ORTA DOĞU TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜ

TEZ ŞABLONU ONAY FORMU
THESIS TEMPLATE CONFIRMATION FORM

1. Şablonda verilen yerleşim ve boşluklar değiştirilmemelidir.
2. **Jüri tarihi** Başlık Sayfası, İmza Sayfası, Abstract ve Öz'de ilgili yerlere yazılmalıdır.
3. İmza sayfasında jüri üyelerinin unvanları doğru olarak yazılmalıdır. Tüm imzalar **mavi pilot kalemle** atılmalıdır.
4. **Disiplinlerarası** programlarda görevlendirilen öğretim üyeleri için jüri üyeleri kısmında tam zamanlı olarak çalıştıkları anabilim dalı başkanlığının ismi yazılmalıdır. Örneğin: bir öğretim üyesi Biyoteknoloji programında görev yapıyor ve biyoloji bölümünde tam zamanlı çalışıyorsa, İmza sayfasına biyoloji bölümü yazılmalıdır. İstisnai olarak, disiplinler arası program başkanı ve tez danışmanı için disiplinlerarası program adı yazılmalıdır.
5. Tezin **son sayfasının sayfa** numarası Abstract ve Öz'de ilgili yerlere yazılmalıdır.
6. Bütün chapterlar, referanslar, ekler ve CV sağ sayfada başlamalıdır. Bunun için **kesmeler** kullanılmıştır. **Kesmelerin kayması** fazladan boş sayfaların oluşmasına sebep olabilir. Bu gibi durumlarda paragraf (¶) işaretine tıklayarak kesmeleri görünür hale getirin ve yerlerini **kontrol edin**.
7. Figürler ve tablolar kenar boşluklarına taşmamalıdır.
8. Şablonda yorum olarak eklenen uyarılar dikkatle okunmalı ve uygulanmalıdır.
9. Tez yazdırılmadan önce PDF olarak kaydedilmelidir. Şablonda yorum olarak eklenen uyarılar PDF dokümanında yer almamalıdır.
10. Tez taslaklarının kontrol işlemleri tamamlandığında, bu durum öğrencilere METU uzantılı öğrenci e-posta adresleri aracılığıyla duyurulacaktır.
11. Tez yazım süreci ile ilgili herhangi bir sıkıntı yaşarsanız, Sıkça Sorulan Sorular (SSS) sayfamızı ziyaret ederek yaşadığınız sıkıntıyla ilgili bir çözüm bulabilirsiniz.

1. Do not change the spacing and placement in the template.
2. Write **defense date** to the related places given on Title page, Approval page, Abstract and Öz.
3. Write the titles of the examining committee members correctly on Approval Page. **Blue ink** must be used for all signatures.
4. For faculty members working in **interdisciplinary programs**, the name of the department that they work full-time should be written on the Approval page. For example, if a faculty member staffs in the biotechnology program and works full-time in the biology department, the department of biology should be written on the approval page. Exceptionally, for the interdisciplinary program chair and your thesis supervisor, the interdisciplinary program name should be written.
5. Write **the page number of the last page** in the related places given on Abstract and Öz pages.
6. All chapters, references, appendices and CV must be started on the right page. **Section Breaks** were used for this. **Change in the placement** of section breaks can result in extra blank pages. In such cases, make the section breaks visible by clicking paragraph (¶) mark and **check their position.**
7. All figures and tables must be given inside the page. Nothing must appear in the margins.
8. All the warnings given on the comments section through the thesis template must be read and applied.
9. Save your thesis as pdf and Disable all the comments before taking the printout.
10. This will be announced to the students via their METU students e-mail addresses when the control of the thesis drafts has been completed.
11. If you have any problems with the thesis writing process, you may visit our Frequently Asked Questions (FAQ) page and find a solution to your problem.

☒ **Yukarıda bulunan tüm maddeleri okudum, anladım ve kabul ediyorum. / I have read, understand and accept all of the items above.**

Name : **Tanner**
Surname : **Muturi**
E-Mail : **tanner.muturi@metu.edu.tr**
Date :
Signature : _____

RIGID PAVEMENT CRACK DETECTION UTILIZING GENERATIVE ADVERSARIAL NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

TANNER WAMBUI MUTURI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CIVIL ENGINEERING

JANUARY 2023

Approval of the thesis:

RIGID PAVEMENT CRACK DETECTION UTILIZING GENERATIVE ADVERSARIAL NETWORKS

submitted by **TANNER WAMBUI MUTURI** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Civil Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Erdem Canbay
Head of the Department, **Civil Engineering.** _____

Assoc. Prof. Dr. Onur Pekcan
Supervisor, **Civil Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Hande Işık Öztürk
Civil Engineering, METU _____

Assoc. Prof. Dr. Onur Pekcan
Civil Engineering, METU _____

Assist. Prof. Dr. Güzide Atasoy Özcan
Civil Engineering, METU _____

Prof. Dr. Yusuf Sahillioğlu
Computer Engineering, METU _____

Assist. Prof. Dr. Seda Selçuk
Civil Engineering, Çankaya University. _____

Date: 27.01.2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name: TANNER MUTURI

Signature:

ABSTRACT

RIGID PAVEMENT CRACK DETECTION UTILIZING GENERATIVE ADVERSARIAL NETWORKS

Muturi, Tanner Wambui
Master of Science, Civil Engineering
Supervisor: Assoc. Prof. Dr. Onur Pekcan

January 2023, 77 pages

Cracks are observed to be an initial sign of the degradation of the pavement and should therefore be detected and repaired to prevent further disintegrations. With the growth in technology, various image processing, machine learning, and deep learning methods have been applied to detect cracks on road pavements. The research leans towards using deep learning models for the pixel-wise segmentation of pavement image cracks among these models. However, most deep-learning models adopted in the literature are supervised and thus require enormous amounts of images with their corresponding ground-truth labels, which are expensive to obtain. Therefore, this study proposes using Cycle Generative Adversarial Network (CycleGAN), an unsupervised image translation model, for the pixel-wise segmentation of crack regions. A novel training procedure is adopted, with the forward and reverse cycle generators trained for every odd epoch and the discriminators for the even ones. A rigid pavement dataset of Fully Convolutional Network (FCN) (X. Yang et al., 2018) and drone (Ersoz et al., 2017) images are collected for training. As the model does not require accurate ground truth labels, labor-free unpaired image labels are obtained by compiling the ground truth crack labels from CrackForest, GAPS384, CrackTree200, and Crack500 public datasets.

The proposed model achieved an overall F1 score of 0.84, achieving comparable results with the CrackForest algorithm (Shi et al., 2016), FCN algorithm (X. Yang et al., 2018), and pix2pix model (Isola et al., 2017). In addition, the model outperforms the CrackForest algorithm and FCN-supervised model when testing the drone dataset. Finally, the effect of change in discriminator architecture and the application of transfer learning in the generator is investigated. A one-class discriminator architecture and loading of ImageNet pre-trained weights to the generator were observed to achieve the best performance.

Key words: Unsupervised Crack Detection, Generative Adversarial Networks, Cycle Generative Adversarial Networks, Image Translation, Segmentation.

ÖZ

BETON YOLLARDA ÇEKİŞMELİ ÜRETİCİ AĞLAR KULLANARAK YOL ÇATLAK TESPİTİ

Muturi, Tanner Wambui
Yüksek Lisans, İnşaat Mühendisliği
Tez Yöneticisi: Doç. Dr. Onur Pekcan

Ocak 2023, 77 sayfa

Çatlaklar, kaplamanın bozulmasının ilk işareti olarak gözlenir ve bu nedenle kaplamanın daha fazla parçalanmasını önlemek için tespit edilmeli ve onarılmalıdır. Teknolojideki büyüme ile yol kaplamalarındaki çatlakları tespit etmek için çeşitli görüntü işleme, makine öğrenimi ve derin öğrenme yöntemleri uygulanmıştır. Bu yöntemlerden araştırmalar, görüntülerdeki çatlakların piksel bazında bölütlenmesi için derin öğrenme modellerinin kullanımına yöneliktir. Bununla birlikte, literatürde benimsenen derin öğrenme modellerinin çoğu denetlenir ve bu nedenle, elde edilmesi pahalı olan, karşılık gelen yer-gerçeği etiketleriyle birlikte çok büyük miktarda görüntü gerektirir. Bu nedenle, bu çalışma, çatlak bölgelerinin piksel bazında bölütlenmesi için denetimsiz bir görüntü çeviri modeli olan Döngüsel Çekişmeli Üretici Ağlar'ın (CycleGANs) kullanılmasını önermektedir. Her tek dönem için eğitilmiş ileri ve geri çevrim üreteçleri ve çift dönem için ayrımcılar ile yeni bir eğitim prosedürü benimsenmiştir. Eğitim için FCN (X. Yang ve diğerleri, 2018) ve drone (Ersoz ve diğerleri, 2017) görüntülerinden oluşan rijit bir kaplama veri seti toplanır. Model, doğru kesin bilgi etiketleri gerektirmediğinden, CrackForest, GAPS384, CrackTree200 ve Crack500 genel veri kümelerinden temel doğruluk çatlak etiketlerinin derlenmesiyle emek gerektirmeyen eşleştirilmemiş

görüntü etiketleri elde edilir. Önerilen model, CrackForest algoritması (Shi ve diğerleri, 2016), FCN algoritması (X. Yang ve diğerleri, 2018) ve pix2pix modeli (Isola ve diğerleri, 2017) ile karşılaştırılabilir sonuçlar elde ederek 0,84'lük bir genel F1 puanı elde etmiştir.). Ayrıca geliştirilen model, insansız hava aracı veri setini test ederken CrackForest algoritması ve FCN denetimli modelden daha iyi performans göstermektedir. Son olarak, diskriminatör mimarisindeki değişimin etkisi ve üreteçte transfer öğrenme uygulaması araştırılmıştır. Tek sınıf bir ayrımcı mimarisi ve ImageNet'in önceden eğitilmiş ağırlıkların üretece yüklenmesinin en iyi performansı sağladığı gözlemlenmiştir.

Anahtar Kelimeler: Denetimsiz Çatlak Tespiti, Çekişmeli Üretken Ağlar, Döngüsel Çekişmeli Üretken Ağlar, Görüntü Çevirisi, Segmentasyon

To my dearest mother and sister

ACKNOWLEDGMENTS

The author would like to express gratitude towards her supervisor, Assoc. Prof. Onur Pekcan for his guidance throughout the research process. The author hopes he and his loved ones be blessed plentifully.

The author greatly appreciates the assistance and guidance of Asst. Prof. Seda Selçuk. For all the hours spent in zoom meetings and patience during the experimental phase of this study, words could not sum up the profound thanks she has. The author hopes she and her loved ones be blessed abundantly.

Profound thanks to each examining committee member for sparing their precious time to be a part of the thesis defence jury. Once more, the author hopes that they and their loved ones be exceedingly blessed.

The author would like to thank God for the strength to finish writing the thesis and for guidance in choosing the words and the methodology adopted.

The assistance of Ahmet Bahaddin Ersöz and Serhat Erinmez in the collection of the pavement images used in this study is acknowledged.

Finally, the author would like to thank her wonderful family for their continued encouragement and support throughout the study. Her deepest gratitude goes to her mother for providing moral and financial support. Therefore, the thesis is dedicated to the author's mother and sister, who taught the author to be brave and work hard in each situation.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xiv

LIST OF FIGURES xv

LIST OF ABBREVIATIONS xvii

CHAPTERS

1 INTRODUCTION 1

1.1 Overview 1

1.2 Objectives of the Research 7

1.3 Thesis Organization 8

2 LITERATURE REVIEW 9

2.1 Cluster-based algorithms 9

2.2 Minimal Path Selection (MPS) Based Algorithms 13

2.3 Novel Algorithms 14

2.4 Deep Learning-Based Algorithms 17

3 GENERATIVE ADVERSARIAL NETWORKS (GANs) 23

3.1 Overview of GANs 23

3.2 Image-to-Image Translation 23

3.3 Training of GANs 27

3.4 Types of GANs 28

3.5 CycleGAN 29
3.5.1 Training of CycleGAN 30
4 IMPLEMENTATION 35
4.1 Overview 35
4.2 Model Architecture 36
4.3 Hyperparameter Search 38
4.4 Model Training 39
5 DATASETS 43
5.1 Overview 43
5.2 Compiling Crack Dataset 43
5.3 Compiling Unpaired Groundtruth Labels 44
5.4 Preprocessing 45
6 PERFORMANCE EVALUATION 47
6.1 Evaluation Metrics 47
6.2 Comparative Methodologies: 50
6.3 Metric Results 51
6.3.1 Results on the FCN Dataset 51
6.3.2 Results on the Drone Dataset 56
6.4 Investigative Study 59
6.4.1 Change in Discriminator architecture. 59
6.4.2 Application of Transfer Learning in the Generator 61
7 SUMMARY, CONCLUSION, AND RESEARCH PROSPECTS 65
7.1 Summary 65
7.2 Conclusions 67

7.3 Research Prospects.....68
REFERENCES 69

LIST OF TABLES

TABLES

Table 4.1 Hyperparameter Values 39
Table 5.1 Number of ground truth labels collected from different crack image datasets 45
Table 6.1 Training and test combinations summary 51
Table 6.2 Region-based metric results on the FCN dataset under different training combinations 52
Table 6.3 Comparative results on the FCN dataset (Region-based metrics) 54
Table 6.4 Comparative results on the FCN dataset (Pixel-wise metrics) 54
Table 6.5 Region-based metric results on the drone dataset with different training combinations 56
Table 6.6 Comparative results on drone dataset (Region-based metrics) 57

LIST OF FIGURES

FIGURES

Figure 1.1.Vehicle-fitted imaging systems (a) Agile-RN system (b) LCMS system (c) VIAPIX system showing the acquisition module, exploration module, and the vehicle fitted with the acquisition module (Kaddah et al., 2020) 2
Figure 1.2. DJI Phantom 5 drone 3
Figure 1.3. Different deep learning crack analysis methods. (a) The input image (b) Patch-based classification (c) Object detection (d) Crack segmentation. (Hsieh & Tsai, 2020) 5
Figure 1.4. Distribution of published articles utilizing deep learning from 2015 to the year 2020. (Hsieh & Tsai, 2020)..... 6
Figure 3.1. Grayscale image → colored image 24
Figure 3.2. Examples of image-to-image translation applications (Pang et al., 2021) 25
Figure 3.3. Example of a dataset that would be utilized in a multi-domain I2I task (Pang et al., 2021) 26
Figure 3.4. Example of paired images and unpaired images used in training of a two-domain I2I task (Zhu et al., 2017)..... 26
Figure 3.5. GAN algorithm training pseudocode (Goodfellow et al., 2014) 28
Figure 3.6. Examples of the application of CycleGAN (Zhu et al., 2017) 30
Figure 3.7. Representation of the introduction of a forward and reverse cycle, (b) Forward cycle consistency loss, (c) Reverse cycle consistency loss (Zhu et al., 2017) 32
Figure 4.1. Schematic representation of the proposed model..... 36
Figure 4.2. Sample of the (a) Crack image fed to generator G, which aims to translate it into image (b) Representing the ground truth pixel segmentation. Feeding generator F with the segmented images results in the image (c) A generated crack image..... 36
Figure 4.3. CycleGAN implementation flow chart..... 40

Figure 4.4. CycleGAN implementation pseudocode 40
Figure 4.5. Generator architecture 41
Figure 4.6. Discriminator architecture 42
Figure 5.1. Division of datasets for training and testing 44
Figure 5.2. Sample crack images from different datasets (a) Crack500 Dataset, (b) CFD Dataset, and (c) CrackTree200 Dataset sample 45
Figure 6.1. Representation of the process of creating labels for region-wise metric analysis (a) The original image, (b) The original image overlaid by the grid representation of crack cells 49
Figure 6.2. The pictorial result on the FCN dataset (a) Represents the original crack image (b) The ground truth mask (c) Prediction of the original image when training on the Drone + FCN dataset (d) Prediction when training on the FCN dataset only (e) Prediction when training on the drone dataset only 52
Figure 6.3. The pictorial results on the FCN dataset (a) Crack image (b) Ground truth label (c) CrackForest (d) X. Yang et al. (2018) (e) Pix2pix model (f) Proposed model 55
Figure 6.4. Pictorial results on the drone dataset (a) Crack image (b) Ground truth label (c) CrackForest (d) X. Yang et al. (2018) (e) Pix2pix model (f) Proposed model 58
Figure 6.5. Loss comparison between the adoption of the One-class discriminator and the PatchGAN discriminator. Graphs (a) to (g) represent the different loss values calculated during training for the first 50 epochs 60
Figure 6.6. Pictorial results of the two different discriminator models. (a) Original Crack images (b) Ground truth labels (c) Model predictions 61
Figure 6.7. Loss comparison of the effect of utilizing vgg16 pre-trained weights as opposed to random uniform initialized. Graphs (a) to (b) represent the different losses calculated for the first 50 epochs 62
Figure 6.8. Pictorial results showing the effect of utilizing pre-trained weights instead of uniform random initialized weights. (a) Original Crack images (b) Ground truth labels (c) Model predictions 63

LIST OF ABBREVIATIONS

ABBREVIATIONS

CycleGAN	Cycle Generative Adversarial Network
FCN	Fully Convolutional Network
GANs	Generative Adversarial Networks
I2I	Image-to-Image translation
UAVs	Unmanned Air Vehicles

CHAPTER 1

INTRODUCTION

1.1 Overview

Infrastructure maintenance forms a significant part of the lifecycle of any built structure. Monitoring directs the maintenance process and allows the maintenance works to perform regular checks and identify defects. Road pavements are exposed to different weather conditions that could lead to the degradation of their lifespan; hence they should be regularly monitored. Lack of regular road maintenance could lead to transport inefficiency and pose a problem to vehicle safety. Furthermore, higher costs could be incurred if observed defects are not mitigated, resulting in the need for complete road reconstruction. Hence the importance of carrying out regular monitoring of the pavement. The Federal Highway manual (Miller & Bellinger, 2014) identifies road pavement defects such as cracks, potholes, patches, and depressions. Of these, cracks are observed to be the initial sign of damage to the road pavement. The crack in analogy could be viewed as the initial symptoms of a 'sick' pavement.

Pavement Management Systems (PMS) have been developed to monitor pavements effectively. These systems aim to collect images from pavements, process the information, and analyze the road's status.

Various collection methods exist, with the oldest involving going on field surveys, to manually note and measure defects on the road. However, manual survey methods are time-consuming, labor-intensive, and put the surveyor at risk. Furthermore, they are subjective, leading to poor reproducibility and repeatability. Vehicles fitted with cameras and laser scanners have also been utilized with imaging systems such as the Agile-RN system (Figure 1.1a), the Laser Crack Measurement System (LCMS)

(Figure 1.1b), and the VIAPIX system (Figure 1.1c) being developed (Kaddah et al., 2020). Beyond this, Unmanned Air Vehicles (UAVs) (Figure 1.2) have been explored for collecting pavement information. Silva et al. (2020) employed drones in collecting pavement information, citing their advantage in collecting large amounts of data over a short period. Furthermore, the use of drones is relatively cheaper compared to vehicles fitted with road imaging systems.

