniques, which relies essentially on machine learning (ML), places emphasis on crack identification (Talab et al., 2016), categorization (Nguyen et al., 2022), crack length and width measurement (Carrasco et al., 2021), etc. ML algorithms have become a popular technique in almost every field of discipline. This is because it has the ability to perform various tasks with outstanding performance. ML algorithms can automatically digest intrinsic knowledge of a data, such as the hidden structures or relationships in the data and generate output that can be relied upon. Broadly, ML can be divided into two groups based on their algorithms’ principle of operation namely; traditional ML and deep learning (DL)algorithms. Traditional ML techniques require a predefined feature extraction stage to reduce the complexity of a data and make patterns more visible to learning algorithms. However, it limits the performance of the models developed with it, even if more data is provided.

DL, on the other hand, is a subset of ML that uses multilayer neural networks. It is an extremely powerful ML technique that has gained traction in most disciplines owning to its ability to extract features in some cases over traditional ML algorithms. For example, in image classification tasks where the popular convolutional neural networks (CNNs), a type of DL technique, are employed, it has the ability to extract image features without human intervention. And to achieve this, techniques such as scaling of images and data augmentation are employed. The data augmentation technique is used essentially to reduce overfitting of the model on the training data (Perez & Wang, 2017), (Mumuni & Mumuni, 2022). In certain scenarios where domain expertise plays a crucial role in identifying high-value features, particularly in structured (tabular) data, human involvement in feature extraction can be valuable, and the application of DL might not be essential under such circumstances. However, in cases where domain expertise is not of help, compared with traditional ML, DL techniques are more intelligent as the features of the data are automatically learned through the training process especially when large amount of data is used in model training.

With the success of the ML techniques, a large number of research efforts on ML-based crack detection have been conducted (Kheradmandi & Mehranfar, 2022). ML, especially DL, has become a mainstream technology for developing enhanced crack detection algorithms (Joshi et al., 2022). However, using CNNs, which was described earlier as a type of DL model, can be computationally expensive and less efficient in some cases. Furthermore, most existing solutions focused on segmenting cracks using datasets containing unfinished concrete surfaces (surfaces without any form of painting) but in the real world, cracks will occur on painted surfaces as well as on unpainted surfaces. Also, there is not much study on the performance of impact of low-quality images on crack detection or how the preprocessing of images can facilitate overcoming these impacts since after preprocessing, all images take a more or less similar meaning. To advance the course of the new methods in practice, this present study aimed to develop a new preprocessing technique and compare its results with a DL approach (UNET VGG 19) on raw images to ascertain if crack detection process can be handled efficiently on both painted and plastered wall surfaces. The principal objective of crack detection technique in the present study is to efficiently and accurately segment cracks on painted and plastered wall surfaces.

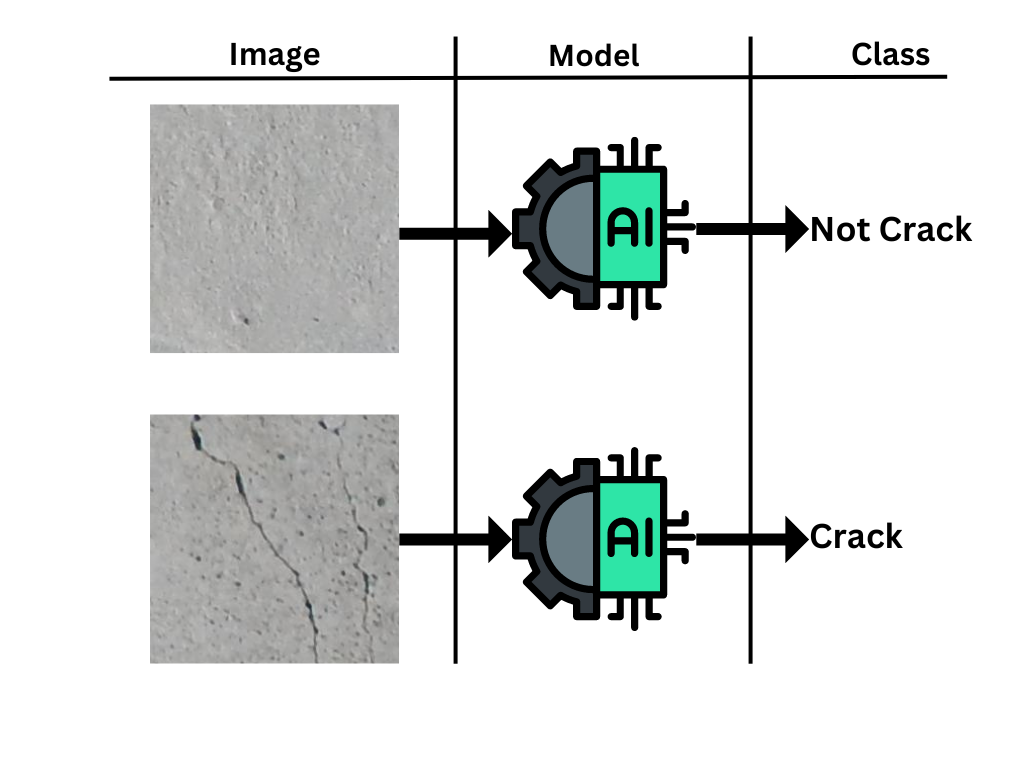
The structure of the paper is as follows: Section 2 presents the Background of the Literature, Section 3 presents the proposed methodology, it then describes and provides details of the proposed algorithm while Section 4 deals with the experimental results and discussion. The final section presents the conclusion.

1. **Background Literature**

While machine learning (ML) and deep learning (DL) approaches have gained popularity in crack identification, there has been limited research that explores the use of IP techniques. IP techniques are of various kinds namely: integrated algorithm, morphological approach, percolation-based method, thresholding, edge-detection, grayscaling etc. The integrated algorithm begins with a preprocessing step which involves the elimination of noise and the enhancement of crack features. Thereafter, thresholding is employed for the segmentation of the cracks. The morphological approach uses mathematical morphology and curvature evaluation to detect crack structures. The percolation-based approach determines the presence of cracks through analyzing the neighboring pixels and assessing the extent of the crack (Gao et al., 2020). Advanced Preprocessing methods combines one or more of the aforementioned IP techniques (Nnolim, 2020), (Salman et al., 2013).

