CONCRETE STRUCTURE DAMAGE CLASSIFICATION AND DETECTION USING CONVOLUTIONAL NEURAL NETWORK

by

Palisa Arafin

A thesis
submitted to the Faculty of Graduate Studies
in partial fulfilment of the requirements for the
Degree of Master of Science
in
Civil Engineering

Supervisor
Dr. AHM Muntasir Billah
Assistant Professor – Dept. of Civil Engineering
Co-Supervisor
Dr. Anas Salem Issa
Assistant Professor – Dept. of Civil Engineering

Lakehead University
Thunder Bay, Ontario
August 2022

Author's Declaration Page

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I understand that my thesis may be made electronically available to the public.

Abstract

A resilient infrastructure system remains a top priority for Canada as it is hinged to strong economies. Corrosion, ageing, aggressive environments, material defects, and unforeseen mechanical loads can compromise the serviceability and safety of existing infrastructures. Introducing Artificial Intelligence (AI) in smart structural health monitoring (SHM) can assist in building an automated and efficient infrastructure condition monitoring method to facilitate an effortless inspection and accurate evaluation of deteriorated infrastructure. Over the last decades, vision-based AI has proven successful in pattern recognition applications and motivated this current research to assemble a data-driven damage detection technique. To further explore the possibilities of deep-learning (DL) applications, this dissertation research aims to develop an autonomous damage assessment process using DL techniques to classify and detect two types of defects- crack and spalling on concrete structures. This research started with reviewing existing application of various DL-based technologies for damage detection of concrete structures and identifying the challenges and limitations. One major challenge in DL-based SHM technique is the lack of adequate real image database obtained from field inspection. To address these challenges, this research has created a diverse dataset with concrete crack (4087) and spalling (1100) images and used it for damage detection and classification by applying convolutional neural network (CNN) algorithms. For defects classification VGG19, ResNet50, InceptionV3, MobileNetV2, and Xception CNN models were used, whereas semantic segmentation process adopted Encoder-Decoder Models- U-Net and PSPnet. For both cases a detailed sensitivity analysis of hyper-parameters (i.e., batch size, optimizers, learning rate) was performed to compare their performances and identify the best-performed model. After assessing all the criteria, the best performance for defects classification was achieved by InceptionV3. On the other hand, for crack and spalling segmentation U-net outperformed the other models. Overall, the developed algorithms achieved an excellent performance in damage classification and localization and proved to be successful enough to offer an automated inspection platform for ageing infrastructures. The outcomes of this study signify that the data-driven CNN methods could be a promising solution for the condition assessment of deteriorating concrete structures. The research outcomes can be implemented by practitioners for condition assessment of existing infrastructure and recommending proper rehabilitation measures.

Acknowledgements

First of all, I want to express my sincere gratitude and appreciation to the almighty Allah for the successful completion of my graduate study. I express my deepest gratitude to my advisor Dr. Muntasir Billah for trusting me and giving me an opportunity to work with this project at Lakehead University, Thunder Bay. I greatly acknowledge his enormous support, guidance, motivation and valuable suggestions without which this thesis would have been impossible. I am also immensely grateful to my co-supervisor Dr. Anas Salem Issa for his invaluable suggestions and encouragement involved in completion of thesis work. I gratefully acknowledge the financial support from Natural Science and Engineering Research Council of Canada (NSERC), Dr. Muntasir Billah and Dr. Anas Salem Issa to pursue my graduate degree.

I would also like to appreciate the contribution of the industrial partner of this project, TBT Engineering Limited for providing me all the resources to prepare my dataset in competition of my project. My especial thanks to Dani Rhodes, P.Eng. from TBT Engineering for her prompt response to all my queries and making the collaboration a success.

The period spent at Lakehead University will always be an amazing memory for me. During my studies I have met some impressive teammates from Resilient and Innovative Bridges and Structure (RIBS) research group who have always showered me with positive energy and passion. Finishing my thesis project marks an amazing end of my graduate studies with a beginning of new possibility.

I am deeply indebted to my husband Zakaria, for his endless support in my bad days and having faith on me. Also, my love and heartwarming gratitude to my parents for being an inspiration to me and guide me through all the decisions that I have ever made. I would also like to thank my friends and well-wishers for their support and encouragement throughout my graduate study period.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

1.1 Research Background and Motivation

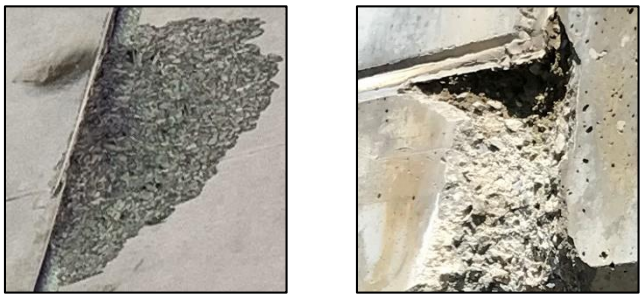
Maintaining a fully functional and healthy infrastructure is crucial to ensure proper serviceability and operation of any society. After a certain service life, these infrastructures including bridges, buildings, dams, power station, etc. faces deterioration while continuously going through extreme environmental changes and loading conditions and ultimately disturbing the reliability of the structures. Nations all around the world face a great challenge in economic and environmental sectors making it imperative for the public and private sectors to provide uninterrupted service. According to ASCE's 2021 Infrastructure Report Card at least 42% of all the bridges in the United States are over 50 years old and 7.5% of these bridges are in “poor” condition implicating the structural deficiency. In total 231,000 bridges acquired a grade “C” indicating a requirement of $125 billion worth repair and preservation work. The 2019 Canadian Infrastructure Report Card stated that Over 40% bridges in Canada were built between 1940 to 1970 making them over 40 years old, and 39% bridges and tunnels are in fair, poor, and very poor condition. The repair and replacement cost of these bridges alone was estimated over CAD 21 billion. The main reasons behind the increment of structure’s life-cycle costs are lack of systematical inspection, proper maintenance, and limited data on deterioration process, which ultimately disrupts the functionality and safety of the structures [Anai et. al. (2019)]. In quest of early detection and safety assessment, Canadian government has considered to inspect bridges on every 540 days or less [The Government of Canada, 2018], while the US government imposed a regulation of biannual bridge inspection [Federal Highway Administration, 2004].

In Canada for any kind of construction and repair work, concrete has always been considered as the primary material [Cusson and Isgor (2004)]. Due to the continuous exposure to aggressive environment, these concrete structures are facing durability issues which is affecting the desired service life of the structure. For deterioration of reinforced concrete structures, carbonation and chloride-induced corrosion are mostly responsible. Later on these corrosions results in defects on concrete surface, such as- crack and spalling, which are the main reason behind the subsequent failure of the structures [Yamane and Chun (2020)]. Figure 1 shows the typical nature of crack and

spalling on concrete surface. With the increasing service time, these defects show a pattern of gradual increment both in number and quantitatively [Chao and Wenjun 2020].

Even though the periodical inspection is highly encouraged for infrastructure condition assessment, the conventional inspection system has some serious drawbacks. Firstly, due to variance in technical skills, expertise and subjective interpretation, the inspection output lacks the consistency [Phares et al., 2001]. Also, the aged infrastructures require more frequent inspections than the recently constructed ones. Moreover, these manual inspections are time-consuming, laborious, expensive and can be health hazardous in case of inaccessible parts of the infrastructures [Wells and Lovelace, 2018].

(a) Crack on concrete surface



(a) Spalling on concrete surface

Figure 1.1: Type os defects (Image Courtesy: TBT Engineering)

Infrastructure inspection has historically been depended on visual inspection method. In general, visual inspection refers to finding the visual changes on structures' surface showing the obvious warning of deterioration, and the visual assessment is considered as the initial form of evaluation in support of decision making in terms of safety, maintenance, and rehabilitation. Based on the collected visual information visual assessment involves a detailed inspection of the damaged area, including data collection, data processing and analysis, and post-event reconnaissance covering

the decision making process and documentation. Visual inspection and assessment are the primary requirement of a successful structural health monitoring (SHM) system. Researchers have immensely worked on creating various smart SHM modules to establish a resilient infrastructure system, as the stakeholders are more indulgent to finding an innovative, effective and economical ways to investigate the infrastructures [Agdas et al. (2015), Shamshirband et al. (2019), Zhang et al. (2017)]. With the introduction of computer-vision based analysis engineers are now capable to build a comprehensive solution for detecting and diagnosing the structural defects [Azimi et al. (2020)].

Recently, deep learning (DL) based techniques have discovered a potential solution with a promise to reduce the subjectivity yet increase the accuracy of the damage diagnoses and accessibility in SHM system. Also, DL algorithms are proved to be quiet effective in understanding the hard-to-describe features in high-dimensional data automatically without prior definition of the features. Moreover, DL approach helps overcoming the limitations of manual visual inspection. Among different types of DL methods convolutional neural network (CNN) is a type of neural network that works by processing data from an image. CNN has gained a high confidence in image classification and object recognition by automatically analyzing the different features from an image.

1.2 Research Objective and Contribution

Currently, DL approaches, more specifically CNN has proved to be exceedingly successful in image classification, segmentation, detection, recognition and many other process by using automatic feature extraction method. An in-depth review of prior studies (described in detail in Chapter-2) has showed the existing limitations, such as- only a few studies worked with multiple damage detection whereas multiple detection is essential to comprehend the real scenario of damage condition of any structure. Even though some studies have worked with different types of damages (Dunphy et. al. (2022), Ghosh Mondal et. al. (2020)), the image dataset is very limited for some case, i.e. spalling, rebar exposure. Also, most of the prior research lack multiple CNN model analysis and detailed sensitivity analysis of hyper-parameters, whereas a comprehensive comparison of the different CNN models' performance based on a variety of hyper-parameters can actually provide a good understanding on how a well-tuned model can help building an automatic DL based damage classification and segmentation solution. Considering the aforementioned

challenges, this study has outlined some specific improvements for all these challenges and evaluated the effectiveness and feasibility of the developed DL based visual assessment technique. In summary, this study has used CNN-based DL methods to classify the different defect types, and later on used segmentation process to recognize the specific defect areas. In defects classification, using the CNN algorithms the models learn the different patterns and features from the multiple defects image database and finally categorizes the different types, such as- concrete crack and concrete spalling. Segmentation is the process to localize the defects area on pixel level, and labels the exact area by outlining the different features from an image. This dissertation research aims to develop a comprehensive module for concrete defects analysis from visual inspection images, specifically including the following objectives-

i. Performing a literature review on prior contribution of CNN-based defects identification and segmentation in SHM system for civil infrastructures and understand the concept of DL transfer learning to frame a solution for CNN based defects detection. The review has helped understanding the specific difficulties and solutions to adopt for certain scenarios. Also, analyzing the prior studies have facilitated the idea of identifying the limitations of the current studies and explore the existing DL transfer learning techniques to use in this study's benefit to achieve the research scope.
ii. Building a large dataset of labeled images for two different types of defects representing the diversity of the defects' physical parameters and image architectures. For CNN models, the successful completion of pattern recognition and object detection highly depends on a comprehensive and diverse dataset.
iii. Avoiding augmented images for the developed damage detection and classification algorithm. Augmented images are avoided as the augmented dataset can give a false presentation of good performance with a specific dataset while in real-word application the models do not achieve a successful evaluation.
iv. Developing an automatic and user-friendly concrete damage detection (i.e. crack and spalling) AI model that can be utilized for any site condition.
v. Performing a detailed sensitivity analysis to identify optimized hyper-parameters for CNN-classifiers and segmentation models. This study aims to analyze different pre-built CNN models for defects classification and segmentation. Also, the hyper-parameters are tuned

during the training process to achieve an optimized CNN model. For sensitivity analysis different type of hyper-parameters are selected and implemented in all the CNN models to find the best tuned hyper-parameters values for defects classification and detection.

To achieve the aforementioned objectives, this study has collected defect images from various sources including actual industrial inspection reports (courtesy to TBT Engineering), web-based resources and pre-developed dataset by other researchers. This study has developed a crack dataset of 4087 images and spalling dataset of 1100 images without implementing any augmentation process. This dataset is developed to train and evaluate the CNN models for defects identification and segmentation and to the best of authors knowledge this dataset is the largest concrete defects dataset consisting the original images from inspection reports without applying any image augmentation process. As the site condition of concrete structure is not same everywhere, this study has focused on building a damage dataset of different concrete structures from different locations. This way a generic AI model can be standardized for any site condition to avoid the complexity of using condition-based solution. Also, the AI model developed in this study is trained on real site conditions targeting the generic data collection system to provide a user-friendly approach for engineers. Moreover, by introducing this automated AI solution for data categorization and segmentation professionals can reduce the resources required for site reconnaissance and focus on rehabilitation work more preciously. Also, for classification and segmentation purpose this study has worked with five different types of CNN models to make a performance comparison. Simultaneously, a substantial amount of sensitivity analysis is done for each CNN model with the significant hyper-parameters, like- optimization function, loss function, learning rate, batch size, and epoch. The final selection of the hyper-parameters values is done based on the optimized model performance. Another highlight of this research is the Encode-Decoder models- U-net and PSP-net used for segmentation process. For both U-net and PSP-net four different backbone CNN models are considered and to this author's knowledge no previous studies used these four backbone models for concrete crack and spalling detection. In short, this research has contributed by pushing some boundaries of the existing studies and developed a comprehensive defects assessment technique by addressing the real-world complexity and comparison of multiple CNN models in a feasible way.

1.3 Organization of Thesis

In this dissertation, a comprehensive vision-based defects assessment technique is introduced, including- categorizing the defects type and recognizing the exact area of the defects on the image through semantic segmentation enabling the successful implementation of this project in real-world problems. The first half of this dissertation has listed some of the previous works with CNN models in defects detection and illustrated the core techniques of the CNN models used in this study. The other half of the dissertation has portrayed the sensitivity analysis of the CNN models based on the hyper-parameters tuning and made some graphical representation of the final outputs. The details of the chapters are organized as follows-

Chapter 2 reviews the previous studies worked on artificial intelligence (AI) based SHM system for condition assessment of civil infrastructures. These reviews started from digital image processing techniques (IPTs) to various DL-based models.

Chapter 3 presents the overall methodology of how the DL-based defects condition assessment process works and the detailed techniques of the CNN-models considered in this study. The detailed process starts with data collection process to CNN layer organization, model selection and finally the feature extraction methods. This chapter also includes the various methods adapted for hyper-parameters tuning, model training and validation, model's optimization process, and performance evaluation metrics.

Chapter 4 exhibits the final performance output of the models for classification models by means of tabulation and graphical content. This chapter displays the accuracy of the models and the superiority of this research over previous works.

Chapter 5 discusses the basic methodology of semantic segmentation process and presents a summary of two types of encoder-decoder models used in this study. Also, this chapter illustrates the sensitivity analysis of hyperparameters for all the segmentation models and the final performance results of the encoder-decoder models.

Chapter 6 summarizes the research methodology and findings from the previous chapters and put on generalized conclusion. It also focuses on the advantages and limitations of this study while mentioning the future scope of this research.

Chapter 2 Literature Review

2.1 Introduction

Safety, reliability and uninterrupted performance are the universal concerns for any kind of in-service infrastructure. To highlight the construction materials, in today's world, concrete is one of the most regularly used materials for any construction work. However, under the extreme weathering effect, the serviceability of the concrete structures is subjected to disruption. Hence it is crucial to employ a systematic inspection system to ensure the perseverance of the structures, i.e., bridges, buildings, dams, high-rises and power stations. With the increasing number of ageing infrastructures globally, the significance of SHM systems to visually inspect the structures are prevalent. Conventionally the SHM system is highly dependent on on-site manual inspection, which can be expensive, laborious, time-consuming and inconsistent in the condition assessment decision-making process. To support the professionals in the structural diagnosis and advance the qualitative assessment, researchers have introduced computer-vision based technology in the SHM system. Moreover, some researchers have built a systematic review of the advancement of smart SHM systems in the application of structural performance assessment [Mohsen et. al. (2020), Han et. al. (2021) and Sandeep et. al. (2021)]. Chapter 2 reviews the challenges and advancements of some of these previous works on computer-vision based defects analysis, starting from image processing technology and following machine learning employed defects analysis, DL-based condition assessment, and finally highlighting the CNN-focused studies.

2.2 Related Works: Image Processing Techniques (IPTs)

Vision-based techniques in damage detection have been studied extensively for the past few decades. One of the early advancements of the vision-based method was image processing techniques (IPTs), which has shown credibility in crack detection on concrete and asphalt surfaces [Zenghui et. al. (2021)]. Some of the most widely known approaches to IPTs for defects detection are- the thresholding-based approach [Cheng et. al. (2003), Ying and Salari, (2010), Yusuke and Yoshihiko (2011), Takfumi et. al. (2012)], and morphological methods [Shivprakash and Sunil, 2006; Hoang-Nam et. al. (2014)]. The thresholding method converts an image into a binary image from a colour or grayscale image. In contrast, in morphological processing, a structuring element

is added to an image to outline the shapes of the objects. These methods are employed for concrete crack detection [Yusuke et. al. (2006) and Takfumi et. al. (2012)] and concrete spalling [Stephanie et. al. (2012)]. In summary, IPTs are an assumption-based approach that considers the contrast of the gray level of the image objects between crack and background. These assumptions are made on two concepts- first, the defects' area is thinner than the other texture patterns, and the image background has a lighter pixel shade than the area of the defects [Tomoyuki et. al. (2008)]. Therefore, the performance of IPTs is highly reliant on photo-shooting conditions avoiding the shading variance and noises. Later, researchers built some methods like Bayesian decision theory [Mohan and Poobal (2018) and Argyris et. al. (2018)], edge detection techniques [Hoang et. al. (2018)] and fuzzy C-means clustering [Ouma and Hahn (2017)] to detect cracks. In a study of crack detection, Ikhlas et al. (2003) developed an evaluation with four techniques, i.e., Fast Haar Transform (FHT), Fast Fourier Transform (FFT), Sobel and Canny, where FHT outperformed the other three techniques in crack identification. Even though assumption-based models are fast and reliable in crack detection, the performance is negatively affected by lightning conditions and background obstructions of the images [Michal and Boguslaw (2017)]. To address this limitation, Leonid et. al. (1992) adopted denoising techniques to remove the distortion and noises from the images to escalate the model's training performance. Yet the improvement of IPTs is limited as the inspection images are collected from extensively varying environmental conditions.

2.3 Related Works: Machine Learning in Vision-based Methods

Considering the generic site condition of structures, preparing a dataset with standard lightening conditions and without any disruption is quite impossible. Therefore, to further improve the damage detection methods, some researchers have worked on combining Machine learning (ML) approaches with IPTs. ML methods are a subsection of artificial intelligence (AI) which are widely used in defects detection for their prominent characteristics of both supervised and unsupervised learning of features and noise elimination, including techniques like feature engineering, classification and data clustering. Some of the previous studies included IPTs, ML method (i.e., Support vector machine) and neural networks to categorize the damage features from other features [Mohammad et al. (2013) and Young-Jin et al. (2016)]. Ikhlas et al. (2006) studied three special principal component analysis (PCA) methods while working on automatic crack extraction in concrete structures. First, they employed PCA to analyze the raw data, then enhanced the results

of linear feature detection and finally focused on the features detected on the specific block of the data. In another study, Gajanan and Sayan (2012) combined the fuzzy logic and artificial neural network (ANN) approach for crack extraction from digital images based on two principles (a) image approach: classifying the crack and crack-free images globally, (b) object approach: analyzing an image to locally classify the crack and background obstruction.

Over the past decades, ML has proved its potential in reliability and flexibility in feature detection applications through feature engineering, where geometrical properties and colour distributions are analyzed and calculated to identify a segmented area in images. Later, different ML algorithms are applied to categorize or cluster the specific feature index areas based on the pre-defined damage classes. Researchers used the k-nearest neighbours [(Jahanshahi et. al. (2013); Oliveira & Correia (2008)], support vector machine (SVM) [Fu-Chen et. al. (2017), Po-Han et.al. (2012), Michael et.al. (2014)] and ML algorithm to identify and detect the crack area. Based on a type of ML, SVM, Wei and Wang (2012) built a proximal support vector machine (PSVM) method for pavement surface damage detection. Later, Gang et. al. (2017) developed an automatic crack detection algorithm established on the base algorithm of linear SVM. This novel algorithm performed well with the capacity of noise elimination by applying a unique, wide-ranging search strategy.

Hyunjun et.al. (2018) proposed a two-step classification ML approach for crack and non-crack identification. First, they applied the image banalization technique to highlight the crack-containing zone. They later used a previously trained classification model to extract the crack and crack-free features separately. Another novel approach was achieved by Bao et. al. (2019), where they used the genetic optimized online sequential extreme learning machine algorithm. This method does not need to restart the training from scratch when a new dataset is uploaded. However, although the ML methods successfully attempted defect classification, the feature-based approaches have certain drawbacks as the performance degrades when applied in the complex situation for arbitrary defect categories. In addition, the models require reasonable training with domain knowledge and pre-defined feature indexes to achieve a better performance, which gets more complicated with a complex dataset.

2.4 Related Works: Deep Learning in Vision-based Methods

In recent times, DL based detection process has become profound for its automatic and optimized feature extraction potentiality [Wu et. al. (2021)]. Also, DL can analyze a large amount of data at

a time and multiple categorizations, which have facilitated the damage evaluation in the SHM system [Azimi et. al. (2020)]. In the context of SHM, DL techniques are implemented in the damage detection of infrastructure in three steps: classification, localization and segmentation. DL has several techniques, such as deep belief networks (DBNs), recurrent neural networks (RNNs), encoder-decoder model and convolutional neural networks (CNNs). Among all CNNs have shown efficacy for defects detection purpose [Zhao et. al. (2019)].