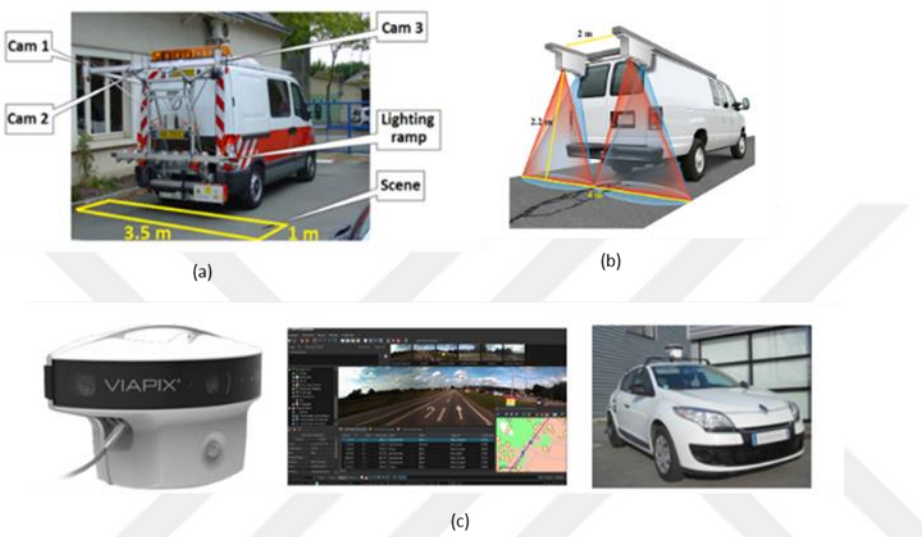

Figure 1.1.Vehicle-fitted imaging systems (a) Agile-RN system (b) LCMS system (c) VIAPIX system showing the acquisition module, exploration module, and the vehicle fitted with the acquisition module (Kaddah et al., 2020)



Figure 1.2. DJI Phantom 5 drone

Following the collection of images, data processing and analyses are performed. Tasks employed during analyses include detecting, classifying, or measuring pavement defects. The growth in computing capability has boosted this process by introducing various algorithms. The oldest are image processing techniques such as filters, thresholding, and edge detection algorithms. In addition, image processing methods have been complemented by the introduction of machine learning models, which shall be the focus of the study.

Machine Learning models can broadly be divided into traditional and deep learning models. Traditional machine learning techniques require prior extraction of learning features. Examples of such models include Artificial Neural Networks (ANN), Support Vector Machines (SVM), and Random Forest. First, image processing techniques such as edge detection methods, image thresholding, or image are applied for crack classification and detection. After processing, statistical features representing line locations, orientations, lengths, and thicknesses are extracted for classification using machine learning. Various authors have adopted traditional machine learning techniques. For example, Kaseko and Ritchie (1993) utilized ANN to classify cracks on pavement surfaces. Hoang et al. (2018) proposed using SVM and the artificial bee colony optimization algorithm to classify cracks. Beyond this, Shi et al. (2016) proposed the CrackForest crack detection framework, which utilized

the Random Forest algorithm to detect and classify cracks on pavement surfaces. However, traditional machine learning techniques are heavily affected by false crack detection for images with shadows, low contrast, and discontinuous crack regions. Furthermore, shallow learning techniques adopted are deemed unsuitable for complex information in the images (Hsieh & Tsai, 2020).

Consequently, deep learning techniques have been utilized increasingly in literature. In exploiting the deep learning methods, the task of crack detection could be divided into three primary functions, i.e., (i) classification, (ii) object detection, and (iii) pixel segmentation (Figure 1.3). Classification could present itself in the form of the model identifying image patches (Figure 1.3b) or the entire image as either a crack or non-crack image. To accomplish this task, Cha and Choi. (2017) proposed a Convolutional Neural Network (CNN) for the classification of image patches achieving an accuracy of 98%.

Object detection can be characterized as the localization of regions containing cracks, as seen in (Figure 1.3c). Silva et al. (2020) adopted the YOLO model for detecting potholes and cracks, achieving an average precision (AP) of 94.67%. Tran et al. (2020) trained RetinaNet in the localization of cracks. The authors achieved a detection accuracy of 89.1% considering crack type only and 84.9% considering the crack type and severity level on images obtained from the survey vehicle.

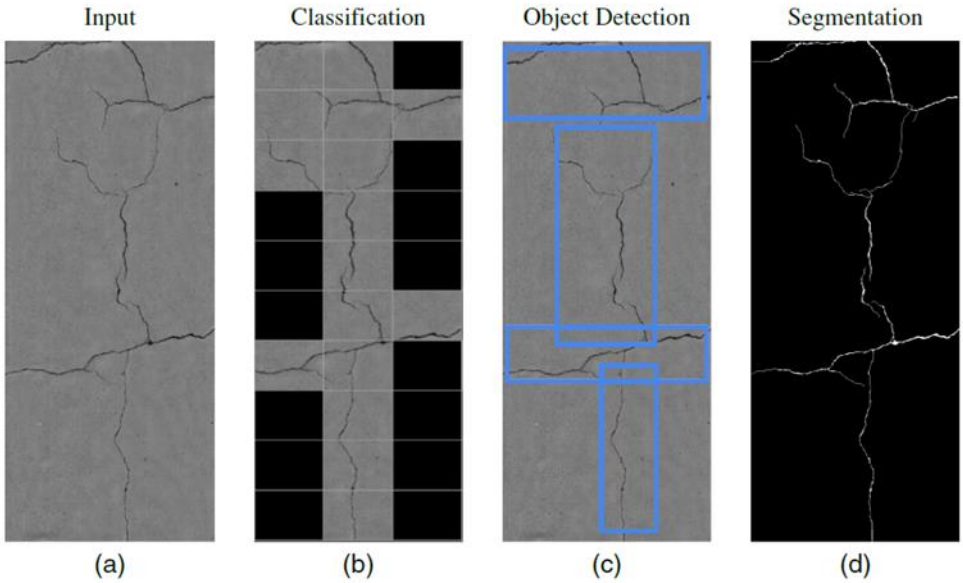

Figure 1.3. Different deep learning crack analysis methods. (a) The input image (b) Patch-based classification (c) Object detection (d) Crack segmentation. (Hsieh & Tsai, 2020)

The task of pixel segmentation of cracks could be defined as the pixel-wise labeling of an image as either crack or no crack, resulting in a binary mask (Figure 1.3d). As seen in Figure 1.4 of the analysis methods, most research articles are geared toward pixel segmentation of cracks. Pixel-wise labeling allows for identifying the crack width and pattern, which could help extract crack parameters and severity levels. Deep learning models employed in pixel segmentation can be divided into supervised and unsupervised machine learning models. Supervised models require ground truth labels for training, but unsupervised models do not.

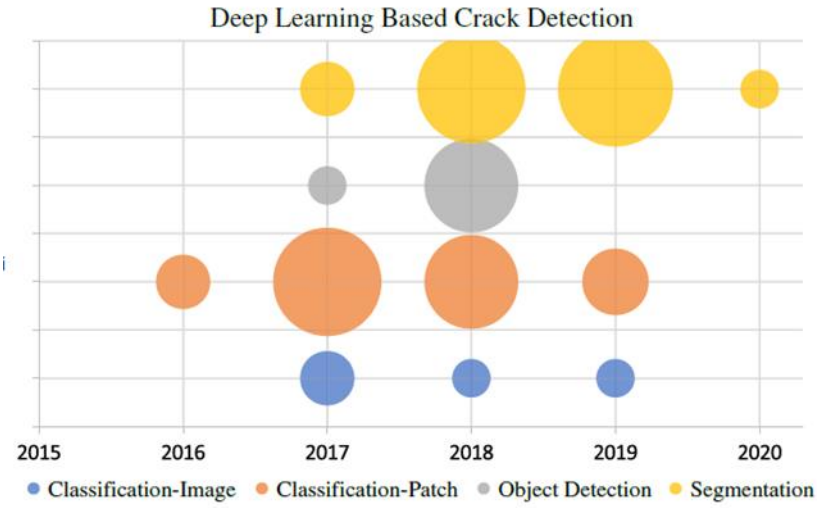

Figure 1.4. Distribution of published articles utilizing deep learning from 2015 to the year 2020. (Hsieh & Tsai, 2020)

Various supervised deep-learning architectures for the task of crack segmentation have been investigated. Cheng et al. (2019) proposed a U-Net architecture with their methodology achieving 6-9% accuracy higher than popular algorithms such as CrackTree, CrackIT, and CrackForestSVM. Wu et al. (2019) adopted the focal loss and self-attention mechanism with the U-Net architecture for crack detection to achieve better segmentation results. Their methodology reached 43.3% and 76.23% precision and recall metrics on the CrackForest dataset. Augustauskas and Lipnickas (2020) proposed a modified pixel-wise segmentation network based on the U-Net architecture autoencoder. The authors utilized residual connections, atrous spatial pyramid pooling with parallel and "Waterfall" connections, and attention gates to perform better defect extraction. At 0-pixel tolerance, the authors' model achieved accuracy, recall, and precision scores of 99%, 74.9%, and 68.9% on the CrackForest dataset. Beyond this, Tang et al. (2021) proposed an encoder-decoder network (EDNet) tested on 3D and 2D images. Authors attained an overall precision of 96.15%, recall of 99.56%, and F1-score of 97.82% on the CrackForest dataset outperforming the U-Net.

Despite achieving high-performance results, supervised methodologies are disadvantageous as model training heavily depends on accurate pixel-wise labels. Moreover, with the introduction of new images, the model would require retraining and hence the development of new tags, which is time-consuming, especially with large amounts of data (Duan et al., 2020). Beyond this, unsupervised methodologies best mimic human learning ability and are thus the focus of research in deep learning. Following this, the use of unsupervised algorithms has been adopted in the detection and classification of cracks. Autoencoders (Generative Adversarial Networks, Variational Autoencoders), K-means clustering, and Minimal Path Selection are common algorithms adopted in the literature.

1.2 Objectives of the Research

As observed, pavements require regular monitoring, with crack detection paramount for the early mitigation of defects. Current research on detecting cracks is geared toward developing better segmentation techniques, as seen in Figure 1.4. Of the methodologies presented, the use of traditional machine learning models and supervised deep learning models were unfavorable for the segmentation task. Following this, the objective of this thesis is to utilize unsupervised deep learning models for the task of pixel-wise identification of cracks in rigid pavements. The thesis presented is novel in that it shall:

- Apply the CycleGAN model to segment a rigid pavement dataset, which has yet to be explored in the literature.
- Introduce an unpaired ground truth label dataset by collecting ground truth labels from other crack datasets.
- Adopt a model architecture based on VGG16 and a novel training procedure to improve performance on the dataset collected.
- Investigate the effects of change in discriminator architecture and application of transfer learning in the generator, which has not been explored in literature yet.

Subject to the following limitations:

- Images during training are resized to 128x128 pixels, which though it reduces crack information, speeds up the time taken during training.
- An exact description of thin cracks at the pixel level is not defined in the literature and is thus subject to visual inspection.

With the novelty and limitations of the study set, this study is significant in that:

- Pixel-wise segmentation of cracks shall be performed, thus relaying information on the crack network. This information could subsequently be used to calculate crack width and severity.
- An unsupervised deep learning model will be adopted; thus, the model is insensitive to ground truth labels. Furthermore, the model can work with a large amount of unlabeled data. Beyond this, if additional crack images are collected, the model can easily be retrained, as no additional labels are required.

1.3 Thesis Organization

Following the description of the research objective above, the thesis has been divided so that Chapter 2 reviews current unsupervised crack detection models adopted in the literature. Chapter 3 then introduces the use of Generative Adversarial Networks (GANs), the concept of image-to-image translation, and CycleGAN used to accomplish the research objective. With this discussed, Chapter 4 introduces the implementation of CycleGAN to the task of crack pixel segmentation. Chapter 5 discusses the datasets utilized in the deep learning model training, and Chapter 6 provides the results obtained. Finally, the summary, conclusions, and future works are highlighted in Chapter 7.

CHAPTER 2

LITERATURE REVIEW

A comprehensive survey of unsupervised models and algorithms for crack detection is done in the literature. The author emphasizes the keywords such as self-supervised crack detection, unsupervised crack detection, and semi-supervised crack detection during the literature survey. The unsupervised algorithms found are divided into cluster-based methodologies, Minimal Path Selection (MPS) methodologies, novel unsupervised algorithms, and deep learning-based algorithms. The sources discovered are described below:

In subsequent sections, detection shall be defined as the classification of either patches/pixels as either crack or non-crack. Classification of cracks shall be defined as evaluating the crack patterns according to a defined system such as longitudinal, transverse, or fatigue.

2.1 Cluster-based algorithms

In their 2013 article, Oliveira and Correia (2013) proposed using clustering to detect road pavement cracks. In their methodology, firstly, the images were divided into 75X75 pixel blocks and prelabelled using the mean and standard deviation of the block's matrices. This preliminary labeling is further utilized in dividing the images into training and test sets, ensuring representability in both data sets. The images are pre-processed through normalization to ensure blocks classified as non-crack have a uniform illumination in the image. Following normalization, pixel saturation is performed using a set threshold value such that the brightest pixels in a block are either below the value or equal to it. Pixel saturation will allow the standard deviation values of crack blocks to stand out in the feature matrix. The standard deviation and

mean matrix are computed for the normalized blocks and then further normalized for easier processing. In calculating the feature space normalization, each image in the training database is first clustered, and a global centroid is computed for the target cluster (non-crack images). A linear fit on the target class is then performed to calculate a reference angle for each 2D space. The feature space is normalized using these results in the training set to ensure the target cluster is conspicuous. K-means, Hierarchical, and the mixture of two Gaussian-clustering are utilized in the 2-class classification.

In contrast, the one-class classification strategy utilizes the Gaussian density method, Prazen density estimator, and minimum covariance determinant gaussian classifier (MCDG). Following training, the computed boundaries are superimposed into the feature distribution of the test image to classify the blocks. The authors develop a set of rules to classify the image based on the standard deviation calculated for each row and column of crack-connected components. Furthermore, the severity level of the cracks is calculated as the pixels' spatial resolution is known. In the development of severity level, the blocks containing crack pixels are thresholded through the Otsu method to define the crack pixels enabling the calculation of crack width. The performance of the selected algorithm is evaluated on 56 grey-level images of 1536X2048 pixel images obtained while avoiding shadows cast by objects during sunny weather. Five images containing cracks were used in training, and 51 were used in testing, with eight containing no cracks. The precision, recall, f-measure, error rate, and crack error rate are calculated. The mixture of the two gaussian-clustering achieved the best overall performance with an F measure (93.5%), the best global error- rate (0.6%), and the second-best performance in terms of recall (95.5%). The model took about 2min to process 56 images. The authors proposed future work in developing denoising techniques and entropy reduction. Furthermore, the authors proposed adopting different crack types and severity levels characterization.

Mubashshira et al. (2020) proposed crack detection by adopting k-means clustering. Firstly, a color histogram is utilized to segment the road section from other regions in the picture. To separate the road segment from the rest of the image, mean shift

filtering was performed, and a 3x3 window was used to detect the color of the pavements. The pixel color is converted to black if it does not achieve the set threshold. After this, the images were resized to 450x500 pixels and converted to greyscale. A Gaussian filter and log transform were further employed to smooth the image and enhance the brightness of the crack pixels. K-means clustering with k=2 was employed, and similar pixels were grouped in this stage to classify the pixels as crack and non-crack. Finally, the Otsu method was used to binarize the image. In dealing with noise pixels, morphological dilation, and erosion were performed, and a feasible contour was drawn to connect the crack pixels in the original image. Contours help eliminate noise. Utilizing 120 images from the internet, the authors achieved an accuracy of 97.75%

Vignesh Mohanraj et al. (2018) utilized k-means clustering to segment images. In the methodology, images were obtained from a video sequence and pre-processed using a mean filter to eliminate bright pixels, a median filter to eliminate salt and pepper noise, and an adaptive Weiner filter to remove additive white noise. Subsequently, the images were divided into n-by-n blocks, with the mean and standard deviation being utilized as features to classify blocks as crack or non-crack. Beyond these features, the authors carried out canny edge detection. For each pixel, an analog canny edge value was assigned as the magnitude of the gradient. Each block's maximum analog edge value was chosen as the third feature in classifying blocks as crack or non-crack. Following feature extraction, the blocks are classified as crack or non-crack using clustering. Finally, post-processing was performed to improve crack connectivity. The methodology developed is disadvantageous as it offers low computational efficiency, and the authors did not evaluate their method at the pixel level.

Mucolli et al. (2019) carried out a comparative study between k-means and k-medians in classifying underwater structures as either cracked or not cracked using Haralick texture features derived from the image. Bridge foundations and dams were inspected with 490, 256x256 pixels images obtained with the GoPro Hero 7 camera. Obtained images are subjected to a median filter of kernel size 5 to reduce noise.

Following this, the image is divided into 16x16 pixel non-overlapping boxes for which the grey level co-occurrence matrix is computed. Features computed included the angular second moment, contrast, inverse difference model, entropy, correlation, and variance. Following clustering, images are post-processed to remove outliers. The histogram of the candidate block is drawn up and treated as a bimodal Gaussian distribution. Utilizing Bayes theory and average greyscale values, outliers are eliminated. The authors concluded that k-median outperformed k-means, though with a higher processing time. The authors obtained F1 scores between 0.63 and 0.9 for different images.

Ji et al. (2020) proposed using clustering with feature extraction by resnet50 in labeling microcracks in solar cells. 640, 300X300 grayscale image samples are obtained. The images are normalized concerning size and perspective and subsequently annotated as 1 or 0 as either containing defect or not. An equal selection of crack and non-crack images is obtained. Data augmentation is performed at two levels, the first involving resizing and horizontal flipping, and the second, in addition to the first level, contrast and saturation adjustment are performed. The methodology applied included first applying initial labels with the clustering head. The feature extractor is then trained, and labels are assigned by measuring the distance between the image and label utilizing the Sinkhorn Knopp algorithm. The model is constrained to produce an equal number of crack and non-crack labels. Results showed that level one augmentation showed better performance compared to level 2. In addition, utilizing 2 cluster heads resulted in an accuracy rate of 74.53%. The methodology is compared to k-means, with the self-labeling system achieving better performance. Furthermore, an unbalanced dataset showed lower performance.

Though clustering offers an unsupervised detection of cracks, detection is generally performed at the patch level with the need for extraction of features or the adoption of post-processing techniques such as edge detectors to segment the cracks. The use of statistical features is not suitable for increasingly complex images. Furthermore, patch-level classification offers blocky results that would not be suitable for

extracting features such as crack width. Minimal path selection (MPS) methods have been investigated to combat these disadvantages.

2.2 Minimal Path Selection (MPS) Based Algorithms

Amhaz et al. (2016) proposed minimal path selection in detecting cracks on pavement surfaces. The methodology proposed is divided into the selection of endpoints, estimation of a minimal path, minimal path selection, and post-processing procedures. In selecting endpoints, the image is subdivided into squares from which the darkest pixel is chosen. Next, this pixel is subjected to a threshold value and is discarded if it exceeds it. The minimal path between endpoint pixels is then estimated using the Dijkstra algorithm. A cost function based on pixel intensities determines the minimal way. Post-processing is further applied to eliminate spikes and loops that may be retained in the final path selection. The image is subdivided into linear segments, and thresholding is applied hence eliminating spikes and loops.

