DEEPCRACK: LEARNING HYPERCONVOLUTIONAL FEATURES FOR AUTOMATIC CRACK DETECTION AND MEASUREMENT

PROJECT REPORT

Submitted by

SATHYA KUMAR.A

REG NO : TKM21CSCE07

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
Prof. Shameem Ansar

Thangal Kunju Musaliar College of Engineering
Kerala

JULY 2023

DECLARATION

I hereby declare that the project report on **"DeepCrack: Learning Hyper-convolutional Features For Automatic Crack Detection And Measurement,"** submitted for partial fulfillment of the requirements for the award of the degree of Master of Technology of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done under the supervision of **Prof. Shameem Ansar**, Assistant Professor of the Computer Science and Engineering Department, TKMCE. This submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact, or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Kollam
06/07/2023

SATHYA KUMAR.A

Thangal Kunju Musaliar College of Engineering
Dept. of Computer Science & Engineering

C E R T I F I C A T E

This is to certify that,this report titled ***Deepcrack: learning hierarchical convolutional features for crack detection*** is a bonafide record of the **Main Project** presented by **SATHYA KUMAR.A(TKM21CSCE07)**, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **M.Tech Computer Science & Engineering** in **APJ Abdul Kalam Technological University** .

Project Guide
Prof. Shameem Ansar
Assistant Professor
Computer Science & Engineering

Head Of The Department
Dr. Dimple A Shajahan
Professor
Computer Science & Engineering

Project Coordinator
Dr. Aneesh G Nath
Associate Professor
Computer Science & Engineering

ACKNOWLEDGEMENT

A successful main project is a fruitful culmination of efforts by many people, some directly involved and some others indirectly, by providing support and encouragement. Firstly I would like to thank the almighty for giving me the wisdom and grace for making my project a memorable one. I thank him for steering me to the shore of fulfillment under his protective wings.

I express my sincere gratitude to **Dr. T A Shahul Hameed**, Principal of T.K.M College of Engineering for giving me an opportunity to present my main project. I would like to thank **Dr. Dimple A Shajahan,** Professor and Head of the Department, CSE, TKMCE, for her constant support and encouragement throughout the work.

With a profound sense of gratitude, I would like to express my heartfelt thanks to my guide **Prof. Shameem Ansar,** Assistant Professor, CSE, TKMCE for their expert guidance, cooperation and immense encouragement. I also extend my thanks to the entire faculty members and staffs of the Department of Computer Engineering, TKMCE, who has encouraged me throughout this work.

I also express my thanks to my loving parents, brother and friends, for their support and encouragement in the successful completion of this main project work.

Kollam SATHYA KUMAR.A
06/07/2023

Abstract

DeepCrack is a continuously trainable deep convolutional neural network designed for automatic crack detection and measurement. Cracks are linear features that hold significant importance in various imaging applications, but their detection is challenging due to factors such as poor road conditions, low line quality, and low contrast. This work addresses these challenges by introducing hyperconvolution techniques in DeepCrack. The proposed method utilizes multi-scale depth convolution features learned through hyperconvolution stages to effectively capture line structures. By combining a larger feature map for a detailed perspective and a smaller feature map for a global view, DeepCrack can accurately identify cracks. The network architecture of DeepCrack is based on the encoder/decoder design of SegNet, allowing the fusion of convolution functions from both networks on the same scale. To evaluate the performance of DeepCrack, it was trained on one crack dataset and assessed on three additional crack datasets. The morphological characteristics of cracks, including length and breadth, were measured using a skeleton extraction approach. The experimental results demonstrate the effectiveness of the proposed technique, surpassing the performance of current state-of-the-art methods. DeepCrack achieves an average F-Measure exceeding 0.90 on the challenging datasets, showcasing its superior performance. The proposed technique demonstrates its potential for practical crack detection and measurement tasks, accurately capturing different types of cracks, including multiple cracks, thin cracks, and cross cracks. The findings highlight the successful application of the suggested technique for crack measurement. The superior performance of DeepCrack compared to existing methods underscores its significance in the field of crack detection.

Keywords: Crack Detection And Measurement, Deep Learning, DCNN, Crack Detection, segmentation, morphological.

Contents

List of Figures

Chapter 1

Introduction

Cracks are a common fault that may be noticed in a range of physical constructions, including the pavement of a road, the wall of a nuclear power plant, the ceiling of a tunnel, etc. Crack repair is a critical obligation for the security of engineering infrastructures and to prevent harm from spreading. For instance, a crack in a highway's pavement may easily develop into a hole in only one night of rain, making it risky for fast-moving automobiles. Automation of testing methods is greatly desired to improve testing efficiency and reduce testing expenses. A fracture is among the most common faults. The cost of maintenance can be considerably reduced by stopping a fracture from deteriorating. Even now, fully automated fracture detection in a noisy environment remains challenging.

Crack detection may be defined as line detection, a fundamental issue in computer vision, because a crack is a visibly linear or curved structure. Two viewpoints allow us to describe a fracture visually. It is narrow and frequently preserves the intensity of the jumps in the background, which causes the image to seem from a distance as having a one-pixel wide border. It appears to be a narrow line from a local perspective. Edge detection-based and picture segmentation-based approaches are hence the two basic categories into which fracture detection techniques may be split. Ideally, classic edge detection and picture segmentation algorithms might detect the fracture with high accuracy if it had strong continuity and high contrast. However, background noise can frequently lead to cracks, which have poor consistency and low contrast. The quality of the slit's picture can also be impacted by the exposure direction. Due to these issues, traditional low-level feature-based fracture detection algorithms frequently perform less well.

Deep convolutional neural networks (DCNN) have lately demonstrated state-of-the-art, human-competitive, and occasionally better-than-human performance in handling a number of computer vision problems, including image segmentation, picture classification, and object recognition, among others. DCNN-based algorithms have also been recommended for line detection for

applications including edge identification, contour detection, boundary segmentation, and so forth. In order to produce high-level characteristics from low-level primitives, these deep architectures convolve the sensory inputs in a hierarchical fashion.

It has been discovered, in particular, that the performance of edge detection can be enhanced by combining the detailed features in the larger-scale layers and the abstracted features in the smaller-scale layers when using deep learning for edge detection. The convolutional features used in the convolving-pooling pipeline when using deep learning for edge detection also become increasingly coarse. When using deep learning for image segmentation, such as the SegNet, it has been discovered that convolutional features in the decoder network are helpful for enhancing the performance of semantic image segmentation. Additionally, indexing of pooling positions can further improve accuracy of boundary localization.

Propose a novel DeepCrack network for crack identification in response to these discoveries by fusing the convolutional features of encoder and decoder networks. The encoder-decoder architecture stated in SegNet is the foundation for DeepCrack. On the same size, the convolution phase of the decoder network and the encoder network in SegNet are equivalent. In DeepCrack, pairwise convolutions of the encoder network and the decoder network at each scale are utilised to first build a single-scale composite feature map. Then, the fused feature maps at each scale are combined to create a multi-scale composite map for crack detection. Fig.1.1 shows this to illustrate; the bottom row shows the combined map objects at different scales.Improved crack-detection performance is achieved by fusing the continuous feature at larger scales and the sparse feature at smaller scales.

The analysis of crack images in structures provides valuable insights into their geometry, including characteristics such as width, length, and direction. This information is crucial for evaluating the necessary safety precautions and maintenance measures required for the structure. To extract morphological components and skeletonize crack images, various techniques can be employed. These techniques include the use of algorithms such as medial-

axis, Hilditch's, and 3D-medial axis thinning. The goal of these algorithms is to simplify the crack image representation while preserving its essential geometric features. One common approach is the medial-axis algorithm, which identifies the centerline or skeleton of the crack. The algorithm determines the thinnest path within the crack image, representing its width and allowing for accurate width measurement. Similarly, Hilditch's algorithm is another technique used for skeletonization, aiming to extract the crack's central axis by iteratively thinning the crack regions. In cases where crack images are captured in three-dimensional space, the 3D-medial axis thinning algorithm can be applied. This algorithm extends the medial-axis approach to three dimensions, enabling the extraction of a 3D skeleton representing the crack's geometric characteristics.

By skeletonizing the crack images using these algorithms, it becomes possible to measure the crack width accurately. Furthermore, a crack defragmentation method can be employed to determine the average width by segmenting the crack image into distinct fragments and then calculating the average width across these fragments. Through the use of morphological processes such as segmentation and skeleton extraction, essential morphological characteristics of cracks can be quantified. This quantification allows for a more detailed understanding of crack geometry, enabling engineers and maintenance personnel to assess the severity of cracks accurately and determine the appropriate safety and maintenance measures needed.

The DeepCrack network, which is a specialized system designed specifically for crack identification. It aims to improve the accuracy and efficiency of crack detection by utilizing deep learning techniques. Deep learning is a subfield of artificial intelligence that focuses on training neural networks with multiple layers to recognize patterns and extract features from data. In the context of crack detection, deep learning allows the network to automatically learn and recognize complex patterns and characteristics associated with cracks, which traditional methods may struggle to capture effectively.