2.4.1 Pre-built base CNN models

Even though ML methods' performance was relatively decent, since 2012 introduction of CNN gave a robust solution for object recognition and classification [Liu et. al. (2020)]. A CNN-based DL method was developed to detect concrete cracks by Young-Jin and Wooram (2017) and Pan and Yang (2020) and other defects by Lin et. al. (2017). The CNN model has different algorithms based on the depth and width of the learning layers. Researchers continuously explore pre-built and own-developed CNN algorithms to develop an effective damage assessment model, such as- VGGNet [Simonyan and Andrew (2015)], ResNet [He et. al. (2016)] and Inception [Szegedy et. al. (2015)] models are used as damage pattern recognition and detection process. VGGNet- built-in 2014 by Visual Geometry Group (VGG) from Oxford University used 3x3 convolutional filters with 16 and 19 convolutional layers and achieved an outstanding performance without having too deep layers. However, VGGNet faced a vanishing gradient problem where the learning rate becomes smaller within the deeper layer, increasing the training time considerably. In 2015, researchers proposed a Residual Network (ResNet) to solve the vanishing gradient issue. ResNet uses the “skip connection” concept within the residual block, allowing the gradient an alternate shortcut without hindering the performance of the deeper layers. In 2014, in the ImageNet visual recognition competition, the Inception model put forward a breakthrough performance to solve recognition and detection problems [Sandeep et. al. (2021)]. The objective of the Inception model was to increase the depth and width of the model while avoiding overfitting and extensive computational cost. Table 2.1 presents a comprehensive summary of the previous studies that worked with base CNN models for infrastructure defects detection.

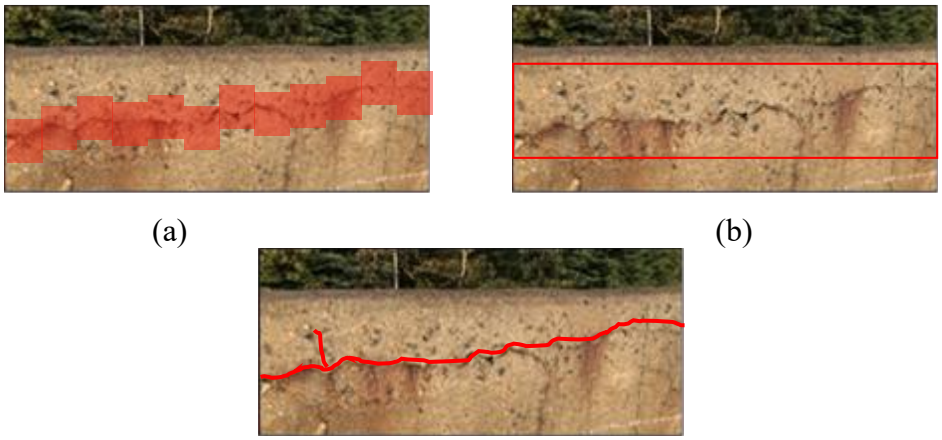
Table 2.1 A summary of previous studies on defects classification and detection with pre-built CNN models

References	Application	CNN architecture	Specifications
Yang et. al. (2021)	Concrete surface damage- crack	AlexNet VGGNet13 ResNet18	• 227x227 pixels • Data augmentation • 10,000 cracks and 10,000 crack-free images • Accuracy- AlexNet, 97.6%, VGGNet13, 98.3%, ResNet18, 98.8%,
Niannian et al. (2021)	Brick surface damage- crack, spalling, and efflorescence	AlexNet GoogLeNet	• 480x105 & 210x 105 pixels • 24000 images • Accuracy- GoogLeNet, 94%,
Dimitris et. al. (2021)	Masonry surface-Crack	VGGNet ResNet DenseNet Inception MobileNet	• 224x224 pixels • 351 cracks and 118 crack-free images • Accuracy- MobileNet 95.3%,
Yadong and Yicheng (2018)	Tunnel concrete surface damage	GoogLeNet VGGNet Faster R-CNN	• 256x256 pixels • 9520 images • Accuracy- 95%,
Dhananjay et al. (2019)	Building damage assessment	VGG16	• 224x224 pixels • 1200 images • Accuracy- 97.85%
Li et al. (2018)	Automatic pixel-level crack detection	GoogleNet	• 256x256 pixels • 1250 images • Accuacy 99.39%

A. Transfer learning of CNN in damage classification: Conventionally, the damage assessment process works in two steps: a) classify the types of damages using the feature extraction and b) object area recognition technique (also known as segmentation) to recongnize the area of the damages on the images. In the first step, CNN models identify the differences among the images and categorize the damages. After the categorization, CNN models do the quantitative analysis of the images to show the exact defect areas for the specific defect category. For example, Yang et. al. (2021) compared three neural networks, AlexNet, VGGNet13, and ResNet18, to identify the concrete cracks and no-crack surface, with images of 227x227 pixels. The models were trained with 10,000 cracks and 10,000 crack-free images and got an accuracy of 97.6%, 98.3%, and 98.8%, respectively. This study used the data augmentation process to create the 10000 crack images database from 2000 original images. In the study of Niannian et. al. (2021), AlexNet and GoogLeNet base models were used to identify the brick crack, spalling, and efflorescence. They used 1466, 1830, 865, and 984 original images of intact brick, crack, spalling and efflorescence, respectively. Both the models had an accuracy of over 90%. In another research, Dimitris et. al. (2021) compared five different types of base algorithms- VGGNet, ResNet, DenseNet, Inception and MobileNet to identify the crack images of masonry surfaces. The dataset contained 351 images of cracks and 118 without any cracks and had a resolution of 224x224 pixels. The implementation of transfer learning brought a significant boost in the model's performance, and with an accuracy of 95.3%, MobileNet outperformed all the other models.

B. Damage detection process with CNN: CNN–based damage detection can be categorized into three sections: Image patch method, boundary box regression and semantic segmentation [Mahtab et. al. (2020)]. Figure 2.1(a) shows the small patches used to identify the damage on the image. For the boundary box regression method, the model uses a box to bound the area of the damage [Figure 2.1b)]. These methods were successful in predicting the damaged area with promising accuracy. CNN method with the image patch technique was adopted by [Pan and Yang (2020), Chuncheng et. al. (2019), Yadong and Yicheng (2018)] to localize the damages on the images. Subsequently, region-based CNN models (R-CNN) were proposed, with simple modifications such as Faster R-CNN [Byunghyun and Soojin (2019), Young et. al. (2018), Yadong and Yicheng (2018), Jianghua et. al. (2019) and Tarutal et. al. (2020)] for multiple damage detection. Employing a Faster R-CNN, built on the basic architecture of ResNet-101, Niannian et al. (2019) developed

the efflorescence and spalling detection model for historic masonry buildings. They used a dataset of 500 images with 500x500 pixels splitting the dataset into 80-20% for training and testing. They concluded that despite having some minor errors, the developed model was able to obtain substantial success in predicting efflorescence and spalling. Nevertheless, the image patch method and R-CNN were incapable of extracting the exact geometry of the damages as it works by dividing the images into patches and boxes, respectively. To get a more quantitative assessment of damage characteristics, researchers worked with pixel-wise analysis using CNN-based semantic segmentation [Figure 1(c)]. Table 2.2 shows a summarized version of previous research on defects detection with the help of CNN models.



(a) (b)

(c)

Figure 2.1: Defect detection with (a) image patch classification, (b) boundary box regression and (c) semantic segmentation (Image Courtesy: TBT Engineering)

Table 2.2 A summary of previous studies on defects detection with pre-built CNN models

References	Application	CNN architecture	Specifications
Lei et. al. (2016)	Road surface damage- crack	ConvNet	• 99 × 99 pixels • Data augmentation • 500 images • F1-score 89.6%
Pan and Yang (2020)	Concrete building surface- post-disaster damage	AlexNet VGG19 ResNet50 YOLOv2	• 224 × 224 pixels • Precision- YOLOv2, 94%,
Chuncheng et. al. (2019)	Hydro-junction infrastructure- Crack, Spalling, Seepage, Rebar Expose	InceptionV3	• 300x300 pixels • 435 images • Accuracy- 96.8%,
Yadong and Yicheng (2018)	Tunnel concrete surface damage	GoogLeNet VGGNet Faster R-CNN	• 256x256 pixels • 9520 images • Accuracy- 95%,
Byunghyun and Soojin (2019)	Concrete surface- crack	Mask R-CNN	• 800x800 pixels • 453 images • Recall 76.5%
Young et. al. (2018)	Concrete and Steel surface- crack, corrosion	Faster R-CNN	• 500x375 pixels • 297 images • Accuracy 89.7%
Jianghua et. al. (2019)	Concrete surface- cracks	Faster R-CNN YOLOv2	• 500 × 375 pixels • 160 images • Accuracy- Faster R-CNN 79%
Tarutal et. al. (2020)	Concrete infrastructure- Post-disaster reconnaissance- crack and spalling	Faster R-CNN Inception-ResNet-v2	• 1024 x 1024 pixels • Overall Accuracy- Proposed model 80%
Niannian et. al. (2019)	Historic masonry building- Efflorescence and spalling	Faster R-CNN	• 500x500 pixels • 500 images • Average Precision 90%

2.4.2 Proposed CNN Models for Classification and Segmentation

A. Proposed CNN-classifiers using backbone CNN models: Using the MobileNet as the backbone model, Perez and Tah (2021) used SSDNet to develop a mobile device application to detect concrete damages, such as mold, crack, stains and paint deterioration. For training and testing purposes, 700 and 176 images were used, respectively. Finally, the accuracy for detecting crack, deterioration, mold and stain was 61%, 83%, 81% and 95%, respectively. Another study by Majdi et. al. (2020) built a distinctive CNN-classifier with three convolutional layers, ReLu (Rectified Linear Unit) activation function, and a softmax layer. This model used an available online dataset of crack images with a resolution of 227x227 pixels. The proposed classifier achieved a high accuracy of 99.57%. However, the dataset used in this study was highly modified without any noise, which does not replicate the real-time inspection condition. To achieve a quantitative assessment of multiple seismic damages of reinforced concrete (RC) structures, such as cracks, spalling and crushing, reinforcement exposure, buckling and fracture, Zenghui et. al. (2021) built two unique CNN models. First, the Crack-Net was developed to detect cracks, and finally, 4Category-Net was optimized to detect the other four damage categories. These models performed satisfactorily with a mean IoU of 70% (Crack-Net) and 71% (4Category-Net).

An improved crack detection and recognition process with Batch Normalized (BN) technique was adopted by Chen et al. (2019) to detect cracks in historic masonry buildings. They used a dataset of crack and no-crack with the RGB pixels of 227x227 and gained an average accuracy of 99.71%.

B. Proposed CNN models for segmentation: To upgrade the damage evaluation process, researchers have worked on developing their own CNN model while using the pre-built base models as the backbone of the structure. In recent years, DeepCrack [Zou et. al. (2019)], SDDNet [Choi and Cha (2020)], STRNet [Dong and Young-Jin (2021)], and U-Net [Ronneberger et. al. (2015)] have gained popularity over the base models with their robust performance in crack detection. The studies above found that these models were successful in crack detection with a performance evaluation matric IoU value of 0.88, 0,846, 0.926 and 0.923 for DeepCrack, SDDNet, STRNet and U-Net, respectively. Also, Fully Convolutional Network (FCN) DeepLab and YOLO have shown good performance in defect identification and detection. For crack detection, Anai et al. (2019) used publicly available pre-built models, such as R-CNN, Shot Multibox Detector (SSD), YOLO, and RetinaNet, with different numbers of layers. The models were trained with a

dataset of 230 images, and validation was done with 208 images. After the training at the testing stage with randomly split images, YOLOv3 showed a mean average precision of 91.1%.

A detailed review of CNN-based semantic segmentation with various application techniques was presented in the study by Alberto et. al. (2017). Liu et. al. (2020) proposed U-Net architecture to build an autonomous pavement crack segmentation model. ResNet-34 was applied as the backbone base model, pre-trained by ImageNet parameters. The proposed segmentation model detected cracks with the precision, recall and F1-score 97.24%, 94.31%, and 95.75%, respectively. Next, DeepLab3+ CNN model was applied by Jia-ji et. al. (2022) for semantic segmentation of cracks. This proposed model used ResNet as the base model to compare the crack detection performance, and the DeepLab3+ with the backbone ResNet101 outperformed the others. The dataset was collected from previous research resources, consisting of crack images of the building, pavement, and concrete structure. Finally, a dataset of 2446 images was developed with a resolution of 512x512 pixels, and the whole dataset was divided into 1827 and 619 images for training and validation purposes. The best-performed model had the highest IoU, recall, and F1-score of 0.6298, 0.6834, and 0.7732, respectively.

In another semantic segmentation of defects, Pozzer et al. (2020) used two different types of images, regular and thermographic, to compare the performance of different CNN model architectures, VGG-16, ResNet, and MobileNetV2. The dataset included images of delamination, crack, spalling, and patches. The authors achieved the best performance from MobileNetV2, detecting the defects with 79.7% accuracy.

C. FCN-based semantic segmentation: Recently, FCN-based CNN algorithms have been extensively analyzed for semantic segmentation [Li et. al. (2019)]. FCNs are implemented as an end-to-end pixel-level process where the outcome is semantic segmentation instead of damage categorization. This study considered four types of concrete defects: crack, spall, efflorescence and hole, and achieved a mean pixel accuracy of 91.59%. In the end, they also compared their own-developed FCN model's performance with the SegNet-based method and concluded that their proposed method performed better in realistic conditions. Table 2.3 summarises proposed CNN models used in different research for classification and segmentation, respectively.

Table 2.3 A summary of previous studies on defects classification and segmentation with proposed CNN models

References	Application	CNN architecture	Specifications
Perez and Tah (2021)	Concrete surface damage- mould, crack, stains and paint deterioration	SSDNet	• 224 x 224 pixels • 1890 images • Overall accuracy- Crack 61%, deterioration 83%, mould 81% and stain 95%.
Majdi et. al. (2020)	Concrete surface- crack	Distinctive CNN-classifier	• 227x227 pixels • 458 images • Accuracy- 99.57%
Zenghui et. al. (2021)	Concrete surface- crack, spalling, rebar exposure, fracture.	Crack-Net 4Category-Net	• 300x300 pixels • 76 images • Overall IoU- Crack-Net 70.11% 4Category-Net 71.12%
Xiao-Wei et. al. (2019)	Concrete surface damage- crack	Ci-Net	• 500 x 300 pixels • 762 images • Overall precision 84%
Yang et. al. (2019)	Seismic damage detection of reinforced concrete columns	Faster R-CNN	• 640 × 640 pixels • Data augmentation • Overall precision 80%
Zou et. al. (2019)	Crack detection	DeepCrack	• 512 × 512 pixels • Backbone model- SegNet • Pre-built dataset • Overall precision 85-95%
Choi and Cha (2020)	Concrete surface- cracks	SDDNet	• 513× 513 pixels • Backbone model- DenSep • 44 imges • IoU- 84.6%, F1 score 81.9%

Dong and Young-Jin (2021)	Crack segmentation	STRNet	• 512 × 512 pixels • 1784 images • Precision- 91.7%,
Fangzheng et. al. (2015)	Structural crack segmentation	U-Net	• 386×256 pixels • 101 images • Accuracy- 90.02%,
Anai et al. (2019)	Concrete surface-deterioration	R-CNN, Shot Multibox Detector (SSD), YOLOv3, RetinaNet	• 800x800 pixels • 438 images • Average Precision 91.1%
Liu et. al. (2020)	Concrete and Steel surface- crack, corrosion	YOLO U-net	• 320x320 pixels • Backbone model- ResNet • 297 images • Segmentation 90.5%
Jia-ji et. al. (2022)	Concrete surface-cracks	DeepLab3+	• 512 × 512 pixels • Backbone model- ResNet • 2446 images • IoU- 62.98%
Li et. al. (2019)	Concrete infrastructure-crack, spalling, efflorescence, hole	FCN	• 504 x 376 pixels • 2750 images • Overall Accuracy- 91.59%

2.5 Summary

This chapter provides a detailed summary of past research on different types of defects classification and detection in the context of database size, image quality and image processing techniques. Furthermore, this section provides an insight into previous applications' achievements and scopes of future works indicating the existing demand for automatic SHM techniques. Comparing to other techniques deep learning methods have provided a promising result in SHM applications with automated defects pattern recognition.

In terms of deep learning algorithms CNN proves to have the most success in damage pattern recognition in computer vision. This literature review contains an in-depth discussion of structural

condition assessment by making a list for both the classification CNN models and the segmentation CNN models and identifies the behaviour of the different CNN models in context of damage classification and detection for different types of structures. Finally, this review has summarized three concepts- first, transfer learning can facilitate the idea of developing a sophisticated CNN algorithm to build a successful defects detection model, especially the deeper architectures are able to recognize the patterns more accurately. Also, having a sufficiently large and complex image dataset can aid in build a better performing and efficient CNN model capable enough to solving complex problem in real-time condition. Finally, multiple damage detection with CNN is a considerably possible approach which is not yet explored extensively.

Based on the findings of this chapter this study has decided on exploring multiple damage detection with CNN models by developing an image dataset of concrete crack and spalling replicating the real-world structural conditions. Also, this study has implemented multiple CNN algorithms to compare the damage detection accuracy and outline the best output. Overall, this review work of previous studies has helped this research to identify the unexplored segments of CNN models and built a comparative study for defects assessment to understand how new approaches can assist in achieving a better prediction quality.

Chapter 3 Deep Learning Methodologies

3.1 Introduction

Visual damage detection of infrastructure using deep learning (DL) based computational approaches has developed an interdisciplinary area of interest for engineers. DL has the potential to process a large scale of complex data which can be engaged in civil infrastructure health monitoring system. As DL is a very complex method it can be very challenging to train a DL model and to overcome some common issues, i.e. overfitting, vanishing/exploding gradient. To build a fine tuned model it is very important to choose the appropriate components and understand the basic work principles of those components. This chapter explains how a DL model extracts features, is trained and tuned to reach an optimized position. Furthermore, this chapter contains the description of the specific modules used in this study for defects classification and segmentation process.

3.2 Overview of Deep Learning

DL has the potential for automatic detection and identification of patterns and features from a dataset. DL operations combine deep and hierarchical layers working toward feature extraction and identification [Bengio et. al. (2017)]. One of the prominent approaches of DL is CNN, works with the base learning layers called convolutional layers and supports the automatic image analysis processing. CNN was first introduced in a computer vision competition in 2012 called ImageNet Large Scale Visual Recognition Competition (ILSVRC) and continuously had a dominant implementation in various fields for its scalable approach to object recognition tasks and image classification [Yamashita et. al. (2018)]. A schematic diagram of basic CNN architecture is presented in Figure 3.1.

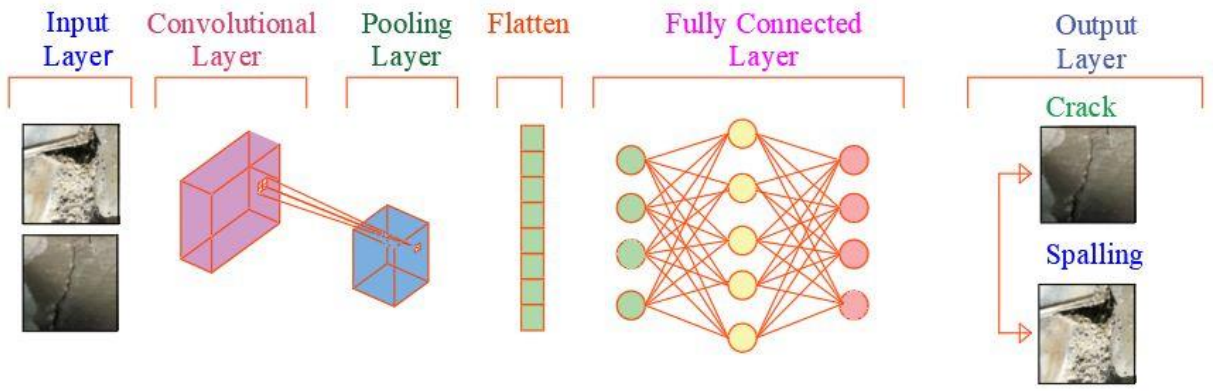


Figure 3.1: schematic diagram of basic CNN architecture

As illustrated in Figure 3.1, a CNN model starts with an input layer, followed by a group of learnable layers (convolutional layer, pooling layer, activation layer, fully connected layers etc.) and finally provides the decision in the output layer. These learnable layers are responsible for feature extraction from images where the previous layer's output acts as the next layer's input. DL is an automatic learning process; however, to optimize the learning process and cut the computational cost, several learnable parameters, also independently defined as hyperparameters, are selected and fed into the models. The hyperparameters are initialized randomly, and with the help of the optimization process, each layer tunes these parameters and accumulates the final result, which is referred to as model training. The tuning of hyperparameters is set empirically by imposing a trial and error process. In CNN model training, a training dataset is fed into the model for training purposes. A part of this training dataset is used as a validation dataset to validate an unbiased evaluation of the training data against unseen data while tuning the model's hyperparameters. Once the training and validation are done, hyperparameters are set, and satisfactory performance is achieved, the model is considered trained. A trained model is then tested with a set of images that were not introduced beforehand to the model. This test dataset aids in evaluating the trained model.

3.3 Layer operations of CNN

Generally, a CNN model consists of (a) Input Layer, (b) hidden layer- convolutional layer, pooling layer, fully-connected layer, activation layer, and (c) output layer. This section presents the fundamental operation process of the layers in CNN models. The CNN layer's learning process works by forward-pass and backpropagation in search of best optimized output.

3.2.1 Input Layer

In this study for the CNN models, the input data accumulates a set of images of concrete defects, including crack and spalling. In this case, colored images with red, green, and blue (RGB) channels are used. A digital image with RGB channels is considered to have three dimensional (3D) tensor (i.e., height, h x width, d x channel, c) [Figure 3.2].

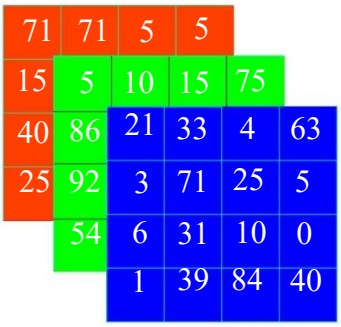


Figure 3.2: Data input style (RGB channel)

RGB channel is an accumulation of different pixel values representing the pixel's brightness, and for this study, the pixel number ranges between 0 to 224. Another significant image processing is called image normalization. A high-resolution image is converted into a small resolution image to reduce the pixel size, ultimately easing the computational cost of model training. Also, a large image with more background can adversely affect the model's performance. Therefore, cutting the original images into small ones is recommended for better performance. According to Flah et. al. (2020) a model trained with relatively small pixels can learn the desired feature more accurately than the original image but not vice-versa.