Furthermore, neighboring pixels with intensity below the threshold are absorbed into the crack image in detecting crack width. In evaluating their methodology, one synthetic image and 269 real images collected using the survey vehicle were used. Of the real images, 68 contained reference segmentations. The authors obtained an average dice score of about 50% to 65% for different databases of the real image. MPS adopted here outperformed other methodologies such as the Markov-based methods, Freeform anisotropy, and geodesic contour method.

Kaddah et al. (2019) proposed an Optimized Minimal Path Selection (OMPS) on pavement images to reduce the computational cost of MPS introduced by Amhaz et al. (2016). The authors introduced the use of local anisotropy (retaining pixels whose anisotropy is high enough for use as candidate endpoints) and adaptive thresholding to reduce the number of bright pixels with a low probability of belonging to the crack during the selection of endpoints. To curb computational cost during the estimation of the minimal path, the authors suggest the adoption of 3P X 2P image subnets to

reduce the number of paths to be computed to 4 instead of the original 8. The authors obtained 33 1900×924-pixel images from the Agile-RN imaging device. Due to the unbalanced lighting, the dataset was divided into 991×462 images and 311×462 images. For 991x462 images, a decrease in processing time was noted from 796 sec to 47 sec if only local anisotropy is applied and 12 sec if adaptive thresholding is applied too. An increase in dice scores from 64% to 74% and 75% was observed. A similar trend was seen in 311X462 images with time values of 234, 16, and 3 seconds and dice scores of 70%, 71%, and 72%.

Kaddah et al. (2020) further developed the Automatic darkest filament detection (ADFD) method to detect cracks on road surfaces in an unsupervised way. Image processing was first performed through normalization and edge detection, increasing contrast while reducing noise and uneven illumination. Secondly, the selected crack candidate pixels are chosen as the darkest in an image. Following this, a 1-pixel width crack is segmented, and the segmentation results are improved through post-processing. Authors claim the difference between ADFD and the MPS lies in the pre-processing of the images, the method of selection of endpoints, and the post-processing steps. The authors utilized images collected through the Agile-RN, Laser Crack Measurement System (LCMS), and VIAPIX system. For the Agile-RN dataset, authors achieved 76% and 77% dice scores on the 991x462 pixel and 311x462 pixel images, respectively.

Despite offering unsupervised detection of cracks on pavement surfaces, MPS requires selecting parameters such as thresholding values that would vary across different image sets. Furthermore, the method is computationally expensive. As a result, other novel methods have been investigated to mitigate the disadvantages.

2.3 Novel Algorithms

In their 2015 article, Shamsabadi et al. (2015) proposed using a hessian-based filtering algorithm to segment cracks on a road surface. In achieving the

segmentation of cracks on the road surface, pre-processing of the images is done to eliminate lane markers and further correct the images for distortion. The hessian method detects tubular/ridge-like structures in an image in all directions. In their algorithm, the authors first detected the area covered by the alligator cracks, and longitudinal and transverse cracks were detected from the remaining area. Any other crack encountered is characterized as other. On 150 images, the authors achieved a detection accuracy of all cracks of between 93% and 98%. Though the methodology does not require training, it falls short in that specific parameters need to be chosen to threshold the eigenvalues.

To develop a method that deals with the disadvantages of edge detection algorithms that are heavily affected by noise and k-means clustering, which requires a large amount of data, Lei et al. (2018) developed the Crack Central Point Method (CCPM). Images obtained by a UAV are firstly pre-processed through converting to grayscale, image filtering, and finally, image enhancement. CCPM assumes that the crack center has the lowest intensity along an image's row or column of pixels. Crack pixels are detected based on this assumption, and a threshold value is introduced restricted by the crack's width. The authors evaluate the performance of CCPM on concrete crack images obtained from a UAV flying 40cm above the ground. Authors report better performance of CCPM compared to that of traditional edge detection methods, the LoG algorithm, and the Prewitt algorithm.

Mathavan et al. (2014) investigated the use of self-organizing maps (SOM) or Kohonen maps to detect cracks on pavement surfaces. The methodology proposed combines texture and color properties to distinguish cracks from the background. Firstly, the co-occurrence matrix is obtained from Haralick features to represent the texture in an image. The coarse-grained textured surface is separated from the finely textured surface. The SOM is subsequently trained on the acceptable aggregate segments to differentiate between the cracks and the background. The authors tested their algorithm on four 3,264×2,448 pixels images captured with the Sony Cybershot DSC-W180 while avoiding shadows. While training, the images were divided into

tiles of 60x60 pixels, obtaining 8,640samples. The model recorded an overall segmentation precision of 75% and a recall of 70%.

Li et al. (2019) proposed using a multi-scale fusion crack detection algorithm (MFCD) to detect pavement cracks. In detecting cracks, authors utilizing a windowed minimal intensity path first extracted candidate cracks with the assumption that cracks exhibit lower intensity values than the background and pixels belonging to the same crack form a continuous path. Candidate cracks found at different scales are fused based on a multivariate statistical test. The algorithm performance is tested on the Agile-RN, CFD, and a self-captured dataset containing 33 images. The algorithm's performance is compared with the performance of Geodesic Contour (GC), Free Form Anisotropy (FFA), Minimal Path Selection (MPS), CrackIT, CrackTree, and CrackForest. MFCD is shown to have better overall performance compared to other algorithms. In the CFD dataset, the model achieves 89.9%, 89.47%, and 88.04% precision, recall, and F1-measure, respectively.

Fang et al. (2020) proposed using video image sequences and different unsupervised machine-learning algorithms to detect faults in sewer lines. The authors first extracted the image sequences from the video. Obtained images are resized to 224x224-pixel size, with features being subsequently extracted. Features extracted include the local binary patterns (LBP), histograms of oriented gradient (HOG), grey level co-occurrence matrices (GLCM), Gabor filter processing, and image feature vectors (IMG-FV) (Obtained using PCA on the image itself). The authors evaluated the performance of iForest, Gaussian distribution, one-class SVM, and Local Outlier Factor (LOF) detection algorithms on these features. The authors further performed a sensitivity analysis on the detection algorithms' choice of features. From a sample size of 8952 images containing 1514 images with faults, the authors attained the best performance with selected features and Gaussian distribution algorithms (accuracy, precision, and recall of 0.88, 0.94, and 0.90, respectively). On the other hand, OV-SVM and LOF show poor performance across all datasets utilized by the authors. Authors further note that though Gaussian-d and iForest offer an overall better

performance, Gaussian-d had the best performance irrespective of the feature combination.

Abdel-Qader et al. (2006) proposed using Principal Component Analysis to detect cracks on a bridge deck. Firstly, the train set images are normalized by subtracting the mean of all the images in the set. From this, the covariance matrix is calculated, to which the PCA algorithm is applied. Euclidean distance is then applied to cluster images with similar features. The authors adopted linear convolutional masks for vertical, horizontal, and diagonal line detection to improve results. Following convolution, the images are passed through a smooth filter to eliminate noise in the form of weak cracks. The authors further implemented local processing by dividing the images into 16 blocks and processing each as an individual image. The training dataset consisted of 5 crack and five non-crack images, and the test set consisted of 40 images, of which 20 were crack and 20 were non-crack. With PCA being applied directly to the test images, the authors achieved an accuracy of 57.5%. The authors achieved an overall accuracy of 60% with the introduction of convolution. However, they noted an increase in false negatives. With the introduction of local processing, authors saw an overall accuracy of 73%, with a reduction of false negatives compared to other approaches.

Beyond these novel algorithms, deep learning models such as autoencoders, variational autoencoders, and Generative Adversarial Networks (GAN) have been adopted. These models do not require parameter selection that would vary from one image set to another and are thus more advantageous.

2.4 Deep Learning-Based Algorithms

Chow et al. (2020) trained a convolutional autoencoder with defect-free images to detect defects in concrete structures. First, 42200 256X256-pixel defect-free RGB images in the training set were resized to 320X320 pixels and normalized between -1.0 and 1.0. Next, the training dataset was randomly augmented through flipping and

rotation. Finally, utilizing a mean squared error loss, batch size of 16, and Adam optimizer, an anomaly map was created from the reconstruction and the original image to detect the cracks. The investigated model yielded an average precision of 0.607 and a recall of 0.879 for defect detection.

Y. Wang et al. (2019) proposed using UAVs to detect anomalies on wind turbine blades, aiming to reduce the cost of periodic checking. One-class support vector machine (OCSVM) and an unsupervised learning method with in-depth features learned from a generic data set. VGG-16 is used as the backbone CNN for feature extraction, pre-trained on the ILSVRC2014 ImageNet dataset. PCA (principal component analysis) is used to reduce the dimensionality of the feature map obtained from the CNN. In addition, the min-max operation was used to normalize the feature vector for ease of training in the OCSVM. The OCSVM is trained on normal data to create a hypersurface that separates normal (not containing cracks) from abnormal images. Image patches of 128 x128 pixels are utilized in the network to reduce dimensionality. The training data included 130 images of blades with no damage, and they were divided into 73,918 patches. The test data contained 30 blade images with known damage, divided into 21,085 patches. The authors obtained a precision of 0.63, recall of 0.496, and F1 score of 0.555 with a 1-layer VGG-16 network. Their research found lower layers better-extracted features related to anomalies, especially cracks. The authors experienced limitations in that the model may wrongly detect conspicuous dirt, stains, and patterns of painted lines.

Z. Liu et al. (2020) investigated Variational Autoencoders (VAE) in detecting micro-cracks in photovoltaics, with the significant advantage being that the VAE does not require manually labeled samples. The proposed architecture consists of the encoder that reduces the input scan into an array of latent variables and the decoder that converts the latent variable to a line scan profile. The encoder consists of 2 convolutional layers (Conv1D), a max pooling layer, and a flattened, fully connected layer. The output of the encoder is a 10-pixel array, the encoded latent variable. The VAE is trained on data without any cracks, enabling it to learn how to reconstruct an uncracked line scan profile; hence when faced with a cracked sample, the difference

between a reconstructed sample and an original sample beyond a certain threshold will signal an anomaly is present. The authors achieved precision and recall scores of 0.83 and 0.72, respectively.

Zhai et al. (2018) adopted Generative Adversarial Networks (GAN) to evaluate textured surfaces for defects. The methodology proposed was applied to wood surfaces and road crack surfaces. Authors first train a GAN to generate images similar to a normal textured surface. In doing this, the network learns a good representation of features in a latent vector space. Following this, the first three layers of the discriminator are utilized as feature extractors whose response is sensitive to the abnormal regions. A course vector calculated from the feature maps generated by the convolutional layers is used to distinguish abnormal zones in an image. Beyond this, a multi-scale heatmap fusion strategy is adopted. The authors first resize inspection maps produced by the convolutional layers to the same size and then further apply a weighted average method for fusion. After obtaining the final heatmap, the Otsu method binarizes the image. IoU and pixel accuracy were used as evaluation metrics. This method, compared with other baseline methods, showed superior performance.

K. Zhang et al. (2020) introduced a self-supervised model based on a cycle-generative adversarial network (GAN) to perform pixel-wise crack detection. The proposed model is advantageous in that it does not require ground-truth labels for the training images. Instead, labor-free ground-truth images collected from other sources are used as a structure library in training the network. The model consists of 2 GANs. The first transforms a crack patch image into a Ground Truth-like image, and a second GAN performs the reverse, i.e., it transforms the GT-like image into a crack patch image. The GAN consists further of 2 discriminators, one that compares the generated GT image and the structure image and a second one that compares the translated structure image and the original crack image. A cycle consistency loss with extra constraints is introduced to help the generator create accurate structure images. A U-Net architecture is chosen for the generator, and a classifier is chosen for the discriminator. In training, publicly available CrackForest database (CFD),

FCN data set (X. Yang et al., 2018a), and a personal database were used. The personal database consisted of 600 images collected using a line-scan camera mounted on a survey vehicle. GT values were roughly marked by engineers with a 1-pixel curve which was not accurate. Model performance was compared with state-of-the-art CrackIT, Crack-Forest, MFCD, FCN, and DeepCrack algorithms. The author performed a patch-level evaluation with p-rate, r-rate, F1 score, and Hausdorff distance used as evaluation metrics. The methodology achieved comparable results with other algorithms. Lastly, it boasted a faster processing speed. The authors carried out an ablation study by removing the cycle-consistency loss and the one-class discriminator. These were found to be crucial in structure learning. The authors also found that the methodology developed was not domain specific.

Duan et al. (2020) investigated a method to translate crack images to binary images using Generative Adversarial Networks (GANs) with unpaired data. Eight residual blocks connected convolutional neural network for feature extraction are used as a generator, and a 5-layer fully convolutional network is used as a discriminator. GAN containing two modules, the generator, and discriminator, is advantageous in faster speed, sharper generative sample, and fully fitting data. Residual blocks in the generator are utilized as they are faster to train. Skip connections were added between the encoder and decoder to keep crack details. The discriminator FCN utilized leaky, ReLU nonlinearity, and normalization. The authors' unpaired images are obtained using the SHcracklabel120 with 30 images drawn by hand and then rotated 90 degrees, 180 degrees, and 270 degrees, respectively, and reshaped to a resolution of 320×480 pixels. Cycle consistency is introduced to enhance the accuracy of crack localization. The proposed methodology is validated with the CrackForest database (CFD) containing 118 images. With a 5-pixel tolerance, the methodology proposed achieves a precision of 89.70%, a recall of 82.52%, and an overall F1 score of 85.07%. However, the methodology proposed falls short in that the width of the crack in the binary image does not change, and the binary image is less sensitive to the bending of the crack.

Following the literature analysis above, unsupervised training using CycleGAN is proposed. Although K. Zhang et al. (2020) and Duan et al. (2020) utilized this method in crack segmentation, a new approach is adopted in the thesis in the form of:

- Compiling an unpaired ground truth label dataset by collecting ground truth labels from other crack datasets. This is proposed to allay the disadvantage mentioned by Duan et al. (2020), who utilized drawn labels resulting in cracks insensitive to bending and having uniform widths.
- The model is applied to a rigid pavement dataset, whereas previous authors applied the methodology to flexible pavements. As the construction material of the topmost layer in flexible and rigid pavements is different, this leads to a contrast in the type of cracks and their backgrounds. Flexible pavements have 'noisy' backgrounds due to reflective materials in the asphalt binder, whereas rigid pavements have smooth backgrounds.
- Different model architectures and training procedures are adopted to improve the results of the dataset collected.
- Investigating the effect of change in the discriminator architecture and application of transfer learning in the generator.

The next chapter shall introduce Generative Adversarial Networks (GANs), image-to-image translation, and the proposed model, CycleGAN.

CHAPTER 3

GENERATIVE ADVERSARIAL NETWORKS (GANs)

3.1 Overview of GANs

Generative Adversarial Networks (GANs), a machine learning model, are among recent breakthrough models in machine learning. First proposed by Goodfellow et al. (2014) for the implicit modeling of high dimensional data in either an unsupervised or semi-supervised way has been heavily researched. The authors proposed a framework composed of a generator G that learns the data distribution and discriminator D that estimates the probability of the sample being part of the real distribution or the generated distribution. The training of a GAN hence results in a max-min two-player game that aims to trick such that it cannot tell the difference between real samples and the generated ones (fake samples). Thus, Generator G can be considered an art forger and discriminator D as an appraiser trying to detect counterfeit artwork. The competition between the two networks drives the teams (referring to the generator and discriminator) to improve their approach, hence adversarial networks.

GANs have been used widely for tasks of image synthesis (a core GAN capability), image-to-image translation, and classification. For this thesis, Image-to-image translation is further discussed in depth.

3.2 Image-to-Image Translation

To better understand the definition of image-to-image translation, a question is posed. For example, if one would like to transform a sketched image into a realistic image or have a gray-scale image and would like to color it (Figure 3.1), how would this be possible computationally? The answer to this question led to research in the

broadly deemed domain of image-to-image translation (I2I). Hence, image-to-image translation is defined as the process of transforming an image from the source to the target domain while preserving the content. Therefore, the chosen model aims to learn the mapping between the output and the input image. To achieve the outlined objective, I2I borrows from generative models, which can learn and approximate the underlying distribution of data.



Figure 3.1. Grayscale image → colored image

Variational Autoencoders and GANs are among the most used and efficient deep generative models for the task of I2I translation. Variational Autoencoders (VAE) model data distribution by maximizing the lower bound of the data log-likelihood. However, VAE shall not be discussed further as GANs are the focus of the thesis.

I2I translation applications range from image style transfer (Lee et al., 2020) (Figure 3.2a), semantic segmentation (Park et al., 2019) (Figure 3.2c), image colorization (Suarez et al., 2017) (Figure 3.2d), image super-resolution (Y. Zhang et al., 2020) (Figure 3.2e) to image inpainting (Marinescu et al., 2020) (Figure 3.2d).).

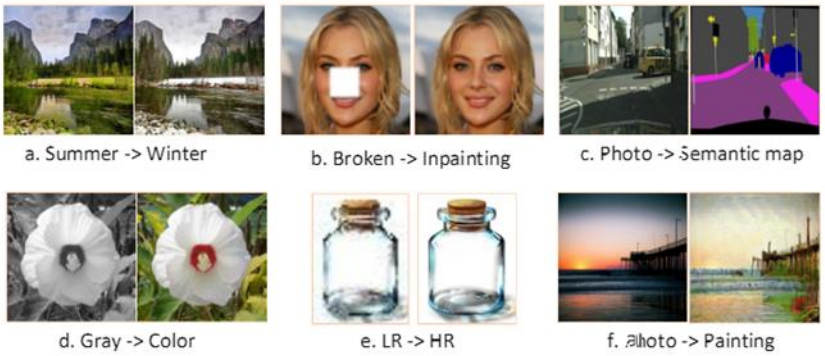

Figure 3.2. Examples of image-to-image translation applications (Pang et al., 2021)

I2I translation can be divided into two-domain I2I tasks and multi-domain I2I tasks. Multi-domain I2I centers on utilizing a single model to handle multiple domains with multiple outputs, different style textures, or semantic contents. An example of the kind of dataset that would be utilized in a model is shown in Figure 3.3 (Pang et al., 2021) below. Conversely, two-domain I2I focuses on the use of a single model which captures the relationship between two domains. The thesis focuses on the two-domain I2I task, which can further be divided into four sub-categories based on leveraged data. These include:

- **Supervised I2I**. Aligned image pairs of the source and target domain (Figure 3.4) are utilized during training and testing. This led to the development of models such as the pix2pix model (Isola et al., 2017)
- **Unsupervised I2I**. Obtaining a large set of paired training data is time-consuming, expensive, and sometimes impossible. This disadvantage led to research on unsupervised I2I tasks, where an unpaired dataset is utilized (Yi et al., 2017; Zhu et al., 2017) (Figure 3.4). Unsupervised I2I translation forms the basis of the thesis research objective.
- **Semi-supervised I2I**. To further improve unsupervised I2I translation results, the authors utilize a few source-target paired images alongside the unpaired data during training. This model is used in fields such as the restoration of old films or genomics (Mustafa & Mantiuk, 2020).

- **Few-shot I2I**. Human beings are observed to learn from only one or a limited number of examples. Beyond that, they are found to utilize prior experience and knowledge in learning a new task. This characteristic of human beings inspired the development of few- or on-shot I2I algorithms (Liu et al., 2019). These models have been proposed for translating a few or even one example in a limited unpaired training dataset.