Interestingly, of the aforementioned IP techniques, it is the grayscaling, thresholding and edge detection techniques that are the widely used IP techniques for crack segmentation. In reviewing various pieces of literature, two categories of crack identification are identified in which researchers employed IP techniques as well as ML and DL approaches for the identification of cracks, namely crack image classification and crack segmentation.

**i. Crack Image Classification**: This involves the classification of an entire image into one of two classes, such as the ones that have cracks or the ones without cracks. The classification process is usually achieved through the utilization of ML or DL models on raw or preprocessed images. For example, several attempts have been made to efficiently classify cracks; using a combination of Stockwell transform and a shallow Neural Network architecture (Nguyen et al., 2022), Convolutional Neural Networks of different architectures (VGG19 for example) with the raw crack images have also been used (Zeeshan et al., 2021).

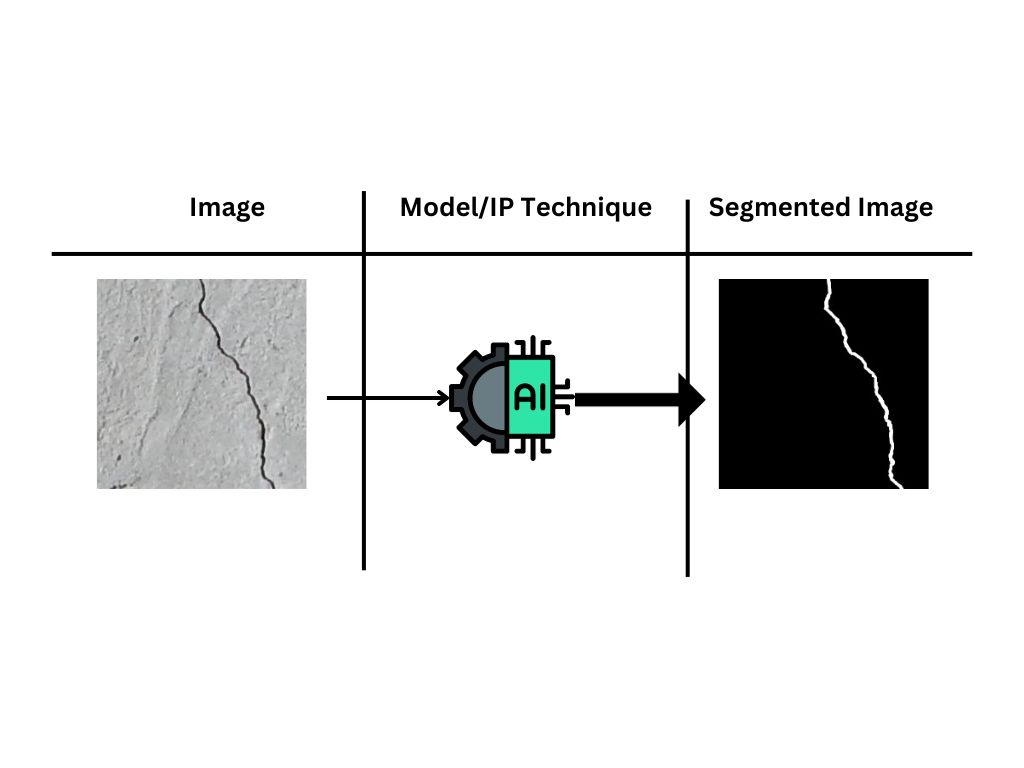
****

**Figure 1**: Diagrammatic representation of Crack Image Classification

**ii. Crack Segmentation**: This involves returning an output image with the crack regions clearly defined. In crack segmentation process, pixels are classified as either containing cracks or being crack-free, resulting in the generation of a segmented image. Various image preprocessing techniques, including thresholding and advanced methods, have been employed to enhance the accuracy of crack segmentation. For instance, the percolation based image processing techniques (Gao et al., 2020) and crack Image enhancement and detection 1 & 2 (CIEAD-1 and CIEAD-2) algorithms (Nnolim, 2020), which are typical examples of advanced image preprocessing techniques have been employed by several researchers.

Furthermore, ML techniques have also been used together with manual feature engineering. For example (Fujita et al., 2017) used manual handcrafted features together with an SVM classifier to do pixel level crack detection. DL techniques have also been used for the detection of cracks. For example (Cha et al., 2017) applied CNNs in a sliding window fashion to classify each window as having cracks or not, hence returning cracked part of an image. (Zhao et al., 2022), on their part, used Faster R-CNN utilizing bounding boxes with Image preprocessing techniques like (linear smooth, Gaussian smooth, thresholding and grayscale) to segment cracked part of an image.

These methods have proven to be accurate in detecting cracks but there have not been so much attempts towards development of less computationally expensive techniques for crack segmentation. Hence, through this study, the potential benefits of IP techniques towards crack segmentation are to be explored. A robust and efficient crack detection system that can provide reliable results in various real-world scenarios is aimed to be developed.



**Figure 2**: Diagrammatic representation of Crack Image Segmentation

**2.1 Evaluation Metrics**

The performance metrics or indicators of machine learning models or algorithms are very important to assess the efficacy of the models in the prediction of target values. In image segmentation tasks, the mean intersection over union (IOU) score is a good metric as it shows how well our technique correctly segments actual crack pixels. Four metrics overall are discussed namely IOU, F1 score, Precision and Recall score.

Crack pixels are denoted as 1 or positive and non-crack pixels as 0 or negative. In image segmentation, a single image produces a score for all metrics, and the average across all images is usually taken to get the general performance of a model or algorithm.

**2.1.1 Intersection Over Union (IOU)**

Intersection over union (IOU) is a fundamental evaluation metric used in computer vision and object detection tasks to assess the accuracy of object localization. It quantifies the overlap between a predicted bounding box or segmented mask and a ground truth bounding box or mask.

IOU is calculated as the ratio of the area of overlap between the predicted and ground truth regions to the area of their union. In mathematical terms, it can be expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

The IOU metric provides a value between 0 and 1, where a higher IOU indicates a better alignment between the predicted and ground truth regions. An IOU of 1 indicates a perfect overlap, meaning the predicted region precisely matches the ground truth. Conversely, an IOU of 0 indicates no overlap at all. IOU is particularly useful for tasks like object detection, instance segmentation, and image segmentation, where it helps quantify the accuracy of localization and segmentation algorithms. It serves as a critical tool for evaluating and comparing the performance of various computer vision models and techniques.