The DeepCrack network takes advantage of the capabilities of deep learning by integrating features from multiple scales. This means that the network

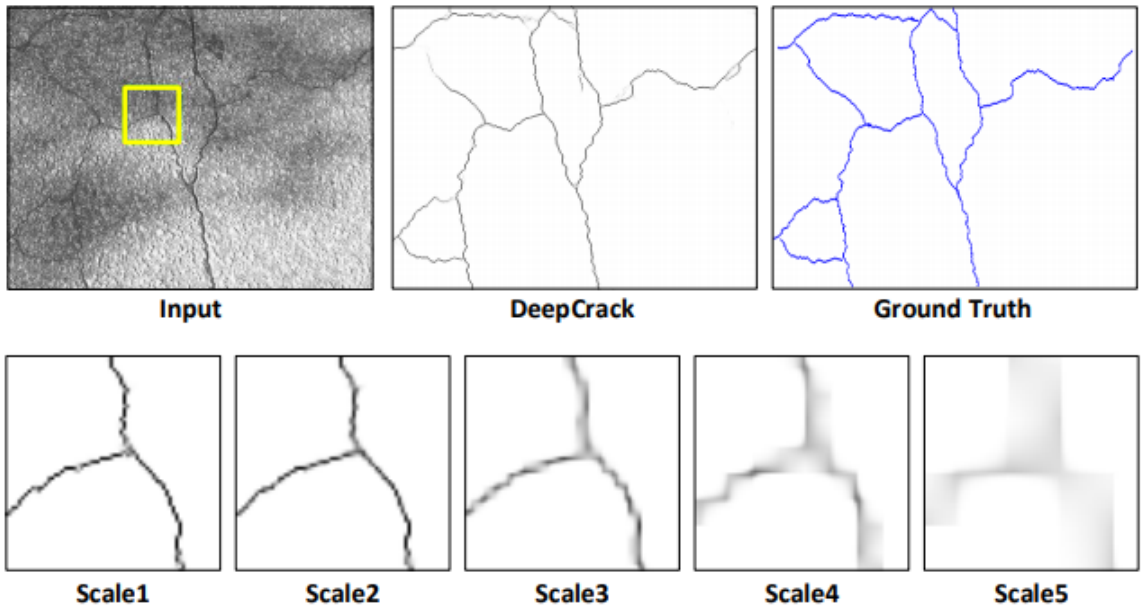

Figure 1.1: A real example of crack detection using DeepCrack.

considers various levels of detail when analyzing crack images or scans. By incorporating features from different scales, the network can capture both global and local characteristics of cracks, improving the overall performance of crack detection. This multi-scale approach allows DeepCrack to detect cracks accurately, even in challenging scenarios where cracks may be subtle or partially obscured.

In addition to crack detection, the project also focuses on the extraction of morphological components for crack analysis. Morphology refers to the study of the shape and structure of objects. In the context of crack analysis, morphological techniques involve examining the geometric properties of cracks, such as their shape, length, width, and orientation. This analysis provides valuable insights into the characteristics and severity of cracks. By combining deep learning techniques with morphological analysis, the DeepCrack network offers a comprehensive approach to crack detection and analysis. It can accurately identify cracks by leveraging the power of deep learning to recognize intricate patterns and features. Furthermore, the morphological analysis provides additional information about crack geometry, enabling engineers and maintenance personnel to make informed decisions about the

severity of cracks and the necessary repair or maintenance actions. Overall, the DeepCrack network represents an innovative solution that harnesses the capabilities of deep learning and integrates morphological analysis to improve crack detection performance and provide valuable insights for safety and maintenance assessments.

Chapter 2

Related Works

2.1 Line Detection

Line detection is a crucial aspect of computer vision, and it can be achieved through different methods such as edge/contour detection and line object detection. In the case where edges and contours are visible as gradients, their identification in a gradient map can be treated as line object detection or line grouping. Over the past few decades, the research on edge and contour detection has gone through three main stages. In the first stage, the widely used Canny edge detector is employed as a representation. This involves computing the first- or second-order gradients on the pixel intensity to identify edges.

The second stage focuses on edge detection and contour grouping, and it is characterized by the use of energy reduction techniques and middle-level feature learning algorithms. Notably, the introduction of techniques like sketch token and structural edge detector has allowed for the utilization of global Pb, a representative learning technique for edge detection, to achieve its maximum learning potential. For contour detection, methods like ratio contour level set and untangling cycles are employed to model line clutters using graphs and minimize the energy function to infer the contour.

In the third stage, deep learning techniques are applied to edge and contour detection. For example, deep learning approaches are used for edge identification, contour detection, and boundary segmentation to recognize edges and contours. In the context of edge identification, a multi-scale version is developed where parts of the image are predicted from picture patches under a deep learning architecture. Deep Convolutional Neural Networks (DCNN) are used to handle edge detection and line object extraction by combining feature abstraction with neighbor search. End-to-end edge detection is achieved through a deep convolutional network that exploits convolutional features in multiple convolutional stages, resulting in improved edge detection results. Similarly, richer convolutional features generated by a fully convolutional network are combined to enhance performance further.

Overall, the research on edge and contour detection has progressed through different stages, from traditional techniques like the Canny edge detector to energy reduction approaches and middle-level feature learning algorithms, and finally to the utilization of deep learning methods. These advancements have improved the accuracy and effectiveness of edge and contour detection in computer vision applications.

2.2 Crack Detection

In crack detection, cracks typically have a darker background compared to the surrounding area under normal illumination. This characteristic makes the picture thresholding method a simple approach for fracture detection. A heuristic formula is often used to determine the threshold value, making it easier to separate the cracks from the background. However, the robustness of thresholding-based approaches can be compromised by factors such as

pavement shadows and uneven lighting conditions. To address this issue, various techniques derived from edge detection and wavelet transformation have been developed for crack identification. Since cracks are narrow and appear as edges, these techniques aim to detect the edges corresponding to cracks. However, the presence of noise in the images can interfere with accurate edge detection and potentially affect the results.

Minimal path searching, which is a subset of energy reduction methods, has also been studied for crack identification. Techniques based on minimal path searching were proposed for detecting pavement cracks. Path voting and searching were employed to locate cracks in complex backgrounds. However, a disadvantage of these minimal-path-based approaches is that they require presetting seed locations for the route tracking, which can be limiting in certain situations.

Machine learning-based techniques have also been investigated for crack identification. For example, a deep convolutional neural network was used to classify image patches into crack blocks and non-crack areas. An updated active contour model and a support vector machine based on greedy search

were employed for the identification of bridge cracks. In nuclear power plants, fully convolutional neural networks were tested using multi-view photos to detect flaws. Other techniques, such as saliency detection, structure analysis employing the least spanning tree, and random structure forest, have also been proposed for crack identification. Overall, deep learning-based methods have shown superior performance compared to conventional techniques in terms of crack identification. These methods leverage the power of deep convolutional neural networks to learn and extract features that are effective for crack detection. By training on large datasets, deep learning models can generalize well and provide accurate crack identification results.

2.3 Machine Learning

In recent times, several methods have been proposed for crack detection. One such method involves using a support vector machine (SVM) to detect fractures in aircraft skin. The SVM is a supervised learning algorithm that can classify data points into different classes. In this case, the SVM is trained on a dataset of crack images to learn the characteristics of cracks and then used to classify new images and identify cracks in the aircraft skin.

Another approach, introduced by Oliveira and Correia, is a system called CrackIT. It is an unsupervised learning system designed to find cracks. The system utilizes various techniques and algorithms to automatically detect cracks in images without the need for manual labeling or supervision. Building upon their initial work, Oliveira and Correia further developed the CrackIT toolkit, which expands the capabilities and functionalities of the original system. This toolkit enables more efficient and effective crack detection. To differentiate between crack and non-crack pixels, a novel descriptor was developed using a random structure forests approach. The random structure forests approach is a machine learning technique that combines random forests with structured data representations. In this context, the descriptor is designed to capture the unique characteristics and patterns associated with crack pixels, enabling accurate differentiation between crack and non-crack areas in an image.

Inspecting road cracks poses challenges for operators due to factors such as feature descriptor layering and complicated road conditions. Road surfaces often have multiple layers, and cracks can occur at different depths, making their detection more complex. Additionally, the presence of debris, shadows, and various road conditions can further complicate the identification of cracks. Therefore, developing robust methods and tools for road crack inspection is crucial for effective maintenance and safety.

In summary, recent advancements in crack detection have introduced various methods and tools. These include using SVMs for fracture detection in aircraft skin, the development of the CrackIT system for unsupervised crack detection, the expansion of the CrackIT toolkit, and the use of novel descriptors based on random structure forests. Addressing challenges in road crack inspection remains a focus, considering the complexity of feature descriptor layering and diverse road conditions. Continued research and development in these areas contribute to improved crack detection techniques for various applications.

2.4 Ensemble Learning

In the field of medical image classification, an ensemble network has been proposed as a method to improve the accuracy of classification. Wen et al. designed an ensemble network based on probability fusion for facial expression recognition. This means that multiple networks are trained independently, and their predictions are combined or fused using probability fusion techniques. By combining the predictions of multiple networks, the ensemble network can leverage the diverse perspectives and knowledge learned by each individual network, resulting in improved classification performance for medical images, specifically in the context of facial expression recognition.

Similarly, Maji et al. proposed an ensemble network for the detection of retinal vessels. Retinal vessels play a crucial role in various medical conditions and can provide valuable information for diagnosing and monitoring diseases. The ensemble network developed by Maji et al. combines the predictions of multiple networks to enhance the accuracy and reliability of vessel detection

in retinal images. By aggregating the outputs of individual networks, the ensemble network can capture the complementary information from each network, improving the overall performance of vessel detection.

Ensemble networks offer a powerful approach to medical image classification by harnessing the collective wisdom of multiple networks. Each network within the ensemble may have different architectures, training strategies, or input representations, resulting in diverse and complementary insights. By combining the predictions of these networks, either through probability fusion or other fusion techniques, the ensemble network can achieve higher accuracy, robustness, and generalization in medical image classification tasks. These ensemble-based approaches contribute to advancements in medical diagnosis, treatment planning, and overall patient care.

Chapter 3

Proposed Network

- Designing a novel, neural network architecture for fracture identification and crack measuring was our proposed system. This network creates a trainable end-to-end network for crack identification and measurement by making full use of the encoder and decoder network's information.
- In the proposed network, a convolutional layer of the encoder network and a convolutional layer of the decoder network at the same size are fused to compute the training loss at the corresponding scale. For inferring the cracks out from the backdrop of the picture, it is discovered that the fusion of hierarchical convolutional features is extremely successful.
- Extraction of the morphological components, which may be segmented into a thinned crack skeleton, allows for measurement of the expected crack image's width. To determine the average crack width, a crack defragmentation method was suggested. Finally, the skeleton extraction technique was used to measure the expected morphological characteristics of cracks. The anticipated crack maps can be used to calculate the crack's width and length.