3.2.2 Convolutional Layer

Ian et. al. (2016) introduced a simple convolutional operation method [Figure 3.3]. In general, convolution is a linear function where multiplication of a set of weights is multiplied with the input tensors. This set of weights is called filter/kernel and has a two-dimensional array. The filter is a smaller tensor in size than the input array, and the output result of the multiplication of the input array and filet array is a dot product reflecting a single value. Within the forwarding pass training period, the filters keep multiplying and learning the features of the local dependents on their own. Notably, the higher the number of filters in a network, the more features are extracted from the

image, and the better chance for the network to excel in recognizing diverse patterns. To simplify, the deeper the CNN network goes, the better the performance. The following mathematical term can represent the convolutional process for a single image-

$$Y_{ij} = \sum_{(m,n)\epsilon f} \omega_{m,n} X + b_{ij} \quad (1)$$

Here, *X* and *Y* represent the input and the output values, respectively. f is a filter used for training. w and b are the weight and bias, continuously updated with the ongoing learning process. On the forward training process, the filters move along the width and height of the input image. The steps followed by the filters to convolve around the image is called stride. With each layer input image losing much information, padding is considered to keep the volume same for the input and output image.

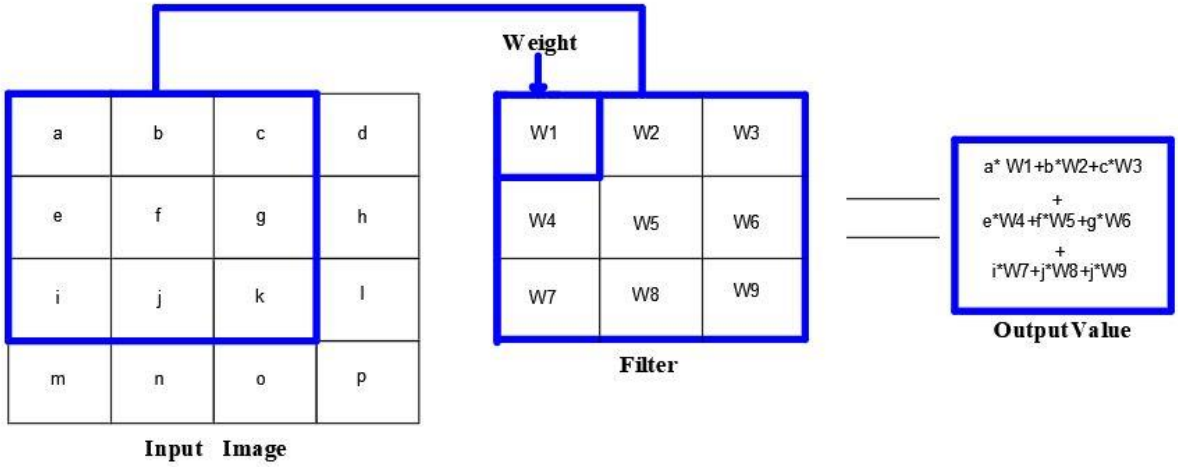


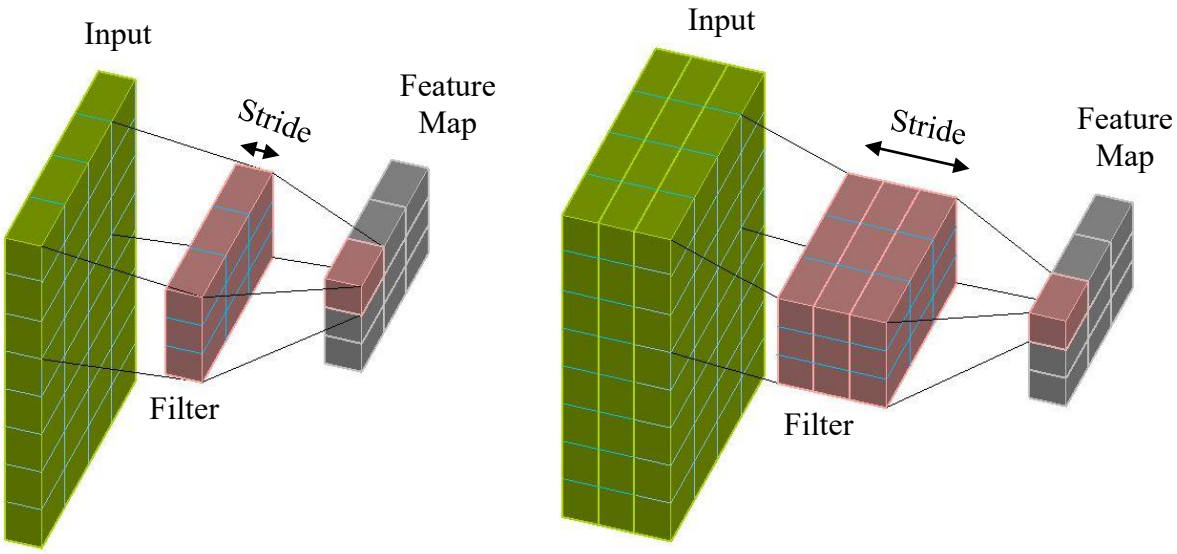
Figure 3.3: Convolutional operation method with 1-D tensor

In mathematical explanation, if for an actual conv layer (*L*), *f* is the filter size, *p* is the padding, *s* is the stride and n_f is the number of filters. The input image size is (m^{L-1}, n^{L-1}, c^{L-1}) where *m, n, c* represents the height, width, and channel of the image and the output layer will be a two-dimensional matrix. Following Eqn. 2 and Eqn. 3 show the mathematical formulation for the size of the output layer-

$$m^L = \frac{m^{L-1} + 2 * p^L - f^L}{s^L} + 1 \quad (2)$$

$$n^{L} = \frac{n^{L-1}+2*p^{L}-f^{L}}{s^{L}} + 1 \quad (3)$$

In the convolutional layer, stride function can be for both input and filter multidimensional. Figure 3.4(a) shows an example of the multiplication of 2-D stride input with a spatial dimension of 8x4 and a 2-D filter array with a spatial dimension of 3x3. The filter moves along the column by column and row by row on the input array and provides a weighted sum, and the output is a 2D array with a spatial dimension 3x3. Following a similar process, an input with three channels works with a three-channel filter, resulting in a 2D array with one channel [Figure 3.4(b)].



(a) 1-D channel (b) 3-D Channel

Figure 3.4: Convolution example- (a) 1-D channel and (b) 3-D channel

3.2.3 Pooling Layer

The Pooling layer operates by summarizing the features of a targeted region in an image by using a pooling filter (selecting the active array of each region) [Ciresan et.al. (2012), Zeiler and Fergus (2013)]. It helps the model reduce the size of the feature maps, which eventually makes the model computation faster by reducing the number of training parameters. After the pooling layer, the dimension of the output layer can be determined using the Eqn. 4 and Eqn. 5.

$$m^{output} = \frac{m-f}{s} + 1 \quad (4)$$

$$n^{output} = \frac{n-f}{s} + 1 \quad (5)$$

Where (m^{output}, n^{output}) represents (height, width) of the output dimension, *s* and *f* is the size of the stride and filter.

In practice, maximum and average pooling are the most employed pooling methods [Song et. al. (2018)]. Average pooling considers all the elements in the pooling region to avoid variance increase [Song et. al. (2018)]. On the other hand, maximum pooling only grabs the foreground element from the featured map as the representative feature for the next layer [Ciresan et.al. (2012)]. *Figure 3.5* shows the example of the maximum and average pooling process.

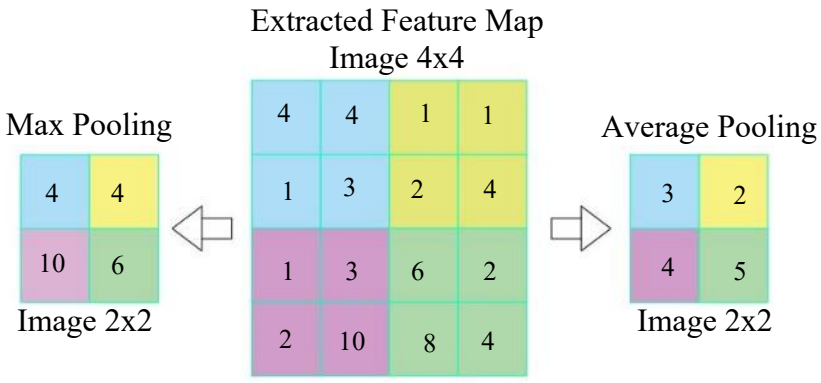


Figure 3.5: Example of max and average pooling

3.2.4 Activation Layer

In DL, Convolution and pooling are solely responsible for linear operations of the model and proceed the feature map with linear expression. However, the feature extraction in DL is a nonlinear process and requires the application of nonlinear functions called the activation function. In a neural network, the activation layer uses an activation function (nonlinear) to navigate how the weighted sum of the input transforms from nodes to output. In this study, ReLu (Rectified Linear Activation) function [Figure 3.6] is used for all the CNN models, showed in Eqn. 6. ReLu is a linear function that gives output only if the input is positive otherwise the output is zero

meaning the neuron is deactivated. This gives advantages on computational efficiency as not all the neurons are activated at once instance.

$$f(x) = \max(0, x) \tag{6}$$

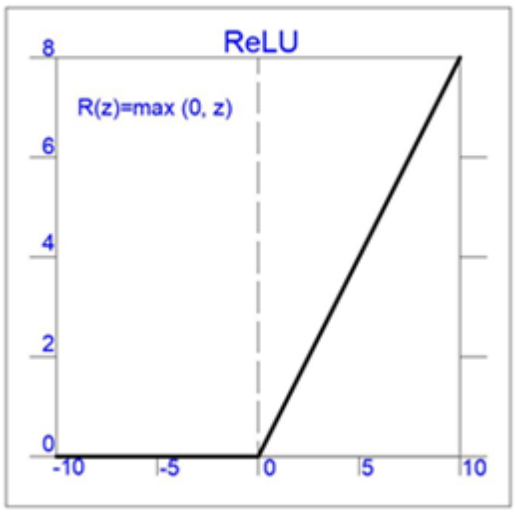


Figure 3.6: Activation Function (ReLu)

3.2.5 Fully Connected (FC) Layer

Fully connected layers are one of the essential components of the CNN model because of their successful application for classification and recognition in vision-based analysis. In practice, this layer works as decision making layer to predict the best label for the image. In numerical terms, all the input neurons from the previous layer (Eqn. 7) are connected with the corresponding output layer (Eqn. 8) through the FC layer.

$$\text{Input layer, } \psi^{(l-1)} = \{\psi_j^{(l-1)} \mathrm{I}\, j \in \{1,2, \dots \dots \dots, n_{l-1}\}\} \tag{7}$$

$$\text{Output layer, } \psi^{(l)} = \{\psi_i^{(l)} \mathrm{I}\, i \in \{1,2, \dots \dots \dots, n_l\}\} \tag{8}$$

Figure 3.7 shows how each input neuron goes through the learning process using weights and biases to give a weighted sum as output.

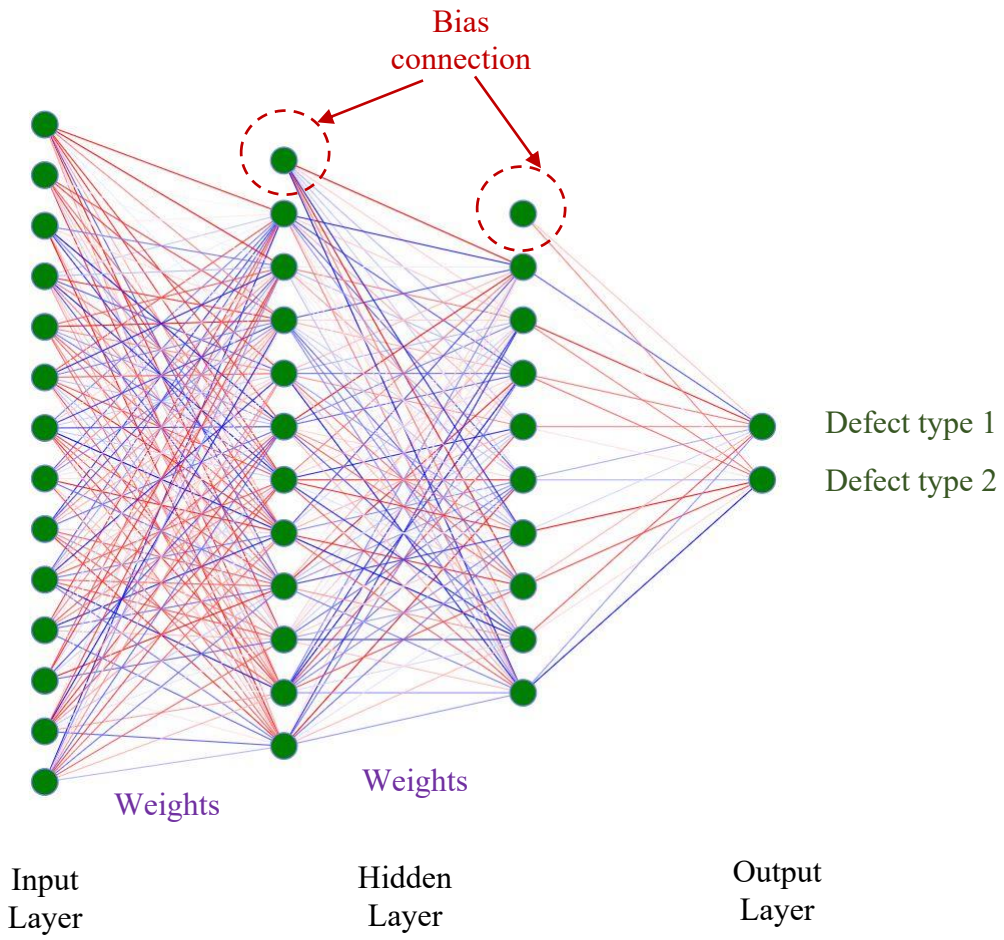


Figure 3.7: Example of a fully connected layer

3.3 CNN Model Optimization

The previous section discusses the operations of CNN layers and the parameters are tuned over numerous forward and backward pass actions to achieve a meaningful result. Forward propagation is the calculation and accumulation of intermediate variables of CNN layers from the input layer to the output layer. In backpropagation, the variable parameters are penalized by calculating the gradient of the network in reverse order from the output layer to the input layer. A complete forward-pass and backpropagation are referred to as iterations. With each iteration, the trained model updates its' parameters (i.e. filters, weights, biases) by minimizing the loss and the process

is defined as model optimization. For model optimization, two functions are required- the loss function and the optimization function.

3.3.1 Loss Functions

To update the model variables, it is crucial to calculate the derivation of the ground truth and the prediction value, and the function that calculates the derivation is referred to as the loss function. In this study, a combination of focal loss and dice loss is considered for the classification binary cross-entropy loss function and segmentation.

Binary cross-entropy (BCE) is a cross-entropy function used to choose between two choices (i.e. concrete crack and spalling). This loss function is usually considered to achieve prediction by the sigmoid activation function. Cross-entropy (CE) is a pixel-wise loss function and has performed prominently for various object detection applications [Ju et. al. (2020)]. Also, using this loss function in the CNN model gives the model the highest compatibility to be employed in the new dataset. Eqn. 9 presents the mathematical approach to how the binary cross-entropy loss function (L_{BCE}) calculates the average loss, where y_j is the scaler value of output, y_i is the corresponding target value, and n is the output size.

$$\text{Loss} = -\frac{1}{n}\sum_{i=1}^{n} y_i * \log y_j + (1 - y_i) * \log(1 - y_j) \tag{9}$$

Focal loss is an updated version of binary cross-entropy. This loss function solves class imbalance issues by down-weighting the easy example learning while focusing more on harder example [Lin et. a. (2017)]. The focal loss function can be written by rearranging the components of the cross-entropy function [Eqn. 10, Eqn. 11 and Eqn. 12].

$$\text{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{if } y = 0 \end{cases} \tag{10}$$

$$\text{Ground truth probability } (p_{gt}) = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{if } y = 0 \end{cases} \tag{11}$$

$$L_{BCE}(p, y) = \text{CE}(p_{gt}) = -\log(p_{gt}) \tag{12}$$

Finally, the focal loss (L_F) can be presented with a modulation factor consisting of parameters α and λ shown in Eqn. 13. α and λ represent the class weights and degree of down-weighting scale.

$$L_F(p_{gt}) = \alpha (1 - p_{gt})^{\lambda} * L_{BCE}(p, y) \quad (13)$$

Dice loss (L_{DSC}) is one of the most commonly used loss functions for segmentation. Dice loss comes from an evaluation metric, Dice similarity coefficient (DSC), applied in Boolean data, which is also known as the Sørensen-Dice index [Michael et. al. (2021)]. DSC is defined with the terms true positive (TP), false positive(FP) and false negative (FN) prediction [Eqn. 14 and Eqn. 15].

$$DSC = \frac{2*TP}{2*TP+FP+FN} \quad (14)$$

$$L_{DSC} = 1 - DSC \quad (15)$$

Dice loss function is also adapted to manage class imbalance. In this study, for segmentation, a total loss is calculated with a summation of focal loss and dice loss.

3.3.2 Optimizers

The optimization technique in a neural network works as finding the minimum or maximum output of the function depending on the input parameters or arguments. While updating the variable parameters through the forward pass and backpropagation process, the model emphasizes minimizing the loss function and optimizing the model accuracy. The loss function guides the optimizers by quantifying the difference between the expected result and the predicted result of the model. For classification CNN, two optimizers are used- Stochastic Gradient Descent (SGD) and Root Mean Square Propagation (RMSprop). The segmentation models are analyzed using SGD and Adaptive Moment Estimation (ADAM) optimizers.

SGD is a type of gradient descent process that is linked with a random probability. SGD takes a single random data to update its parameters for each iteration. To summarize, SGD uses a single sample to find the gradient of the loss function rather than using the summation of the loss function of all the samples. As, SGD works with each image individually, the overall computational cost of the model analysis increases. However, a greater performance is easily achieved by SGD as it processes each data separately to reach the local minima.

To the DL researchers, RMSprop is one of the most popular optimizers. RMSprop has a unique feature which restrains swaying in the vertical direction when helping the learning rate to learn

faster in the horizontal direction, making the convergence faster. In mathematical terms, RMSprop takes an exponential average of the gradients to demise the learning rate instead of the cumulative sum of squared gradients [Eqn.16]. Weights are calculated using the formula presented in Eqn. 17.

$$V_t = \gamma\, V_{t-1} + (1-\gamma) * G_t^2 \quad (16)$$

$$W_t = -[\eta/(\sqrt{(V_t + \epsilon)})] * G_t \quad (17)$$

Here, η = learning rate

V_t = Exponential average of squares of gradients

G_t = Gradient at time t

γ = Forgetting factor

ADAM is a gradient descent optimizer that combines two types of gradient descent- Momentum and RMSProp algorithm, which is very efficient and requires very little memory [Diederik and Jimmy (2015)]. Momentum helps speed up the gradient descent of the network by taking the exponentially weighted average. RMSP also accelerates the optimization process by reducing the number of function evaluations. For the ADAM optimizer, the numerical expression is done by taking the average of weights squared gradients and dividing by the square root of the mean square [Eqn. 18, Eqn. 19 and Eqn. 20].

$$V_t = \beta_1\ V_{t-1} + (1-\beta_1) * G_t \quad (18)$$

$$S_t = \beta_2\ S_{t-1} + (1-\beta_2) * G_t^2 \quad (19)$$

$$W_t = -[\eta/(\sqrt{(S_t + \epsilon)})] * G_t \quad (20)$$

V_t = Exponential average of gradients

S_t = Exponential average of squares of gradients

β_1, β_2 = Hyper-parameters

3.3.3 Batch Normalization

According to the developers [Sergey and Christian (2015)], Batch Normalization (BN) can help a neural network model improve the computational speed and performance while constructing a

reliable model. BN standardizes and normalizes the operations of the input layer from the previous layer. Also with BN, the internal covariate shift reduces. Covariate shift implies the interruption in input distribution for the learning process. Two parameters are added to each layer by BN, and the output is multiplied by a parameter for standard deviation purposes. By modifying the two weights in the activation layer, BN helps stabilize the network performance rather than changing the weights of the entire network. Eqn. 21 and Eqn. 22 represent features' mean and variance over a mini-batch.

$$\text{mean } (\mu) = \frac{1}{m} \sum_{i=1}^{m} \psi[i] \tag{21}$$

$$\text{variance} = \frac{1}{m} \sum_{i=1}^{m} (\psi[i] - \mu) \tag{22}$$

Here, ψ and m define the mini batch and size of the mini batch.

3.4 Transfer Learning

In vision-based DL process, deep neural networks learn the features from the dataset by tuning a group of parameters and later on transferring these attributes to solve novel tasks. This phenomenon of transferring the learned data to a new model is referred to as transfer learning [Rich (1995), Yoshua (2012)]. In practical use, transfer learning uses the pre-learned elements from a trained model to initialize the training process of a new DL model. This can be considered as a less resource-intensive approach as the new models do not have to start training from scratch. To consider the pre-trained models for new tasks usually, the original model should have a certain amount of better generalization adaptability to perform satisfactorily with new unseen data [Dai et. al. (2007)]. In general, a novel CNN model requires analyzing a large amount of data resulting in training a few million parameters. However, these training parameters can reduce sizes by implementing a transfer learning process.

Transfer learning has two different methods- (a) feature extraction and (b) fine-tuning. In the feature extraction method, some existing pre-build CNN models are directly used to learn the features and patterns of the input images. In the second method, a group of pre-trained hyper-parameters are considered for training the parameters of the new model. There are many existing CNN models; however, for this study, a few pre-trained models are considered (i.e. VGG-19,

ResNet50, InceptionV3, MobileNetV2, Xception, EfficientB3). To train the CNN models, the pre-trained weights, “ImageNet” (1.2 million images with 1000 categories), was examined to achieve the efficiency in classifying the type of defect- crack and spalling. For instance, ImageNet has paved the way toward pre-trained generic features for transfer learning [Jia et. al. (2009)]. The hyper-parameters from ImageNet have acceptable interpretations and usually require minimal tuning. The architecture of the proposed methods is discussed in the following subchapters.