* + 1. **F1 Macro Score**

The F1 score is a widely employed metric within the parametric family of F-measures, specifically for the parameter value β=1. It is mathematically defined as the harmonic mean of precision and recall. It is expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

The F1 score falls within the range of 0 to 1. The lowest value (0) occurs when there are no true positive (TP) instances, indicating that all positive samples have been incorrectly classified. Conversely, the highest value (1) is achieved when there are no false negatives (FN) or false positives (FP), representing a perfect classification (Chicco & Jurman, 2020).

* + 1. **Precision Score**

The precision score is a measure of how many positive class data points were correctly classified compared to the total amount of data points predicted to belong to the positive class. It is expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

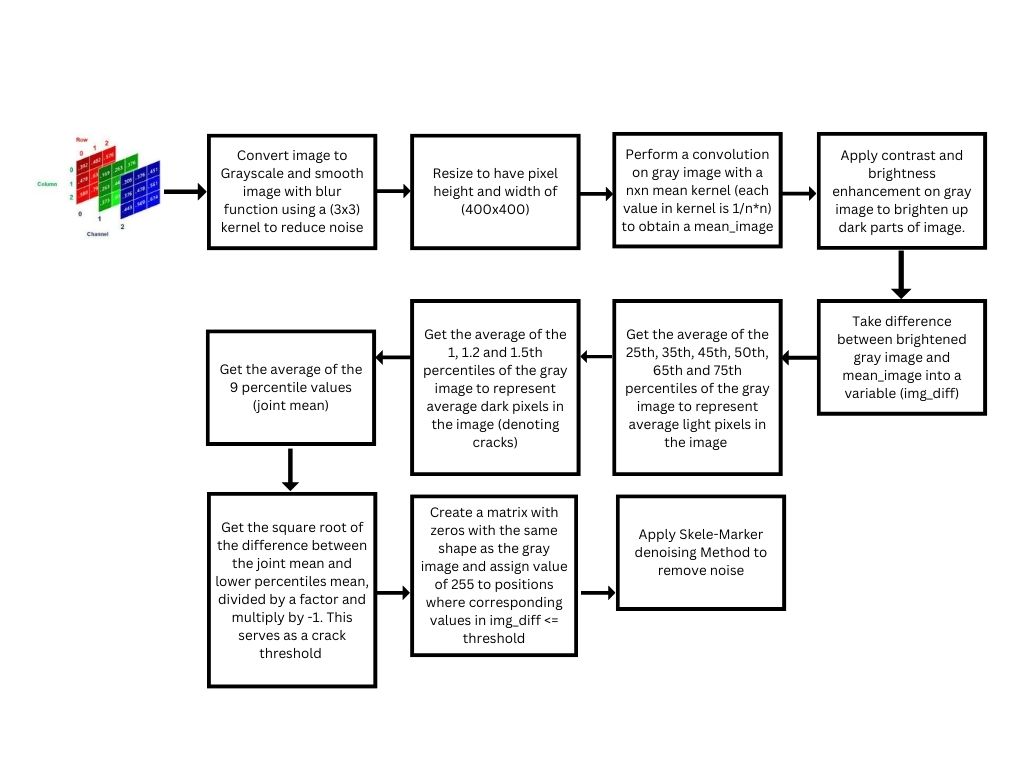
* + 1. **Recall Score (True positive rate)**

In a binary classification scenario involving two classes, the recall score serves as an indicator of the proportion of correctly identified positive class data points in relation to the total number of actual positive class data points. In other words, it is a measure of how many crack pixels were labelled correctly.

|  |  |  |
| --- | --- | --- |
|  |  | (4) |
|  |  |  |

**3. Methodology**

The Generally, the technique employed smoothening, contrast and brightness enhancement, thresholding, and pixel intensity distribution analysis towards crack visibility enhancement on concrete surfaces. The algorithm discussed and proposed is termed Crack Contrast Enhancement and Local Deviation (CCELD).



**Figure 3:** Processes involved in the CCELD algorithm

* 1. **Dataset Collection and Labelling**

The images were gotten from soil samples in our region. dataset size and content greatly affect the performance of the models. The pictures used for this study was obtained from two publicly available datasets and from our local environment (Marc Maguire, Sattar Dorafshan et Al., 2018) and (DatasetNinja, 2019). This first dataset (DS1), which had no segmentation masks, contains 56,092 two-dimensional color images of different surfaces (all unpainted) with .JPG extension. The second dataset (DS2) contained 445 images with masks. In total, 585 images were used in this study (80 from DS1, 445 from the DS2 and 60 from our local environment). 68% was used for training, 15% was used for validation and 17% (80 images from DS1 and 20 from environment) served as our test data for performance comparisons.

The cracks from the DS1 were selected based on two criteria:

1. The cracks on the images should at least be recognized by humans.
2. Crack widths considered were of a certain width (width size was based on cracks that were generally obtainable in our environment).

**3.2 Preprocessing the Images towards Training with ADABoost**

The CCELD technique primarily takes advantage of the fact that the darkest region of an image with a crack will most likely be the cracked region. It involves utilizing a convolution operation on a grayscale image with a smoothening mean filter of size determined from experimentation. The grayscale image is then subtracted from the result of the convolution to obtain a local deviation image. Using each grayscale images’ pixel distribution, a unique threshold value is determined with the expression written in [Eq. (11).](#formula) Based on this value, thresholding is done and the pixels from the grayscale image whose value in the local deviation image are less than or equal to the unique threshold is set as 255 while those greater than the unique threshold are set as 0. The procedures involve in the proposed CCELD algorithm are discussed explicitly in Sub-sections [3.21](#start_techn)-[3.24](#end_techn). All the computation executed in the CCELD algorithm were implemented with the OpenCV library using the python programming language.

**3.2.1 Smoothening, grayscale conversion and resizing**

Smoothing and blurring is a very important step in image preprocessing. By smoothing an image prior to applying techniques such as edge detection or thresholding the amount of high-frequency content, such as noise and edges (i.e., the “detail” of an image) is reduced (Rosebrock, 2021). The first step involved the conversion of the RGB image into a grayscale image. Grayscaling is a significant step in IP techniques especially in crack detection. It involves the conversion of 3D color image into a 2D black and white image. The image was then resized to have a height and width of 400 by 400. This size was chosen after experimentation as very small and very large sizes did not yield good results. Prior to resizing, blur smoothening was applied on the images with a kernel which aided denoising. The blur smoothening operation involved assigning a pixel a value based on the mean values in the neighboring kernel with the pixel in the center of the kernel. Considering any neighborhood, the operation on the center pixel value can be expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Where; represents the th row; represents the *th* column, and is any center pixel surrounded by 8 neighboring pixels.