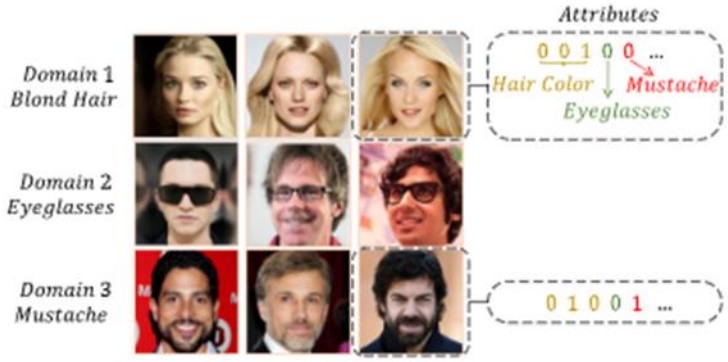

Figure 3.3. Example of a dataset that would be utilized in a multi-domain I2I task (Pang et al., 2021)

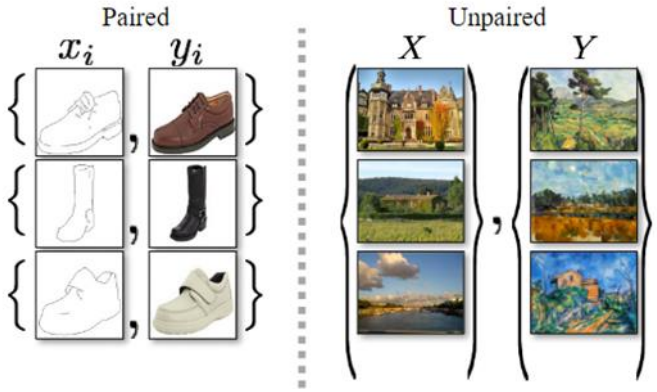

Figure 3.4. Example of paired images and unpaired images used in training of a two-domain I2I task (Zhu et al., 2017)

3.3 Training of GANs

Following the introduction to the task of image-to-image translation and GANs, this section shall discuss the process of training GANs, which forms the basis of the task tackled in the thesis.

As described, the training of GANs aims to find values in the generator G that effectively confuse the discriminator D and, conversely, values that allow the discriminator to tell apart generated values effectively. To further understand the training frameworks, given that we would like to learn the generator's distribution p_g over data x, a prior input noise variable $p_z(z)$ is defined, with a final mapping to the dataspace as $G(z, \theta)$, where θ are the parameters of the network G. A second network is defined as $D(x, \theta)$, which outputs a single scalar, with $D(x)$ representing the probability that x came from the data rather than p_g. D is trained to maximize the probability of assigning the correct label to both training and samples from G, whereas G is trained to minimize $\log\,(1 - D\big(G(z)\big))$, i.e., the probability of D being able to tell apart the generated sample. This results in the value function $V(G, D)$ outlined below:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim P_{data}(x)} \log D(x) + \mathbb{E}_{z \sim P_Z(z)}\ \log\,(1 - D\big(G(z)\big)) \quad (1)$$

Early in training, Goodfellow et al. (2014) notes G is poor, and D can reject sample with a high degree of accuracy; hence the equation $\log\Big(1 - D\big(G(z)\big)\Big)$ saturates. Therefore, to alleviate this problem, the authors suggest that instead of minimizing the equation, instead, G should be trained to maximize $\log D(G(z))$. Therefore, equation (1) can be broken down into equations (2) and (3):

$$max_D V_D(D, G) = \mathbb{E}_{x \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim P_Z(z)}\ [\log\,(1 - D\big(G(z)\big))] \quad (2)$$

$$max_G V_G(D, G) = \mathbb{E}_{z \sim P_Z(z)}\ [\log\ D\big(G(z)\big)] \quad (3)$$

Figure 3.5 (Goodfellow et al., 2014) below presents the pseudocode of the training procedure of a GAN network.

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Figure 3.5. GAN algorithm training pseudocode (Goodfellow et al., 2014)

3.4 Types of GANs

Training of GANs is often unstable. In literature, evaluating the 'symptoms' of unstable training is recommended to train a GAN properly. These 'symptoms' could include:

1. Failure of the models to converge.
2. Mode collapse. This is a situation observed when the generator produces similar images with different inputs.
3. The discriminator loss converges to 0. This prevents the update of the generator.

Different GAN models have been developed to alleviate the problems encountered above or produce higher quality results and growing interest and research in GANs. These include models such as conditional GAN (Mirza & Osindero, 2014), InfoGAN (Chen et al., 2020), Wasserstein GAN (Arjovsky et al., 2017), StackGAN (H. Zhang et al., 2017), CycleGAN (Zhu et al., 2017) and more. The subject in this thesis takes advantage of CycleGAN in accomplishing crack segmentation. CycleGAN is discussed in detail below.

3.5 CycleGAN

In accomplishing the objective outlined in the Introduction, a variation of GANs known as CycleGAN is utilized. The CycleGAN, first introduced by Zhu et al. (2017), aims to perform image-to-image translation using unpaired image samples Figure 3.4. Thus, given an image in domain X, the model attempts to map it to the target domain Y. To achieve this, a cycle consistency loss is introduced to preserve the original image.

CycleGAN relishes being an unsupervised image-to-image translation model. This fact has numerous applications in style transfer, such as transforming an image from a realistic view to Monet or Van Gogh style, object transfiguration (zebras to horses), season transfer (summer to winter), and photo enhancement, as seen in Figure 3.6 below.

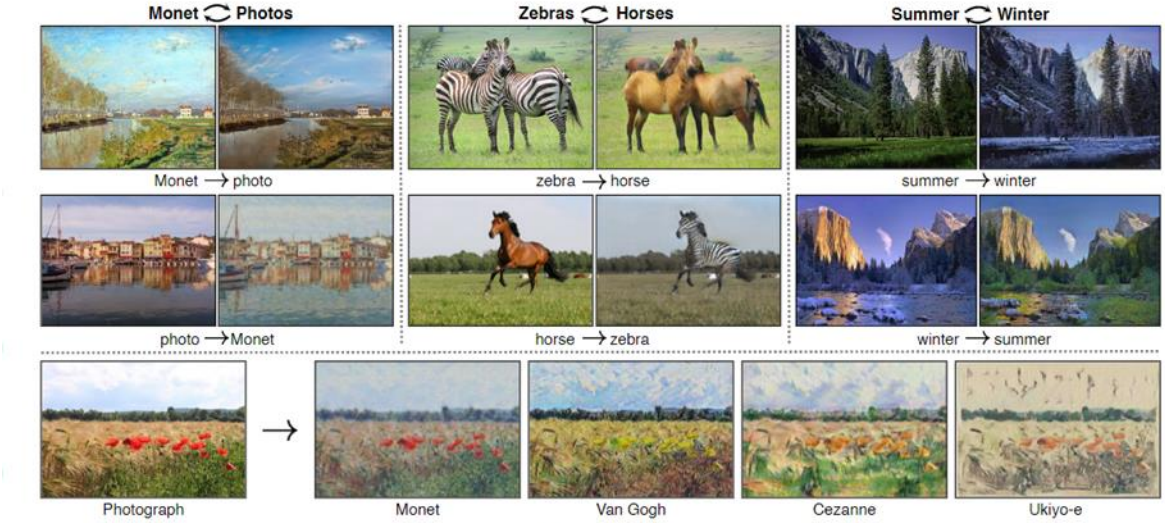

Figure 3.6. Examples of the application of CycleGAN (Zhu et al., 2017)

As proposed by the authors, the CycleGAN model consists of two GANs, commonly referred to as the forward and reverse cycle GAN. The section below outlines the training objective in CycleGAN.

3.5.1 Training of CycleGAN

When broken down, a CycleGAN model consists of two GAN models (forward and reverse cycle GAN) trained for the task at hand. In the forward cycle, generator G performs the mapping of images from domain X to Y ($G: X \rightarrow Y$), while the discriminator D_y, aims to distinguish between real samples in Y and generated images $G(x)$. In the reverse cycle, generator F does the mapping $F: Y \rightarrow X$, while the discriminator D_x, aims to distinguish generated images $F(y)$ and samples in X. The two GAN models utilize adversarial loss discussed in section 3.3. Beyond this, the authors introduce a second loss, the Cycle Consistency Loss.

3.5.1.1 Adversarial loss

The Adversarial loss, described in equation (1) in the training of GANs, is applied to both mapping functions. For the function $G: X \rightarrow Y$ and its discriminator D_y, the adversarial loss is given as:

$$L_{GAN}(G, Dy, X, Y) = \mathbb{E}_{y \sim P_{data}(y)}[\log\ D_Y(y)] + \mathbb{E}_{x \sim P_{data}(x)}\ [\log\ (1 - D_Y(G(x)))] \quad (4)$$

G aims to minimize this function whereas D_y aims to maximize it, leading to the $min_G max_{D_y} L_{GAN}(G, D_y, X, Y)$ adversarial loss objective (equation (4)). This similar objective is applied to the reverse cycle as well, which does the mapping $G: Y \rightarrow X$, ($min_F max_{D_x} L_{GAN}(F, D_x, X, Y)$

3.5.1.2 Cycle Consistency Loss

Given the adoption of adversarial loss only, a well-trained model on the data distribution in either domain X or Y could produce several random permutations of the source image in the target domain. This unconstrained behavior would not lead to the desired mapping of $X \rightarrow Y$ and vice versa. To diminish this occurrence, the authors adopt transitivity through cycle consistency.

The use of transitivity in the regularization of models has been a long-established practice. In the domain of language translation, translators have carried out "back translation and reconciliation" to improve and verify translation (Xia et al., 2016). In visual tracking, a forward-backward consistency has been utilized (Kalal et al., 2010). More recently, higher order cycle consistency has also been used in the task of depth estimation(Godard et al., 2017), dense semantic alignment(Zhou et al., 2015), co-segmentation (Wang et al., 2013) as well as structure from motion (Zach et al., 2010). Zhu et al. (2017) adopt the transitivity approach and introduce the cycle consistency loss.

To further explain this, for an image x, the mapping $G(x)$ gives a fake y, and the application of $F(G(x))$ should lead to a sample comparable to the original x ($x \approx F(G(x))$. Similarly, for an image y, $G(F(y))$ should be equal to y ($y \approx G(F(x))$ (Figure 3.7). This behavior is therefore parameterized through the objective function (5) (Zhu et al., 2017).

$$L_{cyc} = \mathbb{E}_{x \sim P_{data}(x)} \left[\left|\left| F(G(x)) - x \right|\right|_1 \right] + \mathbb{E}_{y \sim P_{data}(y)} \left[\left|\left| G(F(y)) - y \right|\right|_1 \right] \quad (5)$$

The final objective function (6) defined by Zhu et al. (2017) is shown below:

$$L(G, F, D_y, D_x) = L_{GAN}(G, D_y, X, Y) + L_{GAN}(F, D_x, X, Y) + \lambda L_{cyc} \quad (6)$$

The lambda value, introduced as a multiplier of the cycle loss, controls the relative importance of the two objectives.

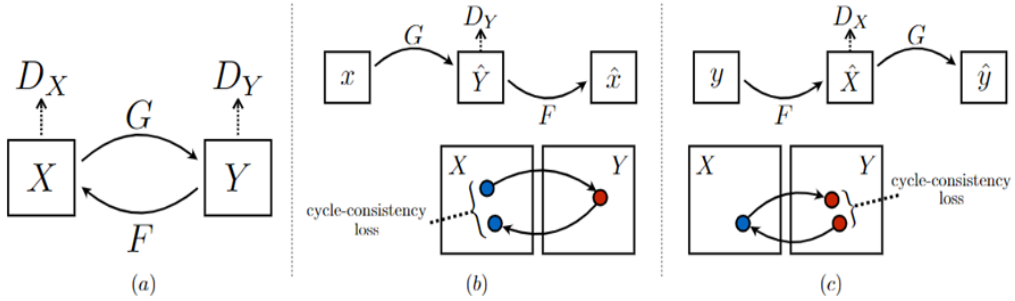

Figure 3.7. Representation of the introduction of a forward and reverse cycle, (b) Forward cycle consistency loss, (c) Reverse cycle consistency loss (Zhu et al., 2017)

3.5.1.3 Training Parameters

In the training of the CycleGAN, Zhu et al. (2017) proposed replacing the negative log-likelihood objective with the least-squares loss, which provides more stability. Furthermore, the authors offer a solution to the oscillation problem by updating the discriminator on a history of generated images, not the immediately generated ones. Moreover, a lambda value of 10 with the Adam optimizer is suggested during

training. An initial learning rate of 0.0002 was chosen for the first 100 epochs and linearly decaying the rate to 0 over the subsequent 100 epochs. A batch size of 1 is utilized, with instance normalization and reflection padding employed.

Despite positive results, especially in texture and color change, Zhu et al. (2017) noted a limitation in carrying out geometric changes and situations where the distribution characteristics of the training dataset caused failure.

An introduction to Generative Adversarial Networks was shared. Image-to-image (I2I) translation was focused on two-domain I2I tasks. Consequently, the training of GANs was discussed, and the different derivations of the original GAN model were introduced. A study of the CycleGAN model was then carried out, focusing on the training objective and parameters adopted in the original article. This foundation directs to the next chapter, which discusses implementing the CycleGAN model for unsupervised crack image segmentation.

CHAPTER 4

IMPLEMENTATION

This chapter discusses the implementation of CycleGAN in the pixel-wise segmentation of crack images. The chapter has been divided into a discussion of the application of CycleGAN, the model architecture adopted, hyperparameters chosen for the model, and the training pseudocode.

4.1 Overview

As expounded in the previous chapter, the CycleGAN model consists of a forward and reverse cycle, resulting in a model with two generators and two discriminators. Following the notation established in section 3.5.2, Figure 4.1 below represents the task assigned to the discriminators and generators. In the image, the generator G learns the mapping from the original crack image (Figure 4.2a) in domain X to the target segmented images Y (Figure 4.2b). In contrast, the generator F learns the mapping of the segmented images in domain Y (Figure 4.2b) to the crack images in domain X (Figure 4.2c). The discriminator D_y, learns to tell apart the generated images, $G(x)$, and real images in Y, whereas the discriminator D_x learns to tell apart the generated crack images, $F(y)$, and real crack images in X. Following training, the crack detection model is chosen as the generator G

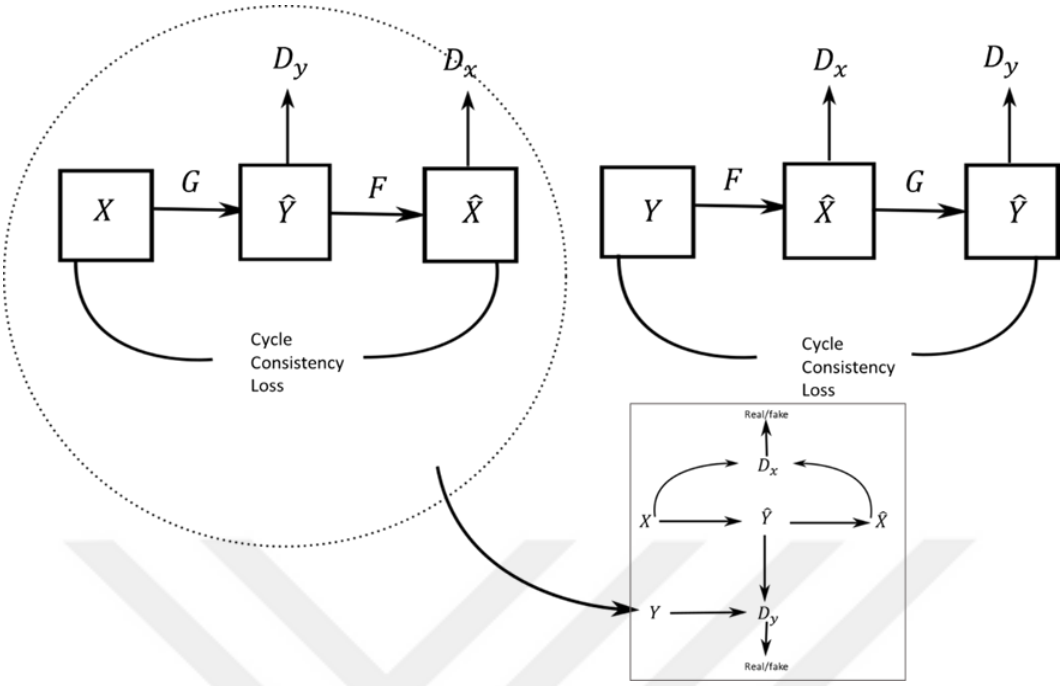

Figure 4.1. Schematic representation of the proposed model

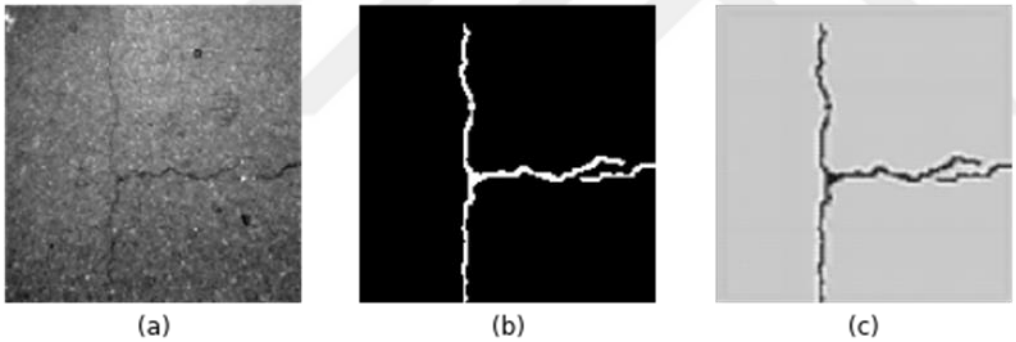

Figure 4.2. Sample of the (a) Crack image fed to generator G, which aims to translate it into image (b) Representing the ground truth pixel segmentation. Feeding generator F with the segmented images results in the image (c) A generated crack image.

4.2 Model Architecture

The same generator architecture is adopted for both the forward and reverse cycle, likewise for the discriminator, as also observed by Zhu et al. (2017). However, the

individual generator and discriminator architectures differ. The paragraphs below expound on the architectures adopted in the thesis.

The generator model adopts a fully convolutional architecture. In the convolutional section, generally referred to as the encoder, the VGG16 backbone is utilized. Pooled results are stored and added during deconvolution as a skip connection. The VGG16 backbone is loaded with ImageNet (Deng et al., 2010) pre-trained weights. Zhu et al. (2017) proposed the use of Instance normalization and reflection padding to reduce artifacts during training. Therefore, the generator adopts reflection padding in all layers except the final deconvolution layer. Reflection padding in the final deconvolution layer resulted in sporadic results and was consequently dropped. Instance normalization is only adopted during deconvolution. Regularization through applying dropout during deconvolution and an L2 regularization with a factor of 0.5 is applied to all layers. Figure 4.5 displays the architecture of the generator.