3.1 DeepCrack Architecture

The DeepCrack network (Figure 3.1) is constructed based on the SegNet architecture, which is a pixel-wise semantic segmentation algorithm. SegNet combines a deep convolutional encoder-decoder structure with a decoder network. The encoder network in SegNet draws inspiration from the convolutional layers of the VGG16 network. The VGG16 network comprises a total of 13 convolutional layers and 5 down-sampling pooling layers. Similarly, the encoder network in SegNet also consists of 13 convolutional layers. The decoder network in SegNet is designed to be symmetric to the encoder network, with each layer corresponding to a layer in the encoder. The key distinction lies in the first encoder layer, where the initial convolution operation generates a multi-channel feature map. Conversely, the final decoder layer, representing the last convolution operation, produces a c-channel fea-

ture map, where 'c' signifies the number of classes involved in the image segmentation task.

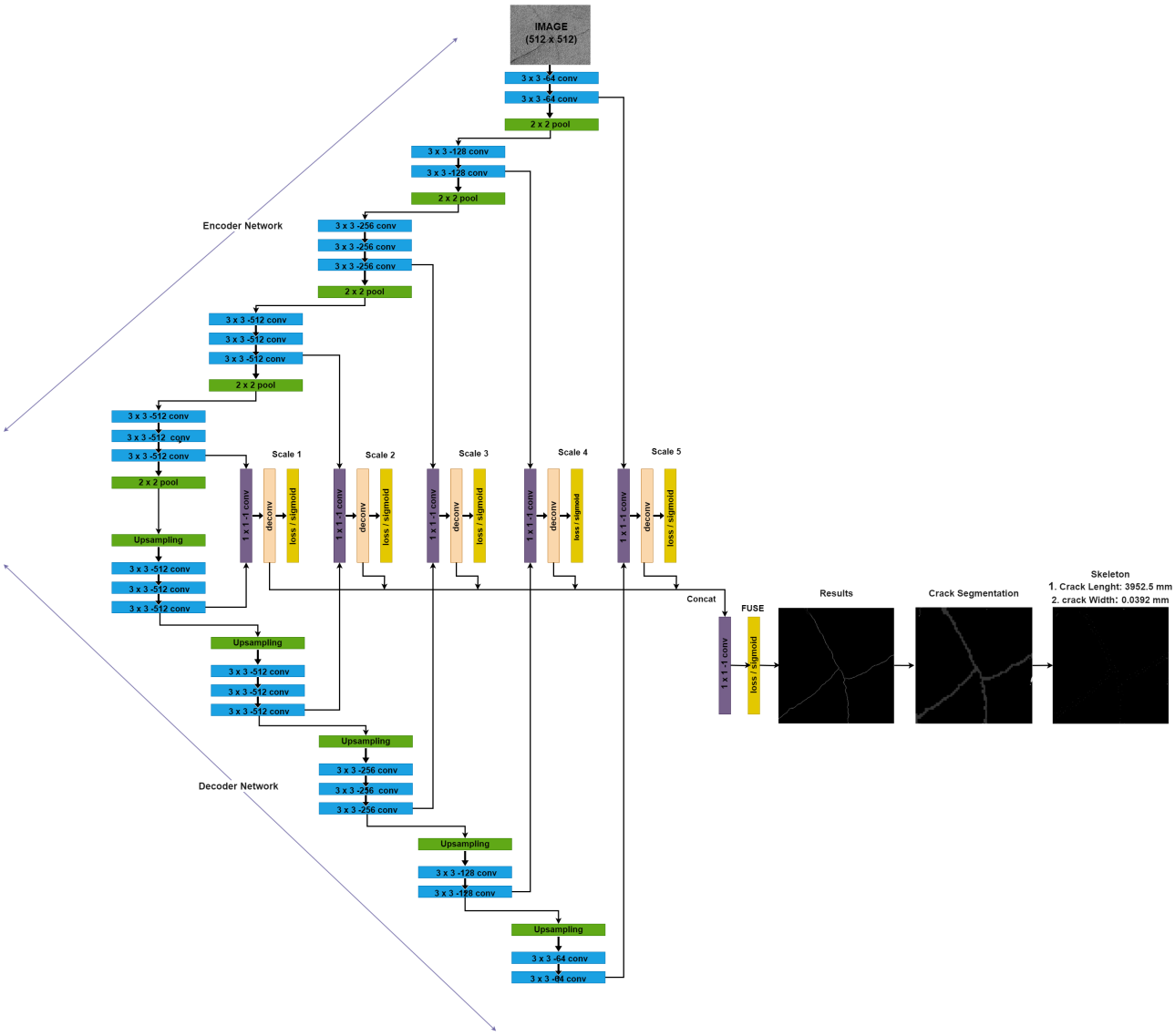

Figure 3.1: DeepCrack Architecture

After each convolution step, the feature maps go through a batch normalization procedure. The max-pooling procedure with a stride bigger than 1 can lower the scale of feature maps without creating translation variance over minute spatial shifts, despite the loss of spatial resolution and potential border bias caused by sub-sampling. To avoid a lack of detail representa-

tion during sub-sampling, the boundary information in the encoder feature maps is acquired and recorded using max-pooling indices. The appropriate decoder layer in the decoder network then uses the max-pooling indices to execute non-linear upsampling. Sparse feature maps will be produced by this upsampling step. However, the sparse feature maps achieve more accurate localization of region boundaries in comparison to continuous and dense feature maps.

In the meanwhile, due to the nature of hierarchical learning in deep convolutional neural networks, multi-scale convolutional features can be learnt in the down-sampled layers as increasingly bigger receptive fields. It has been demonstrated that combining multiscale convolutional features improves the performance of line detectors. Each scale's convolutional layer prior to the pooling layer in the encoder network is concatenated with that scale's final convolutional layer in the decoder network. The skip-layer fusion performs a series of computations on the concatenated convolutional features.

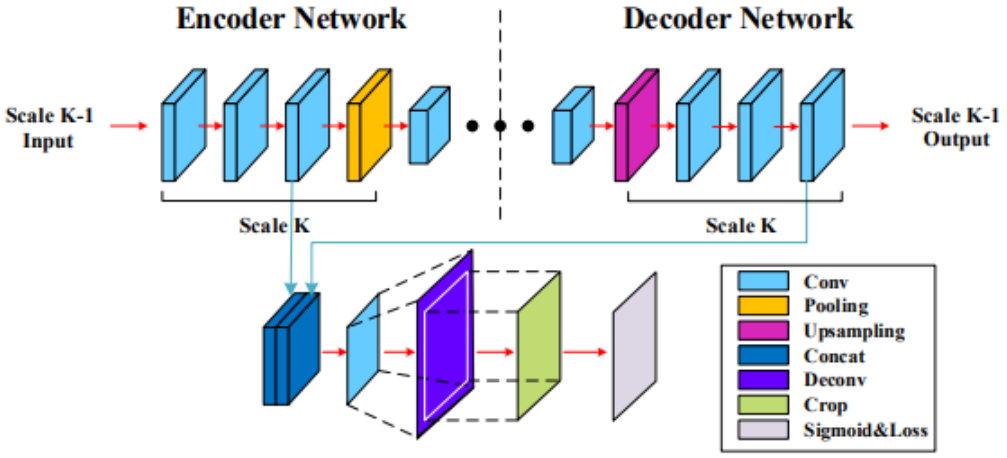

Figure 3.2: An illustration of skip-layer fusion at scale K.

Figure 3.2 depicts the skip-layer fusion in excellent detail. The feature maps from the encoder and decoder networks are first concatenated, and then the multi-channel feature maps are condensed into a single channel using a 1x1 convolution layer. In order to compute the pixel-wise prediction

loss in each scale, the feature map is then up-sampled with a deconv layer and cropped with a crop layer to suit the size of the input picture. With the use of these techniques, we can produce prediction maps at each scale that are the same size as ground-truth crack maps. The five different scales' prediction maps are further concatenated, and a 1x1 convolution layer is added to combine the results from all scales. The prediction maps will eventually be available to us.

3.2 Crack Measurements

In this section, the main experimental results are presented together with a discussion of the fracture assessment procedure in detail. The morphological elements could be estimated and retrieved from binary crack images once the detected crack images had been obtained by the DeepCrack network. For the purpose of producing segmented crack images, the expected binary crack images are labelled. The segmented crack photos were thinned in order to create crack skeleton images, which only displayed the crack's outline in one pixel. Using crack skeleton images, calculate crack morphological characteristics. Based on the expected crack maps, the width and length of the crack may be calculated.

3.2.1 Crack Segmentation

Each crack pixel from the crack photos has to be labelled in order to segment them and distinguish them from the backdrop. To fill small gaps in a crack image, the closure operation based on morphological operations is utilized (Figure 3.3). This operation aids in the removal of tiny holes and noisy pixels, allowing for better segmentation and labeling of each crack pixel. The closure operation can be represented by the following equation:

$$Closing = ((f \oplus \psi) \ominus \psi) \tag{3.1}$$

where $\oplus$ and $\ominus$ are the dilation and erosion operations for the morphological operation, respectively. f and ψ are the crack image and the structure element, respectively. The operation $\oplus$ is used to increase the regions of the crack pixels and the operation erodes the boundary regions of the crack pixels.

Warping off noisy pixels uses an opening procedure based on morphological operations. The opening operation has a different priority order for the erosion and dilation operation compared to the closing operation. The definition of the opening action is,

$$opening = ((f \ominus \psi) \oplus \psi) \tag{3.2}$$

The process of segmenting a picture entails assigning various labels to various areas of the image. As a result, segmentation images were created by labelling the individual cracks. After identifying each crack, the crack images can be divided into segments.