**3.2.2 Contrast and Brightness Enhancement**

Image contrast and brightness enhancement are fundamental techniques that improve the quality of images. These methods adjust the intensity of pixels in an image to make it more visually appealing and easier to interpret. Contrast enhancement emphasizes the differences between light and dark areas, leading to the enhancement of details. Brightness adjustment ensures that images are neither too dark nor too bright.

Some crack images on finished surfaces could have different color of paints and if a crack exists in both parts of the picture, the threshold determined might not work well in the dark region, hence the brightness of the image is enhanced using a simple linear formula.

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

Where; is the returned brightened image, and is the original gray scale image

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| (a) | **(b)** | **(c)** |

**Figure 4:** **(a)** Original Image **(b)** Grayscale image **(c)** Brightness and Contrast enhanced Image

**3.2.3 Image Differencing**

This process employed another smoothening operation on the grayscale image using a kernel determined after experimentation to yield a mean image. Thereafter, a difference between corresponding pixels in the grayscale image and the mean image is obtained. This differencing yields negative results for possible crack regions since after finding the average value, the corresponding value in the mean image is significantly larger than the pixel value in the gray image. Hence, the result from the subtraction of the mean image from the gray image is negative for all possible crack regions. This optimal smoothening kernel was determined from comparison which involved tracking the average IOU score of 100 CCELD plus Skele-Marker preprocessed images against their ground truth mask at different kernels and the kernel that gave the best results together with other parameters of CCELD was selected.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| (a) | **(b)** |

**Figure 5:** **(a)** Grayscale image **(b)** Pixel Value for selected region (red square) after image differencing

**3.2.4 Automated thresholding and determination of crack mask**

The next step involved the determination of a threshold in which pixel values less or equal to that threshold is taken to be a crack pixel and hence is assigned a pixel value of 255 (whitest pixel) and pixel values greater were assigned a pixel value of 0 (darkest pixel) resulting in a crack mask image with pixel values having 0 or 255 (the segmented image). Towards automated threshold selection, a histogram analysis of image pixel distribution was done. In images [1-3 of Fig. (6)](#img123), the distribution of pixels values falls on the low side of pixel values indicating a dark colored image with a lot more darker pixels generally than light ones. In making a comparison, an analysis of specific areas (area within the red box) in [Fig. (7)](#img3_comparison) is conducted.

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| (a) | (b) |

**Figure 6:** **(a)** Grayscale image **(b)** Histogram Showing Grayscale Pixel Distribution

In Image 3 [of Fig. (7)](#img3_comparison), pixel values are generally small as pixels outside the crack regions ranges from 60 - 80 and pixels in the crack regions have values in range 25-40. In image 4 [of Fig. (7)](#img3_comparison), however, pixel values are considerably larger as values outside the crack region ranges from 145 - 190 and pixels in the crack regions have values in range 140 - 130. Due to this difference in range of values, image differencing will yield larger negative difference in image 3 when compared to image 4. As demonstrated in [3.2.3](#img_differencing), difference in range of -30 to -40 [Fig. (5)](#fig6) is obtained in image 3 and a difference in range of -50 to -75 [Fig. (5)](#fig6b) is obtained in image 4. With this, it is evident that a single threshold will not be optimal for all kinds of images. This prompted the analysis for an automated threshold selection.

|  |  |
| --- | --- |
| 3 |  |
| 4 |  |
| (a) | (b) |

**Figure 7:** **(a)** Grayscale image with crop area indicator **(b)** Cropped area of image showing Pixel values

Using percentiles and other statistical means, a formula was obtained. The division variable was found to be the most important in the formula. Thus, with the aid of tracking, the average IOU score of CCELD results augmented with Skele-Marker against the ground truth for varying values of the parameter, the best value for the parameter was obtained as 0.4 to determine the best threshold for a picture. The statistical steps towards thresholding are defined;

**i. Mean Computations**

From the initial smoothened grayscale Image before resizing, three variables containing average of percentiles of pixel values are computed.

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

The can be said to capture the mean of the crack intensity while the can be said to capture the average of pixels close to the crack and far away from the crack but not so far away. The then gets the average of all 9 percentiles. This imitates using a kernel in obtaining the mean image as 9 percentile values are considered.

**ii. Threshold Initialization 1**

The variable initial threshold is computed. This initial threshold ( was computed as a subtraction between the join mean (mean of all pixels) and low mean (mean of pixels denoting crack).

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

**iii. Final Threshold Computation**

The final threshold for each image was then determined using expression shown in Equation (11).

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

In total two parameters were tuned for namely; Kernel size, and division .

**3.3 Denoising using Skele-Marker Method**

Binary image denoising is the process of removing unwanted artifacts and random variations in binary (black and white) images to reveal the true structural features of objects in the image. In binary images, each pixel is either entirely black (0) or entirely white (1), with no grayscale values in between. Noise in binary images can obscure object boundaries and affect the accuracy of image analysis and processing. Denoising binary crack images is thus a crucial step in enhancing the accuracy of crack detection and analysis. The presence of noise complicates the identification and segmentation of cracks (Chianese et al., 2021). By addressing noise, denoising processes play a vital role in ensuring the reliability and accuracy of crack detection algorithms. There are several methods that can be used for denoising, however, in the present study, the Skele-Marker was selected because it has demonstrated good performance as evident in literature (Hamish Dow et al., 2023)

The Skele-Marker method, which is a recent innovation in denoising binary crack images, starts with area thresholding process and then proceeds to streamlining pixel groups into skeletons. Each skeleton is then linked with its closest neighbor. Subsequently, short skeletons are eliminated using a length threshold. Thereafter, it then employs a morphological reconstruction aimed at removing elements in the original noisy image that lacks intersection with the skeleton. Finally, it does a radius-based restoration, such that pixel’s groups close to the endpoints of pixel group in the reconstructed image are reinstated (Dow et al., 2023).