3.4.1 VGG-19

In 2015, Simonyan and Andrew (2015) proposed the VGG-16 and VGG-19 models and analyzed the effect of the depth of the CNN model for the classification purpose. For this study, VGG-19 is considered as one of the backbone models, and Figure 3.8 illustrates the architecture of the VGG-19. VGG-19 consists of 19 layers with convolutional layers, pooling layers, fully connected layers, and softmax layer. The images are passed down these layers by applying a fixed filter size (3x3), and the max-pooling function is used at pooling layers. There are two distinctive characteristics of the VGG network- (a) the filter size remains the same for all the feature map sizes, and (b) using the max-pooling function, the feature map size is reduced to half, and with that number of filter gets doubled. According to the researchers, VGG-19 had 19.6 billion floating-point operations (FLOPS) and 144 million parameters, and after training, the model improved remarkably in the CNN-based image classification task.

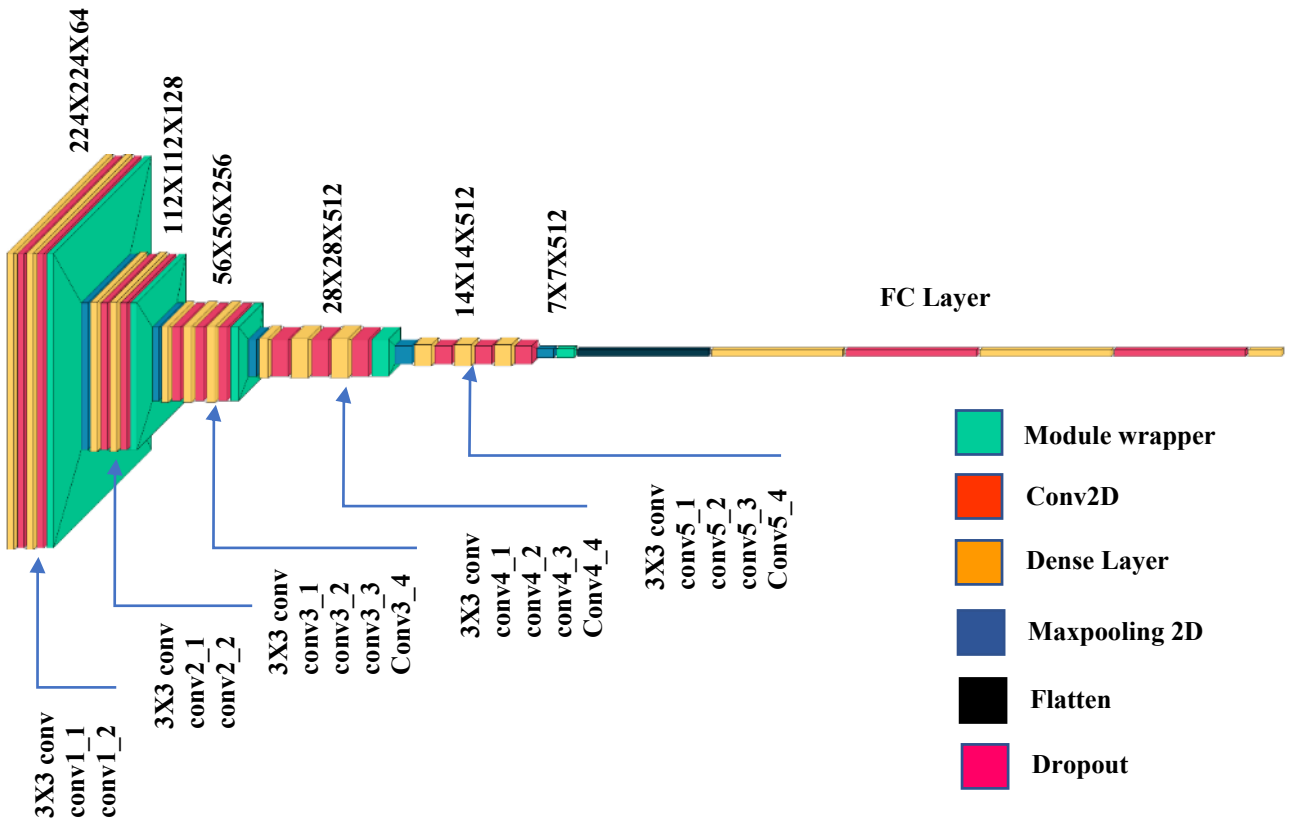


Figure 3.8: CNN base model architecture- VGG-19

3.4.2 ResNet-50

ResNet was first introduced by He et. al. (2016) where they described a residual learning algorithm with the advantage of going deeper without encountering performance degradation. ResNet was also proved effective in solving the problem with vanishing gradient descent by decreasing the error within the deeper layer. In each layer of the convolutional layer, a residual learning block was added, which worked as a "skip connection". Figure 3.9 shows the building block for the ResNet, where the formulation works as shown in Eqn. 23 and Eqn. 24.

$$y = F(x, \{w_i\}) + x \tag{23}$$

$$F = w_2 * б(w_1 x) \tag{24}$$

From the formulation, *x* and *y* represent the input and output function, and *F* denotes the residual mapping to be trained. w_1 and w_2 are the weights for the subsequent convolutional layers, and б is

the activation function. This research adopts ResNet50 for the classification and semantic segmentation process.

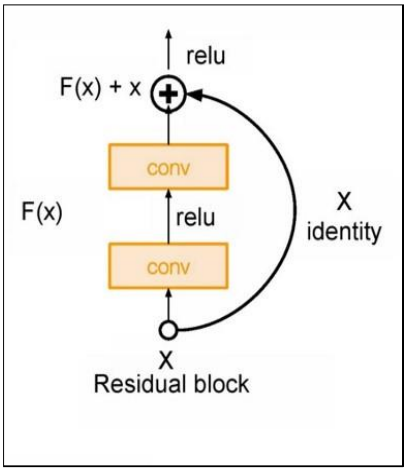


Figure 3.9: CNN base model architecture- ResNet

3.4.3 EfficientNet

In 2019, Tan and Quoc (2019) reviewed the performance of CNN based on the correlation of width and height of CNN models. They found an efficient CNN architecture with a limited parameter that achieved higher accuracy. The authors named the model EfficientNet and proposed a novel model scaling method for scaling up CNN models with an effective compound coefficient. Unlike the conventional models that scale up the dimensions of the networks with width, height, and resolution, EfficientNet uses a fixed set of scaling coefficients to scale up the dimensions uniformly. Generally, the model's performance improves with an individual dimension scale. However, by balancing all the network dimensions, the model's whole performance can be improved highly. Figure 3.10 illustrates the compound scaling architecture of EfficientNet.

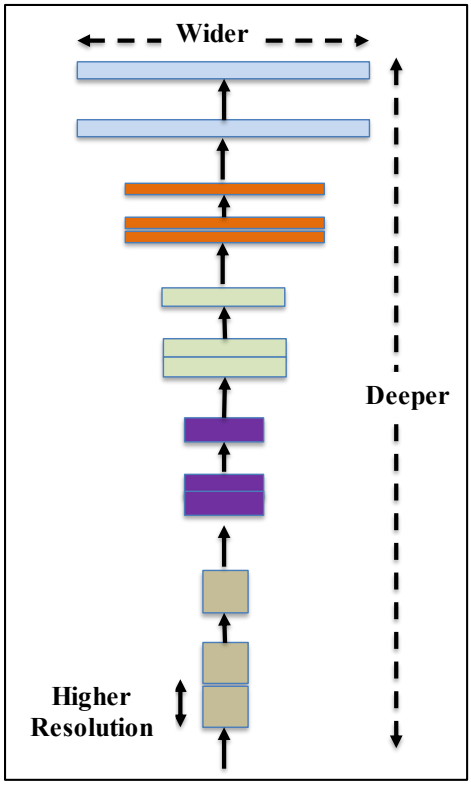


Figure 3.10: CNN base model architecture- EfficientNetB3

3.4.4 Inception

The inception model was first introduced by Szegedy et. al. (2015) and put forward a remarkable performance on the ImageNet Visual Recognition Challenge (2014). This model was once regarded as the state-of-the-art deep learning model for its' noteworthy performance in image recognition and detection. The main objective of this model is to connect the model sparsely, replacing the fully connected networks of the convolutional layers. The sparsely connected network is the core concept of the inception layer. The inception layer has its convolutional layers (conv) and pooling layer, including a 1x1 conv layer, 3x3 conv layer, 5x5 conv layer, and 3x max-pooling layer. The output of these layers is converted into a single output vector and concatenated into the next stage as input. Figure 3.11 shows the work method of the inception layer in the Inception CNN model. One of the major advantages of the Inception model is that the model's computational cost is maintained while increasing the height and depth of the network. For the damage assessment purpose, InceptionV3 is considered for this study.

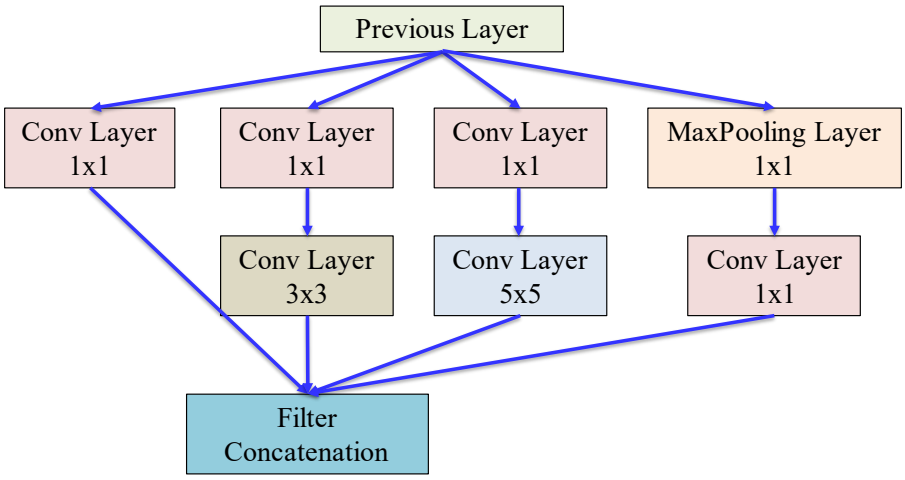
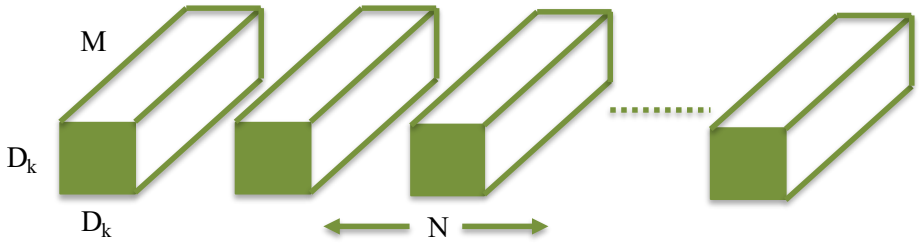


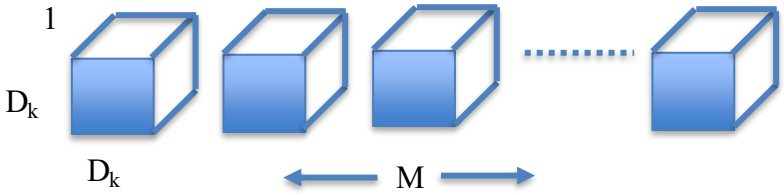
Figure 3.11: CNN base model architecture- Inception

3.4.5 MobileNet

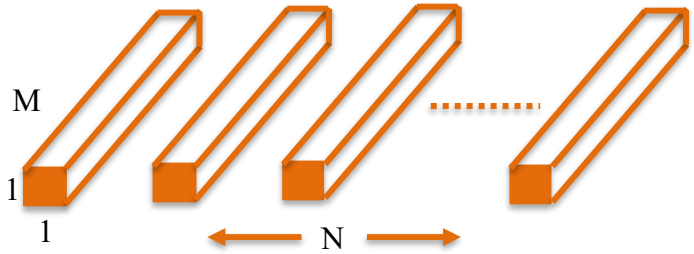
MobileNet CNN model is the first-ever computer vision model solely designed for mobile and embedded applications [Andrew et. al. (2017)]. This model takes a unique approach called depth-wise separable convolutions to build a lightweight neural network. Depth-wise separable convolution was first introduced by Laurent (2014) and showed a reduction in computational cost in the first few CNN layers. In practice, using the Depth-wise separable convolutions, MobileNet significantly reduces its quantity of the learnable parameters making the model smaller and faster. This unique convolution works in two steps- (a) Depthwise convolution and (b) Pointwise convolution. In depthwise convolution, the filters' depth and spatial dimension (input channel) are separated, and a single filter is applied for each input channel. Finally, the pointwise convolution, a 1x1 convolution, combines the outputs of the depthwise convolution. Figure 3.12(a) (b) shows the transformation process from standard convolution filter to two-layer depthwise convolution. Figure 3.12(c) presents the 1X1 convolution that grabs the output of the depthwise convolution to build a depthwise separable filter. From Figure 11 (a) (b) (c), DK is the size of the spatial dimension with a convolution filter K, M is the number of input channels, and N is the number of output channel.



(a) Standard convolution filters



(b) Depthwise convolution filters



(c) 1x1 Pointwise convolution filters

Figure 3.12: Difference between the standard convolution filer and the depthwise separable convolutions.

3.4.6 Xception

The basic concept of Xception is based on the Inception and refers to "extreme inception". Inception takes a 1x1 convolution to map the cross-channel correlations of the input images and uses a different type of filter for each depth space [Francois (2017)]. However, Xception works in a reverse way. Firstly, Xception applies the filters on each depth map, and a 1x1 convolution is used to compress the input space across the depth [Figure 3.13]. This model's working method is almost similar to depthwise separable convolution. Like Xception, depthwise convolution first works with taking the mapping of 1x1 convolution and later cross-channel wise spatial

convolution, whereas Inception works in vice versa. Another notable difference between the Inception and Xception model is the presence of non-linearity. Inception uses non-linearity throughout all its operations, followed by ReLu non-linearity; however, Xception avoids any type of non-linearity in its architecture.

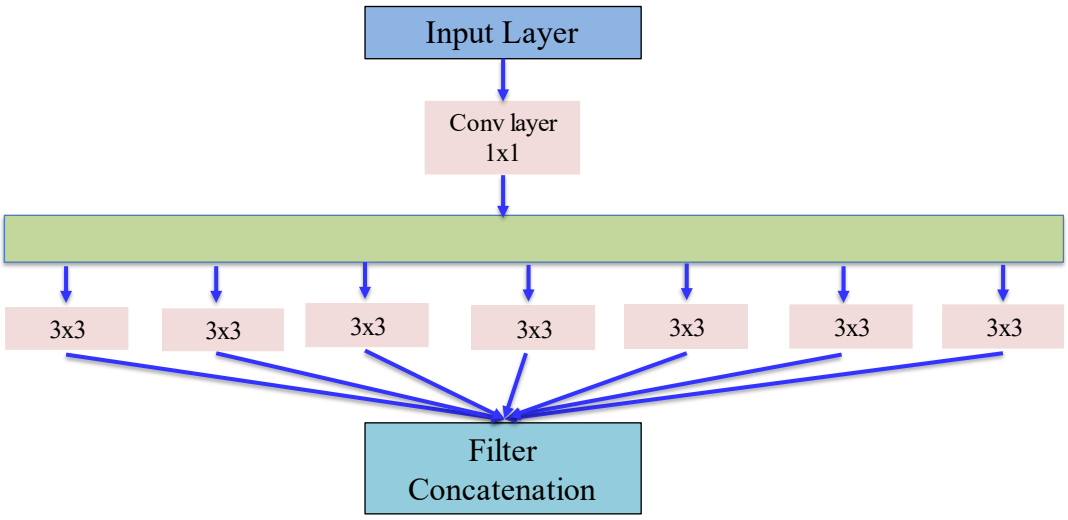


Figure 3.13: CNN base model architecture- Xception

3.5 Evaluation Metrics

In CNN model analysis, evaluation matrices are considered to quantify the statistical performance of the output results of the trained models. Evaluating the DL models is an essential part of understanding the output results and comparing various models' performance to select an appropriate model for different tasks. In this study, four different metrics are considered to evaluate the performance of defects classification: Accuracy, Precision, Recall and Confusion matrix. Following are formulations for these evaluation metrics-

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{23}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{24}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{25}$$

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

Here, *TP* denotes if the crack image is classified correctly while *TN* shows if the spalling image is classified correctly. *FP* means if the crack image is classified incorrectly while *FN* represents if the spalling image is classified incorrectly.

The confusion matrix is a type of matrix which presents the numerical summary of the final predictions (*TP, TN, FP*, and *FN*). This model uses a binary confusion matrix as the dataset is divided into two classes. For this study, "0" represents the "crack" and "1" is termed as "spalling". Generally, Intersection over Union (IoU) and F1-Score are considered for semantic segmentation. Apart from these two, precision and recall metrics are also applied here. In simple words, IoU [*Figure 3.14*] quantifies the overlap area between the original and predicted image, and the F1-score is a harmonic mean of the combination of precision and recall metric. F1-Score is one of the most indicative metrics to identify the best-performing network. Eqn. 26 and Eqn. 27 presents the formulation of IoU and F1-Score.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{26}$$

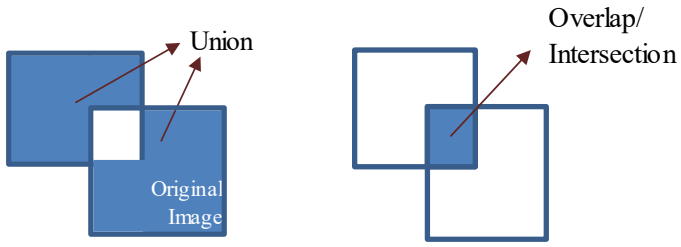


Figure 3.14: Schematic presentation of IoU

$$\text{F1} - \text{Score} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \tag{27}$$

3.6 Summary

This chapter presents a detailed summary of all the component of CNN layers and parameters used for this study. Also the chapter includes a short description of all the CNN models used as transfer learning methods for both classification and segmentation. Moreover, few mathematical elements

are used to explain the structure of the model algorithms. The application of these mathematical operations are demonstrated in chapter 4 and 5.

Chapter 4 Concrete Defects Classification using Deep Learning

4.1 Introduction

Concrete is one of the most popular material for any kind of infrastructure construction and within the life-cycle of an infrastructure it requires systematical maintenance. Crack and spalling are the most common defects types of concrete infrastructures and detecting these defects are the primary scope of any structural management and maintenance. Conventionally the on-site human inspections are time-consuming, expensive and certainly not error-free. In recent decades' researchers are working on implementing artificial intelligence (AI) in SHM to build an automatic detection method while avoiding the inconvenience of manual inspection. Vision-based DL methods have gained popularity for their significant efficiency in defects detection. To start an automatic detection process identification of defects is the preliminary requirement. This study has considered concrete crack and spalling for the damage detection process and used those defects' images for CNN model training. In classification process CNN layers learn the features from the images and categorizes the images based on the training. Later on the trained model is evaluated with a new dataset to examine the prediction accuracy of the model.

4.2 Research Method

CNN classifiers analyze the spectral information of pixels in the images and classify the pixels into multiple classes. For this study, the classification model is constructed to categorize two types of concrete defects- crack and spalling. In this study, the overall CNN classification model is built on three components – a) data processing, b) CNN models training, and c) performance evaluation of the trained models. A schematic diagram of the workflow chart for defect identification is presented in Figure 4.1. As shown in Figure 4.1, the overall process starts with data preparation, including image acquisition from available resources, image processed into desired resolution and randomly split into training, validation, and testing datasets. After data processing, the next step is the implementation of CNN models and model training. The hyper-parameters are continuously updated and tuned to achieve the best performance in the training and validation process. Later, the trained model is evaluated with the testing dataset to predict defects classification.