Figure 3.3: Crack Segmentation.

3.3 Crack Skeleton

The purpose of crack skeletons, as illustrated in Figure 3.4, is to reveal the topology of cracks by representing them as a single-pixel width fracture. These crack skeletons are valuable for road maintenance and structural health monitoring as they provide a reference value. In this paper, a medial-axis technique is employed to achieve real-time detection and skeletonization

of fractures. When the crack skeletons are only one pixel wide, the crack length can be calculated using the following formula:

$$L_{\text{crack}} = \sum f(x, y)\mathrm{d}l \tag{3.3}$$

where f(x, y) and dl are the calibrated displacements of the pixels in the crack images and the finite length of the crack skeleton elements, respectively. Since there won't be any geometric distortion in this project, this is what we'll assume. Thus, the uniqueness of f(x, y) is defined.

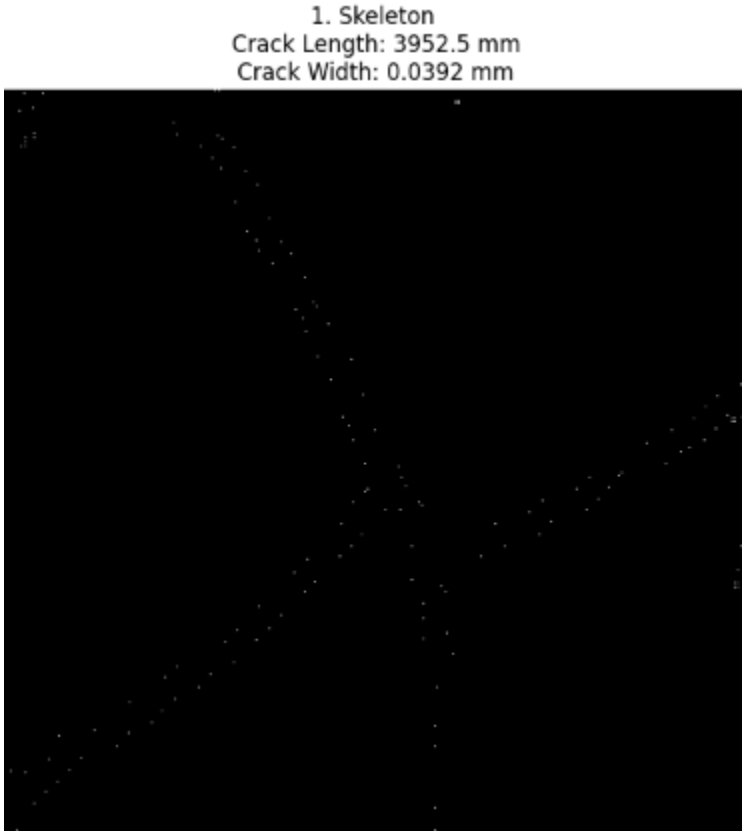

Figure 3.4: Crack Skeleton.

Calculate the length of the cracks by dispersing the pixels for the crack skeletons at the same time. The following equation can be used to simulate the average crack width:

$$W_{\text{avg}} = \frac{\sum f^2(x,y)\, ds}{L_{\text{crack}}} \tag{3.4}$$

where ds is the finite area of the crack elements. Decide the true length and width of a fracture depending on the image resolution. The security performance of a pavement crack, for example, can be evaluated and estimated with the aid of these reference values by engineers.

3.4 Loss Function

Given a training data set containing N images as:

$$S = (x^n, y^n), n = 1...N \tag{3.5}$$

$$Where,\ x^n = x_i^n, i = 1, \ldots, I \tag{3.6}$$

$$y^n = \{y_i^n, i = 1, \ldots, I, y_i^n \in \{0, 1\}\} \tag{3.7}$$

Eq 3.6 denotes the raw input image. Eq 3.7 denotes the ground-truth crack label map. I denotes the number of pixel in every image, our goal is to train the network to produce prediction maps approaching the ground truth.

In the encoder-decoder architecture, let K be the number of convolution stages, then at the stage k, the feature map generated by the skip-layer fusion may be written as, Additionally, the multi-scale fusion map is described as:

$$f^k = \{f_i^k, i = 1, \ldots, I\} \tag{3.8}$$

$$f^{\text{fuse}} = \{f_i^{\text{fuse}}, i = 1, \ldots, I\} \tag{3.9}$$

Define the pixel-wise prediction loss as:

$$l(F_i;W) = \begin{cases} log(1 - p(F_i;W)) & \text{,if } \mathrm{y}_i = 0 \ , \\ log(p(F_i;W)) & \text{,otherwise,} \end{cases} \tag{3.10}$$

It is defined as follows: when the ground truth label yi for pixel i is 0 (indicating a non-crack pixel), the loss is calculated as the logarithm of 1 minus the predicted crack probability p(Fi; W); otherwise, when yi is 1 (indicating a crack pixel), the loss is computed as the logarithm of the predicted crack probability. Here, Fi represents the output feature map of the network at pixel i, W denotes the standard parameters in the network layers, and p(F) is the sigmoid function used to convert the feature map into a crack probability map. The overall loss may then be calculated as follows:

$$L(w) = \sum_{i=0}^{I} (\sum_{k=1}^{k} l(f_i^k;W) + l(f_i^f use;W)) \tag{3.11}$$

In this equation, $L(w)$ denotes the total loss, which is calculated by summing over i from 0 to I, where I represents the total number of pixels or samples in the dataset. For each pixel i, the equation comprises two terms. The first term involves an inner summation over k, ranging from 1 to K, representing the sum of individual pixel-wise prediction losses, denoted as $l(f_i^k;W)$. These losses are obtained by applying a specific prediction loss function to the feature maps f_i^k, which correspond to different layers or stages in the network.

The second term in the equation, $l(f_i^f use;W)$, accounts for the prediction loss of a fused or combined feature map, $f_i^f use$. This fused feature map is generated by aggregating or combining the individual feature maps from various layers. By summing up the losses for each feature map and the fused feature map loss across all pixels, the overall loss function $L(w)$ quantifies the overall discrepancy between the predicted crack probabilities and the ground truth labels. By using this loss function, the network is trained to minimize the deviation between predicted and ground truth crack probabilities, thus enhancing its crack detection performance.

Chapter 4

Experiments And Results

4.1 Experimental Settings

Introduce the experimental conditions first in this part, and then discuss the crack detection and measurement findings produced by the DeepCrack network. Finally, research how DeepCrack performs under various conditions.

4.1.1 Implementations details

The implementation of the network can utilize the image processing facility available in the college research lab. This facility provides the necessary resources and infrastructure to process and analyze images efficiently.
In this network, batch normalization is applied after each convolutional layer in both the encoder and decoder networks. Batch normalization is a technique that helps accelerate the training process by normalizing the input to each layer. It reduces the internal covariate shift, which is the change in the distribution of layer inputs during training. By normalizing the inputs, batch normalization helps stabilize and speed up the convergence of the training process. The biases in the network are initialized to 0, and the weights of the convolutional layers are initialized for the entire network. Weight initialization is an important step in training neural networks, and initializing the weights properly can significantly impact the learning process and the final performance of the network.

The initial global learning rate, which determines the step size during optimization, is set to 1e-3 throughout the training process. This learning rate controls how much the network's parameters are adjusted in response to the estimated error. The decay rates for momentum and weight are set at 0.9 and 0.0005, respectively. These decay rates control the rate at which the parameters are updated during training, helping to prevent overfitting and improving the network's generalization ability. During training, the network parameters are updated using optimization methods such as stochastic gradient descent (SGD) and adaptive moment estimation (adam). These optimization methods adjust the weights and biases based on the gradients of

the loss function, iteratively improving the network's performance.

A mini-batch size of two is used in each iteration, which means that two samples are processed simultaneously in each training step. This helps speed up the training process and allows for more efficient memory usage. The model is trained for a total of 500 epochs, which refers to the number of times the entire training dataset is passed through the network. Training the network for multiple epochs allows it to learn and refine its parameters iteratively, improving its performance over time. The DeepCrack network is implemented using Python and built on PyTorch libraries. PyTorch is a popular deep learning framework that provides tools and utilities for building and training neural networks efficiently. For testing purposes, the HP Z8 G4 workstation with 32GB(4) DP Graphics in the GPU is used. This workstation provides high-performance computing capabilities, allowing for efficient testing and evaluation of the DeepCrack network.

Overall, the provided information describes the implementation details and the resources used for training and testing the DeepCrack network, including the specific configurations, libraries, and hardware utilized in the process.

4.2 Datasets

This study uses four crack datasets, with the pavement crack dataset CrackTree260 being used for training deep networks while the other three are utilised for testing. The test datasets' pictures all have the same 512 x 512 size. Using a specialised labelling tool, the ground-truth flaws are highlighted.

4.2.1 CrackTree260

It contains 260 road pavement images. These pavement images are captured by an area-array camera under visible light illumination. utilise all 260 pictures for instruction. The training set's size has been increased by data augmentation. Then flip the picture in both the vertical and horizontal directions at each angle, rotate the images with 9 different angles (from 0-90 degrees at an interval of 10), and crop 5 subimages (with 4 at the corners

and 1 in the centre) on each flipped image with a size of 512 x 512. Get a training set of 35,100 photos overall after augmentation.

4.2.2 CRKWH100

It includes 100 road pavement pictures that a line-array camera under visible light illumination took. The pavement is captured by the line-array camera at a ground sampling distance of 1 millimetre.

4.2.3 CrackLS315

It includes 315 photos of road pavement taken with a laser. At the same ground sampling distance, a line-array camera likewise records these pictures.

4.2.4 Stone331

There are 331 pictures of stone surfaces in it. On the cutting surface, fractures may appear when cutting the stone. An area-array camera using visible light for illumination obtained these photos. then create a mask for each stone surface's respective region in the image. The performance assessment can thus be limited inside the stone surface.