The discriminator architecture in K. Zhang et al. (2020) is adopted. The architecture consists of 4 3X3 convolutional layers followed by a fully connected layer with a final single output. Whereas K. Zhang et al. (2020) adopt a SoftMax last activation function, this activation function is disregarded as a lack of training convergence was observed when included. Moreover, the ReLU activation function in intermediate layers is replaced by leakyReLU with a negative slope coefficient of 0.3. To improve training convergence, a dropout rate of 0.3 and L2 regularization, with a factor of 0.5, are introduced at each layer. Figure 4.6 displays the architecture of the discriminator. Initial weights are set to random uniform values with a mean of 0 and standard deviation of 0.02 (similarly for generator layers that do not have pre-trained weights)

Zhu et al. (2017) proposed the PatchGAN discriminator architecture during training. An investigation on the effect of using the PatchGAN discriminator is also carried out.

4.3 Hyperparameter Search

In the training of the model, different hyperparameters are investigated. These include:

- Batch size
- Cycle Consistency loss multiplier
- L2 regularization factor
- Dropout probability
- Initial learning rate
- Epochs

Optimal Hyperparameter values are chosen based on a grid search between the upper and lower limit values. In selecting the batch size and cycle consistency lower limit, values of 1 and 0.35 are set, as proposed by Zhu et al. (2017) and K. Zhang et al. (2020), respectively. The lower limit for the L2 regularization factor is determined as TensorFlow API's default value given as 0.01. In selecting the initial learning rate, a value of 0.0002 is chosen, as suggested by Zhu et al. (2017). Beyond these values, other upper and lower limits are determined experimentally, observing the training speed and convergence of the model.

Table 4.1 below displays the list of hyperparameters, the search boundaries, and the values chosen.

Table 4.1 Hyperparameter Values

Hyperparameter	***Grid Search Range***		***Final Value Chosen***
	Lower Limit	***Upper Limit***	
Batch size	1	20	20
Cycle consistency loss multiplier	0.35	100	0.35
L2 regularization factor	0.01	1	0.5
Dropout probability	0.3	0.5	0.3 - discriminator 0.5 - generator
Initial learning rate	0.0002		0.0002
Number of epochs	50	150	100

4.4 Model Training

Adopting an Adam optimizer (Kingma & Lei Ba, 2014), all generators are first trained, and following an entire epoch, the discriminators are then trained. The flow chart (Figure 4.3) represents the training procedure. Training the generators for every odd epoch and discriminators for every even epoch was found to reduce the training time and promote faster convergence. While training, the initial learning rate is halved every 50 epochs, and checkpoints are saved after every ten epochs. After 100 epochs, training activity is terminated, with the best performance observed at the final epoch.

The model is written in the TensorFlow 2.8.2 environment and trained in Google Collab, offering users free GPU resources. The pseudocode (Figure 4.4) displays the training procedure and losses calculated at each step.

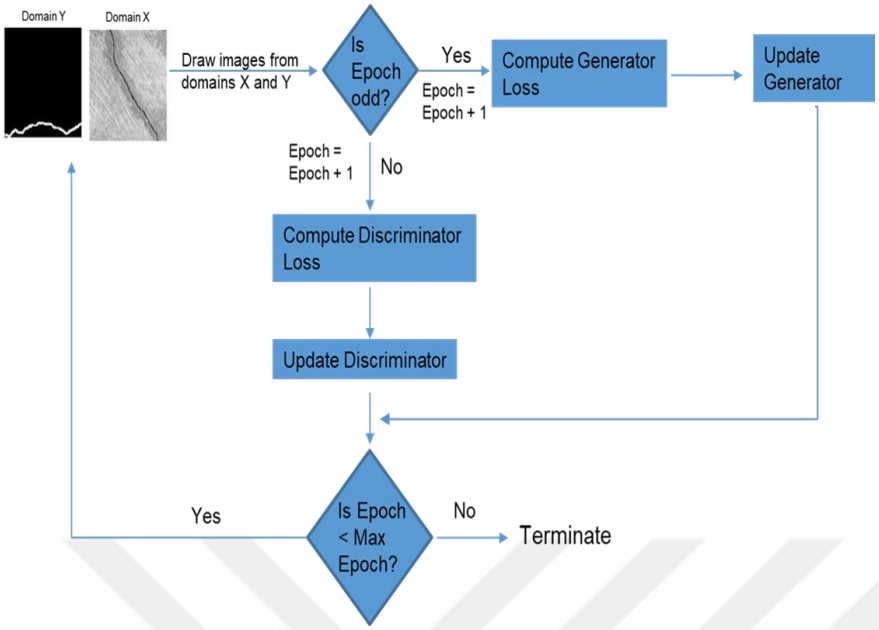

Figure 4.3. CycleGAN implementation flow chart

Algorithm 2. Training Procedure for the Cycle GAN

Input: Crack image set X, unpaired image labels Y, forward cycle generator G, discriminator D_y and reverse cycle generator F, discriminator D_x

Procedure:

$Epoch$ – set epoch to an integer value

for $i \leftarrow 0$ *to* $Epoch$ **do**

Draw m samples from Domain X $\{x_1, x_2, \ldots, x_m\}$

Draw n samples from Domain Y $\{y_1, y_2, \ldots, y_m\}$

if $i + 1 \; mod \; 2 \; == 1$ **do**

Compute the $X \rightarrow Y$ generator loss:

$$L^{G_{X\rightarrow Y}} = \frac{1}{m}\sum_{i=1}^{m}\left(D_y\left(G_{X\rightarrow Y}(x^i)\right)-1\right)^2 + \frac{1}{m}\sum_{i=1}^{m}\|x_i - \hat{x}_i\|_1$$

Compute the $Y \rightarrow X$ generator loss:

$$L^{F_{Y\rightarrow X}} = \frac{1}{n}\sum_{i=1}^{n}\left(D_x\left(F_{Y\rightarrow X}(y^i)\right)-1\right)^2 + \frac{1}{n}\sum_{i=1}^{n}\|y_i - \hat{y}_i\|_1$$

Update generator according to equation (3)

else

Compute discriminator loss on real images:

$$L^{D}_{real} = \frac{1}{m}\sum_{i=1}^{m}\left(D_x(x^i)-1\right)^2 + \frac{1}{n}\sum_{i=1}^{n}\left(D_y(y^i)-1\right)^2$$

Compute discriminator loss on fake images:

$$L^{D}_{fake} = \frac{1}{m}\sum_{i=1}^{m}\left(D_y(G_{X\rightarrow Y}(x^i))\right)^2 + \frac{1}{n}\sum_{i=1}^{n}\left(D_x(F_{Y\rightarrow X}(y^i))\right)^2$$

Update discriminators according to equation (2)

Figure 4.4. CycleGAN implementation pseudocode

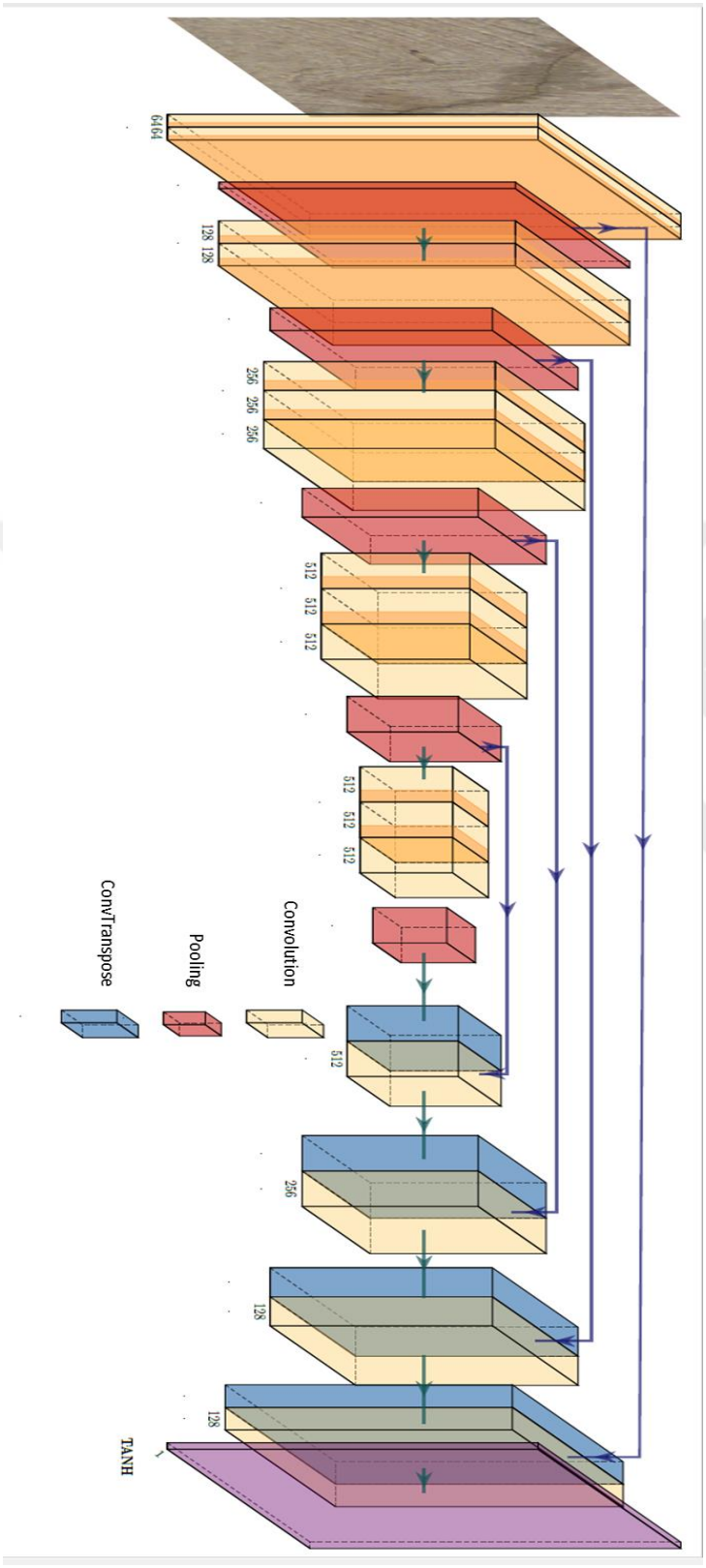

Figure 4.5. Generator architecture

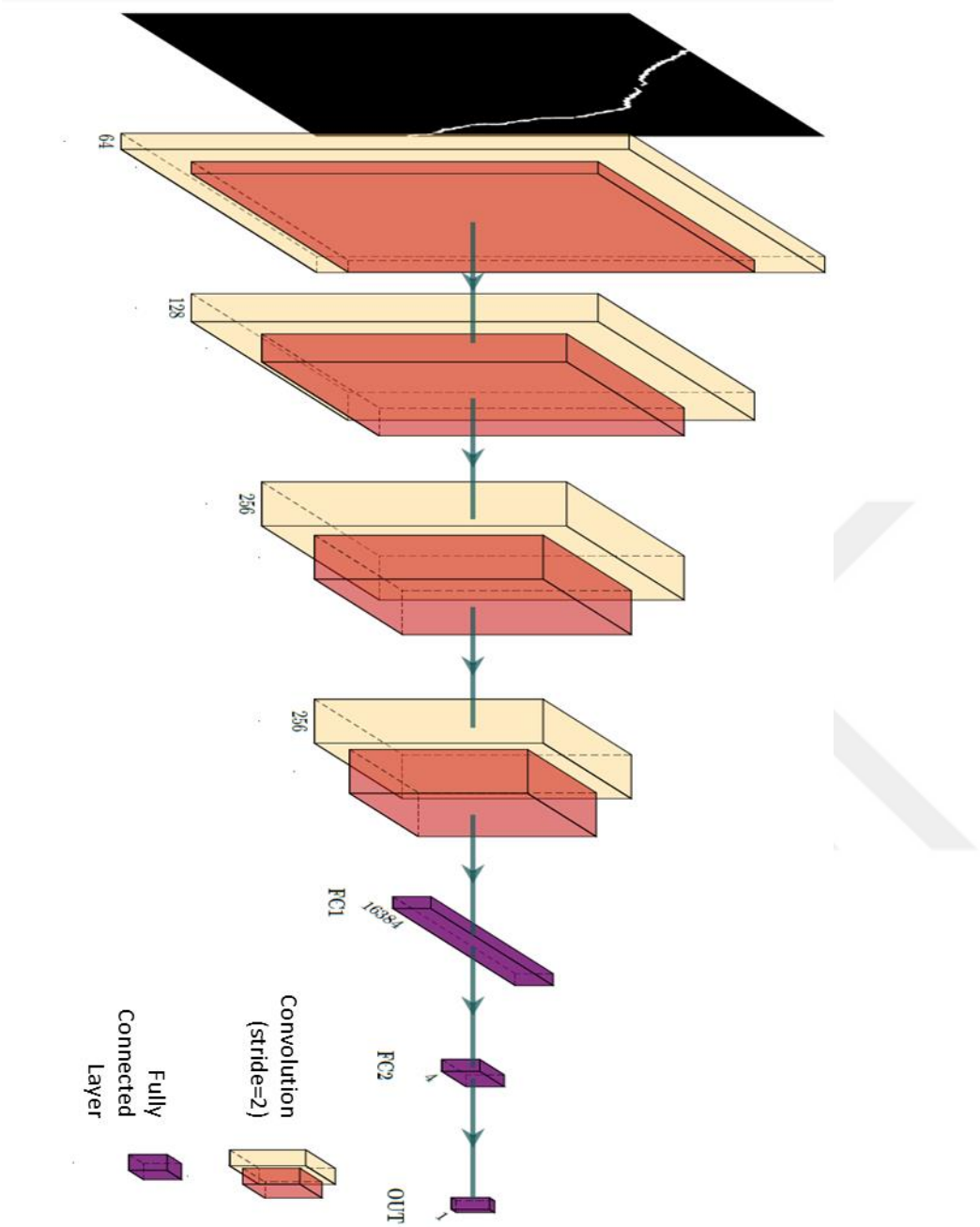

Figure 4.6. Discriminator architecture

CHAPTER 5

DATASETS

5.1 Overview

As mentioned in the previous chapter, the thesis aims to propose an unsupervised method for identifying surface cracks, emphasizing cracked pavement surfaces. Road pavements can be broadly divided into flexible and rigid pavements based on the nature of the constituent layers. Their topmost layer is constructed from asphaltic concrete for flexible pavements, whereas their top layer is built from cementitious concrete for rigid pavements. This difference leads to a difference in cracks formed as well as the nature of the background, i.e., flexible pavements have 'noisy' backgrounds due to the presence of reflective materials in the asphalt binder, whereas rigid pavements have smooth backgrounds. The methodology adopted is trained and tested using a collected rigid pavement dataset. The paragraphs below outline the collection method, source, division of data, and size of the dataset utilized.

5.2 Compiling Crack Dataset

A rigid pavement dataset is collected and composed of images from the FCN dataset (X. Yang et al., 2018) and the drone dataset (Ersoz et al., 2017). The FCN dataset (X. Yang et al., 2018) is a public dataset consisting of 776 images obtained from pavements and buildings in China of size 327x306 pixels. The drone dataset (Ersoz et al., 2017) contains 154 images, 4000x3000 pixels in size, collected by flying a drone 0.5 to 3 m above the ground along a rigid pavement at Middle East Technical University. In preparation for training, the full-size images are divided into crops of size 800 X 500 pixels. Following this division, images presenting no cracks or foreign bodies are eliminated. This augmentation results in a total of 1678 images

for training and testing. After collecting all pictures, these are divided into 4:1 for training and testing, totaling 1960 images for training. Figure 5.1 below displays this division:

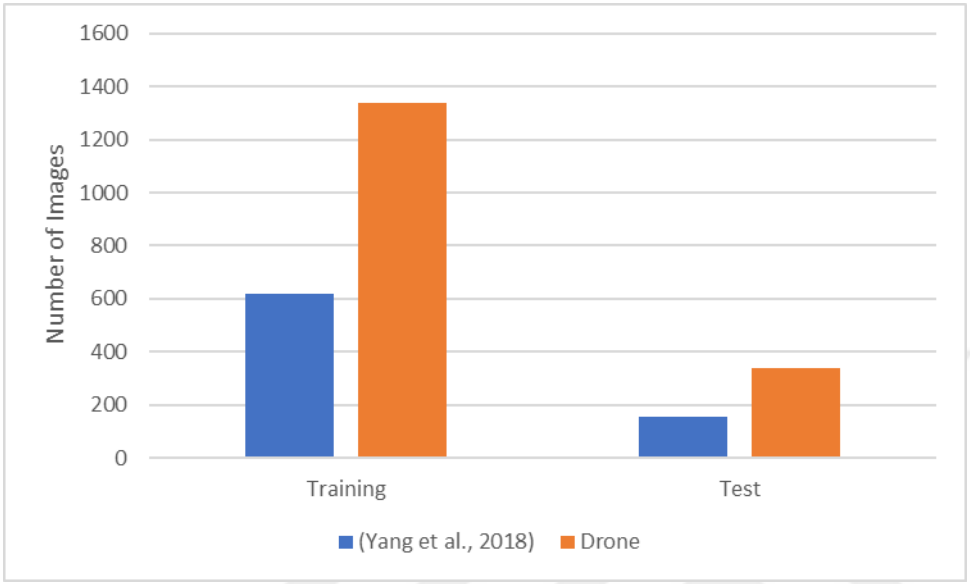

Figure 5.1. Division of datasets for training and testing

5.3 Compiling Unpaired Groundtruth Labels

In their article, Duan et al. (2020) collected unpaired ground truth labels by drawing crack labels on a white background. A total of 30 images are drawn, which are then augmented to create a label set of 120 images used in training. The model is then trained on 118 crack images chosen from the CrackForest dataset and tested on the remaining ten images. K. Zhang et al. (2020) assembled their ground truth labels by cropping 64X64 blocks from full-sized binary images with different structure patterns. Crack images are prepared by cropping 64x64 image blocks from the CFD, CrackTree, and FCN datasets. Following the abovementioned methods, the thesis adopts a different method for ground truth label collection. Unpaired ground truth labels of different crack datasets are collected and augmented through flipping to become equivalent to the training crack images. Thus, during the training of the rigid pavement dataset, the ground truth labels from the CFD dataset (Shi et al., 2016),

CrackTree200 dataset (Zou et al., 2012), Gaps348 dataset (Eisenbach et al., 2017), and Crack500 dataset (F. Yang et al., 2019) are utilized during the training (Figure 5.2). Table 5.1 below represents the number of images collected from the datasets.

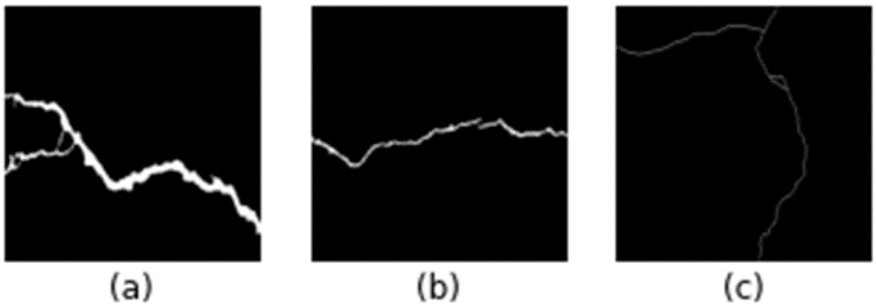

Figure 5.2. Sample crack images from different datasets (a) Crack500 Dataset, (b) CFD Dataset, and (c) CrackTree200 Dataset sample

Table 5.1 Number of ground truth labels collected from different crack image datasets

Dataset	***Number of Images***
CFD	118
CrackTree200	206
Crack500	494
GAPS384	169
Augmentation	973
Total	**1960**

5.4 Preprocessing

Before training, crack images are converted to grayscale and resized to 128 X 128-pixel size. Grayscale conversion reduces the training complexity by getting rid of unnecessary color information. Resizing the image to 128 X128 allows the image to retain crack information while simultaneously speeding up the training process. Beyond this, the images are normalized to values between -1 and 1. Normalization is necessary as it improves model performance. Furthermore, normalization between -1 and 1 in combination with the Tanh activation function in the final layer of the

generator aims to constrain the generator output and allows for faster training and saturation over a color space (Radford et al., 2016).