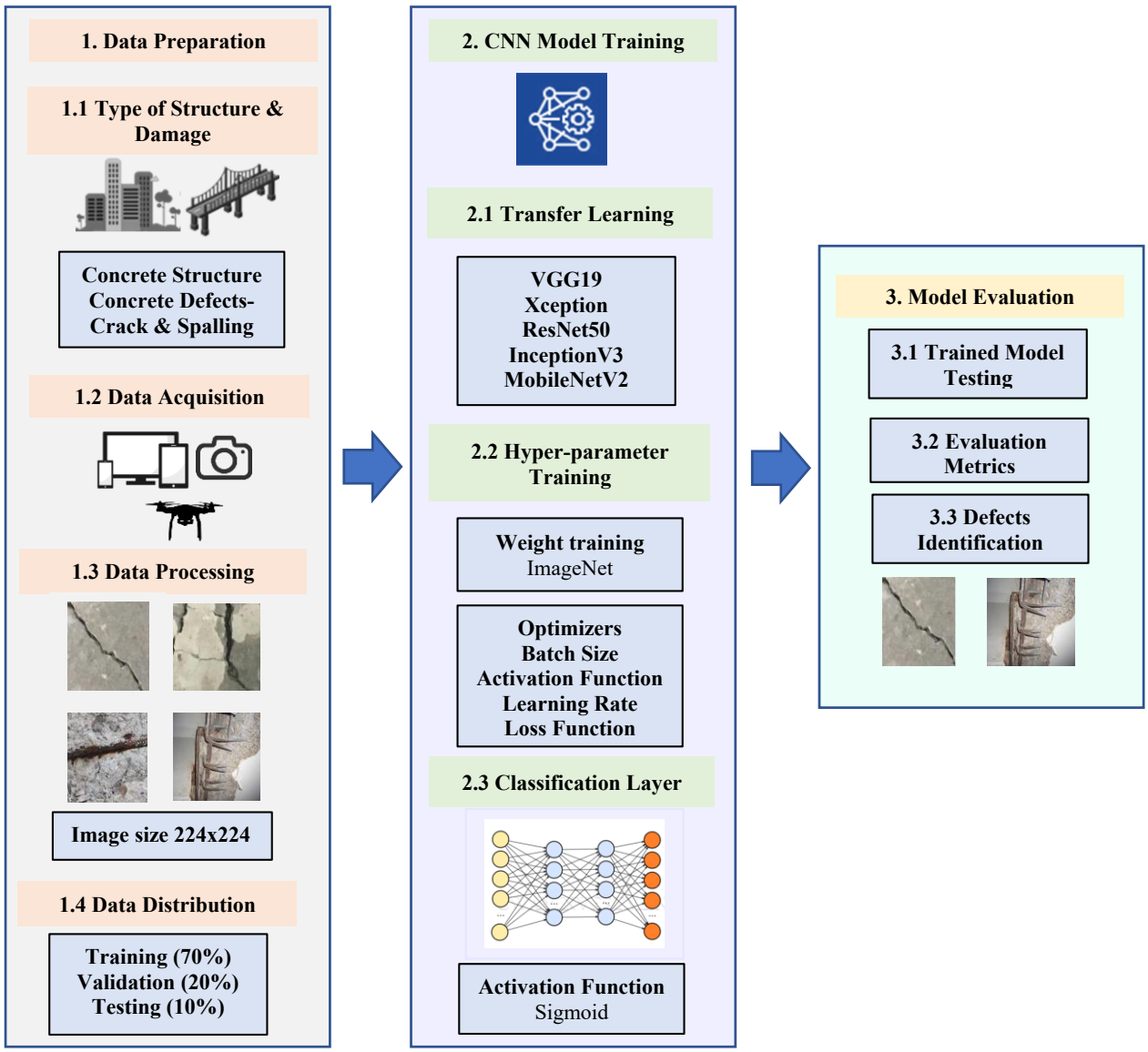


Figure 4.1: Workflow chart for defects classification

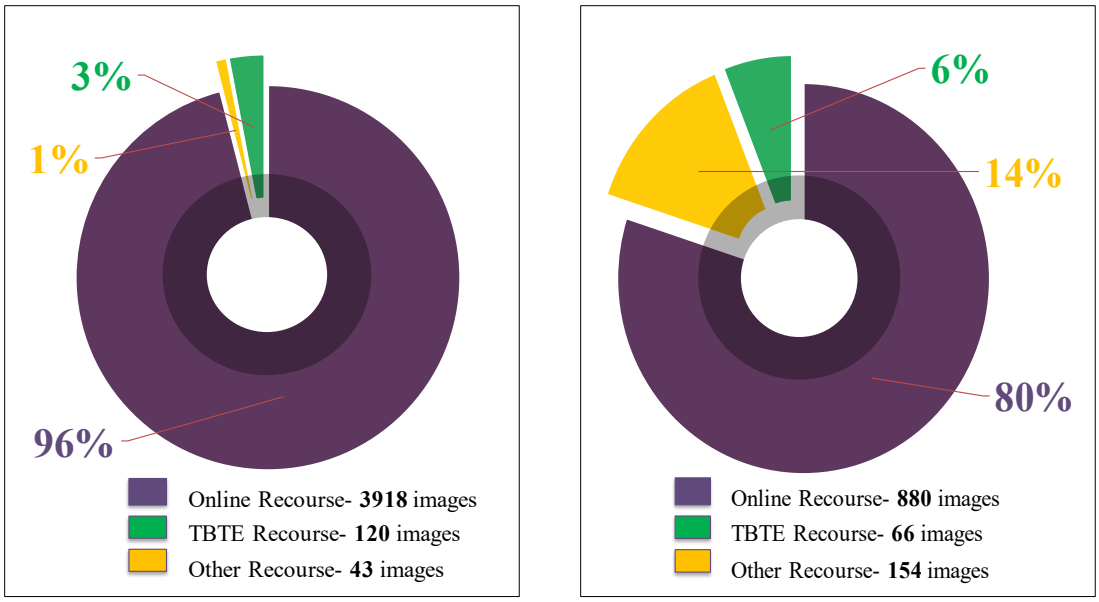
4.3 Data Preparation

4.3.1 Data Collection

CNN models' robust performance is highly dependent on developing an organized dataset. To maintain a certain significant performance of CNN models, a dataset should have high-quality images with various background noises, replicating the real-world conditions. Such conditions include surface roughness (i.e., scaling, edges, holes), lightening condition, background debris etc. According to Kaige et al. (2020), the quality and quantity of the dataset highly influence the

performance of the CNN models and are certainly affected by the low-quality images. Some studies have shown that the models trained with monotonous background images can perform better. However, when these models are evaluated in terms of the new dataset with a complex background, the prediction accuracy easily gets impaired. For example, Choi and Cha (2020) observed that when a CNN model was analyzed with a dataset of targeted images and noise-free background and subsequently tested on images with a rough surface, the model's performance severely declined, condensing the precision from 87.4% to 23.1%.

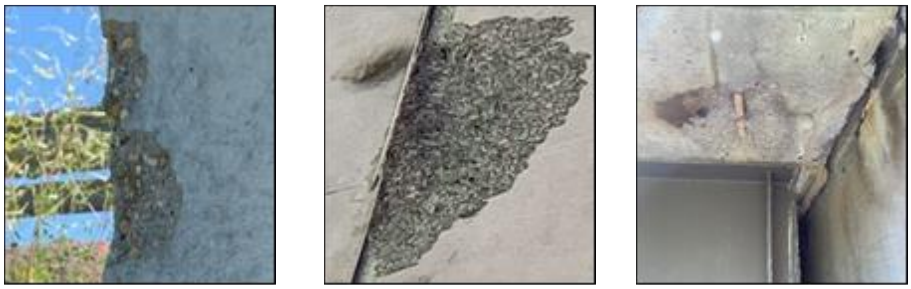
As mentioned in chapter 1, this study has worked with defects on any concrete infrastructure, such as bridges, buildings, dams etc. For defect types, concrete cracks and spalling are considered. One of the primary focuses of this study is to build a comparative dataset collecting images from various resources to imitate the real structural site conditions. Firstly, defects images are collected from actual infrastructure inspection reports executed by a local industry partner, TBT Engineering. These images have served as an exact replication of an actual event that occurs at defected structure site. However, the number of images collected from the inspection reports is inadequate to run a successful DL-based automated defect condition assessment project. Therefore, this study has taken advantage of the online resources to deal with the challenge mentioned above, as some previous studies have explored DL applications in concrete defect identification. Part of the concrete crack and spalling images are retrieved from a freely available annotated dataset created by Çağlar (2021). Apart from these sources, some images are collected from open-source online sources and experimental test results conducted on concrete sections. Finally, a dataset of 4087 crack images and 1100 spalling images are incorporated for this study [Figure 4.2]. A few data samples of crack and spalling images are presented in Figure 4.3. The developed dataset has a wide range of defects characteristics, such as different areas, lengths, widths, and shapes, including horizontal, vertical and zigzag shapes on the various concrete surface. These realities in defects' area and shape are supposed to aid the CNN models in learning the versatile patterns of the defects to make a more accurate prediction with untrained images.



(a) Crack dataset (b) Spalling dataset

Figure 4.2: Size of dataset

(a) crack (b) crack (c) crack



(d) spalling (e) spalling (f) spalling

Figure 4.3: Sample images for defects dataset (a, b, c) crack and (d, e, f) spalling (Image Courtesy: TBT Engineering)

To the author's best knowledge, this is one of the largest datasets of both concrete defects without applying any image augmentation process. Image augmentation is used to artificially expand a dataset for training purposes when the available dataset is minimal by generating new images from existing dataset. Augmentation process creates new dataset by applying minor alteration to the existing dataset using some techniques, such as- cropping, flipping, rotation, color adjustment etc.

4.3.2 Data Processing

As mentioned earlier, the images are collected from multiple sources, meaning it's natural that the image properties are not the same for the entire dataset. This is where detailed image processing is required to standardize the entire dataset's image format. At first, the image resolutions are systematized by converting all the images into a resolution of 224x224 pixels. As per the previous study by Majdi et al. (2020), when the DL models are trained using images with comparatively small pixels, the models can learn and identify the desired features more precisely. Even though the original images of this dataset had higher resolutions, to facilitate the CNN model's learning capability the images are turned into a much smaller pixel. For this study, colourful images are considered input images, implicating that the images have three channels Red-Green-Blue (RGB). These images are also called 3D images as they have three dimensions- height (h) x width (d) x channel (c). So the final input image size can be defined as 224x224x3.

The entire dataset is divided into input and testing images for a model's learning process. The input dataset is used to develop a prediction model, whereas the function of the testing dataset is to determine the model's prediction quality. The input images have two components- the training dataset and the validation dataset. While the training dataset is used for the learning process, the validation dataset offers an unbiased evaluation of the training dataset by subsequently tuning the hyper-parameters. Conventionally, while splitting the entire dataset, the input dataset is considered to have a larger portion of images while the rest is used for testing purposes. However, there is no universal approach to dataset splitting ratio. For instance, most of the researchers [Xincong et al. (2018), Shengyuan and Xuefeng (2020) and Jia-ji et al. (2022)] have considered an 80%-20% train-test split ratio for their CNN models. On the other hand, Youzhi et al. (2021) has adopted the 70% of the entire dataset as a train and validation dataset and the rest of the 30% as a test dataset. Jacob et al. (2021) have divided the dataset into 60%-40% ratio to use the 60% as input images and 40% for evaluating the models. As the crack dataset and spalling dataset has a big difference

in size this study has decided on using the maximum images for training and validation purpose for the CNNs classification, and split the dataset into 70%-20%-10% ratios for training, validation and testing purposes. Table 1 presents the summary of data distribution for train, validation and testing.

Table 4.1: Image distribution for classification CNNs

Defect Classes	Total	Train Dataset	Validation Dataset	Test Dataset
Crack	4087	2861 (70%)	817 (20%)	409 (10%)
Spalling	1100	770 (70%)	220 (20%)	110 (10%)

4.4 CNN-classifier Model configuration

In this study, five different CNN-classifiers are considered: (a) VGG19, (b) ResNet50, (c) InceptionV3, (d) Xception and (e) MobileNetV2. The details of these models are discussed in section 3.4. One of the main reason behind choosing these five models is that after analysing the previous studies, it is perceived that these models have consistent difference in their trainable layers and their performance potentiality. For example, while the VGG-19 followed by ResNet-50 has the lowest trainable layers, they have shown acceptable performance prospects with their unique architecture. As MobileNetV2 was built to perform faster in a mobile application system, this model is considered in this study to evaluate the model's damage identification performance if the model is implemented in mobile application. Moreover, InceptionV3 and Xception has a large amount of trainable layers which helped this study to comprehend the variation in model's performance with a change in trainable layers. The algorithms of these networks are developed using Keras applications [Francois (2015)]. Keras application includes the pre-built deep learning models which can be used for training the model and make prediction. For the coding language, Python is used backend by TensorFlow. After building the CNN-classifier application, the model simulations are run using Google Collaboratory.

4.5 Sensitivity Analysis of Hyper-parameters

As mentioned earlier in chapter 3, pre-trained ImageNet weights are considered to start the training process of CNN models, followed by a continuous trial-error method to reach the optimized point

of hyper-parameters. A sensitivity analysis is done to train the hyper-parameters and find the best-performed models. This study considers a few hyper-parameters, such as batch size, activation function, optimization function, loss function and learning rates, for sensitivity analysis. As mentioned by Goodfellow (2016) these parameters are the most important parameters that guide the models towards the optimized convergence. The details of these hyper-parameters are presented in Table 4.2.

Table 4.2: Details of hyper-parameters

Name of Parameters	Value of Parameters
Batch Size (CNN-classifiers)	10
Learning rate (CNN-classifiers)	0.1, 0.001, 0.0001
Optimization function	SGD, RMSprop
Activation function	ReLu
Evaluation metrics threshold	0.5
Loss function (CNN-classifiers)	Binary cross-entropy
Pre-trained weights	ImageNet
Callbacks	Early-stopping
Epoch	100

To achieve the best output result, the values of hyper-parameters for CNN-classifiers are designated after carefully analyzing the learning process. According to Table 4.2, a batch size of 10 is considered, and the models are trained for 100 epochs. An epoch refers to the one complete training cycle of a forward pass and backpropagation. To finalize the epoch size, two functions called early-stopping and the reduced learning rate is applied in these models. These two functions help the models avoid over-fitting by stopping the model's training process when the best accuracy is achieved. This also helps reduce the models' computational cost (time and computer memory). After completing all the combinations of sensitivity analysis, it is found that the models reach their

optimized performance condition within the 100 epochs. Therefore, this study has considered 100 epochs for model training. Also, for batch size, it is observed that with a group of 10 images, the model learns the features with a minimal computational cost. Moreover, as an activation function, ReLu has functioned to have a positive impact on the model's performance. According to some previous studies SGD and RMSprop are some commonly used optimizers to train the CNN models [Poojary and Pal (2019), Kumar et. al. (2019), Agarwal et. al. (2021)]. Also, some studies used learning rate 0.001 [Poojary and Pal (2019)] and 0.0001 [Verma et. al. (2021)] to control the learning process of the CNN model to achieve the best performance Hence this study has explored two different types of optimization functions- SGD and RMSprop along with three different learning rates 0.1, 0.001 and 0.0001 for each of the five models separately and summarized the results in section 4.6. Finally, the best hyper-parameters values are decided on by evaluating the trained model with the testing dataset and comparing their results using the evaluation matrices.

4.6 Result and Discussion

In accordance with section 4.5 for the sensitivity analysis, at first, each CNN-classifier model- VGG19, ResNet50, IncptionV3, Xception and MobileNetV2 has considered two different optimizers- SGD and RMSprop. Later, each CNN-classifier with both SGD and RMSprop optimizer is evaluated for three learning rates 0.0001, 0.001 and 0.1. With the combination of two optimizers and three learning rates for five CNN classifiers, thirty models are analyzed and evaluated separately. Table 4.3 represent the performance of all the CNN-classifiers for learning rate 0.0001, 0.001 and 0.1, respectively. Three evaluation matrices, accuracy, precision and recall, are considered to evaluate the model's performance.

From Table 4.3, it can be established that InceptionV3 has outperformed all the other models in the case of both optimizers. For a learning rate of 0.001, SGD optimizer IncpetionV3 has achieved the best accuracy, precision and recall value of 91%, 82% and 100%, respectively. Xception attains the second-best performance by adopting SGD optimizer with an accuracy of 89%, precision of 82% and recall of 94%. As mentioned in chapter 3, the architecture of Xception is based on Inception model, which is one of the possible reason for the performance resemblance of these two models. Inception model is considered to have better performance than ResNet as Inception model focuses on reducing computation cost while doing deeper eventually increasing optimization

accuracy. However, ResNet only works on computational accuracy without concerning the optimization which can lead to overfitting the training process and ultimately affecting the prediction performance. In case of MobileNet, this model has less learnable parameters than Inception model which can be an advantage to achieve reasonable performance with lower memory capacity, but then with higher learnable parameters Inception model performs better than MobileNet. So, it is evident that InceptionV3 model has outranked the other models.

Table 4.3: Summary results of defects classification models

CNN models	Learning rate	**Accuracy**		**Precision**		**Recall**	
		SGD	***RMSProp***	***SGD***	***RMSProp***	***SGD***	***RMSProp***
****InceptionV3***	0.1	86%	88%	78%	82%	100%	97%
	0.001	**91%**	**89%**	**83%**	**79%**	**100%**	**100%**
	0.0001	84%	89%	81%	84%	94%	100%
Xception	0.1	89%	87%	79%	76%	100%	100%
	0.001	90%	88%	81%	78%	100%	100%
	0.0001	89%	88%	82%	78%	94%	100%
MobileNetV2	0.1	81%	79%	71%	73%	94%	76%
	0.001	82%	83%	71%	72%	94%	100%
	0.0001	82%	84%	71%	70%	94%	100%
ResNet-50	0.1	85%	87%	72%	76%	97%	100%
	0.001	82%	87%	69%	77%	89%	97%
	0.0001	79%	85%	69%	74%	89%	97%
VGG-19	0.1	63%	65%	60%	62%	81%	82%
	0.001	61%	62%	64%	68%	80%	84%
	0.0001	63%	67%	61%	62%	82%	86%

*Note: *Best Performance*

In the case of a learning rate of 0.001, InceptionV3 has shown the best output, followed by Xception model [Figure 4.4]. With learning rate 0.1 the training process has skipped many learning features and converged faster towards a suboptimal position, whereas learning rate 0.0001 has

taken a slow pace to update the models' weights and increasing the computational cost without improving the model's performance significantly. Between two optimizers, the optimizer SGD has aided in obtained the best performance for defects classification for IncpetionV3 with the accuracy, precision and recall values of InceptionV3 are found at 91%, 83% and 100%, respectively. Like InceptionV3, Xception has the best performance with the SGD optimizer. According to [Hardt et. al. (2016)] SGD has better stability and generalization capacity than other adaptive optimization methods (i.e., RMSprop), which helps the models to reach its optimization point better than others. Wilson et. al. (2019) studied experimental and empirical analysis to prove that for classification task SGD converged better than other adaptive methods. They also stated that even with faster initial training progress, in validation the performance did not improve much.

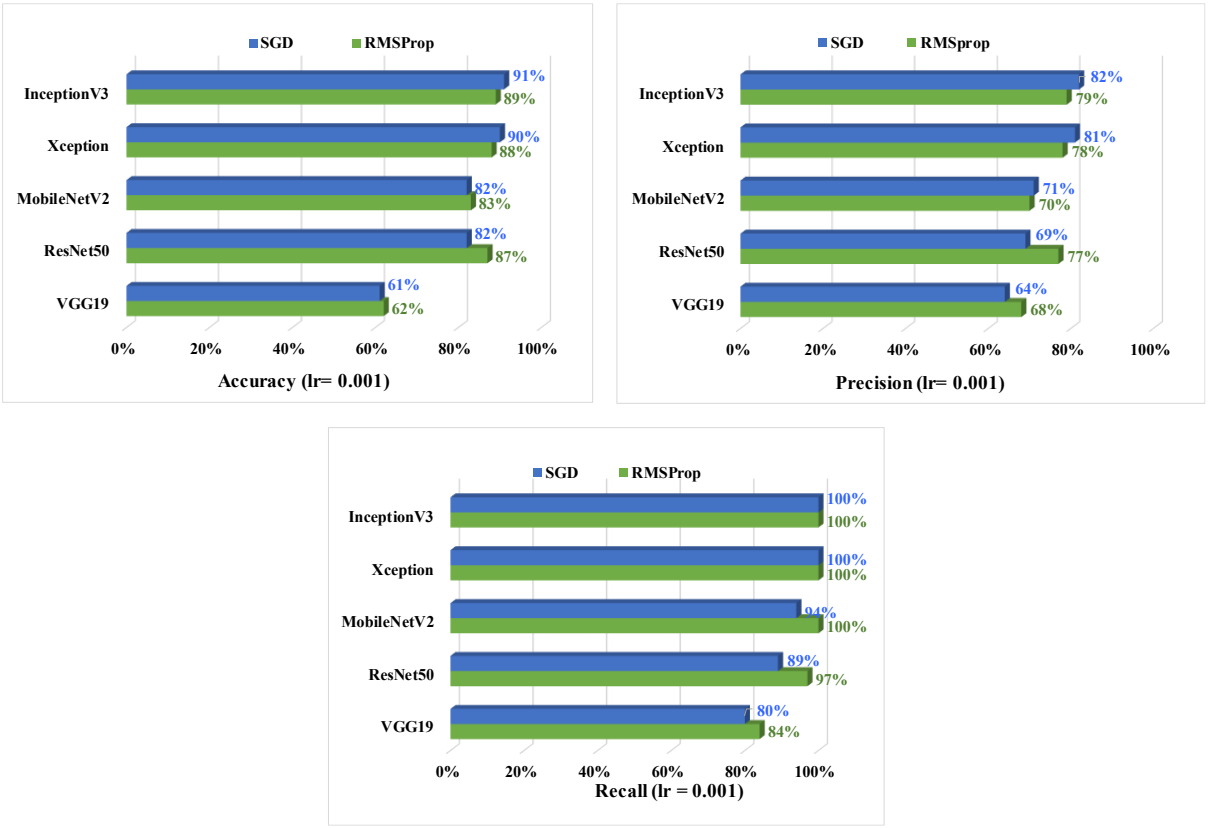
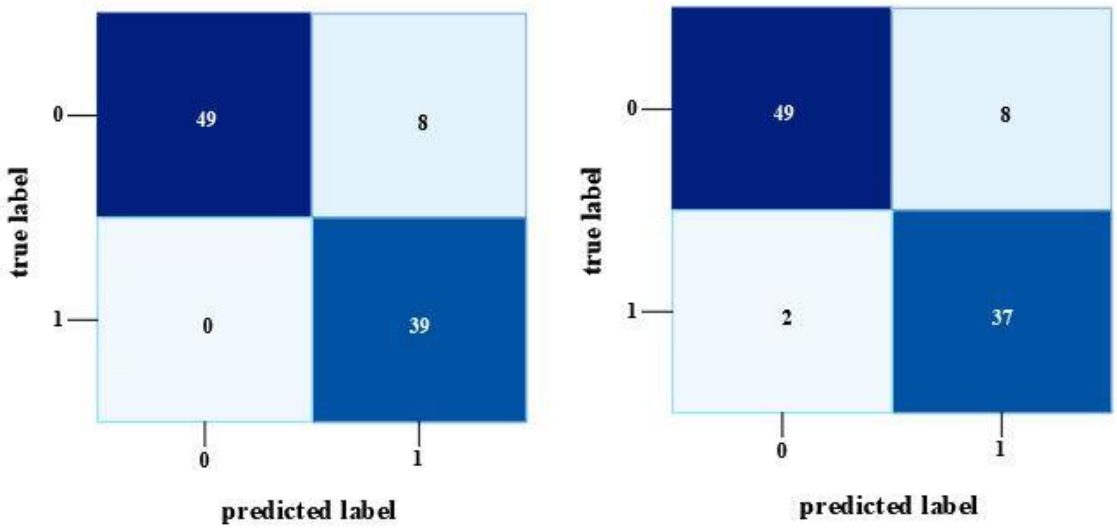


Figure 4.4: Defects classification: comparison of the evaluation metrics based on SGD and RMSprop optimization function for learning rate 0.001

After analyzing the models with evaluation metrics, another evaluation is done based on the true label vs. prediction label of crack and spalling to determine which CNN-classifier has attained better performance on defects identification. As mentioned in chapter 3, the confusion matrix helps to understand the image's true and predicted label. Figure 4.5 portraits the confusion matrix for InceptionV3 and Xception model. For all the confusion matrix diagrams, the x-axis and y-axis represent the true label and predicted label, where "0" denotes the crack, and "1" refers to "spalling". As mentioned earlier, the InceptonV3 and Xception model has its' best performance with optimizer SGD and learning rate 0.001. Therefore, this study has illustrated the confusion matrix graphs only for those conditions.