4.3 Results

The cornerstone of this study is an automated fracture identification and measurement system for photos. The main objective is to build a deep learning model capable of accurately separating fractures in test photos(Figure 4.1). The output of this system encompasses crack segmentation, storage of predicted results, calculation of crack length and width, and determination of segmentation correctness by comparing with ground truth pictures. These operations are executed on a specialized PyTorch dataset known as Test-Dataset, which handles test image loading and transformation. This dataset performs tasks such as resizing, format conversion, and normalization. Subsequently, a data loader is created to efficiently process test data in batches, enabling seamless handling of a large number of test images.

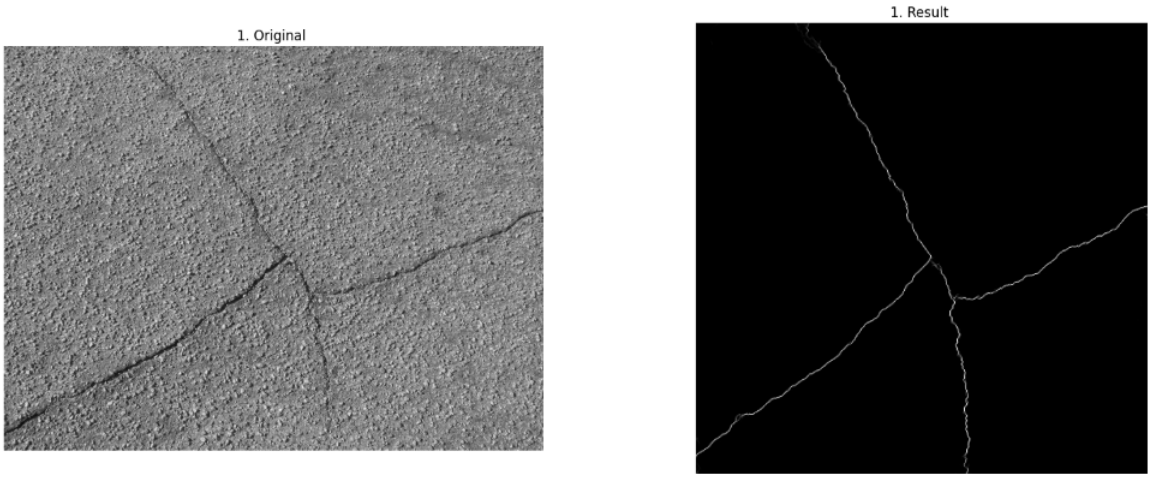

Figure 4.1: Result on Predicted Maps.

The many morphological processes to enhance the photographs quality. Small gaps in the image are filled in with the first operation, and noisy pixels are removed with the second. The system improves the crack segmentation's accuracy by using these techniques. There is a part of the that allows for further analysis by labelling the related components (crack segments) in the picture. The crack segmentation picture is additionally transformed into a skeleton representation using a crack skeletonization procedure.

Using the skeletonized image, the system calculates the length of cracks. Furthermore, it determines the average width of cracks by analyzing the width of cracks in the skeletonized image. These measurements provide valuable quantitative information about the fractures. To ensure the reliability of the segmentation and measurement results, the system compares them to ground truth pictures. This validation process assesses the correctness of the segmentation algorithm and validates the model's performance.

In summary, this study presents a deep learning model for automated fracture identification and measurement in photos. The model accomplishes accurate crack segmentation, precise crack length and width calculations(Figure 4.2), and validation against ground truth images. The utilization of Test-Dataset and the data loader facilitates efficient processing of a large number of test images. Morphological operations, including filling small holes and

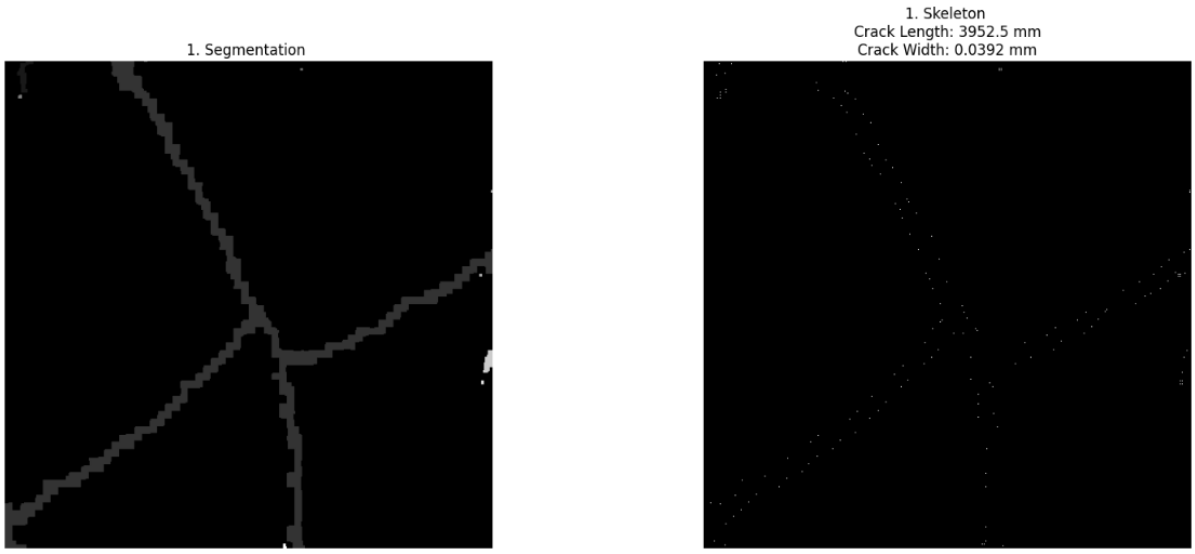

Figure 4.2: Result on Crack Segmentation and Calculates the length and width of cracks in the skeletonized image.

discarding noisy pixels, enhance the quality of the segmented images.

4.4 Evaluation and Accuracy Calculation

4.4.1 Accuracy

The accuracy of the crack segmentation is determined by calculating the ratio of the total number of matching pixels to the total number of pixels in the evaluation set. It is calculated as:

$$accuracy = \left(\frac{\text{total\_matching\_pixels}}{\text{total\_pixels}} \right) \times 100 \tag{4.1}$$

The model has been tested on three different datasets: CRKWH100, CrackLS315, and Stone331. Notably, each dataset yields varying accuracy results. For the CRKWH100 dataset, the model achieved an accuracy of 99.81 percentage. The CrackLS315 dataset exhibited slightly higher accuracy, with a score of 99.75 percentage. However, the Stone331 dataset outperformed the others, attaining an impressive accuracy of 99.83 percentage. The accuracy results clearly indicate that the model performs exceptionally well in crack segmentation. Among the tested datasets, the Stone331 dataset achieved the highest

accuracy, demonstrating the model's effectiveness in accurately detecting and segmenting cracks within this dataset. This outcome suggests that the model has a strong capability to successfully identify and delineate cracks specifically in the Stone331 dataset. The accuracies of three test datasets are plotted in the figure 4.3.

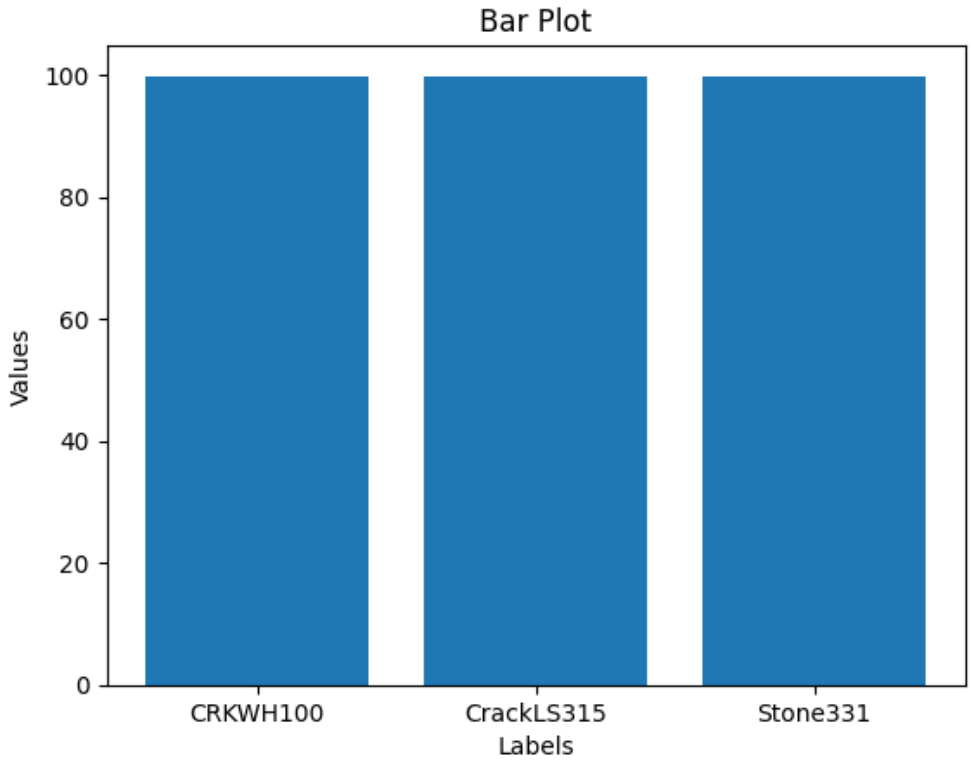

Figure 4.3: The accuracies of three test datasets.

In image and video processing, a more significant measure of prediction is considered to be F -measure rather than accuracy. In image and video processing tasks such as object detection, segmentation, or classification, accuracy alone may not provide a comprehensive evaluation of the prediction quality. Accuracy measures the overall correctness of the predictions by comparing the total number of correctly classified pixels or objects to the total number of pixels or objects in the dataset. However, accuracy does not account for the trade-off between precision and recall. Precision represents the proportion of correctly predicted positive instances (true positives) out of all

predicted positive instances (true positives + false positives). Recall, also known as sensitivity, measures the proportion of correctly predicted positive instances (true positives) out of all actual positive instances (true positives + false negatives).