**3.3.1 Area thresholding**

This is the application of a predefined threshold based on the size of connected pixel groups or components in a binary image. Area thresholding helps to distinguish between significant crack structures and unwanted noise or artifacts. By setting a maximum threshold value, components that are too small to be considered part of the crack pattern are filtered out.

It involves a connected components analysis utilizing eight-neighbors round a given pixel such that connected groups of pixels,  = G1, . . ., , each with area = A1, . . ., is created. Regions consisting of pixel groups with an area smaller than a predefined threshold, , are labeled as “uncracked regions.” Within these regions, all pixels are assigned a binary value of 0, making them part of the background.

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

Where; is a given pixel in Image; is a Pixel group ; and is the Area of group

**3.3.2 Skeleton mask Creation**

This is a reduction of the of all Group pixel widths to have a width of one pixel. This process is iterative and entails thinning down the regions of interest to their core structure. In the resulting skeleton, the centerline of the original object is preserved, and finer details are removed. This step plays a key role towards denoising because when pixels have 1 pixel width, a sum of all pixels within a connected group (group length) can be taken and a threshold can be used to keep some.

Thus, the remaining connected components from Section [3.3.1](#Areathresh) are reduced to 1 pixel thin using the Zhang–Suen thinning algorithm. This is an iterative process that creates a skeleton of the crack by removing pixels with fewer than eight neighbors (RosettaCode, 2023).

**3.3.3 Length removal**

The thinned pixel groups = G1, . . ., , each with length = L1, . . ., are then considered such that groups with length smaller than a predefined length threshold () are removed leaving possible crack skeletons left in the image. The length of a pixel groups is really just the sum of the pixels in a connected skeleton.

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

, Retain Skeleton

, Discard Skeleton

**3.3.4 Morphological Reconstruction**

Morphological reconstruction is an image processing technique used to extract features and patterns from images. It involves iteratively expanding a marker image while limiting it by a mask image, ensuring the final result retains only the common features of both images. This process is useful in enhancing and segmenting image structures.

Towards maintaining the pixel groups from the original image after CCELD that intersect with the remaining skeleton groups, a morphological reconstruction process is employed [Fig. (8)](#morphrecon1). A Python function is utilized with a radius of 1 by default, creating a 3 by 3 structuring kernel (Stack Overflow, 2020). The function operates on two input images: 'mask,' representing the original noisy image, and 'marker,' depicting the skeleton image. The 3 by 3 kernel is employed for iterative expansion of the marker image through morphological dilation (Karas, 2011). After each iteration, the expanded image is combined with the mask using a bitwise AND operation. This step ensures that the expanded image only retains pixels present in the mask image. The function continues to iterate until the expanded image reaches a stable state, at which point the final iteration image is returned.

**3.3.5 Radius based restoration**

This is a crucial step in the Skele-Marker method which involves restoring pixels near endpoints of pixel groups after skeleton restoration based on a set radius. This step is really important because based on length chosen to retain skeletons, some possible cracks might be lost and if a good radius is set, they can be restored back in this step.

Towards achieving this, two python functions are created. The first takes the thinned image after short skeletons have been removed as input and finds the coordinate of all existing skeleton group endpoints.

The second takes three arguments; the thinned image after short skeletons have been removed, the original binary image and a radius (), used to implement radius restoration. Using the coordinates of the endpoints found in the first functions, circular white masks with the set radius are drawn at all endpoints and a morphological reconstruction is repeated towards retaining intersecting pixels in the circular regions and the original image. Finally, the restored image from this step is combined with the restored image from [3.3.4](#morphreconstruct)

|  |  |
| --- | --- |
| **Step** | **Image** |
| Binary Image Post CCELD - |  |
| Area Thresholding – | Some Noise is removed |
| Skeletonization |  |
| Length Thresholding – | Most of the Noise is removed |
| Morphological Reconstruction 1 - | **End Crack is missed** |
| Drawing Circles at Endpoints – |  |
| Morphological Reconstruction 2 ­– | **Most of End Crack is Restored** |
| Final Image is derived |  |

**Figure 8:** Visual Representation of Image state at different points of the Skele Marker Technique

**3.4. DL Segmentation Model (UNET)**

UNET is a popular architecture in the field of computer vision, specifically designed for semantic segmentation tasks. Semantic segmentation involves labeling each pixel in an image with a corresponding class label, making it a fundamental technique for tasks like object recognition and scene understanding. The UNET architecture is characterized by its U-shaped design, which consists of an encoding path and a decoding path.

**3.4. 1 Loss functions**

Every DL model is trained with a loss function which are means to update the weights of the model during training. There are several loss functions used such as the binary cross entropy loss, dice loss, focal loss, etc. The Binary Cross-Entropy Loss, uniquely tailored for binary segmentation tasks like crack segmentation, excels in penalizing models based on the probabilistic differences between predicted and actual crack pixels. In this study, the jaccard variant of the BCE loss is used and it’s termed the BCE jaccard loss. The jaccard index or IoU calculates the overlap between predicted and ground truth masks by considering their intersection over their union. It's commonly used to evaluate the similarity or overlap between two sets.

The BCE Jaccard Loss combines these two concepts by incorporating the binary cross-entropy loss to handle pixel-wise predictions and the Jaccard loss to measure the similarity between predicted and true masks. By doing so, it encourages the model to not only accurately predict pixel-wise probabilities but also to strive for better overlap and alignment between predicted and ground truth masks, ultimately aiming for improved segmentation accuracy and precise boundary delineation this makes it better for binary segmentation tasks over the other loss functions.

**The Binary Cross Entropy (BCE) Loss Function**

The binary cross entropy is a loss function that gives a score based on how far or close the probability of pixels in our training images are close to their true values (1 or 0 in the case of binary segmentation problems) (Udoh, 2023). For a flattened (1D) image mask, it is expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

**Jaccard Loss**

The Jaccard loss is calculated as 1-IOU Score. For a flattened (1D) image mask, the loss function can be represented as:

|  |  |  |
| --- | --- | --- |
|  |  | (16) |

The BCE Jaccard loss is a combination of both loss functions;

|  |  |  |
| --- | --- | --- |
|  |  | (17) |

Where

= ground truth pixel value (1 or 0)

= predicted pixel value (probability)

**3.4.2 Model Hyperparameters**

The chosen encoder for the UNET model is based on the literature (Liu & Wang, 2022) and the open-source python framework “segmentation models” built on top of Keras and Tensorflow is used (Iakubovskii, 2018/2023). Towards optimal and accurate learning, the Adam optimization method is employed. Furthermore, the RGB image is resized to 256 × 256 × 3 and the ground truth is resized to 256 × 256 × 1 (binary image). Other hyperparameters are the learning rate (0.001) where a reducing learning rate approach was taken to avoid the model being stuck in local minima, a mini-batch gradient descent approach was taken with a batch size of 32 images.