(a) InceptionV3 (SGD) (b) Xception (SGD)

Figure 4.5: Confusion matrix for InceptionV3 and Xception for optimizer SGD and learning rate 0.001

Figure 4.5 (a) and (b) illustrates the true and false prediction of defects by the InceptionV3 and Xception model, respectively for learning rate 0.001 and optimizer SGD. From the graphs, it is visible that in case of crack prediction with both InceptionV3 and Xception models predicted forty-nine images correctly while making eight false predictions. In case of spalling detection InceptionV3 has predicted all the spalling cases correctly whereas Xception has falsely identified two spalling cases. From the explanation above, it is clear that the InceptionV3 model has the superiority over Xception model.

As mentioned earlier in chapter 3, loss function helps the model reduce the difference between the true value and prediction value for tuning the hyper-parameters, so its' essential to track the training loss and validation loss over the training period. Figure 4.6, Figure 4.7 and Figure 4.8 present the graphical understanding of the InceptionV3 model's performance over the epochs for three learning rates 0.0001, 0.001 and 0.1. From the graphs, it is prominent that IncpetionV3 models have the least amount of loss with a learning rate of 0.001. Also, the trained model has obtained the sharp training accuracy, precision and recall close to 100%.

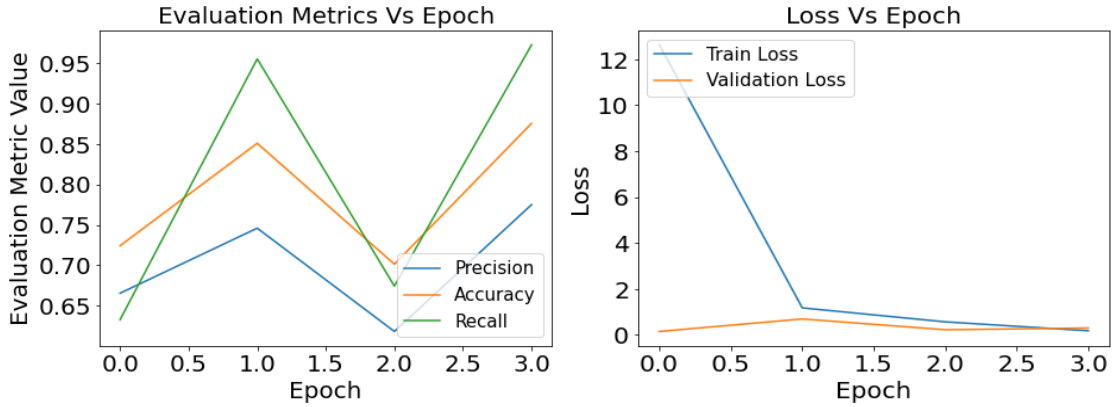


Figure 4.6: Defects classification with InceptionV3: result of the evaluation metrics and model loss for optimizer SGD and learning rate 0.0001

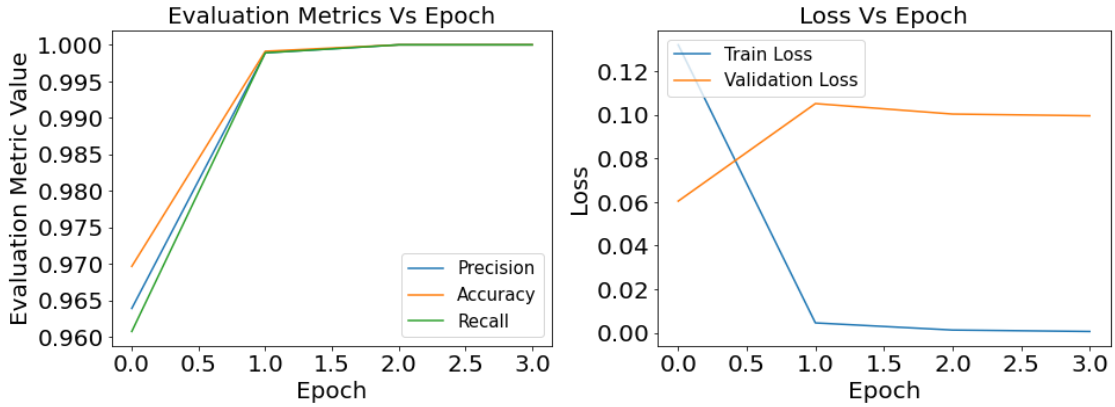


Figure 4.7: Defects classification with InceptionV3: result of the evaluation metrics and model loss for optimizer SGD and learning rate 0.001

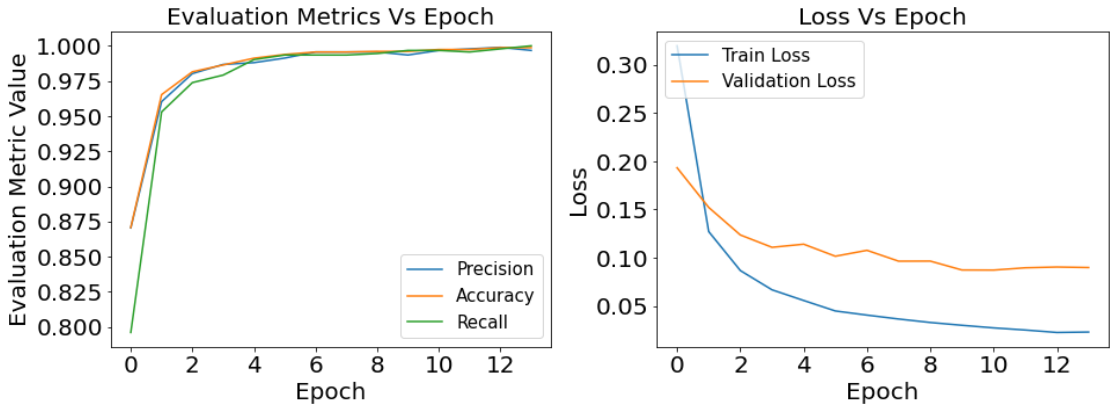


Figure 4.8: Defects classification with InceptionV3: result of the evaluation metrics and model loss for optimizer SGD and learning rate 0.1

After analyzing the model's performances, the IceptionV3 model ranked the best-performed model for defects classification. Also, this model has reached its performance-optimized point owing to the SGD optimization function and learning rate of 0.001. Apart from InceptionV3, the Xception model has also shown a promising ground for defects classification using the SGD optimizer. On the other hand, among all the CNN classifiers VGG19 has ranked the last. One possible reason behind the InceptionV3 model functioning better than other models is that the model has the highest layers of depth for learning, which facilitates the model to gain better performance. On the other hand, VGG19 has the least depth of learning layers, which may have affected its overall performance. The best performance results for each CNN classifier model are presented in Table 4.4. This table summarizes the best result for the models based on their potential optimizer and learning rate.

Table 4.4: Details and results of **defects classification** using **Learning rate 0.001**

Network	Optimizer	Parameters	Accuracy		Precision		Recall	
		(millions)	Train	Test	Train	Test	Train	Test
*InceptionV3	SGD	87.9	100%	89%	100%	80%	100%	100%
Xception	SGD	83.7	96%	89%	70%	82%	100%	94%

*Note: *Best Performed Model*

Figure 4.9 and Figure 4.10 demonstrates some sample results of defects identification of crack and spalling for all the CNN-classifiers. On the image the first sentence describes the prediction result of the defects, and the second line shows the label of defects type. Figure 4.9 indicates that InceptionV3 has predicted most of the cracks with a 100% accuracy. On the other hand, some crack images have an accuracy of around 90% and predicted very few crack images with spalling. Also, the VGG19 model has the least accuracy in crack prediction and even has some false predictions. From Figure 4.10, it is clear that, similar to crack prediction, the InceptionV3 model also performed best for spalling detection and VGG19 has the least accuracy. In both figures red box indicates the prediction inaccuracy. All the probability percentage for each damage cases are the output results of from developed CNN models.

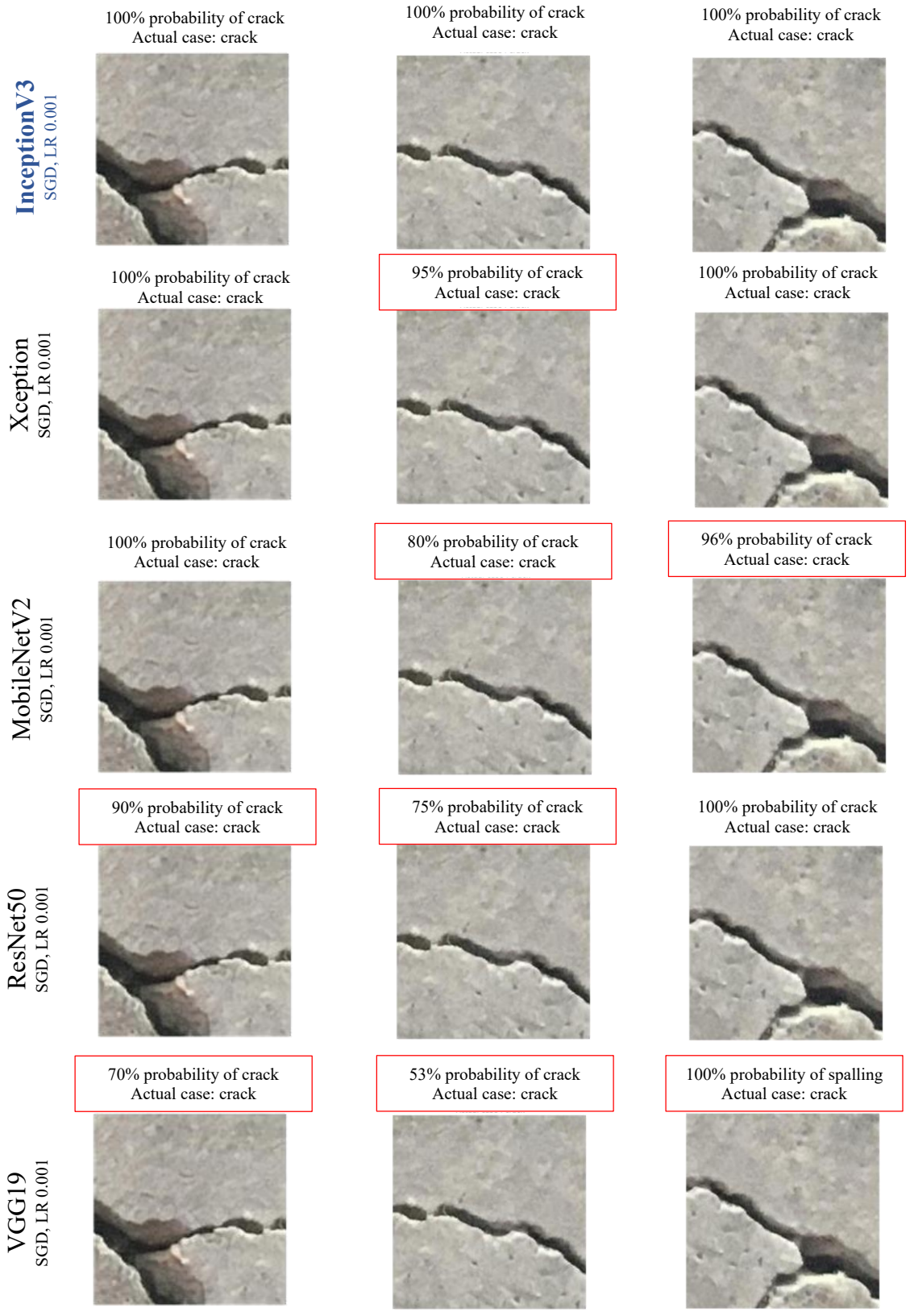


Figure 4.9: Sample images for crack prediction using CNN models

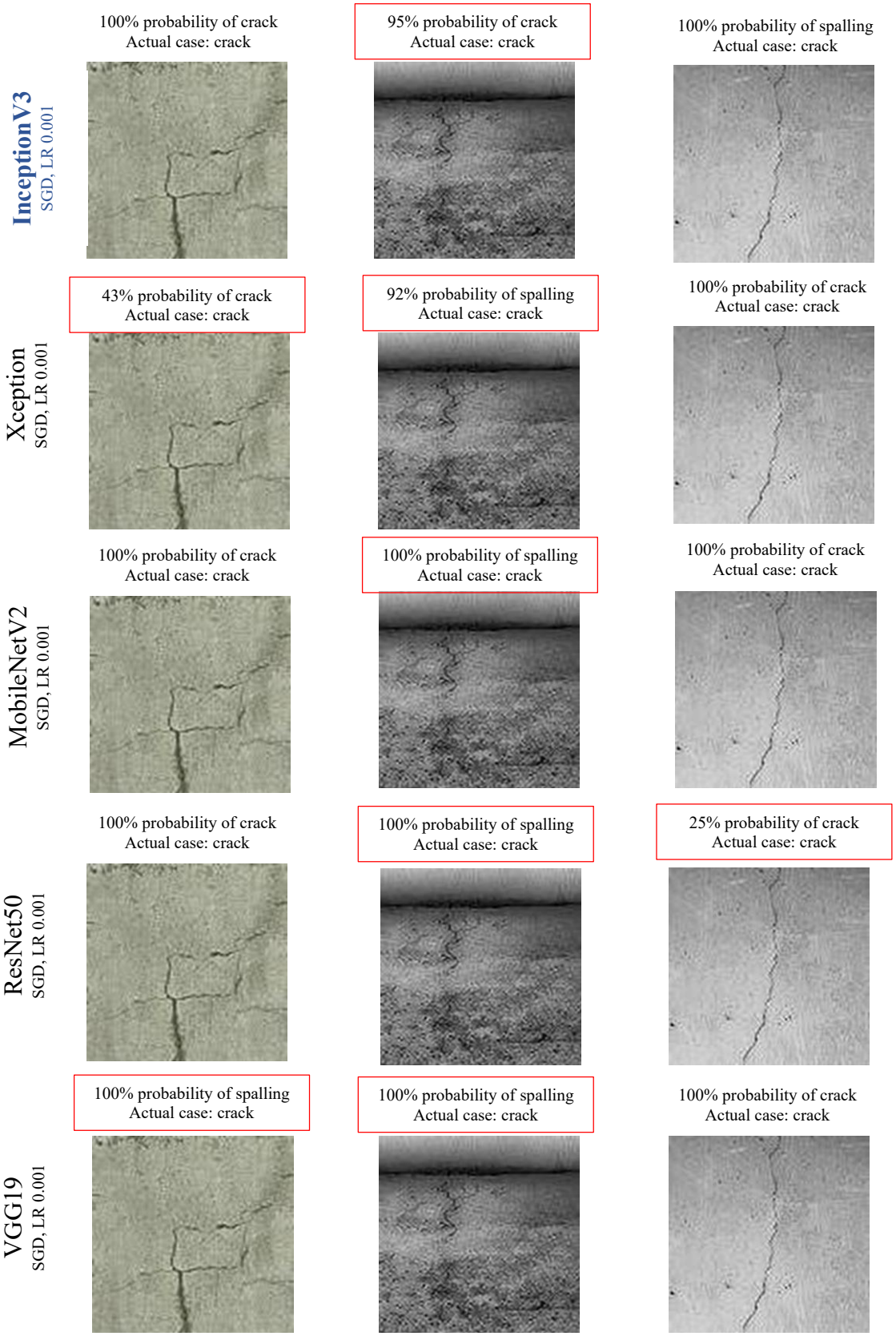


Figure 4.9 (continue): Sample images for crack prediction using CNN models

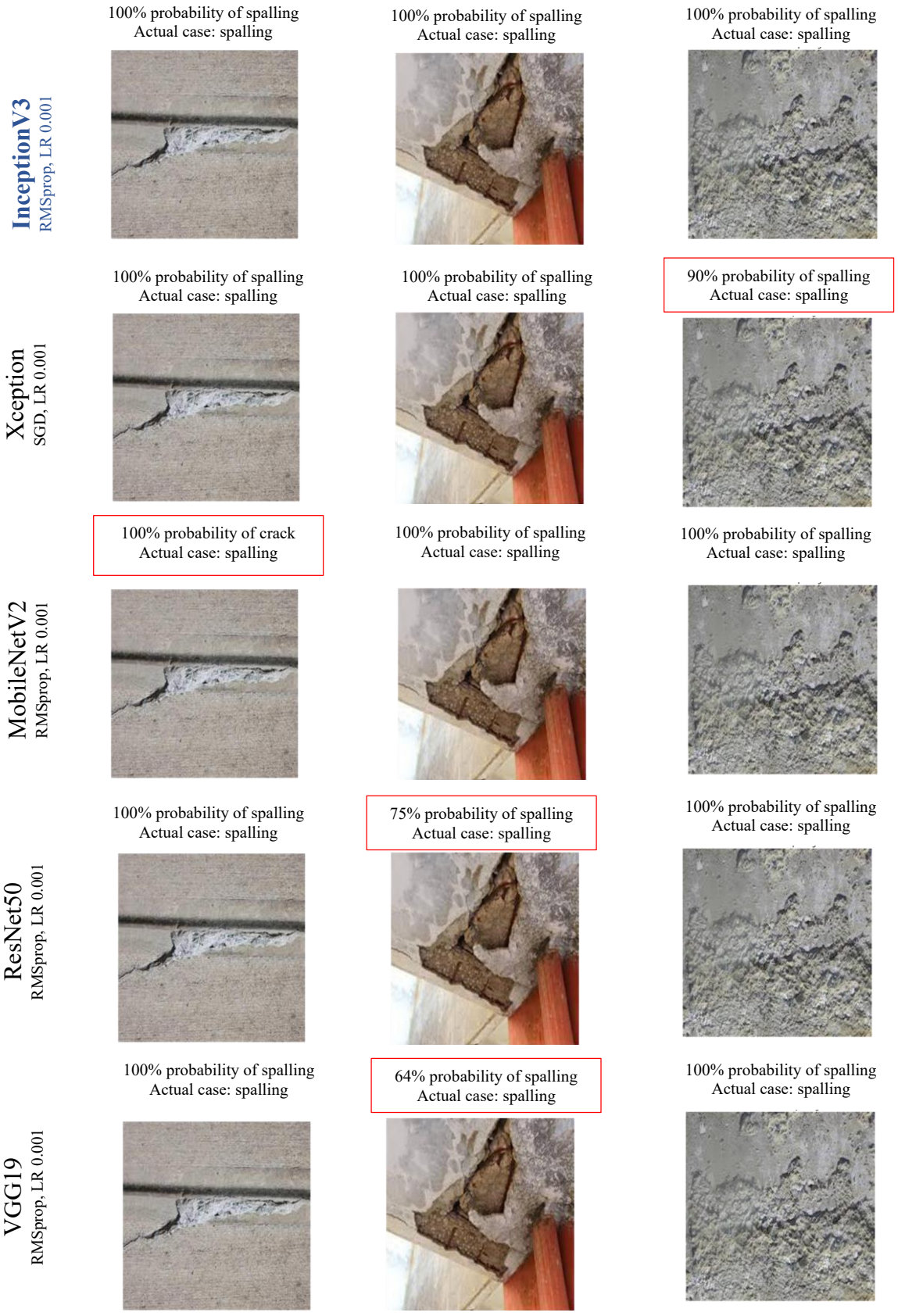


Figure 4.10: Sample images for spalling prediction using CNN

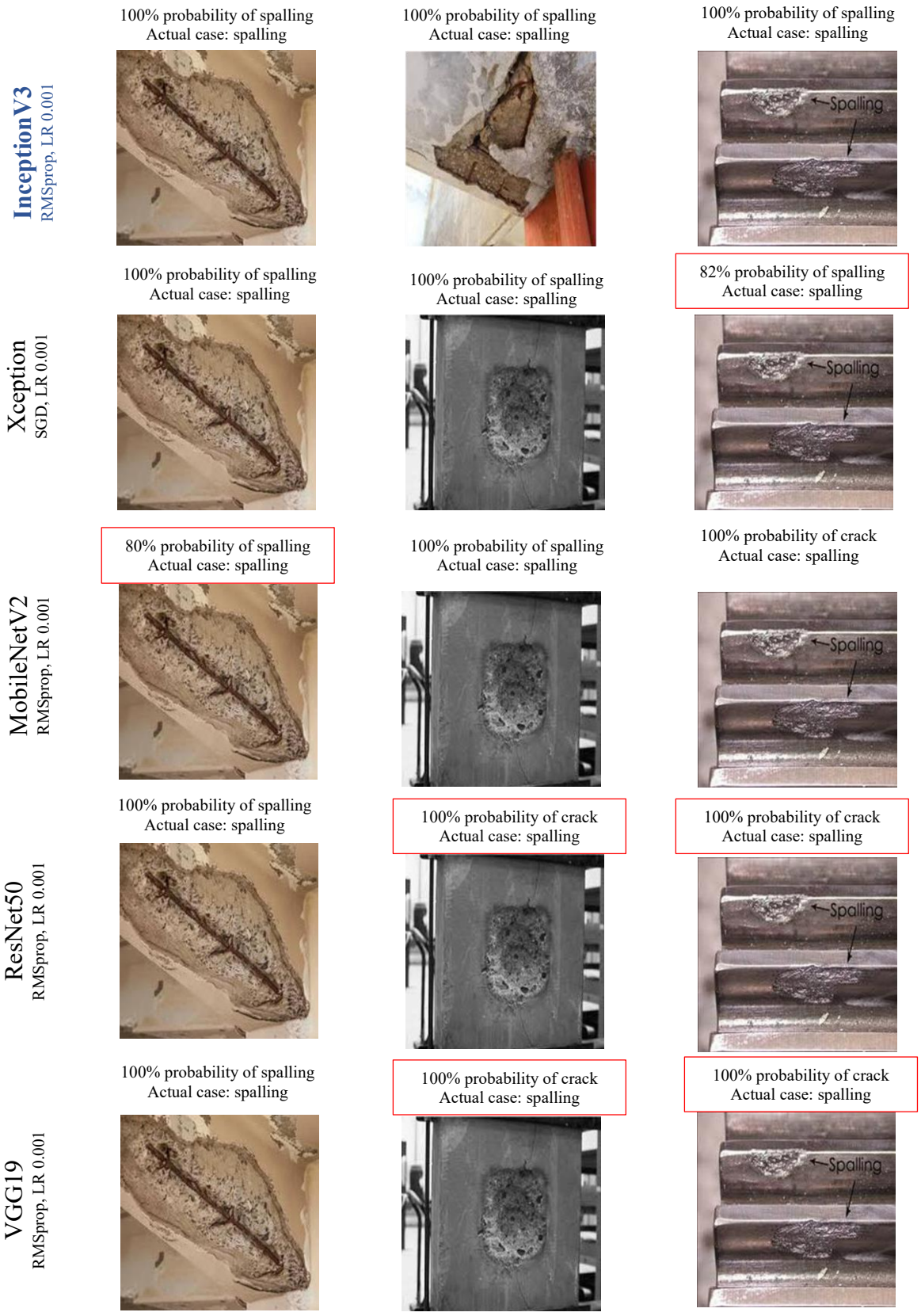


Figure 4.10 (continue): Sample images for spalling prediction using CNN

4.7 Summary

This chapter explores the potentiality of five different CNN classifiers to predict categorizing the defect type. The chapter has two primary goals a) conduct CNN classification for multi-class defects- crack and spalling, and b) train the model with different types and values of hyper-parameters to obtain the best output from the CNN classifiers. To achieve the first goal, this study has collected a dataset of 4080 crack images and 1100 spalling images. As the performance of a CNN model highly depends on the size and quality of the dataset, it is essential to train a model with images replicating the real site condition. This study has collected images from real-world inspection reports to train the model more effectively in real-life infrastructure inspection work. Apart from the inspection reports, many images are obtained from previous research work and other open-source resources. This dataset is solely prepared for this research purpose only. To the author's knowledge, this is one of the largest concrete crack and spalling datasets consisting only of the original images avoiding the image augmentation process. After finalizing the dataset, the CNN classifiers are analyzed for different types of hyper-parameters and evaluated based on their prediction potentiality. The highlights of this task can be summarized as follows:

a) A detailed sensitivity analysis is conducted by combining different types of hyper-parameters, and a total of thirty models are evaluated to decide on the best-performed model for defects identification. Among other hyper-parameters, this study has mostly focused on two optimization function-SGD and RMSprop, and three learning rates, 0.0001. 0.001, 0.1.
b) After a comparative analysis of all these models InceptionV3 model has outranked the other models with accuracy, precision and recall of 91%, 83% and 100%, respectively. Furthermore, Xception has also gained a profound accuracy with defects classification, whereas VGG19 has the least prospect with defects identification.
c) Regarding optimization function, SGD has shown the acute characteristic to guide the IncpetionV3 and Xception model towards its optimized trained condition. However, for the ResNet-50 and VGG-19 model, RMSprop has aided in adequate prediction quality.
d) While training all the models for each optimization function and learning rate, models with a learning rate of 0.001 learnt the trainable parameters more precisely. This phenomenon directs the concept of using a lower learning rate does not always confirm better

performance with CNN models, some cases it can prolong the learning process without significant performance improvement.

e) This study has presented the actual positive and negative defects prediction percentage by adopting the confusion matrix as one of the evaluation metrics for the CNN models. According to the results, IncpetionV3 has made eight false predictions of spalling while the actual cases are crack, and successfully labelled all the spalling cases. After InceptionV3, Xception has the best prediction results- nine falsely labelled crack images and zero false cases for spalling.

Chapter 5 Deep Learning Application: Defects Semantic Segmentation

5.1 Introduction

DL-based automated assessment of infrastructure has significant potential to replenish human visual inspection. In SHM, after categorizing defects, it is crucial to localize the damages to understand the exact pattern and quantify them. Some of the previous studies used the sliding window technique to localize the crack and background in images [Cha et al. (2017), Cha and Choi (2017)]. However, this technique depends on the size of the sliding window rather than the exact size and shape of the damage, which limits the exact qualification of the area of the defect. Also, in the case of thin, diagonal and atypical features residing in defects images, tracking the specific location of the defects can be obscured [Wooram and Young (2019)]. To address this challenge, this study has adopted the pixel level localization method, also named as segmentation process, to detect the defective area in images. Object segmentation serves as the key component of the DL task that aims to learn the features of an image to obtain a complete understating of a scene in images. The base work method of segmentation is similar to classification, except the segmentation process uses each pixel to classify the defects pixel and the image background. In contrast, classification is formulated by considering the entire image. In segmentation, the binary numbers "1" and "0" represents the pixel values of a defect pixel and a general background pixel, respectively.

5.1.1 Semantic Segmentation

There are different types of the segmentation process; however, in this study, semantic segmentation is adopted for defect localization. In DL, to understand an image at pixel-level, two questions must be answered- what object resides in the image, and where is it located in the image? By assigning a class label to each pixel, semantic segmentation can differentiate among the objects and locate each object by creating a segmentation mask inside the boundary line of the same pixel values. In summary, semantic segmentation works in three steps: (a) classifying individual objects on the image at pixel-level, (b) localizing the object by drawing a bounding line, and (c) creating a segmentation mask on a similar group of pixels in a localized image. Figure 5.1 shows a general

overview of semantic image segmentation where the defect (i.e. crack) is labelled as “1” and the background is labelled as “0”.

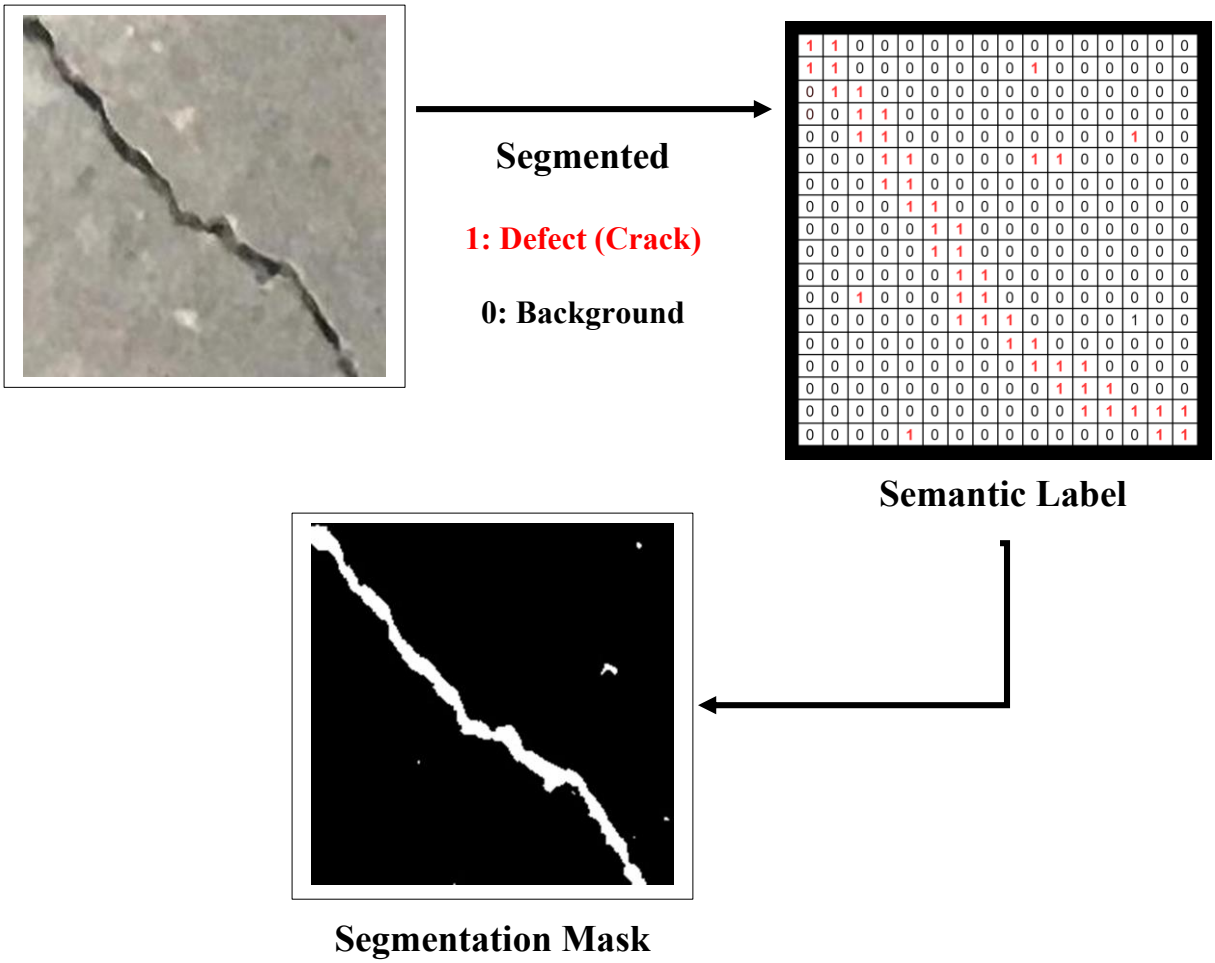


Figure 5.1: Schematic diagram of semantic image segmentation process

5.2 Research Method

In DL, CNN segmentation analyzes each pixel in an image and recognizes a group of pixels that forms a distinct category. In this study, separate semantic segmentation models are constructed for each type of concrete defects: crack and spalling. Similar to CNN-classifiers, the CNN segmentation model is also built on three components: a) data processing, b) CNN models training, and c) performance evaluation of the trained models. Even though the overall work process is the same, each component has a different functioning method from CNN-classifiers. A schematic

diagram of the workflow chart for defect semantic segmentation is presented in Figure 5.2. As shown in Figure 5.2, the overall process started with data preparation, including a separate dataset for crack and spalling, and randomly split into training, validation and testing datasets. Later, both crack and spalling dataset is implemented in encoder-decoder models individually and analyzed with four different CNN base models (VGG19, ResNet50, InceptionV3 and EfficienNetB3). The encoder part of the model converts the image pixels into a two dimensional vector using the CNN base models to focus on the context of the image. On the other hand, the decoder part gives meaning to the context learned from the encoder part. After adopting the CNN models, the models undergo continuous training and validation to learn the pixel data from images. In the training and validation process, the hyper-parameters are updated continuously and tuned to achieve the best performance for defects segmentation. Finally, the test datasets are applied to the trained model to evaluate the performance of the trained model for defects localization.

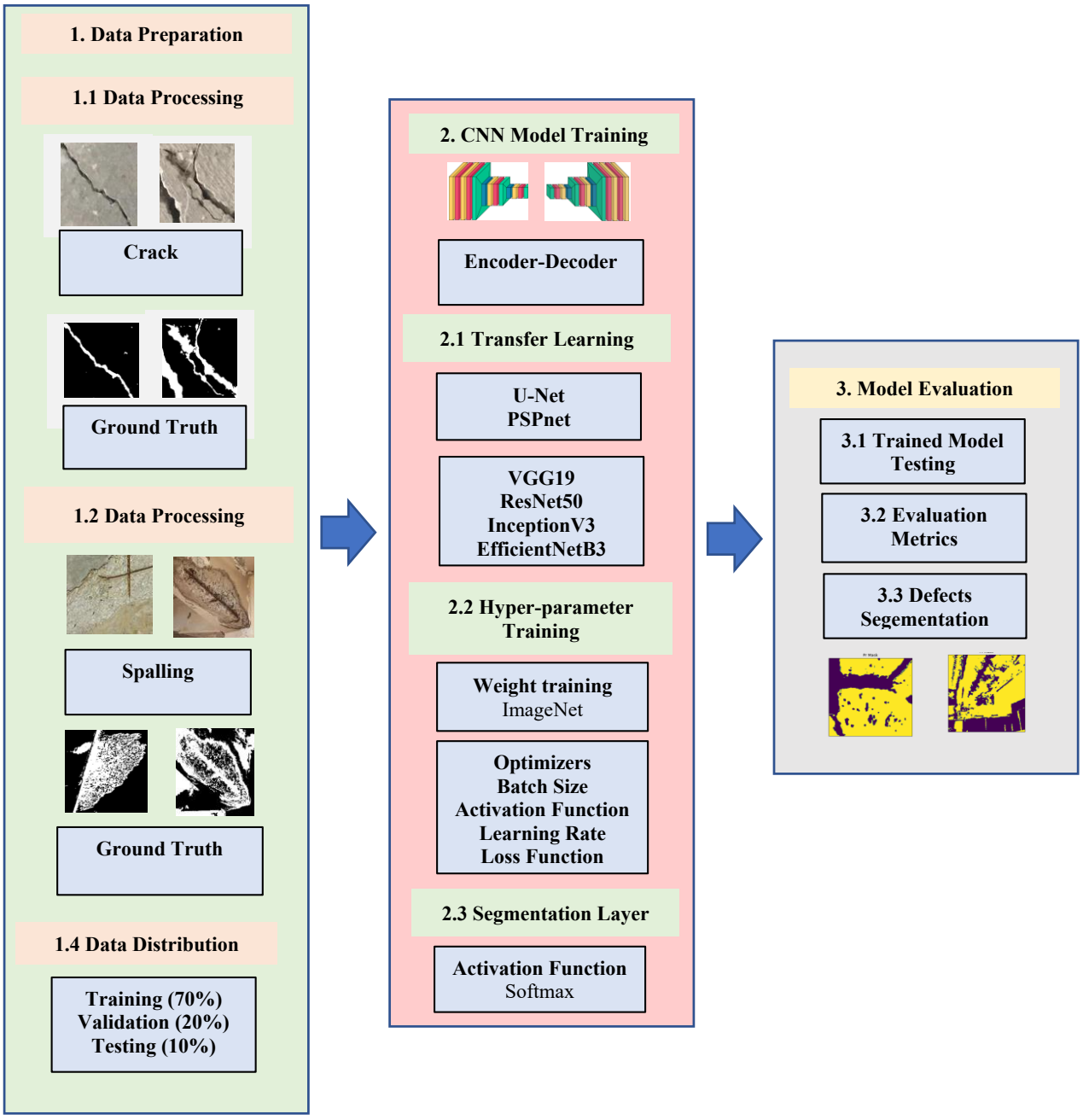


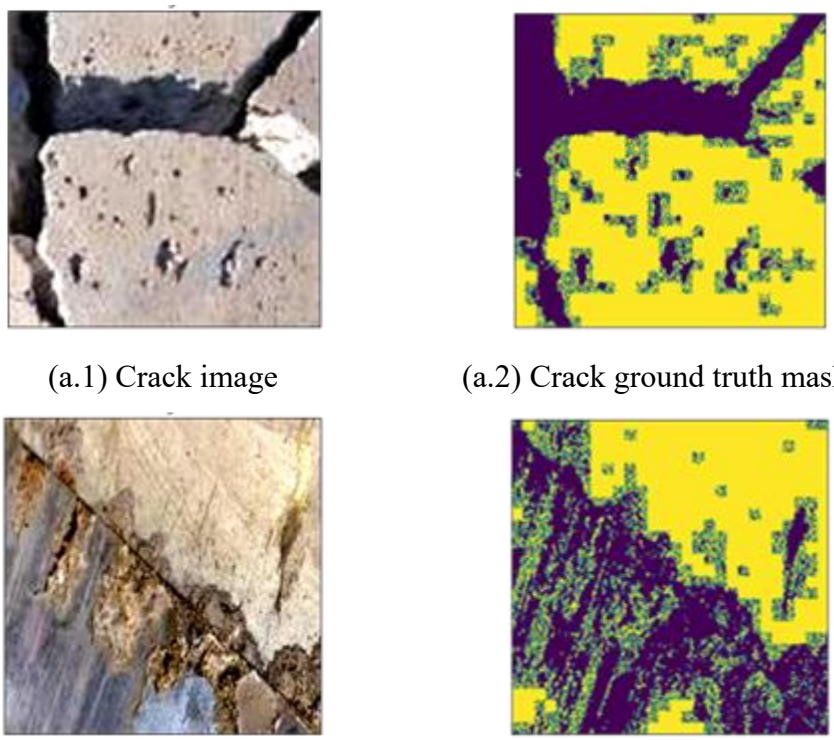
Figure 5.2: Work flow chart for defects segmentation

5.3 Data Preparation

The same dataset as defects (i.e., crack and spalling) classification is used here for CNN segmentation. Therefore, these images contain a variety of background noises replicating the original scenario of the inspection site. The input images have three channels (i.e., RGB channels)

and resolutions of 224x224 pixels, keeping the image dimension 224x224x3. As mentioned in the chapter 4 while splitting the entire dataset, the input dataset is considered to have a larger portion of images while the rest is used for testing purposes. Also, similar to chapter 4 the dataset is divided into three parts: training, validation and testing, with a split ratio of 70%, 20% and 10%, respectively. Table 4.1 from chapter 4 presents the summary of data distribution for train, validation and testing.

For segmentation, apart from the RGB images, a ground truth mask dataset is required as the input dataset for each defect case while keeping the image resolution to 224x224. Ground truth refers to annotations of objects in images from direct observation. Generally, any inspection images of defects contain a variety of objects aside from the defect itself. Therefore, to help the CNN segmentation analysis only focus on defects, the defects' specific area is annotated as defect type and the rest of the objects area is counted as background. The ground truth dataset is the desired output from a CNN algorithm, fed into the model as an input image and expected to predict the same output at the evaluation stage. Figure 5.3 illustrates the example of a ground truth image for crack and spalling.



(a.1) Crack image (a.2) Crack ground truth mask

(b.1) Spalling image (b.2) Spalling ground truth mask

Figure 5.3: Ground truth mask sample for (a) crack and (b) spalling

5.4 CNN Segmentation Model Configuration

Encoder-Decoder models have proven to be the best-performed semantic segmentation model. In the encoder-decoder model, the encoder section focuses on extracting the features from an image. At the end of the model, the decoder part predicts the class of each pixel. This study has considered two types of encoder-decoder models for semantic segmentation: (a) U-net and (b) Pyramid scene parsing network (PSPnet). For semantic segmentation, U-net and PSPnet models are analyzed for four CNN backbone models: VGG19, ResNet50, InceptionV3 and EfficientB3. This subsection presents the details of the segmentation model configuration and the working procedure. For computer computation configuration, the algorithms of these networks are developed using Kears applications [Francois (2015)], and the segmentation models are loaded from the segmentation application. For the coding language, Python is used backend by TensorFlow. After building the CNN-segmentation application, the model simulations are run using Google Collaboratory

5.4.1 Encoder Decoder Model: U-net

U-net [Ronneberger et. al. (2015)] is a type of semantic segmentation method which is based on a Fully connected Neural (FCN) network [Jonathan et. al. (2015)] where numbers increase the feature maps in upsampling part and transmits the image information to higher resolution layers. The name "U-Net" denotes the "U" shape of the network architecture [Figure 5.4]. Like FCN, U-Net is composed of two segments: the encoding component (convolutional and pooling layers) and the decoding component (deconvolutional layers). The base concept of the encoder component is to successfully propagate an input image through convolutional, activation and pooling layers to obtain multilevel feature maps from the image. During this process, the size of the feature map is subsequently reduced, with each layer up to the bottleneck segment of the U-Net. Contrary to the encoder trend, the decoder part aims to recover the lost feature maps of the image through transpose convolution.

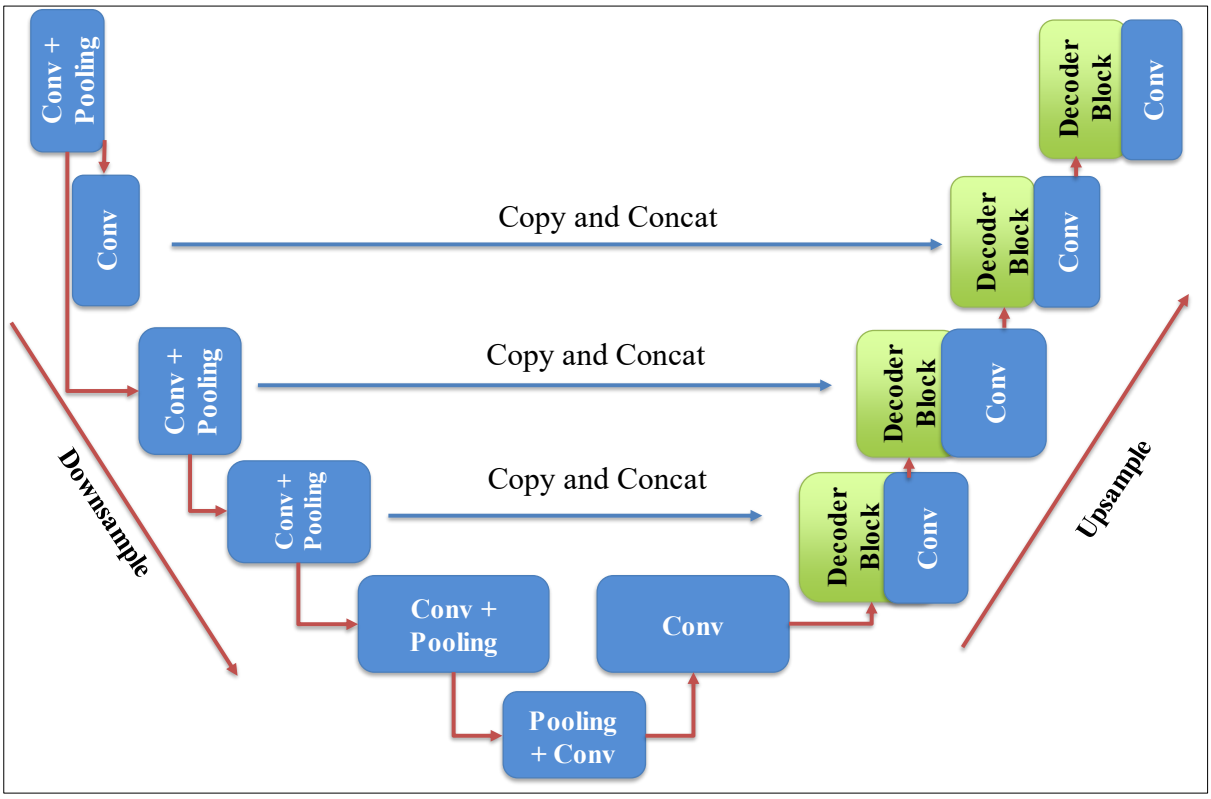


Figure 5.4: Schematic diagram of U-Net architecture

5.4.1.1 U-net: Encoder Architecture

The encoder segment includes two repeated blocks of 3x3 convolutional layers. Each convolutional layer is followed by ReLu activation and batch normalization. Batch normalization helps the CNN model accelerate the convergence and improves the network's performance. These blocks are denoted as ConvBlock. In ConvBlock, a downsampling process is implemented where after the ConvBlock comes to the 2x2 max-pooling layer with a stride size of two. With downsampling, the image's dimensions are halved, and the number of feature maps is doubled. In this study, the encoder network has used four pre-trained backbone models, VGG-19, ResNet-50, InceptionV3 and EfficientNetB3.