4.4.2 F -measure

The F-measure, also known as the F1 score, is a harmonic mean of precision and recall. It provides a balance between these two measures and is often used in image and video processing tasks. The F-measure combines precision and recall into a single metric, giving equal importance to both measures. It is calculated as:

$$F - Measure = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{4.2}$$

This formula calculates the F-measure by taking the harmonic mean of precision and recall, giving equal weight to both measures. The numerator, 2 × precision × recall, represents the product of precision and recall multiplied by 2. The denominator, precision + recall, represents the sum of precision and recall. Dividing the numerator by the denominator gives the F-measure, which provides a balanced evaluation of the prediction performance, considering both precision and recall.

In the evaluation process, three different F-measure-based metrics are utilized. These metrics are used to measure the performance of a model or algorithm in tasks such as object detection or segmentation. The three metrics are:

- ODS (Optimal Dataset Scale): This metric calculates the best F-measure achieved on the dataset using a fixed threshold. It evaluates the model's overall performance on the entire dataset by determining the threshold that maximizes the F-measure.
- OIS (Optimal Image Scale): This metric calculates the aggregate F-measure on the dataset by selecting the best threshold for each individual image. It accounts for variations in optimal thresholds across differ-

ent images and provides an overall evaluation of the model's performance across the dataset.

- AP (Average Precision): This metric measures the performance of the model by calculating the area under the precision-recall curve. Precision represents the proportion of true positive predictions out of all positive predictions, while recall measures the proportion of true positive predictions out of all actual positive instances. The precision-recall curve plots these two measures against different thresholds, and the AP is obtained by calculating the average precision across all thresholds.

These three F-measure-based metrics provide different perspectives on the model's performance, taking into account various aspects such as fixed thresholds, optimal thresholds per image, and precision-recall trade-offs. They offer comprehensive evaluations for image and video processing tasks, allowing for a more thorough analysis of the model's effectiveness. In this work, the accuracy values for three datasets, along with the corresponding metrics, are presented as follows:

- For the 'CRKWH100' dataset, the accuracy metrics are as follows: ODS (Optimal Dataset Scale) achieves a value of 0.9708, OIS (Optimal Image Scale) achieves a value of 0.8464, and AP (Average Precision) achieves a value of 0.7111.

- For the 'CrackLS315' dataset, the accuracy metrics are as follows: ODS achieves a value of 0.8849, OIS achieves a value of 0.6790, and AP achieves a value of 0.4531.

- For the 'Stone331' dataset, the accuracy metrics are as follows: ODS achieves a value of 0.8651, OIS achieves a value of 0.5670, and AP achieves a value of 0.3747.

The accuracy values presented in Figure 4.4 provide valuable insights into the performance of the proposed DeepCrack model on different datasets, using various F-measure-based metrics. These accuracy values allow for a comprehensive evaluation of the model's effectiveness in accurately detecting and classifying cracks in images across different scenarios and datasets. By examining the accuracy metrics, researchers can gain a deeper understanding

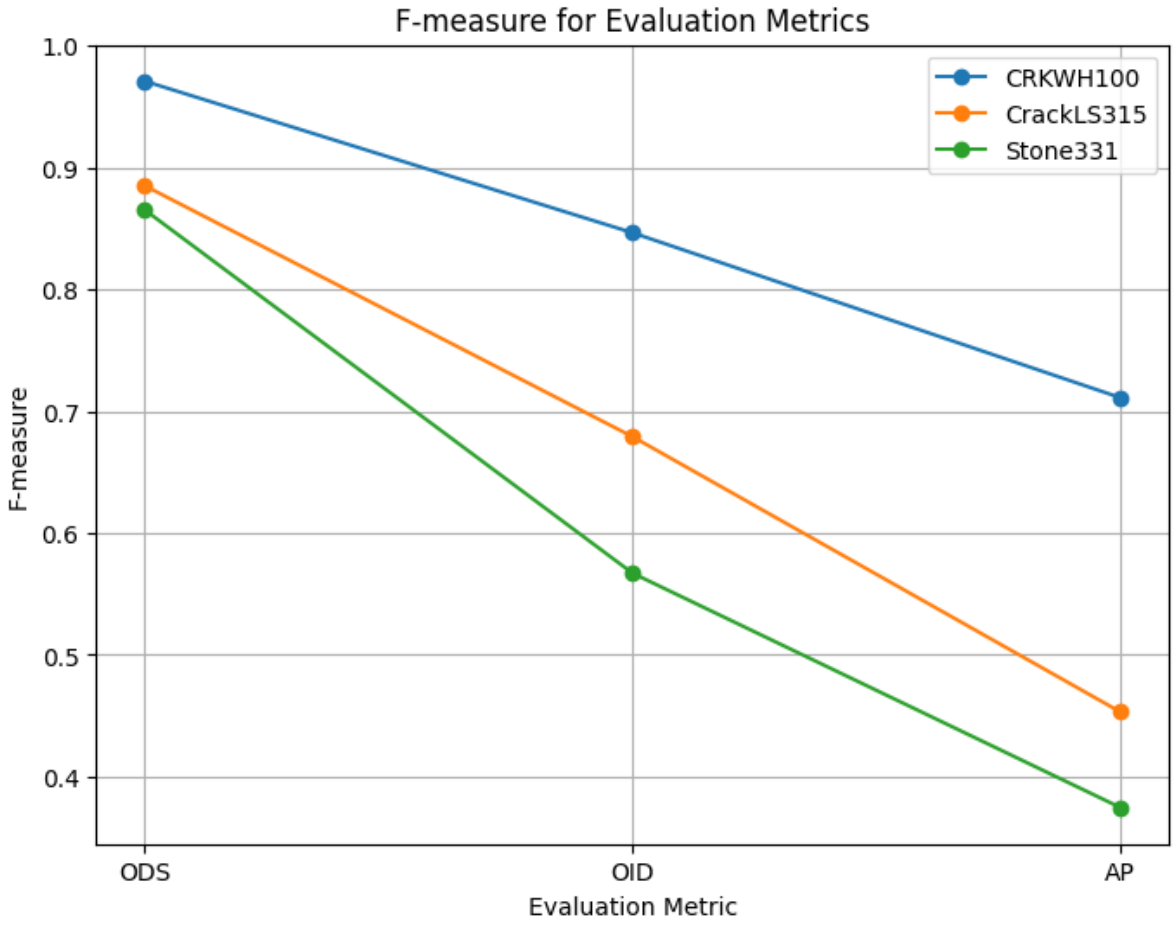

Figure 4.4: F -Measure for Evaluation matrix.

of how well the DeepCrack model performs and how it compares to other methods.

Furthermore, the experimental results, as depicted in Figure 4.5, demonstrate the impressive performance of the DeepCrack model. The model achieves an average ODS (Optimal Dataset Scale) F-measure value exceeding 0.90 on the test datasets. This indicates that the DeepCrack model consistently achieves high accuracy in crack detection tasks. Comparing these results to previous methods, such as SegNet and HCN, DeepCrack outperforms them in terms of accuracy. The model's ability to achieve a high ODS F-measure value showcases its strong capability in accurately detecting and classifying cracks in images.

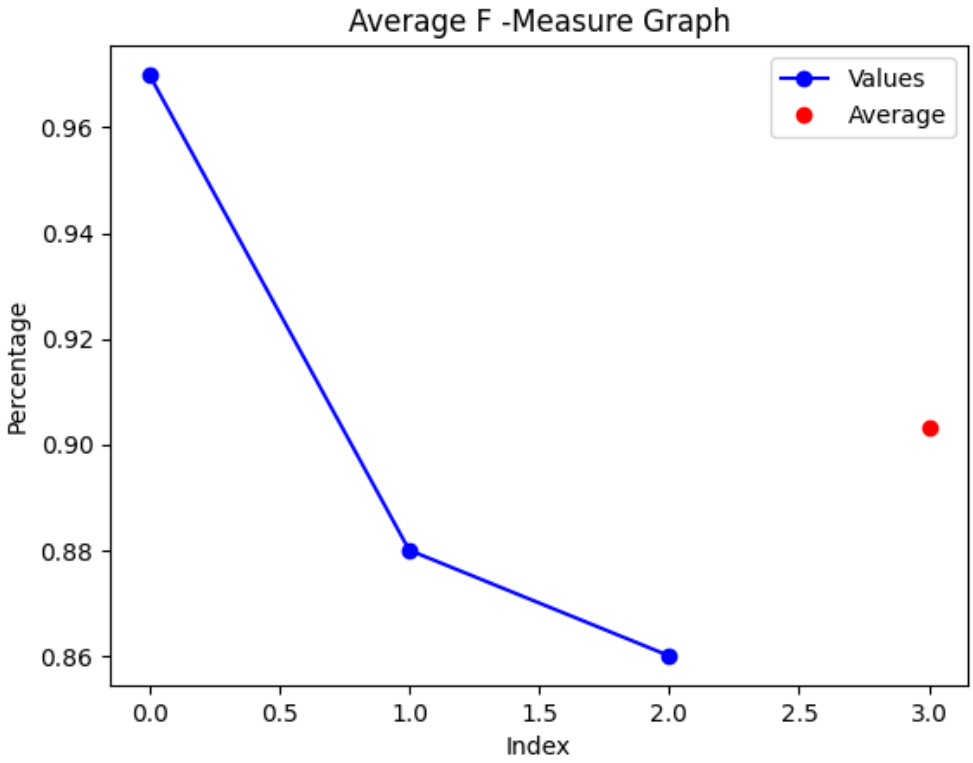

Figure 4.5: Average Accuracy For F -Measure .