**3.5 Model Development and Use**

The training of the UNET model as well as all other codes were run using the python programming Language on Google Colaboratory. Google Colaboratory provides the Tesla T4 Graphics Processing Unit (GPU) which is a GPU card based on the Turing architecture and targeted at deep learning model inference acceleration. The Tesla T4 card contains 40 streaming multiprocessors (SM) with a 6MB L2 cache shared by all SMs. It also has a 16GB high-bandwidth memory Graphics Double Data Rate 6 (GDDR6 RAM) that is connected to the processor (Github- D2l-Ai/D2l-Tvm, 2019). GPUs have become essential in training deep learning models due to their parallel architecture, enabling simultaneous processing of multiple tasks. With thousands of cores, GPUs excel in handling computationally intensive operations like matrix multiplications crucial for neural network training. Their efficiency accelerates the training of complex models, reducing processing times from weeks to hours.

**4. Experimental Results and Discussions**

Towards obtaining the best parameters for the segmentation technique, the correctness of the resulting image was compared with the ground truth using the IoU score after denoising and an average of 0.7 IoU score was obtained for the best set of parameters (*kernel size* of (31, 31), *division* value of 0.4, *Tarea* value of 12, *Tlength* of 43 and *Tradius* of 18. In deciding the parameters to be tuned, few test runs were done to get an idea of the best performing range of values for different kernel size and values for tuning were selected based on this intuition.

**Table 2**. Different parameters of CCELD and the Skele-Marker tuned for each kernel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kernel | Division Value | Area | Length | Radius |
| (10, 10) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (12, 12) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (14, 14) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (16, 16) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (18, 18) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (20, 20) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (22, 22) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (24, 24) | 0.4, 0.5, 0.7 | 6, 8 | 21, 23, 25, 27, 29 | 5, 6, 7, 8, 9 |
| (25, 25) | 0.85, 1.0, 0.4, 0.7, 1.2 | 10, 12, 14, 16, 18, 8 | 30, 32, 34, 36, 38, 25, 28, 31, 37, 40, 43 | 10, 11, 12, 13, 14, 6, 8, 16, 18, 20 |
| (28, 28) | 0.4, 0.7, 1.2 | 8, 10, 12 | 25, 28, 31, 34, 37, 40, 43 | 6, 8, 10, 12, 14, 16, 18, 20 |
| (31, 31) | 0.4, 0.7, 1.2 | 8, 10, 12 | 25, 28, 31, 34, 37, 40, 43 | 6, 8, 10, 12, 14, 16, 18, 20 |
| (34, 34) | 1.5, 2.0, 0.4, 0.7, 1.2 | 8, 10, 12 | 25, 27, 29, 31, 33, 28, 34, 37, 40, 43 | 8, 10, 12, 6, 14, 16, 18, 20 |
| (37, 37) | 0.7, 1.2, 0.4 | 8, 10, 12 | 37, 40, 43, 25, 28, 31, 34 | 14, 16, 18, 20, 6, 8, 10, 12 |

**4.2 Sensitivity analysis of CCELD and Skele parameters**

From [Fig 11](#fig11div), the MIOU as well as MF1 scores seem to be more sensitive to the division parameter which makes sense as it’s the first attempt towards denoising by CCELD before denoising by Skele Marker. One interpretation is that with a good threshold, the noise present in the original binary image after CCELD will be less. Hence, moderate values of Skele-marker will be required to denoise thus values lower than the benchmark from (Hamish Dow et al., 2023) is required. Interactive analysis is performed in [4.3](#interanalysis) to evaluate the influence of the parameters on the output when combined together.

|  |  |
| --- | --- |
| (a) | |
| (b) | **(c)** |
| (d) | **(e)** |

**Figure** **9:** Typical results of Mean IoU score (left) and Mean F1 score (right) vs **(a)** Kernel size **(b)** Division **(c)** Tlength **(d)** Tradius **(e)** Tarea

**4.3** **Interactive influence of all Parameters on MIOU and MF1 scores**

In the domain of image preprocessing, the fine-tuning of various parameters is instrumental in achieving optimal results. These parameters hold the power to profoundly influence the quality and characteristics of processed images. Yet, assessing their individual impact alone often offers an incomplete picture of the overall outcomes. To gain a deeper understanding of how these parameters collectively shape the processed images, an interactive analysis becomes indispensable. In this analysis, the comprehensive effects of the parameters on the preprocessed image are revealed.

This analysis is broken down into 3 based on the magnitude of kernel size;

**1. Small Kernel:** This includes kernels with values ranging from (10,10) to (19, 19)

**2. Medium Kernels:** This includes kernels with values ranging from (19,19) to (25,25)

**3. Large Kernels:** This includes kernels with values ranging from (25,25) to (40,40)

|  |  |
| --- | --- |
| (i) | (i) |
| (ii) | **(ii)** |
| (iii)  (a) | **(iii)**  **(b)** |

**Figure 10:** Planar approximations of **a(i-iii)** CCELD parameters and **b(i-iii)** Skele-Marker parameters influence on MIOU score across the different Kernel segments.

The results of CCELD parameter influence on MIOU are approximated with a linear plane and in the small kernel region, the problem of underfitting is observed as the plane is sloped downwards towards smaller kernel size values, coming to larger kernels, a strict negative correlation between MIOU and the division parameter is immediately noticed. It is important to point that the linear approximation might not be the best and should not be interpreted that much lower division parameter values will yield better results as values of same kernel with similar division values from the medium kernel plot are close indicating a plateau.

The results of the Skele-Marker parameters influence on MIOU shows that across all kernel segments, a larger length indicates better performance. At a given length, there’s a variation of MIOU scores and in each group similar area and radius values exist. This indicates that these parameters themselves don’t significantly influence the results.