CHAPTER 6

PERFORMANCE EVALUATION

Following the detailed introduction to the proposed model and the datasets compiled, this chapter details the evaluation of its performance. First, the chapter shall present the evaluation metrics adopted in crack segmentation. Second, comparative models are used in the model's performance evaluation, and then metric segmentation results are shared. Finally, the results of the investigative study on the effect of change in discriminator architecture and the application of transfer learning in the generator are discussed.

6.1 Evaluation Metrics

Different metrics are utilized to evaluate segmentation results. These metrics range from precision, recall, and F1 score to accuracy. Calculating the scores can be done at either the pixel level or at a regional level. First, at the pixel level, a comparison between pixels in the ground truth mask and those in the predicted mask with an allowable tolerance is made (see equations (7) to (10)). Tolerance most observed in the literature ranges from 0 to 5-pixel tolerance, with a larger tolerance leading to higher accuracy. Regional result-based metrics involve the creation of a matrix whose rows and columns reflect patches in the image (Figure 6.1). If crack pixels are present, the cells in the matrix are assigned the number 1 (representing crack cell); otherwise, 0 (representing non-crack cell) (see Figure 6.1b). This matrix based on the captured image is then compared to the predicted matrix, and region-based precision, recall, and F1 scores are calculated as in equations (11) to (14).

The calculation of precision, recall, F1, and accuracy scores are as follows:

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (9)$$

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

Where:

TP: True Positive

FP: False Positive

FN: False Negative

TN: True Negative

$$Precision_{regional} = \frac{TP_{region}}{TP_{region} + FP_{region}} \quad (11)$$

$$Recall_{regional} = \frac{TP_{region}}{TP_{region} + FN_{region}} \quad (12)$$

$$F1 = \frac{2 * Precision_{regional} * Recall_{regional}}{Precision_{regional} + Recall_{regional}} \quad (13)$$

$$Acc = \frac{TP_{region} + TN_{region}}{TP_{region} + TN_{region} + FP_{region} + FN_{region}} \quad (14)$$

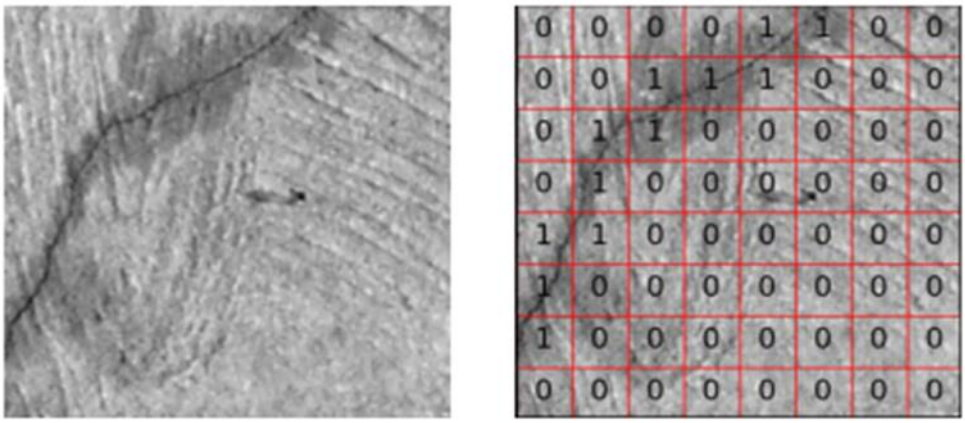

Figure 6.1. Representation of the process of creating labels for region-wise metric analysis (a) The original image, (b) The original image overlaid by the grid representation of crack cells

Each of the scores reveals information about the algorithm or model chosen. For example, a lower precision value indicates the model identifies cracks where there are none in the image (a high number of false positives). A low recall value implies the model does not correctly identify crack pixels where they are observed in the ground truth mask (a high number of false negatives). The F1 score measures the model's overall performance, also referred to as the dice score or harmonic mean.

Apart from the metrics explored above, the Enhanced Hurdsoff distance (EHD) (Tsai & Chatterjee, 2017) is also utilized in displaying the overall localization accuracy of the model. The metric is calculated based on the mean distance between the ground truths and the detected cracks and aims to evaluate model results while reducing the effects of subjectivity involved in developing the ground truth masks. Hsieh and Tsai (2020) and K. Zhang et al. (2020, 2021) utilized this metric to evaluate crack segmentation. Equations (15) to (17) below are used to calculate the EHD score between A and B.

$$score_{BH}(A,B) = 100 - \frac{BH(A,B)}{u} * 100 \quad (15)$$

Where:

$$BH(A,B) = max\left[h_p(A,B)h_p(B,A)\right] \quad (16)$$

Given the penalty $h_p(\mathrm{A},\mathrm{B})$ is defined as

$$h_p(A,B) = \frac{1}{|A|}\sum_{a\epsilon A} sat_u(min_{b\epsilon B}\|a-b\|) \quad (17)$$

u is an upper limit value that is used to eliminate the influence of false positives that are far from the GTs hence reflecting better localization accuracy. Tsai and Chatterjee (2017) proposed setting this value as $1/5$ of the image width. In the equations above, A and B represent the predicted and ground truth crack pixel locations, respectively.

6.2 Comparative Methodologies:

Test results are compared to those obtained by already existing segmentation models to evaluate the overall performance of the adopted methodology. The paragraphs below detail chosen models that shall be compared.

The CrackForest algorithm (Shi et al., 2016) adopts a Random Structured Forest. The authors first identify integral channel features to redefine crack tokens, following which the random structured forests are used to identify cracks which are then characterized.

X. Yang et al. (2018) Fully Convolutional Network (FCN) model provides end-to-end supervised training on a pixel-labeled dataset. In the FCN network, the authors adopt VGG19 architecture loaded with pre-trained weights during downsampling.

The Pix2pix model (Isola et al., 2017) utilizes conditional GANs for image-to-image translation tasks. The authors propose a U-Net-based generator architecture and a 'PatchGAN' classifier network for the discriminator, which penalizes results at a scaled image patch level. As analyzed in the article, a lambda value for the L1 multiplier of 100 is chosen during training.

6.3 Metric Results

This section introduces the metric results of the FCN and Drone datasets.

As explored in chapter 5, the rigid pavement dataset collected consisted of the FCN and drone datasets. During training, three different training dataset combinations are leveraged. Table 6.1 below summarizes the training dataset combinations and the sections and tables where the testing results are discussed.

Table 6.1 Training and test combinations summary

	Training Dataset Combination	***Number of training images***	***Testing Dataset***			
			FCN		***Drone***	
			Discussion Chapter	***Results Table***	***Discussion Chapter***	***Results Table***
1	Drone + FCN	1960	Section **6.2.2.1**	Table 6.2	Section **6.2.2.2**	Table 6.5
2	Drone only	1340				
3	FCN only	620				

While training on the first and second dataset combinations, the investigated and applied hyperparameter values in section 4.3 of chapter 4 were kept constant. However, these parameters were not transferrable while training the model on the third combination (FCN dataset only). Therefore, another grid search was performed, and it was determined that keeping all other hyperparameters constant and changing the cycle consistency loss multiplier to 48 resulted in the best metric results. FCN dataset and Drone dataset results are elaborated below.

6.3.1 Results on the FCN Dataset

A total of 156 images in the FCN dataset are used to test the model's performance. Metric scores when training on the three dataset combinations are shown in Table

6.2 below. The overall best metric results are achieved when training on the Drone + FCN dataset combination, with the lowest results being observed when training on the FCN dataset only. As seen in Figure 6.2d, when training on the FCN dataset only, though the crack is well defined, noise is observed in the areas around the crack hence lower performance.

Table 6.2 Region-based metric results on the FCN dataset under different training combinations

Training Dataset	***Precision***	***Recall***	***F1 score***
Drone + FCN	**0.82**	0.87	**0.84**
FCN only	0.76	0.80	0.78
Drone only	0.76	**0.88**	0.81

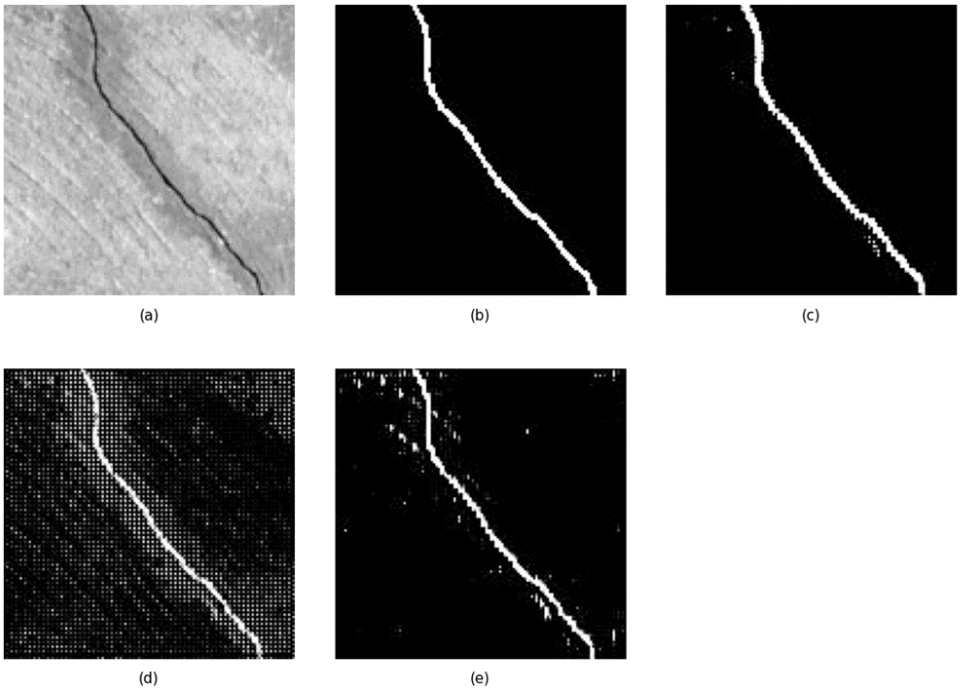

Figure 6.2. The pictorial result on the FCN dataset (a) Represents the original crack image (b) The ground truth mask (c) Prediction of the original image when training on the Drone + FCN dataset (d) Prediction when training on the FCN dataset only (e) Prediction when training on the drone dataset only

The best results in Table 6.2 above are compared with FCN test results from the comparative methodologies discussed in section 6.2.1. The CrackForest (Shi et al., 2016), X. Yang et al. (2018), and Pix2pix models are trained on the 620 FCN training images, predefined during train to test dataset division in section 5.2. The FCN dataset contains pixel-level accurate labels and thus allows for end-to-end training of the supervised model. Following training, these models are tested on the 156 FCN test images (utilized in Table 6.2 above), with precision, recall, F1, and EHD scores displayed in Table 6.3 and Table 6.4 below for region-based and pixel-wise metric scores, respectively. Figure 6.2 further displays the pictorial results of image samples in the test set.

The Pix2pix model achieves the best results with an F1 score of 0.92 and an EHD score of 93.6. As seen in Table 6.3 and Figure 6.3c, the CrackForest algorithm attained the lowest precision score, caused by a high number of false positives observed as the algorithm predicted larger crack widths in the images. As observed in Figure 6.3d, The FCN model failed to recognize thin cracks. A similar issue is reported by X. Yang et al. (2018). The proposed model is observed to achieve comparable results to the supervised algorithms, with the recall score coming second only to the pix2pix model and achieving a higher precision score than the CrackForest model.

Table 6.3 Comparative results on the FCN dataset (Region-based metrics)

Algorithm	***Precision***	***Recall***	***F1 score***	***EHD score***
Supervised Machine Learning based				
CrackForest (Shi et al., 2016)	0.68	0.85	0.76	82
Supervised Deep Learning based				
X. Yang et al. (2018)	**0.96**	0.82	0.88	91
Pix2pix	0.93	**0.91**	**0.92**	**94**
Unsupervised Deep Learning based				
Proposed	0.82	0.87	0.84	89

Table 6.4 Comparative results on the FCN dataset (Pixel-wise metrics)

Algorithm	***Precision***	***Recall***	***F1 score***
Supervised Machine Learning based			
CrackForest (Shi et al., 2016)	0.55	0.87	0.67
Supervised Deep Learning based			
X. Yang et al. (2018)	**0.93** 0.82 - Given in the paper	0.65 0.79 - Given in the paper	0.76
Pix2pix	0.91	**0.91**	**0.91**
Unsupervised Deep Learning based			
Proposed	0.89	0.69	0.78

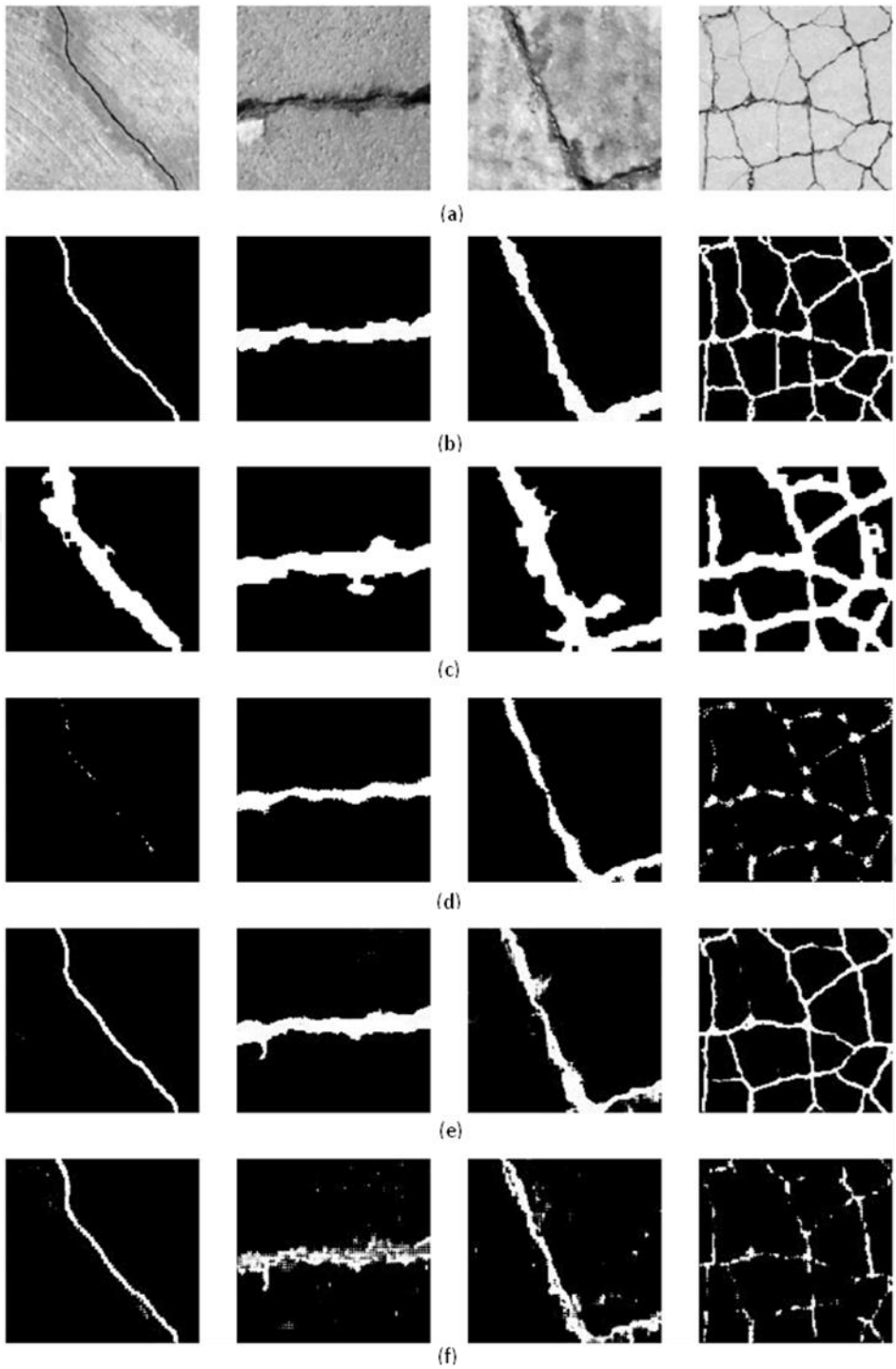

Figure 6.3. The pictorial results on the FCN dataset (a) Crack image (b) Ground truth label (c) CrackForest (d) X. Yang et al. (2018) (e) Pix2pix model (f) Proposed model

6.3.2 Results on the Drone Dataset

As the drone dataset does not contain pixel-accurate labels, simple labels are prepared by creating 2-pixel width labels representing the cracks. Furthermore, since the labels are not pixel accurate, the calculation of region-based metrics is most suitable. Table 6.5 below displays the precision, F1 score, and recall of the testing images when training is done on the three different dataset combinations. Training on the drone dataset only is found to achieve the best overall results.

Table 6.5 Region-based metric results on the drone dataset with different training combinations

Training Dataset	***Precision***	***Recall***	***F1 score***
Drone + FCN	**0.86**	0.72	0.78
FCN only	0.84	0.71	0.77
Drone only	0.84	**0.84**	**0.84**

The best results in Table 6.5 above are compared with those observed in the four comparative methodologies chosen. Supervised deep-learning models utilized in the literature require accurate pixel-level labels for training. However, for the drone dataset, this is not available. Therefore, the CrackForest (Shi et al., 2016), X. Yang et al. (2018), and Pix2pix models, trained on the 620 FCN training images, as defined in section 5.2, are tested on the drone dataset's test images utilized in Table 6.5 above. Each model's precision, recall, F1, and EHD scores are shown in

Table 6.6. Figure 6.4 further displays the pictorial results of image samples in the test set.

The pix2pix model achieves the best metric results, with an overall F1 score of 0.89. CrackForest (Shi et al., 2016) failed to detect thin cracks and was insensitive to the crack's width. Though the FCN (X. Yang et al., 2018) performed better than the

CrackForest, thin crack detection was still poor. The proposed model achieves comparable results with the precision, recall, F1 score, and EHD score only 4% lower than that achieved in pix2pix.