The high ODS F-measure value reflects the model's ability to strike a balance between precision and recall. Precision refers to the model's accuracy in identifying true positives, while recall measures the model's ability to identify all relevant instances. By achieving a high ODS F-measure value, DeepCrack demonstrates its reliability and accuracy in crack detection, as it effectively combines precision and recall. These significant findings highlight the effectiveness of the DeepCrack model and its potential for practical applications in various real-world scenarios. The visual representation in Figure 4.5 further enhances the understanding and interpretation of the comparative performance of DeepCrack against other methods. The visual depiction allows researchers and practitioners to easily grasp and compare the model's performance in a graphical format.

Overall, these results substantiate DeepCrack as an advanced and promising solution for crack detection tasks. The model's impressive accuracy, as evidenced by the high ODS F-measure value, showcases its robustness and

reliability in accurately detecting and classifying cracks in images. These findings validate the effectiveness of DeepCrack and position it as a valuable tool for crack detection in practical applications.

4.4.3 ROC Curve

The Receiver Operating Characteristic (ROC) curve is a graphical representation of the performance of a binary classification model. It illustrates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) at various classification thresholds. The ROC curve provides valuable insights into the model's ability to discriminate between the positive and negative classes.

In the context of crack detection, the ROC curve helps evaluate the performance of the DeepCrack model. The TPR, also known as sensitivity or recall, represents the proportion of actual cracks that are correctly identified by the model. The FPR, on the other hand, indicates the proportion of non-crack pixels incorrectly classified as cracks. To calculate the ROC curve, the model's predictions are compared to the ground truth labels. For each classification threshold, the TPR and FPR values are calculated by varying the decision boundary and assessing the model's performance. These values are then plotted on the ROC curve, with the FPR on the x-axis and the TPR on the y-axis.

The ideal scenario is a ROC curve that hugs the top left corner, indicating a high TPR and low FPR across all thresholds. A diagonal line from the bottom left to the top right represents random classification. The area under the ROC curve (AUC) quantifies the overall performance of the model, with a value of 1 indicating a perfect classifier and a value of 0.5 indicating random classification. By plotting and analyzing the ROC curve for DeepCrack, can assess its ability to accurately detect cracks while minimizing false positives. The curve provides a comprehensive visualization of the model's performance across different thresholds, allowing for a clear comparison with other methods and the calculation of the AUC to measure overall performance.

The ROC curve analysis for the DeepCrack model on three test datasets,

CRKWH100, CrackLS315, and Stone331, yielded different results. The ROC curve with an AUC of 0.80 (Figure 4.6) corresponds to the performance on the CRKWH100 dataset. This indicates that the DeepCrack model has a good ability to discriminate between cracks and non-crack pixels, with a relatively high TPR and a low FPR across different classification thresholds. On

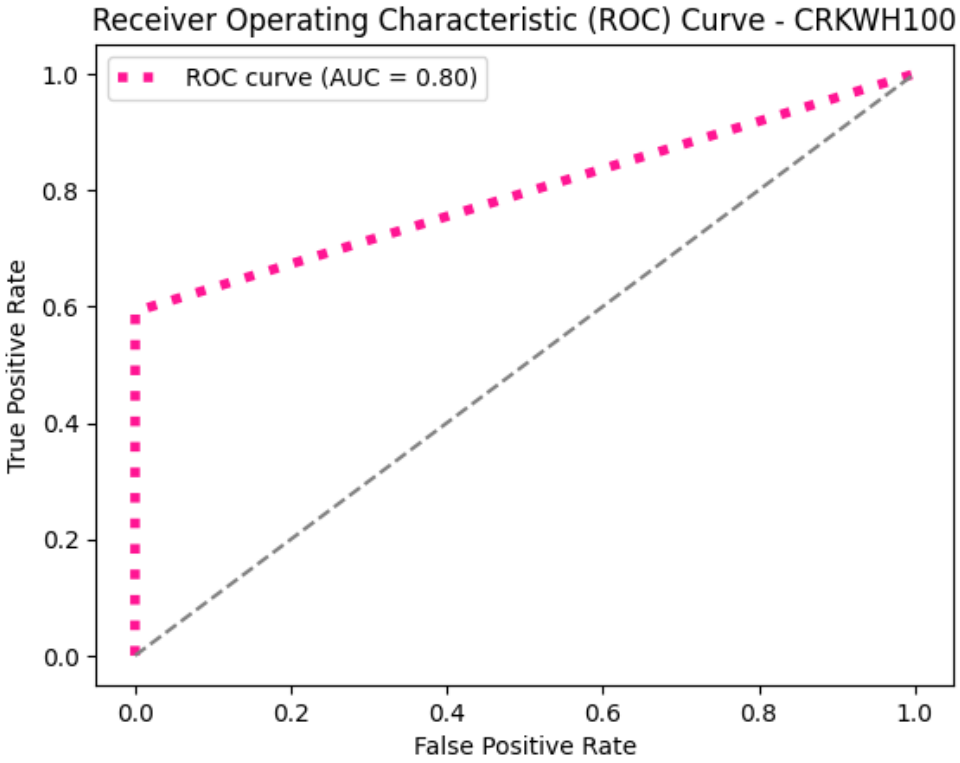

Figure 4.6: ROC Curve - CRKWH100.

the CrackLS315 dataset, the ROC curve obtained an AUC of 0.67 (Figure 4.7). Although lower than the performance on the CRKWH100 dataset, this result still suggests that the DeepCrack model can identify cracks effectively, albeit with a slightly higher false positive rate.

The ROC curve analysis on the Stone331 dataset yielded an AUC value of 0.70 (Figure 4.8). This indicates that the DeepCrack model demonstrates a reasonable level of performance in detecting cracks in this particular dataset. However, when comparing it to the performance on the CRKWH100 dataset, there is still potential for improvement. The AUC of 0.70 suggests that

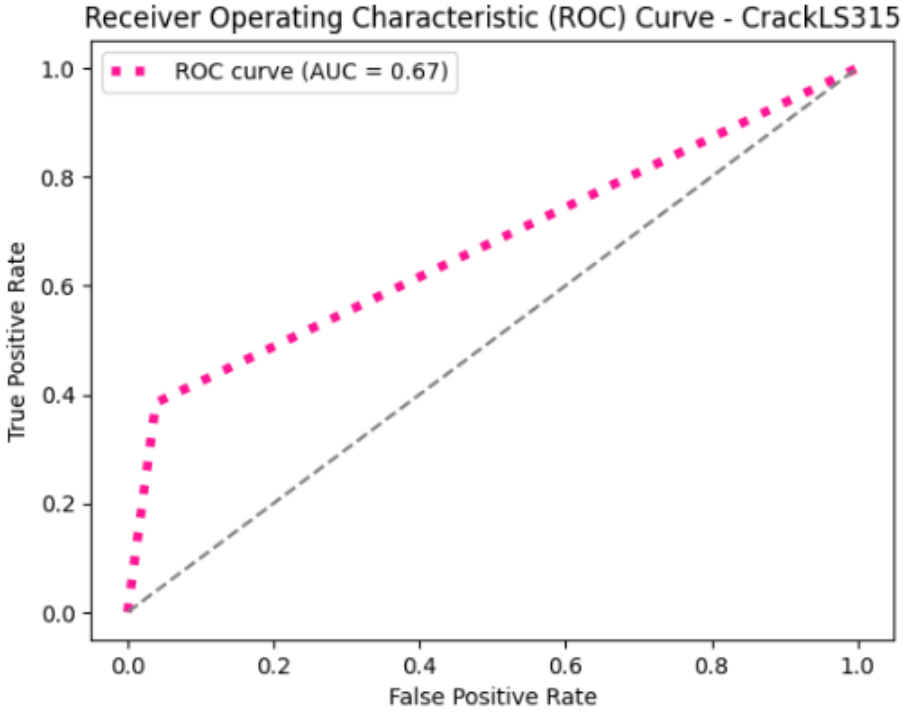

Figure 4.7: ROC Curve - CrackLS315.

there is some room for enhancing the model's ability to accurately distinguish between cracks and non-crack pixels in the Stone331 dataset. This could involve investigating the specific characteristics and challenges present in the Stone331 dataset and considering adjustments to the model architecture, training approach, or data augmentation techniques to address those challenges and improve its performance. By iteratively refining the model based on dataset-specific insights, it is possible to achieve better crack detection results on the Stone331 dataset and similar datasets in the future.

By analyzing the ROC curves and their corresponding AUC values for the DeepCrack model on different test datasets, we can compare its performance. It's important to keep in mind that an AUC value of 0.5 represents random classification. Since all three datasets produced AUC values above 0.5, it indicates that the DeepCrack model has some level of crack detection capability. However, it would be beneficial to conduct further analysis and potentially make adjustments to the model to improve its performance, particularly on datasets where the AUC is lower. These adjustments could involve refining

the model's architecture, fine-tuning hyperparameters, or increasing the diversity of training data to better handle the challenges present in those specific datasets. By iteratively evaluating and optimizing the model, it is possible to enhance its crack detection accuracy and reduce false positives.