From the above results the general deductions are drawn;

1. A small division is a good way to start denoising.

2. For moderate crack widths, a large kernel (between 25,25 and 40,40) is recommended.

3. Large Tlength values are encouraged 55 is recommended from (Hamish Dow et al., 2023) and a value of 43 is used in this study.

4. Small area values (between 6 and 8) will suffice and radius values between (10 and 14) will suffice.

**4.4 CCELD VS OTSU**

To further discuss the efficacy of CCELD, it’s thresholding performance is compared to that of the OTSU algorithm. The Otsu Algorithm takes into consideration the entire image and then based on pixel intensity distribution; it sets a universal threshold. This is a problem because in instances where sudden light shines on parts of the image, this could raise the value of the threshold and the crack region will be omitted [Fig 11](#otsuvscceld). On the other hand, CCELD uses image differencing hence, ensuring a localized form of thresholding and completely erases the possibility of the mistake made by the OTSU algorithm from happening.

|  |  |  |
| --- | --- | --- |
| Image | CCELD | OTSU |
|  |  |  |

**Figure 11:** Comparison of performance of CCELD and OTSU on image thresholding

**4.4 UNET Training and Performance**

Transfer learning was employed and weights from Imagenet was used with the VGG19 as the encoder which was found to be the best encoder by (Liu & Wang, 2022). On the first 2 epochs, the encoder weights were frozen. By freezing the encoder for the first few epochs, the encoder’s weights are kept relatively stable since they are already well-learned from ImageNet and the features learnt by the encoder from ImageNet are generally useful for a wide range of computer vision tasks. This is a way to prevent the encoder from adapting too quickly to the specific task, which could potentially result in the loss of valuable knowledge contained in the ImageNet weights. By doing so, the decoder is then allowed to adapt and fine-tune to the specific segmentation task.

After this initial learning by the decoder, the encoder weight was made trainable. The model is immediately evaluated for the Mean IOU score of the validation dataset for each epoch. During training, a checkpoint was created to save the best model weights after some epochs such that those weights could be reloaded and training can be resumed. Data augmentation of vertical flipping, horizontal flipping, width shifting, height shifting and zooming was also employed and the model results with and without data augmentation are recorded. Both cases were run for 50 epochs which was enough to get good performance since transfer learning was employed and the validation set yielded better results without data augmentation. The best model was saved after which the weights were loaded and used in making predictions.

The UNet model had the best performance from the training without augmentation and achieved a MIOU score of 0.64 and MF1 score of 0.78. [Fig 12](#fig14a) shows the training and validation loss as well as the validation MIOU scores at different epochs with and without data augmentation.

|  |  |
| --- | --- |
| (a) | (b) |
| (c) | |

**Figure 12:** **(a)** Training and validation loss for 50 epochs without Data Augmentation **(b)** Training and validation loss for 50 epochs with Data Augmentation **(c)** MIOU on validation set for model trained with and without Data Augmentation.

It is important to note that while the model trained on data without data augmentation outperformed the one trained with augmented data, data augmentation is still a powerful technique as it aids generalization. It could however involve lots of experimentation to find the best augmentations such that the model performs best.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| (a) | **(b)** | **(c)** | **(d)** |

**Figure 13:** Results of CCELD plus Skele-Marker vs UNet **(a)** Color image **(b)** Ground truth **(c)** CCELD plus Skele-Marker Mask **(d)** UNet Mask

**5. Conclusion and Future Works**

In this study, a novel technique for crack detection in images is introduced, namely “Crack Contrast Enhancement and Local Deviation (CCELD)”. This technique leverage image preprocessing towards obtaining efficient crack segmentation. The CCELD method utilizes the principle that the darkest regions in an image containing cracks can be used for effective identification. By applying convolution and algorithms containing thresholding and denoising, CCELD augmented with the Skele-Marker method successfully highlights crack regions, allowing for accurate detection. However, further experimentation and fine-tuning may be needed to further optimize the size of the mean kernel and other parameters and enhance the thresholding process for a broader range of image conditions. The range of parameter values is usually dependent on crack width.

CCELD augmented with the Skele-Marker denoising technique performed slightly better than the UNet model with an MIOU score of 0.06 and F1 score of 0.02 more than that of the UNet Model which indicates a slightly better performance. However, the CNN models are amazing and with tons of training data they will perform very well but will also require high computational resources which aren’t available to everyone. This study proposes a promising step towards accessible, efficient and automated crack detection for a developing region like Nigeria. Towards a practical implementation, a [web app](https://crack-detection-using-cnns-zd5cust8t9yfertfcammnj.streamlit.app/) that implements video and crack segmentation was created for the end users.

For future improvements, the following will be explored:

**1. Dataset Expansion:** Expansion of the dataset with a wider variety of crack images, including different crack types, sizes, and orientations, will be done to enhance the techniques’ generalization capabilities.

**2. Fine-Tuning:** More extensive experimentation to fine-tune the parameters of CCELD for optimal performance under various conditions will be conducted.

**3. Image wise Analysis:** Did the general parameters perform best for each image? If not, what happened? Can ideal parameter values be determined on an image level using ML? These questions will be answered going forward.

**4. Severity Classification:** Severity classification will be incorporated into the crack detection process. This addition would provide valuable insights into the extent of damage, enabling more informed maintenance decisions.

In conclusion, the presented techniques offer innovative approaches to crack detection that show promise in image preprocessing. By addressing the suggestions for improvement and conducting further validation, these methods have the potential to contribute significantly to the field of image-based crack detection and pave the way for safer and more efficient infrastructure maintenance.

**Acknowledgements**

We would like to extend our gratitude to several individuals whose contributions were integral to the completion of this study. Firstly, we express our sincere appreciation to Engr Prof. Nnolim for his guidance and support throughout the development and implementation of the novel preprocessing crack detection technique – Crack Contrast Enhancement and Local Deviation (CCELD). His expertise and insights greatly enriched the methodology and results of this research.

We are also thankful to AEDJAC Lab for their assistance in providing a comfortable and equipped working environment. Their contributions in providing these resources enabled the proficient completion of the project.

Without the invaluable support and collaboration from these individuals and organizations, this study would not have been accomplished. Their dedication and expertise have contributed immensely to the success of this research endeavor.