Table 6.6 Comparative results on drone dataset (Region-based metrics)

Algorithm	***Precision***	***Recall***	***F1 score***	***EHD score***
Supervised Machine Learning based				
CrackForest (Shi et al., 2016)	0.46	0.67	0.55	69
Supervised Deep Learning based				
X. Yang et al. (2018)	0.80	0.73	0.76	81
Pix2pix	**0.88**	**0.89**	**0.89**	**92**
Unsupervised Deep Learning based				
Proposed	0.84	0.87	0.84	85

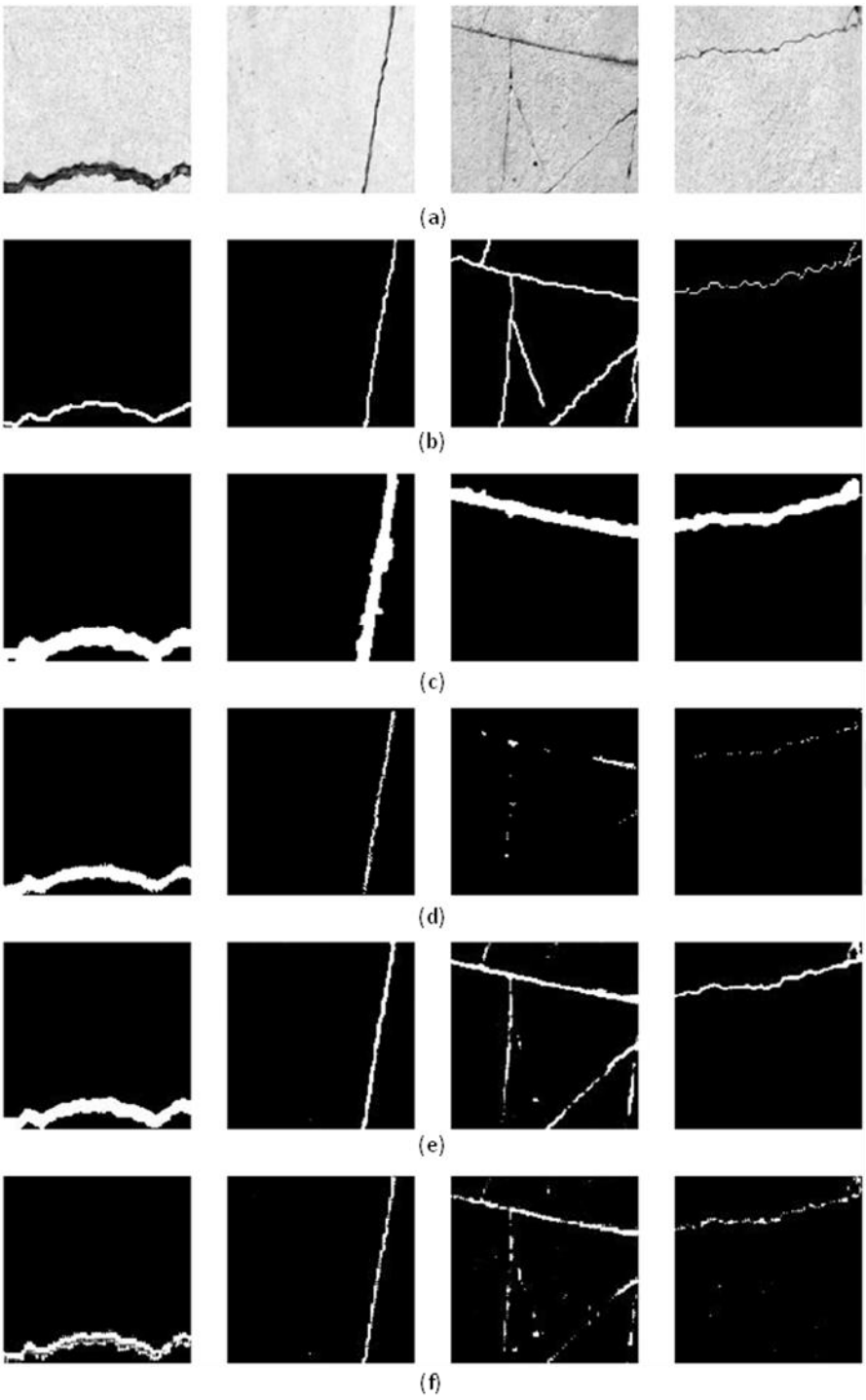

Figure 6.4. Pictorial results on the drone dataset (a) Crack image (b) Ground truth label (c) CrackForest (d) X. Yang et al. (2018) (e) Pix2pix model (f) Proposed model

6.4 Investigative Study

An investigative study is done to determine the effect of the change in discriminator architecture and the adoption of ImageNet (Deng et al., 2010) pre-trained weights in the generator on the proposed model.

6.4.1 Change in Discriminator architecture.

In the original CycleGAN model, Zhu et al. (2017) proposed the use of the PatchGAN architecture (see Chapter 3); however, in the application of CycleGAN to crack detection, K. Zhang et al. (2020) and in this study, a one-class classifier architecture is adopted. Therefore, maintaining all other hyperparameters, the performance of the two different architectures is compared. Figure 6.5 below shows the loss difference between the utilization of the two different discriminator architectures. As seen in Figure 6.5, though a significant difference is observed in the initial loss values, both models achieve Nash equilibrium at almost the same number of epochs. However, despite this equivalence of equilibrium, it is observed that the generated images in the PatchGAN model were sporadic, a situation that could be defined as mode collapse. This situation could be attributed to the choice of the cycle consistency loss multiplier adopted during training (Figure 6.6).

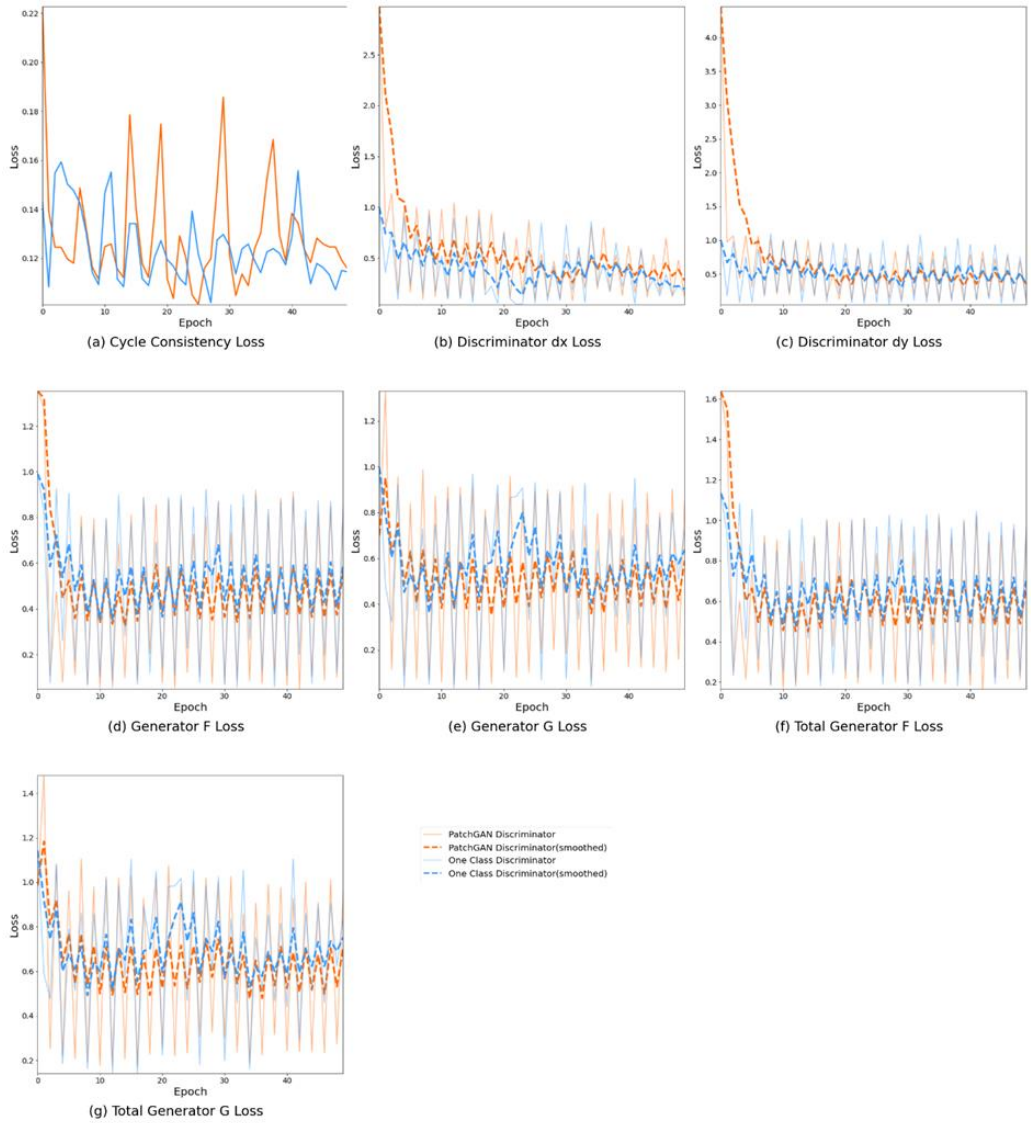

Figure 6.5. Loss comparison between the adoption of the One-class discriminator and the PatchGAN discriminator. Graphs (a) to (g) represent the different loss values calculated during training for the first 50 epochs

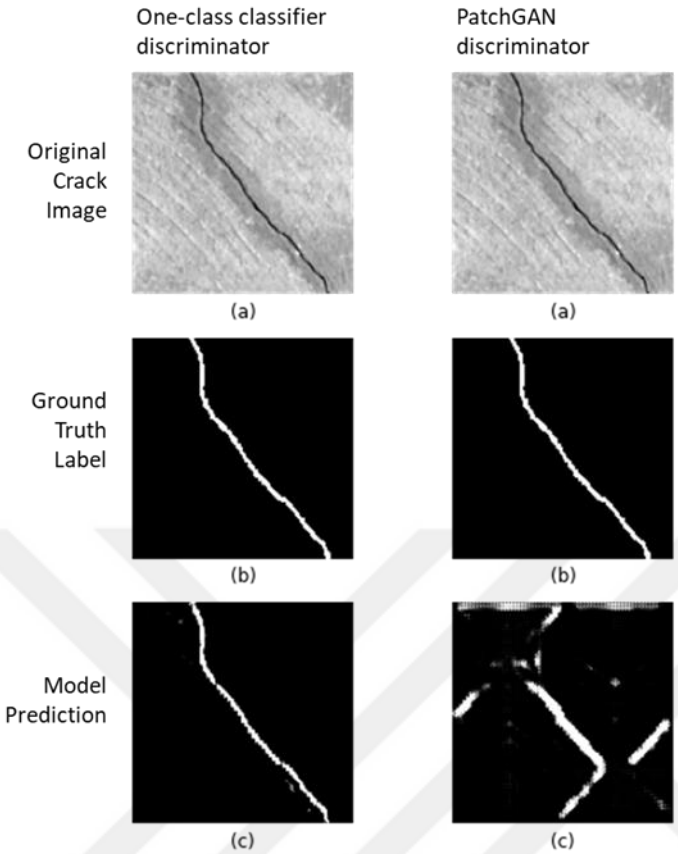

Figure 6.6. Pictorial results of the two different discriminator models. (a) Original Crack images (b) Ground truth labels (c) Model predictions

6.4.2 Application of Transfer Learning in the Generator

Transfer Learning hopes to exploit information learned in a different task to the current task. In deep learning, this is distinguished as a repurposing of trained models loaded with their known weights to the current task (Bengio, 2012). In investigating the effect of transfer learning, the previously loaded generator with ImageNet (Deng et al., 2010) weights is initialized randomly with values exhibiting a uniform distribution with a mean of 0 and a standard deviation of 0.02. The model is then trained for 50 epochs on Drone + FCN dataset combination keeping all hyperparameters constant. The losses observed are displayed in Figure 6.7 below. From the analysis of the losses, no significant difference is observed. However, upon

further inspection of the pictorial results at the final epoch, it was observed that pre-trained weights offer significantly better crack connectivity at the 50th epoch (Figure 6.8).

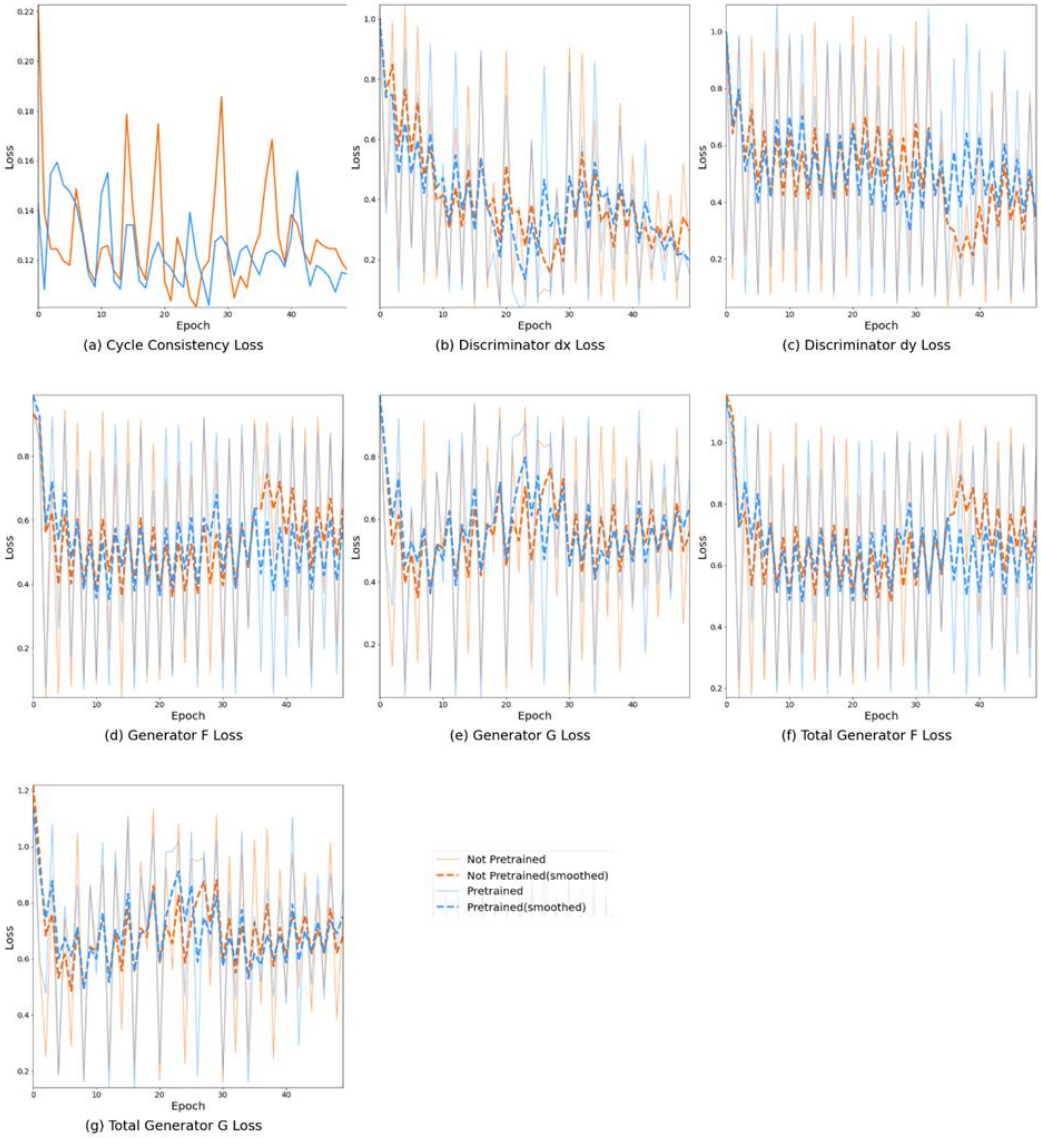

Figure 6.7. Loss comparison of the effect of utilizing vgg16 pre-trained weights as opposed to random uniform initialized. Graphs (a) to (b) represent the different losses calculated for the first 50 epochs

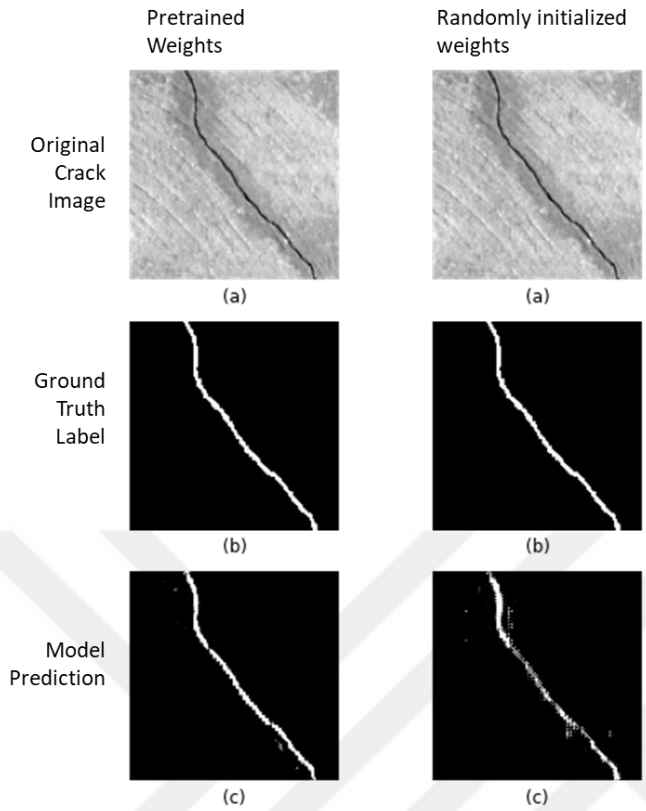

Figure 6.8. Pictorial results showing the effect of utilizing pre-trained weights instead of uniform random initialized weights. (a) Original Crack images (b) Ground truth labels (c) Model predictions

CHAPTER 7

SUMMARY, CONCLUSION, AND RESEARCH PROSPECTS

7.1 Summary

Road pavements are exposed to different loading conditions during their lifespan. These conditions lead to the road structure's degradation, causing defects such as cracks, potholes, or depressions. Degradation of road integrity could pose a problem to vehicle safety and cause transport inefficiency. Furthermore, immense costs could be incurred during repair, especially if a rework of the pavement is required. Therefore, regular monitoring should be done to prevent the detrimental decay of the pavements. The study, therefore, focused on pavement monitoring to detect cracks.

Different algorithms are utilized in the detection of cracks on pavement surfaces. The oldest algorithm used was image processing. With the growth in computing capability, traditional machine learning algorithms such as ANN, SVM, and Random Forests have been adopted alongside image processing methods. However, these techniques are heavily affected by false crack detection in images with shadows, low contrast, and discontinuous crack regions. Furthermore, shallow learning techniques utilized are not suitable for complex information in the images (Hsieh & Tsai, 2020).

Deep learning models have been applied to crack detection to mitigate the disadvantages. Different application levels exist, such as image-based and patch-based classification, object detection, and pixel segmentation. Of these application levels, deep learning has increasingly been applied in pixel segmentation of cracks.

Deep learning models could be divided into supervised and unsupervised deep learning models. Supervised models require image labels for end-to-end training, whereas unsupervised models do not. Supervised methodologies are disadvantageous as the model’s training heavily depends on accurate pixel-wise

labels. Moreover, generating pixel-wise accurate crack labels is expensive. Therefore, the study aimed to utilize unsupervised deep learning methods in pixel-wise segmentation of images.