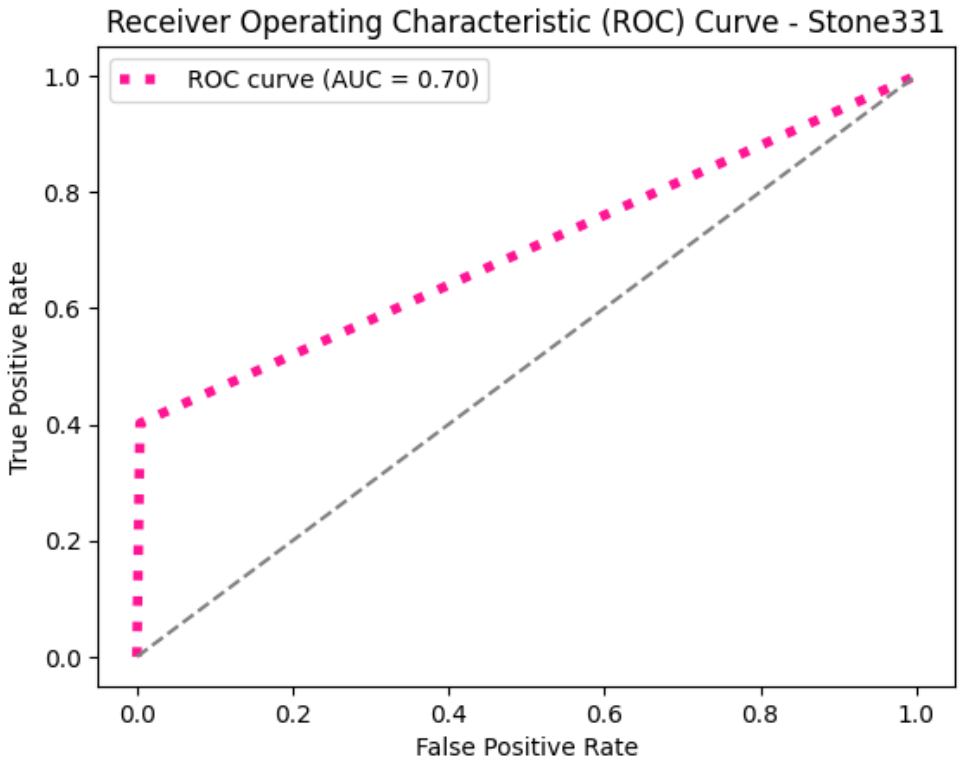

Figure 4.8: ROC Curve - Stone331.

4.5 Comparison with Other Architectures

The proposed DeepCrack model introduces significant modifications compared to the original SegNet and U-Net architectures, enhancing its performance and addressing challenges associated with crack detection.

Firstly, DeepCrack addresses the issue of sparse outputs by connecting the convolutional features in the encoder and decoder networks through skip-layer fusion. In contrast, the original SegNet architecture lacks this connection, potentially resulting in sparse outputs. By establishing this connection, Deepc-

rack ensures that information flows effectively between the encoder and decoder stages of the network. This connectivity allows the decoder network to access high-resolution features from the encoder network, leading to more comprehensive and detailed outputs. This modification improves the model's ability to capture intricate crack patterns and enhances the accuracy of crack detection.

Secondly, DeepCrack differs from U-Net in the implementation of skip-layer fusion. In U-Net, convolutional layers from an earlier stage are copied and inserted into a corresponding later stage of the main network, resulting in a single loss. In contrast, DeepCrack performs skip-layer fusion at each step individually, assigning a loss at each fusion point. This approach enables DeepCrack to effectively capture information related to thin objects at multiple scales. By assigning losses at each fusion point, DeepCrack ensures that the network learns and adjusts its parameters at various levels, optimizing the detection of cracks of different sizes. This modification enhances the network's ability to accurately detect cracks across a range of scales.

Furthermore, while methods like DeepEdge, DeepContour, and N4-Fields perform convolution on individual image patches, DeepCrack conducts convolution over the entire image, enabling an end-to-end production of results. By processing the entire image as a whole, DeepCrack can leverage global contextual information and capture the relationships between different regions. This holistic approach improves the network's crack detection capabilities, as it can consider the complete context and dependencies within the image.

The analysis and findings of this study highlight the significant architectural differences of DeepCrack compared to the original SegNet and U-Net architectures. These modifications provide several advantages that enhance the accuracy of crack detection and measurement in images. By addressing the issue of sparse outputs through skip-layer fusion and utilizing a 1-channel prediction map with cross-entropy loss, DeepCrack demonstrates improved performance compared to the original SegNet architecture. Additionally, the individualized skip-layer fusion in DeepCrack allows for effective capture of information related to thin objects at multiple scales, distinguishing it from

U-Net's approach. Moreover, DeepCrack's convolution over the entire image facilitates an end-to-end production of results, enhancing its crack detection capabilities. These architectural differences contribute to DeepCrack's ability to accurately and comprehensively detect cracks and measurement in images.

Chapter 5

Conclusion

In this study, a brand-new end-to-end trainable convolutional network called DeepCrack was developed for crack detection and measurement. DeepCrack utilized pairwise fusion of convolutional features at each scale to merge feature maps, resulting in a multi-scale feature-fusion map for accurate crack identification. The width and length of predicted crack maps were measured using morphological aspects, including thinning the crack skeleton and applying a crack defragmentation technique.

Experimental results on four crack datasets, with one dataset used for training and three datasets used for testing, demonstrated that DeepCrack achieved an average ODS F-measure value exceeding 0.90. These findings highlight the effectiveness of DeepCrack in accurately detecting and measuring different types of cracks, such as complex, common, thin, and intersecting cracks. The convolutional properties of both the encoder and decoder networks contributed to the success of crack identification. In conclusion, the findings of this study provide strong evidence supporting the effectiveness of DeepCrack as a robust solution for crack detection and measurement. The end-to-end trainable architecture employed in DeepCrack, along with the integration of multi-scale feature fusion and morphological analysis, collectively contribute to its ability to achieve high levels of accuracy in identifying cracks. Furthermore, the successful measurement of crack width and length using DeepCrack highlights its practical utility in assessing the severity and dimensions of cracks.

The results obtained in this study have significant implications for the field of crack detection techniques, as they demonstrate the potential of DeepCrack to address the challenges associated with infrastructure maintenance and monitoring. By leveraging the capabilities of DeepCrack, infrastructure stakeholders can benefit from improved crack detection accuracy, which in turn enables timely intervention and maintenance activities. This advancement in crack detection techniques has the potential to enhance the overall safety and longevity of critical infrastructure systems.

5.0.1 Future Work:

- Extend crack detection techniques to real-time video streaming applications by developing specialized algorithms capable of detecting cracks in dynamic video environments, considering challenges such as motion blur, varying lighting conditions, and complex backgrounds.
- Explore vessel detection for applications in medical imaging and remote sensing, focusing on refining deep learning models to handle variations in vessel appearance, scale, and orientation.
- Invest in refining deep learning models for crack and vessel detection, exploring novel architectures, optimization techniques, and innovative data augmentation methods tailored to video and vessel data.
- Evaluate the performance and applicability of crack detection in video streaming applications and vessel detection tasks, and assess their potential impact in domains such as infrastructure monitoring, medical imaging, and remote sensing.

By focusing on these areas of future work, significant advancements can be made in enhancing crack detection techniques for real-time video streaming applications and vessel detection tasks. This, in turn, can lead to improved decision-making, diagnosis, and analysis in various domains.

REFERENCES

[1] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assess

[2] ment of concrete and asphalt civil infrastructure," Advanced Engineering Informatics, vol. 29, no. 2, pp. 196–210, 2015.

[3] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in IEEE International Conference on Image Processing, 2016, pp. 3708–3712.

[4] S. J. Schmugge, L. Rice, J. Lindberg, R. Grizziy, C. Joffey, and M. C. Shin, "Crack segmentation by leveraging multiple frames of varying illumination," in IEEE Winter Conference on Applications of Computer Vision,, 2017, pp. 1045–1053.

[5] Z. Qu, L. Bai, S.-Q. An, F.-R. Ju, and L. Liu, "Lining seam elimination algorithm and surface crack detection in concrete tunnel lining," Journal of Electronic Imaging, vol. 25, no. 6, pp. 063 004–063 004, 2016.

[6] J. Geusebroek, A. Smeulders, and H. Geerts, "A minimum cost approach for segmenting networks of lines," International Journal of Computer Vision, vol. 43, pp. 99–111, 2001.

[7] A. Sironi, E. Turetken, V. Lepetit, and P. Fua, "Multiscale centerline ¨ detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 7, pp. 1327–1341, 2016.

[8] Z. Zhang, F. Xing, X. Shi, and L. Yang, "Semicontour: A semisupervised learning approach for contour detection," IEEE conference on computer vision and pattern recognition, pp. 251–259, 2016.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[10] R. Girshick, "Fast r-cnn," in IEEE international conference on computer vision, 2015, pp. 1440–1448.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234–241.

[13] S. Xie and Z. Tu, "Holistically-nested edge detection," in IEEE International Conference on Computer Vision, 2015, pp. 1395–1403.

[14] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[15] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3982–3991.

[16] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang, "Object contour detection with a fully convolutional encoder-decoder network," in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 193– 202.

[17] K.-K. Maninis, J. Pont-Tuset, P. Arbelaez, and L. Van Gool, "Con- ´ volutional oriented boundaries," in European Conference on Computer Vision, 2016, pp. 580–596.

[18] A. Khoreva, R. Benenson, M. Omran, M. Hein, and B. Schiele, "Weakly supervised object boundaries," in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 183–192.

[19] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Convolutional channel features," in IEEE International Conference on Computer Vision, 2015, pp. 82–90.

[20] A. Khoreva, R. Benenson, F. Galasso, M. Hein, and B. Schiele, "Improved image boundaries for better video segmentation," in European Conference on Computer Vision Workshops, 2016, pp. 773–788.

[21] J. F. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 6, pp. 679–698, 1986.

[22] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898–916, 2011.

[23] J. J. Lim, C. L. Zitnick, and P. Dollar, "Sketch tokens: A learned mid-level representation for contour and object detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3158–3165.

[24] P. Dollar and C. L. Zitnick, "Fast edge detection using structured forests," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 8, pp. 1558–1570, 2015.

[25] S. Wang, T. Kubota, J. M. Siskind, and J. Wang, "Salient closed boundary extraction with ratio contour," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 4, pp. 546–561, 2005.

[26] P. Martin, P. Refregier, F. Goudail, and F. Guerault, "Influence of the noise model on level set active contour segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 6, pp. 799–803, 2004.

[27] C. Li, C. Xu, C. Gui, and M. D. Fox, "Level set evolution without reinitialization: A new variational formulation," in IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2005, pp. 430–436.