**Declaration of Generative AI and AI-assisted technologies in the writing process**

During the preparation of this work the author(s) used ChatGPT in order to implement few functions in python code. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

**References**

*Udoh,* *Logistics Regression using Gradient Descent | LinkedIn*. (n.d.). Retrieved October 15, 2023, from https://www.linkedin.com/pulse/logistics-regression-using-gradient-descent-udoh-chigozie/?trackingId=j8o2caXZSC2NSnetUCkrTw%3D%3D

*c++—Morphological Reconstruction in OpenCV - Stack Overflow*. (2020). Retrieved October 7, 2023, from https://stackoverflow.com/questions/29104091/morphological-reconstruction-in-opencv

Carrasco, M., Araya-Letelier, G., Velázquez, R., & Visconti, P. (2021). Image-Based Automated Width Measurement of Surface Cracking. *Sensors (Basel, Switzerland)*, *21*(22), 7534. https://doi.org/10.3390/s21227534

Cha, Y.-J., Choi, W., & Büyüköztürk, O. (2017). Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks: Deep learning-based crack damage detection using CNNs. *Computer-Aided Civil and Infrastructure Engineering*, *32*(5), 361–378. https://doi.org/10.1111/mice.12263

Chianese, R., Nguyen, A., Gharehbaghi, V., Aravinthan, T., & Noori, M. (2021). *Influence of image noise on crack detection performance of deep convolutional neural networks*.

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, *21*(1), 6. https://doi.org/10.1186/s12864-019-6413-7

Fujita, Y., Shimada, K., Ichihara, M., & Hamamoto, Y. (2017). A method based on machine learning using hand-crafted features for crack detection from asphalt pavement surface images. *Thirteenth International Conference on Quality Control by Artificial Vision 2017*, *10338*, 117–124. https://doi.org/10.1117/12.2264075

Gao, Q., Wang, Y., Li, J., Sheng, K., & Liu, C. (2020). An Enhanced Percolation Method for Automatic Detection of Cracks in Concrete Bridges. *Advances in Civil Engineering*, *2020*, e8896176. https://doi.org/10.1155/2020/8896176

*Google Colaboratory*. (2019). Retrieved November 12, 2023, from https://colab.research.google.com/github/d2l-ai/d2l-tvm-colab/blob/master/chapter\_gpu\_schedules/arch.ipynb#scrollTo=B7\_DogYoCPgJ

[Dataset] *Concrete Crack Segmentation*. Retrieved November 4, 2023, from https://datasetninja.com/concrete-crack-segmentation-dataset

Iakubovskii, P. (2023). *Qubvel/segmentation\_models* [Python]. https://github.com/qubvel/segmentation\_models (Original work published 2018)

Joshi, D., Singh, T. P., & Sharma, G. (2022). Automatic surface crack detection using segmentation-based deep-learning approach. *Engineering Fracture Mechanics*, *268*, 108467. https://doi.org/10.1016/j.engfracmech.2022.108467

Karas, P. (2011). Efficient Computation of Morphological Greyscale Reconstruction. In L. Matyska, M. Kozubek, T. Vojnar, P. Zemcík, & D. Antos (Eds.), *Sixth Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS’10) – Selected Papers* (Vol. 16, pp. 54–61). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. https://doi.org/10.4230/OASIcs.MEMICS.2010.54

Kheradmandi, N., & Mehranfar, V. (2022). A critical review and comparative study on image segmentation-based techniques for pavement crack detection. *Construction and Building Materials*, *321*, 126162. https://doi.org/10.1016/j.conbuildmat.2021.126162

Liu, F., & Wang, L. (2022). UNet-based model for crack detection integrating visual explanations. *Construction and Building Materials*, *322*, 126265. https://doi.org/10.1016/j.conbuildmat.2021.126265

Mumuni, A., & Mumuni, F. (2022). Data augmentation: A comprehensive survey of modern approaches. *Array*, *16*, 100258. https://doi.org/10.1016/j.array.2022.100258

Nguyen, A., Nguyen, C., Gharehbaghi, V., Perera, R., Brown, J., Yu, Y., & Kalbkhani, H. (2022). A computationally efficient crack detection approach based on deep learning assisted by stockwell transform and linear discriminant analysis. *Structures*, *45*, 1962–1970. https://doi.org/10.1016/j.istruc.2022.09.107

Nnolim, U. A. (2020). Fully adaptive segmentation of cracks on concrete surfaces. *Computers & Electrical Engineering*, *83*, 106561. https://doi.org/10.1016/j.compeleceng.2020.106561

Perez, L., & Wang, J. (2017, December 13). *The Effectiveness of Data Augmentation in Image Classification using Deep Learning*. arXiv.Org. https://arxiv.org/abs/1712.04621v1

Rosebrock, A. (2021, April 28). OpenCV Smoothing and Blurring. *PyImageSearch*. https://pyimagesearch.com/2021/04/28/opencv-smoothing-and-blurring/

Salman, M., Mathavan, S., Kamal, K., & Rahman, M. (2013). Pavement Crack Detection Using the Gabor Filter. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. https://doi.org/10.1109/ITSC.2013.6728529

*[Dataset] “SDNET2018: A concrete crack image dataset for machine learning applica” by Marc Maguire, Sattar Dorafshan et al.* (2018). Retrieved August 19, 2023, from https://digitalcommons.usu.edu/all\_datasets/48/

Skeleton-based noise removal algorithm for binary concrete crack image segmentation. (2023). *Automation in Construction*, *151*, 104867. https://doi.org/10.1016/j.autcon.2023.104867

Talab, A. M. A., Huang, Z., Xi, F., & HaiMing, L. (2016). Detection crack in image using Otsu method and multiple filtering in image processing techniques. *Optik*, *127*(3), 1030–1033. https://doi.org/10.1016/j.ijleo.2015.09.147

Zeeshan, M., Adnan, S. M., Ahmad, W., & Khan, F. Z. (2021). Structural Crack Detection and Classification using Deep Convolutional Neural Network. *Pakistan Journal of Engineering and Technology*, *4*(4), 50–56. https://doi.org/10.51846/vol4iss4pp50-56

*Zhang-Suen thinning algorithm*. (2023, October 18). Rosetta Code. https://rosettacode.org/wiki/Zhang-Suen\_thinning\_algorithm

Zhao, M., Shi, P., Xu, X., Xu, X., Liu, W., & Yang, H. (2022). Improving the Accuracy of an R-CNN-Based Crack Identification System Using Different Preprocessing Algorithms. *Sensors*, *22*(18), Article 18. https://doi.org/10.3390/s22187089