A comprehensive literature survey was conducted using keywords such as unsupervised, semi-supervised, and self-supervised crack detection. In the literature, unsupervised cluster-based algorithms were applied. These algorithms were disadvantageous in that, in contrast to deep learning methods, they require prior parameter extraction. Furthermore, feature extraction is inefficient in increasingly complex images. Minimal Path Selection based models are also utilized in the literature. However, they require selecting parameters such as threshold, which is not interchangeable across datasets. Novel algorithms have also been adopted in literature; however, they face similar disadvantages as MPS and cluster-based algorithms.

CycleGAN offered the best performance with the lowest computational cost of the unsupervised deep learning models researched. CycleGAN is a model consisting of two Generative Adversarial Networks which perform image-to-image translation. The forward cycle maps the crack image to its ground truth image, whereas the reverse cycle maps the ground truth to its crack image. This process requires the calculation of adversarial losses in the generators and cycle consistency loss in a single forward-reverse cycle. The introduction of the cycle consistency loss constrains the model.

Implementing the CycleGAN requires a one-class discriminator with five convolutional blocks. In addition, a fully convolutional Network (FCN) with a VGG16 encoder section is applied for the generators. A hyperparameter grid search is then conducted on the dataset to obtain the best performance. Moreover, to improve performance and training speed, the generators are trained for an entire odd epoch and discriminators for an even one.

The rigid pavement dataset is compiled following the determination of the best training algorithm. First, images from FCN (X. Yang et al., 2018) and drone (Ersoz

et al., 2017) public datasets are collected. The FCN dataset contains 776 images of size 327X306, captured from buildings and pavements in China. The drone dataset includes images taken by flying a drone 0.5 to 3m above the ground along a rigid pavement at Middle East Technical University. A total of 154 images of size 4000x3000 are obtained. The drone images are subsequently cropped to 800x500 pixels resulting in a total of 1678 images. The datasets are individually divided into a ratio of 4:1 for training and testing. Beyond this, unpaired ground truth labels are obtained by collecting ground truth labels from the CFD dataset (Shi et al., 2016), CrackTree200 dataset (Zou et al., 2012), Gaps348 dataset (Eisenbach et al., 2017), and Crack500 dataset (F. Yang et al., 2019) public crack pavement database.

7.2 Conclusions

The study proposed CycleGAN, an unsupervised deep learning algorithm, for the pixel-wise segmentation of a rigid pavement dataset. In doing so, a novel model architecture and training procedure are utilized. Furthermore, an unpaired ground truth label dataset is collected by compiling ground truth labels from the public crack pavement database.

To evaluate the performance of the applied algorithm, precision, recall, F1 score, and EHD score are calculated. While training the algorithm, three different training dataset combinations are investigated. Firstly training is performed on all datasets (1960 images) on the FCN dataset only (620 images) and the drone dataset only (1340 images). Training on 1960 images achieves the best results while testing the FCN dataset with an F1 score of 0.84. On the other hand, while testing on the drone images, training on the drone dataset only attains the best results with an F1 score of 0.84.

The proposed model performance is compared with those in the literature. When testing the model with the FCN dataset, the model achieved comparable results with CrackForest (Shi et al., 2016) and FCN (X. Yang et al., 2018) supervised models.

Furthermore, when testing the model with the drone dataset, the CrackForest and FCN models were outperformed.

Finally, the effect of loading pre-trained weights and changing the discriminator architecture is investigated. Adopting the 'PatchGAN' discriminator architecture results in mode collapse as the discriminator produces sporadic results. Loading pre-trained weights was not observed to affect the training convergence. However, at the 50th epoch, the model loaded with pre-trained weights achieved better crack connectivity.

Despite achieving high performance, the proposed model fails to recognize thinner cracks in pavement images. Furthermore, the model is sensitive to shadows, which is also observed in supervised algorithms.

7.3 Research Prospects

As the model is sensitive to shadows, an unsupervised shadow removal algorithm could be adopted to alleviate this problem. In addition to segmentation, the classification of the different crack patterns through unsupervised models such as K-means clustering or Principal Component Analysis (PCA) could be undertaken.

REFERENCES

Abdel-Qader, I., Pashaie-Rad, S., Abudayyeh, O., & Yehia, S. (2006). PCA-Based algorithm for unsupervised bridge crack detection. Advances in Engineering Software, 37(12), 771–778. https://doi.org/10.1016/J.ADVENGSOFT.2006.06.002

Amhaz, R., Chambon, S., Idier, J., & Baltazart, V. (2016). Automatic Crack Detection on Two-Dimensional Pavement Images: An Algorithm Based on Minimal Path Selection. IEEE Transactions on Intelligent Transportation Systems, 17(10), 2718–2729. https://doi.org/10.1109/TITS.2015.2477675

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein GAN. 34th International Conference on Machine Learning, PMLR, 214–223.

Augustauskas, R., & Lipnickas, A. (2020). Improved Pixel-Level Pavement-Defect Segmentation Using a Deep Autoencoder. Sensors 2020, Vol. 20, Page 2557, 20(9), 2557. https://doi.org/10.3390/S20092557

Cha, Y.-J., & Choi, W. (2017). Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks. Computer-Aided Civil and Infrastructure Engineering, 32, 361–378. https://doi.org/10.1111/mice.12263

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2020). InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. 2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE), 1487–1490.

Cheng, J., Xiong, W., Chen, W., Gu, Y., & Li, Y. (2019). Pixel-level Crack Detection using U-Net. IEEE Region 10 Annual International Conference, Proceedings/TENCON, 2018-Octob, 462–466. https://doi.org/10.1109/TENCON.2018.8650059

Chow, J. K., Su, Z., Wu, J., Tan, P. S., Mao, X., & Wang, Y. H. (2020). Anomaly detection of defects on concrete structures with the convolutional autoencoder. Advanced Engineering Informatics, 45, 101105. https://doi.org/10.1016/J.AEI.2020.101105

Duan, L., Geng, H., Pang, J., & Zeng, J. (2020). Unsupervised Pixel-level Crack Detection Based on Generative Adversarial Network. PervasiveHealth: Pervasive Computing Technologies for Healthcare, 6–10. https://doi.org/10.1145/3404716.3404720

Eisenbach, M., Stricker, R., Seichter, D., Amende, K., Debes, K., Sesselmann, M., Ebersbach, D., Stoeckert, U., & Gross, H. M. (2017). How to get pavement distress detection ready for deep learning? A systematic approach. Proceedings of the International Joint Conference on Neural Networks, 2017-May, 2039–2047. https://doi.org/10.1109/IJCNN.2017.7966101

Ersoz, A. B., Pekcan, O., & Teke, T. (2017). Crack identification for rigid pavements using unmanned aerial vehicles. IOP Conference Series: Materials Science and Engineering, 236(1). https://doi.org/10.1088/1757-899X/236/1/012101

Fang, X., Guo, W., Li, Q., Zhu, J., Chen, Z., Yu, J., Zhou, B., & Yang, H. (2020). Sewer Pipeline Fault Identification Using Anomaly Detection Algorithms on Video Sequences. IEEE Access, 8, 39574–39586. https://doi.org/10.1109/ACCESS.2020.2975887

Godard, C., Mac, O., Gabriel, A., & Brostow, J. (2017). Unsupervised Monocular Depth Estimation with Left-Right Consistency. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6602–6611. http://visual.cs.ucl.ac.uk/pubs/monoDepth/

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. Advances in Neural Information Processing Systems 27 (NIPS 2014). http://www.github.com/goodfeli/adversarial

Hoang, N.-D., Nguyen, Q.-L., & Tien Bui, D. (2018). Image Processing–Based Classification of Asphalt Pavement Cracks Using Support Vector Machine Optimized by Artificial Bee Colony. Journal of Computing in Civil Engineering, 32(5), 04018037. https://doi.org/10.1061/(asce)cp.1943-5487.0000781

Hsieh, Y.-A., & Tsai, Y. J. (2020). Machine Learning for Crack Detection: Review and Model Performance Comparison. Journal of Computing in Civil Engineering, 34(5), 04020038. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000918

Isola, P., Zhu, J.-Y., Zhou, T., Efros, A. A., & Research, B. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5967–5976. https://github.com/phillipi/pix2pix.

Ji, R., Shan, S., Zhang, K., & Wei, H. (2020). The self-labeling of unsupervised poly crystalline solar cell micro-crack images. Proceedings - 2020 5th International Conference on Mechanical, Control and Computer Engineering, ICMCCE 2020, 2237–2240. https://doi.org/10.1109/ICMCCE51767.2020.00485

Kaddah, W., Elbouz, M., Ouerhani, Y., Alfalou, A., & Desthieux, M. (2020). Automatic darkest filament detection (ADFD): a new algorithm for crack extraction on two-dimensional pavement images. Visual Computer, 36(7), 1369–1384. https://doi.org/10.1007/S00371-019-01742-2/FIGURES/21

Kaddah, W., Elbouz, M., Ouerhani, Y., Baltazart, V., Desthieux, M., & Alfalou, A. (2019). Optimized minimal path selection (OMPS) method for automatic and unsupervised crack segmentation within two-dimensional pavement images. Visual Computer, 35(9), 1293–1309. https://doi.org/10.1007/S00371-018-1515-9/FIGURES/23

Kalal, Z., Mikolajczyk, K., & Matas, J. (2010). Forward-backward error: Automatic detection of tracking failures. Proceedings - International Conference on Pattern Recognition, 2756–2759. https://doi.org/10.1109/ICPR.2010.675

Kaseko, M. S., & Ritchie, S. G. (1993). A neural network-based methodology for pavement crack detection and classification. Transportation Research Part C, 1(4), 275–291. https://doi.org/10.1016/0968-090X(93)90002-W

Lee, H.-Y., Tseng, H.-Y., Mao, Q., Huang, J.-B., Lu, Y.-D., Singh, M., Yang, M.-H., Zhu, J.-Y., Li, H., Shechtman, E., Liu, M.-Y., Kautz, J., Hsin-, A. T., & Lee, Y. (2020). DRIT++: Diverse Image-to-Image Translation via Disentangled Representations. International Journal of Computer Vision, 128, 2402–2417. https://doi.org/10.1007/s11263-019-01284-z

Lei, B., Wang, N., Xu, P., & Song, G. (2018). New Crack Detection Method for Bridge Inspection Using UAV Incorporating Image Processing. Journal of Aerospace Engineering, 31(5), 04018058. https://doi.org/10.1061/(ASCE)AS.1943-5525.0000879

Li, H., Song, D., Liu, Y., & Li, B. (2019). Automatic Pavement Crack Detection by Multi-Scale Image Fusion. IEEE Transactions on Intelligent Transportation Systems, 20(6), 2025–2036. https://doi.org/10.1109/TITS.2018.2856928

Liu, M.-Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., & Kautz, J. (2019). Few-Shot Unsupervised Image-to-Image Translation. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 10550–10559. https://github.com/NVlabs/FUNIT.

Liu, Z., Oviedo, F., Sachs, E. M., & Buonassisi, T. (2020). Detecting Microcracks in Photovoltaics Silicon Wafers using Varitional Autoencoder. Conference Record of the IEEE Photovoltaic Specialists Conference, 2020-June, 0139–0142. https://doi.org/10.1109/PVSC45281.2020.9300366

Marinescu, R. v, Moyer, D., & Golland, P. (2020). Bayesian Image Reconstruction using Deep Generative Models. https://doi.org/10.48550/arxiv.2012.04567

Mathavan, S., Rahman, M., & Kamal, K. (2014). Use of a Self-Organizing Map for Crack Detection in Highly Textured Pavement Images. Journal of Infrastructure Systems, 21(3), 04014052. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000237

Miller, J. S., & Bellinger, W. Y. (2014). FHWA, Distress Identification manual for the Long-Term Pavement Performance Program. Report FHWA-HRT-13-092. Federal Highway Administration, May, 142.

Mirza, M., & Osindero, S. (2014). Conditional Generative Adversarial Nets. ArXiv Preprint ArXiv:1411.1784.

Mubashshira, S., Azam, M. M., & Masudul Ahsan, S. M. (2020). An Unsupervised Approach for Road Surface Crack Detection. 2020 IEEE Region 10 Symposium, TENSYMP 2020, 1596–1599. https://doi.org/10.1109/TENSYMP50017.2020.9231023

Mucolli, L., Krupinski, S., Maurelli, F., Mehdi, S. A., & Mazhar, S. (2019). Detecting cracks in underwater concrete structures: An unsupervised learning approach based on local feature clustering. OCEANS 2019 MTS/IEEE Seattle, OCEANS 2019. https://doi.org/10.23919/OCEANS40490.2019.8962401

Mustafa, A., & Mantiuk, R. K. (2020). Transformation Consistency Regularization – A Semi-supervised Paradigm for Image-to-Image Translation. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12363 LNCS, 599–615. https://doi.org/10.1007/978-3-030-58523-5_35

Oliveira, H., & Correia, P. L. (2013). Automatic road crack detection and characterization. IEEE Transactions on Intelligent Transportation Systems, 14(1), 155–168. https://doi.org/10.1109/TITS.2012.2208630

Pang, Y., Lin, J., Qin, T., & Chen, Z. (2021). Image-to-Image Translation: Methods and Applications. IEEE Transactions on Multimedia. https://doi.org/10.1109/TMM.2021.3109419

Park, T., Liu, M. Y., Wang, T. C., & Zhu, J. Y. (2019). Semantic Image Synthesis with Spatially-Adaptive Normalization. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June, 2332–2341. https://doi.org/10.48550/arxiv.1903.07291

Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks. CoRR.

Shamsabadi, S., Sindhu Ghanta, R., Shahini Shamsabadi, S., Dy, J., Wang, M., Birken, R., & Ghanta, S. (2015). A Hessian-based methodology for automatic surface crack detection and classification from pavement images. Https://Doi.Org/10.1117/12.2084370, 9437(1), 524–534. https://doi.org/10.1117/12.2084370

Shi, Y., Cui, L., Qi, Z., Meng, F., & Chen, Z. (2016). Automatic road crack detection using random structured forests. IEEE Transactions on Intelligent Transportation Systems, 17(12), 3434–3445. https://doi.org/10.1109/TITS.2016.2552248

Silva, L. A., Blas, H. S. S., García, D. P., Mendes, A. S., & González, G. V. (2020). An architectural multi-agent system for a pavement monitoring system with pothole recognition in UAV images. Sensors (Switzerland), 20(21), 1–23. https://doi.org/10.3390/s20216205

Suarez, P. L., Sappa, A. D., & Vintimilla, B. X. (2017). Infrared Image Colorization Based on a Triplet DCGAN Architecture. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017-July, 212–217. https://doi.org/10.1109/CVPRW.2017.32

Tang, Y., Zhang, A. A., Luo, L., Wang, G., & Yang, E. (2021). Pixel-level pavement crack segmentation with encoder-decoder network. Measurement, 184, 109914. https://doi.org/10.1016/J.MEASUREMENT.2021.109914

Tran, V. P., Tran, T. S., Lee, H. J., Kim, K. D., Baek, J., & Nguyen, T. T. (2020). One stage detector (RetinaNet)-based crack detection for asphalt pavements considering pavement distresses and surface objects. Journal of Civil Structural Health Monitoring 2020 11:1, 11(1), 205–222. https://doi.org/10.1007/S13349-020-00447-8

Tsai, Y.-C. (James), & Chatterjee, A. (2017). Comprehensive, Quantitative Crack Detection Algorithm Performance Evaluation System. Journal of Computing in Civil Engineering, 31(5), 04017047. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000696

Vignesh Mohanraj, H., Huang, L., Asghari, H., & Mohanraj, V. (2018). Improved automatic road crack detection and classification. Https://Doi.Org/10.1117/12.2504606, 10836, 49–53. https://doi.org/10.1117/12.2504606

Wang, F., Huang, Q., & Guibas, L. J. (2013). Image co-segmentation via consistent functional maps. Proceedings of the IEEE International Conference on Computer Vision, 849–856. https://doi.org/10.1109/ICCV.2013.110

Wang, Y., Yoshihashi, R., Kawakami, R., You, S., Harano, T., Ito, M., Komagome, K., Iida, M., & Naemura, T. (2019). Unsupervised anomaly detection with compact deep features for wind turbine blade images taken by a drone. IPSJ Transactions on Computer Vision and Applications, 11(1), 1–7. https://doi.org/10.1186/S41074-019-0056-0/FIGURES/5

Wu, S., Fang, J., Zheng, X., & Li, X. (2019). Sample and Structure-Guided Network for Road Crack Detection. IEEE Access, 7, 130032–130043. https://doi.org/10.1109/ACCESS.2019.2940767

Xia, Y., He, D., Qin, T., Wang, L., Yu, N., Liu, T.-Y., & Ma, W.-Y. (2016). Dual Learning for Machine Translation. NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems, 820–828.

Yang, F., Zhang, L., Yu, S., Prokhorov, D., Mei, X., & Ling, H. (2019). Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection. IEEE Transactions on Intelligent Transportation Systems, 21(4), 1525–1535. https://doi.org/10.48550/arxiv.1901.06340

Yang, X., Li, H., Yu, Y., Luo, X., Huang, T., & Yang, X. (2018). Automatic Pixel-Level Crack Detection and Measurement Using Fully Convolutional Network. Computer-Aided Civil and Infrastructure Engineering, 33(12), 1090–1109. https://doi.org/10.1111/MICE.12412

Yi, Z., Zhang, H., Tan, P., & Gong, M. (2017). DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. 2017 IEEE International Conference on Computer Vision (ICCV), 2868–2876.

Zach, C., Klopschitz, M., & Pollefeys, M. (2010). Disambiguating visual relations using loop constraints. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1426–1433. https://doi.org/10.1109/CVPR.2010.5539801

Zhai, W., Zhu, J., Cao, Y., & Wang, Z. (2018). A generative adversarial network based framework for unsupervised visual surface inspection. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2018-April, 1283–1287. https://doi.org/10.1109/ICASSP.2018.8462364

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. (2017). StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. 2017 IEEE International Conference on Computer Vision (ICCV), 5908–5916. https://github.com/hanzhanggit/StackGAN.

Zhang, K., Zhang, Y., & Cheng, H. da. (2021). CrackGAN: Pavement Crack Detection Using Partially Accurate Ground Truths Based on Generative Adversarial Learning. IEEE Transactions on Intelligent Transportation Systems, 22(2), 1306–1319. https://doi.org/10.1109/TITS.2020.2990703

Zhang, K., Zhang, Y., & Cheng, H. D. (2020). Self-Supervised Structure Learning for Crack Detection Based on Cycle-Consistent Generative Adversarial Networks. Journal of Computing in Civil Engineering, 34(3), 04020004. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000883

Zhang, Y., Liu, S., Dong, C., Yang, X., & Yuan, Y. (2020). Multiple cycle-in-cycle generative adversarial networks for unsupervised image super-resolution. IEEE Transactions on Image Processing, 29, 1101–1112. https://doi.org/10.1109/TIP.2019.2938347

Zhou, T., Lee, Y. J., Yu, S. X., & Efros, A. A. (2015). FlowWeb: Joint image set alignment by weaving consistent, pixel-wise correspondences. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June-2015, 1191–1200. https://doi.org/10.1109/CVPR.2015.7298723

Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision (Vols. 2017-Octob). https://doi.org/10.1109/ICCV.2017.244

Zou, Q., Cao, Y., Li, Q., Mao, Q., & Wang, S. (2012). CrackTree: Automatic crack detection from pavement images. Pattern Recognition Letters, 33(3), 227–238. https://doi.org/10.1016/J.PATREC.2011.11