5.4.1.2 U-net: Decoder Architecture

In this study, the decoder part of U-Net has two steps: (a) the encoder part- encodes the ground truth image to create a 1/16 feature map, and (b) the decoder part- decodes the 1/16 feature map from the encoder part to yield prediction result. The deconvolutional layer, where the images get upsampled, is followed by a 4x4 transpose convolution layer, referring to increasing the size of the image dimensions and reducing the number of the feature channels by one-fourth. Concatenation is done with a 3x3 convolutional layer, and the model is closed by a 1x1 convolutional layer which generates the final detection. Finally, the decoder segment upsamples the feature channels of the image by five blocks: 1/256, 1/128, 1/64, 1/32 and 1/16, subsequently. Table 5.1 shows the detailed network specification for decoder architecture.

Table 5.1: Details network layers for decoder architecture

Layer name	Layer size
Input	512 X 512 X 1
Conv 1	256 X 256 X16
Conv 2	128 X 128 X 32
Conv 3	64 X 64 X 32
Conv 4	32 X 32 X32
Deconv 1	16 X16 X 32
Deconv 2	32 X 32 X 32
Deconv 3	64 X 64 X 32
Deconv 4	128 X 128 X 16
Deconv 5	256 X 256 X 1

5.4.2 Encoder-Decoder Model: Pyramid Scene Parsing Network (PSPNet)

The base of any semantic segmentation is the scene parsing approach, where the model aims to assign a specific class to each pixel and gives a complete understanding of the image. Using the scene parsing technique, the model predicts the label of an element and gives the geometrical properties, like the location and shape of the element. The overall architecture of the PSPnet is built with the encoder part containing a pyramid pooling layer along with a CNN backbone model and the decoder part that uses the upsampling technique to predict the class of each pixel [Hengshuang et. al. (2016)]. In short, PSPnet is a type of semantic segmentation model that employs a pyramid parsing module that takes advantage of the global context of an image to estimate the local level predictions depending on region-based context aggression.

5.4.2.1 PSP-net: Encoder Architecture

The PSPnet encoder part works with two components: (a) one CNN backbone model with a dilated convolutional layer and (b) a pyramid pooling module. Firstly, the last traditional convolutional

layer of a backbone CNN model is replaced by a dilated convolutional layer, which helps the model upscale the receptive field. In conventional CNN, backbone models lose spatial information with each proceeding convolutional layer. The loss of this spatial information can significantly hamper an image scene's detailed spatial learning process. In some cases of pixel classification, if the desired object's spatial dimension is not dominant rather than the background pixel, the background response can overpower the value of the desired object in a global context. In extreme cases, if the object of interest is lost during the downsampling session, then the upsampling training cannot recover the object anymore. To avoid this situation, the resolution of the network's output is enhanced by replacing the striding layer with a dilation layer [Fisher et. al. (2017)]. In short, dilated convolution is a technique that expands the input image by implementing holes between the sequential elements. This convolutional layer skips pixels while analyzing an image, and skipping some pixels covers a large area of the input image without losing much spatial information.

Along with dilated convolutional layer comes the pyramid pooling module. This is the principal part of the model that helps PSPSnet to obtain the information of the image in a global context, which eventually helps the model classify the pixels in terms of global information of an input. The feature map from the last convolutional layer of the backbone model is pooled at different dimensions and passes through a convolutional layer. After that, the new feature maps are upsampled to match the original size of the input feature maps. Finally, the upsampled feature maps are concatenated with the original feature map from the last backbone convolution layer and passed to the decoder section of the model. By utilizing this technique, the features of different objects are analyzed on different scales hence aggregating the overall context of the pixels in the image. Figure 5.5 illustrates a schematic diagram of how the pyramid pooling module functions. First, the last convolution layer's feature map is pooled into various dimensions. Green, blue, orange and purple represent the feature maps' different sizes, and H & W represents height and weight, respectively. Later, these new feature maps are subsequently passed through a convolutional layer, upsampled and concatenated with the original feature map.

5.4.2.2 PSP-net: Decoder Architecture

After the encoder part, the feature map is inputted into the decoder part and predicts the pixel classes. This decoder part works similarly like the decoder part of U-net. This study uses a simple

8x bilinear upsampling module as the decoder part. The conventionally PSP-net segmentation model only has the encoder part, but a simple upsampling is required to implement the model as a segmentation model. However, this upsampling does not have any learnable perimeters; it just helps the model to classify the pixel output.

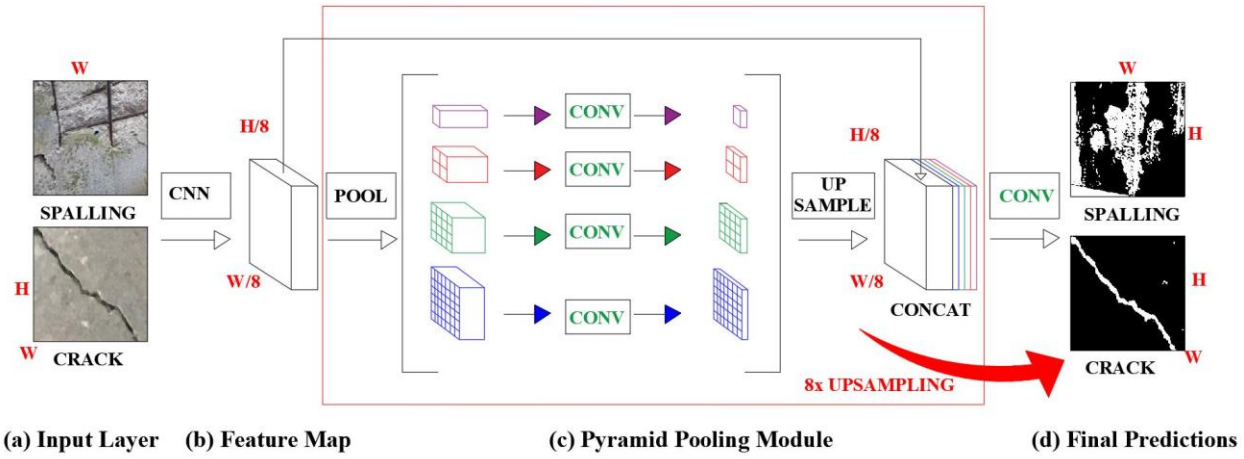


Figure 5.5: Schematic diagram of PSP-net

5.5 Sensitivity Analysis of Hyper-parameters

As mentioned in chapter 3, pre-trained ImageNet weights are considered to start the training process of CNN segmentation models, followed by a continuous trial-error method to reach the optimized point of hyper-parameters. Then, a sensitivity analysis is done to train the hyper-parameters and determine the optimized hyper-parameters for models. In this study, a few hyper-parameters, listed as batch size, activation function, optimization function, loss function and learning rates, are considered for sensitivity analysis, as these hyper-parameters are considered to have a greater influence on the model training process. The details of these hyper-parameters are presented in Table 5.2.

Table 5.2: Details of hyper-parameters

Name of Parameters	Value of Parameters
Batch Size (CNN-classifiers)	10
Learning rate (CNN-classifiers)	0.1, 0.01, 0.0001
Optimization function	SGD, ADAM
Activation function	ReLu
Evaluation metrics threshold	0.5
Loss function (CNN-classifiers)	Dice Loss + Focal Loss
Pre-trained weights	ImageNet
Callbacks	Early-stopping
Epoch	100

To achieve the best output result, the values of hyper-parameters- batch size, the size of the epoch, optimization function, learning rate and loss functions for CNN-classifiers are designated after carefully analyzing the learning process. As Goodfellow (2016) mentioned, these parameters are the most important parameters that guide the models toward optimized convergence. Similar to the classification of CNN models for segmentation, a batch size of 10 is considered, and the models are trained for 100 epochs. As the same dataset is used for classification and segmentation purposes, this study has decided to keep the hyper-parameters similar to classification CNN models. To finalize the epoch size, two functions called early-stopping and the reduced learning rate is applied in these models to reduce the computational cost (time and computer memory) of the models. For the training purpose, this study has used an epoch value of 100, as all the segmentation models reached their optimized point within 100 epochs. Also, it is observed that the model is trained with a batch of 10 images with minimal computational cost and obtained optimized performance. Moreover, for the activation function, ReLu is selected and proved to impact the model's training performance positively. According to some previous studies, it has

been proven that SGD and ADAM optimizers are two of the best optimizers for segmentation purposes; hence this study has decided to follow the statements and used these optimizers to train the models [Zaheer and Shaziya (2019), Desai (2020)]. The learning rate defines the number of weights that will be updated in each step during training and also navigates the degree of convergence frequency of a model to adapt to the problem. This study has explored two different optimization functions: SGD and ADAM along with three different learning rates, 0.1, 0.01 and 0.0001, for each of the two segmentation models, i.e., U-Net and PSP-net. These models separately utilized four different backbone models, i.e., VGG-19, ResNet-50, InceptionV3 and EfficientNetB3 and summarized the results in sections 5.6 and 5.7 for crack segmentation and spalling segmentation, respectively. Finally, the best hyper-parameters values are selected after evaluating the performance of the trained models based on the evaluation metrics. For the hyper-parameters training, this study has finalized.

5.6 Result and Discussion: Crack Segmentation

This section is divided into two subsections where 5.6.1 discusses the results of the crack segmentation model using the U-Net encoder-decoder model, and subsection 5.6.2 illustrates the output of the PSP-net model. Both these models are analyzed based on four backbone models, and their results are summarized in those subsections.

5.6.1 Crack Segmentation: U-Net model

The U-Net model is analyzed with four backbone models- VGG-19, ResNet50, IncpetionV3 and EfficientNetB3 considering two types of optimization functions, SGD and ADAM. Later, these models are evaluated for three different learning rates, i.e., 0.1, 0.01 and 0.0001. In total, twenty-four models are analyzed, and the summarized results are presented in Table 5.3. Finally, all the trained models are tested with the testing dataset and evaluated with four evaluation metrics IoU, F1-score, precision and recall.

Table 5.3 summarizes the outputs of twenty-four models for crack segmentation utilizing U-Net, and it is highlighted that most of the best performance is achieved with a learning rate of 0.01 and optimizer ADAM, meaning the models have identified most of the crack features with a limited computational cost with the combination of these two parameters. With a learning rate of 0.1, the training process has skipped many learning features and converged faster towards a suboptimal

position. In contrast, a learning rate of 0.0001 has taken a slow pace to update the models' weights and increase the computational cost without significantly improving the model's performance. Also, between the two optimizers, ADAM has helped the models to perform better, eventually consenting to the outcome of the research performed by [Kingma and Ba (2014) and Yaqub et. al. (2020)]. Yaqub et. al. (2020) studied the state-of-the-art of nine different CNN optimizers for the segmentation process with an extensive experimental analysis and concluded that ADAM is the best optimizer among all the optimizers, including the SGD optimizer. Also, according to Choi et. al. (2019), the best-tuned model adaptive gradient methods always perform better than the gradient descent optimizers, as the fine-tuned models have better generalization potential for the SGD. Also, ADAM has the unique feature of using momentum and adaptive learning rates to converge faster than any other optimizer.

Another finding from Table 5.3 is that EfficientNetB3 based U-Net model has outranked all the other models with the highest evaluation values (marked in red). The model has an IoU score of 91.98%, F1-score of 95.66%, a precision score of 94.73% and a recall score of 97.59%, with the learning rate of 0.01 and optimizer ADAM. Conventionally, with the certain architectural build of EfficientNetB3, it is supposed to achieve higher accuracy and better efficiency by reducing parameter size over the existing CNN models. The EfficentNet model utilizes a specific technique called compound efficient component, which focuses on scaling up the model is a simple but effective manner [Tan and Le (2019]. The compound technique expands the dimension scale uniformly over a fixed set of scaling coefficients instead of randomly expanding the width, depth and resolution. Also, the training process is faster than other CNN models to reduce computational cost.

Table 5.3 Evaluation summary of U-Net based crack segmentation models' results depending on different optimizers and learning rate

Evaluation Metrics	Learning rate	U-Net *EfficientnetB3*		U-Net *InceptionV3*		U-Net *ResNet 50*		U-Net *VGG 19*	
		SGD	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***
IoU	0.1	87.81	80.64	86.61	86.59	89.50	87.61	86.81	80.64
	***0.01**	89.40	***91.98**	88.81	91.83	91.91	89.55	87.35	89.54
	0.0001	88.67	90.05	88.83	91.46	90.85	82.02	83.59	91.27
F1-score	0.1	92.75	88.17	92.29	92.19	94.11	92.89	92.35	88.17
	***0.01**	94.09	***95.66**	93.66	95.53	95.62	94.16	92.76	94.14
	0.0001	93.66	94.47	93.62	95.32	94.58	89.42	90.42	95.21
Precision	0.1	93.38	80.64	91.97	95.31	92.86	92.99	89.95	80.64
	***0.01**	93.90	***94.73**	95.04	93.82	93.42	94.54	95.81	95.00
	0.0001	92.84	93.86	94.16	94.42	94.58	85.73	92.65	93.97
Recall	0.1	93.72	100	93.14	90.16	95.54	93.45	95.28	100
	***0.01**	94.57	***97.59**	92.74	97.35	97.97	93.94	90.41	93.47
	0.0001	94.42	94.38	93.54	96.25	95.36	94.94	88.55	96.52

*Note: *Best performance*

Figure 5.6 illustrates the output results of four different U-net models for two optimizers using the learning rate of 0.01. It is prominent from the graph that most of the models obtained the best efficient result using the ADAM optimizer except for the ResNet50 backbone model. Also, for all the evaluation metrics EfficientB3 based U-Net has achieved the performance. InceptionV3 based PSP-Net has the second best performance with an IoU of 89.45%, F1-score of 93.7, precision of and recall of 97.4%, followed by ResNet50 and VGG19 based PSP-Net model.

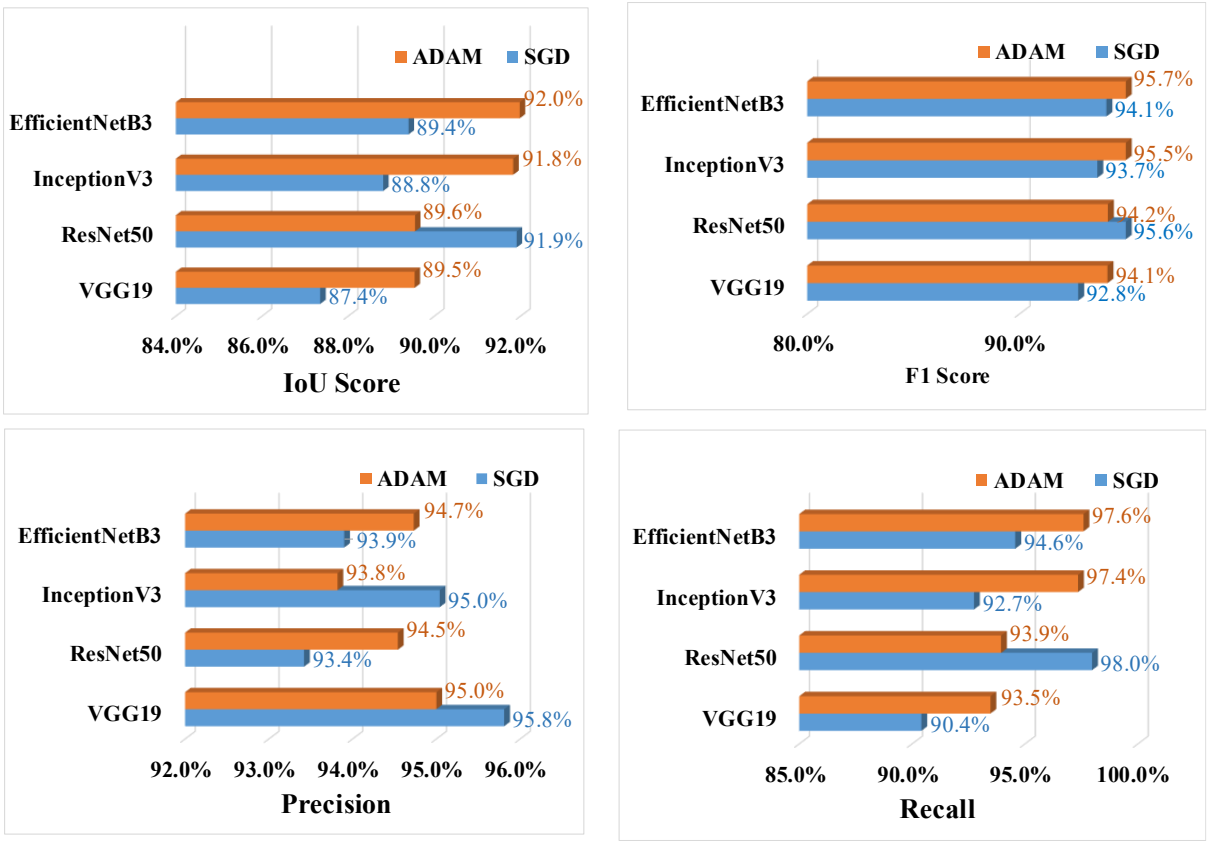


Figure 5.6 Crack segmentation: U-Net- comparison of the evaluation metrics based on SGD and ADAM optimizers for learning rate 0.01

According to Figure 5.7 EfficientB3 based U-Net model with a learning rate of 0.01 has obtained its optimized performance only after 40 epochs. Also, among all the learning rates, 0.01 required most epochs to reach the optimizing point indicating that with a learning rate of 0.01, the trainable model learns the crack features more deeply.

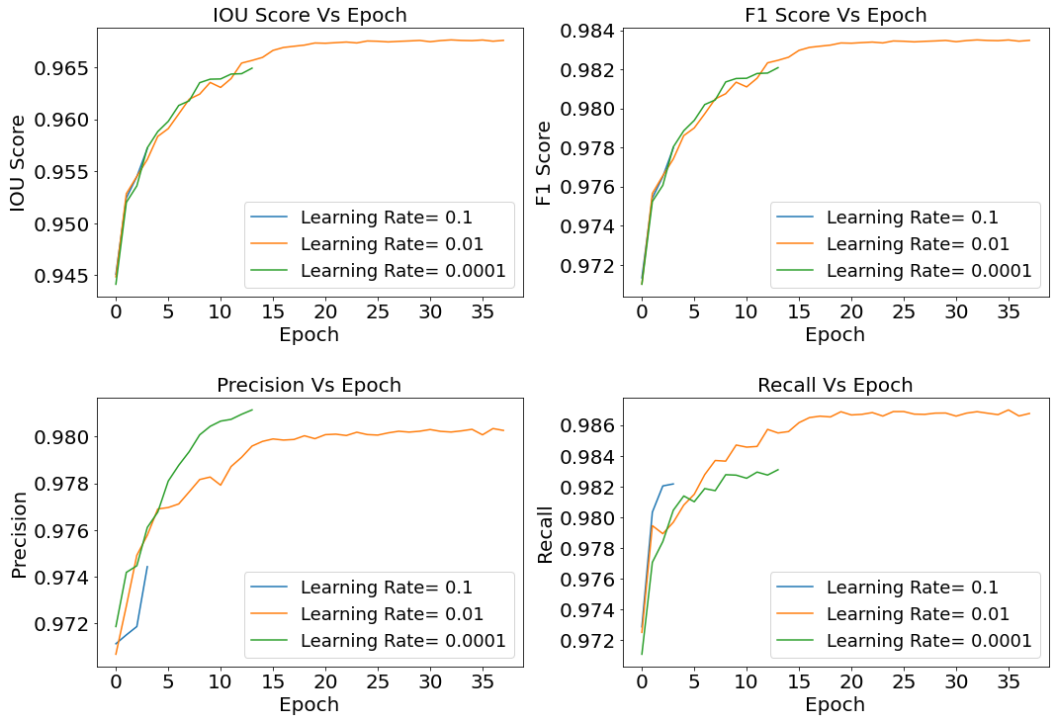


Figure 5.7: Crack segmentation: U-Net- comparison of the evaluation metrics based on ADAM optimizers and learning rate 0.1, 0.01 and 0.0001

5.6.2 Crack Segmentation: PSP-Net model

Like U-net crack segmentation models, PSP-net is also analyzed with four backbone models- VGG19, ResNet50, InceptionV3 and EffiecientNetB3. These models are evaluated for optimizers SGD and ADAM with three learning rates of 0.1,0.01 and 0.0001. Altogether twenty-four models are analyzed and summarized in Table 5.4.

From Table 5.4, EfficientNetB3 based PSP-Net has outranked the other CNN models in crack segmentation with an IoU value of 89.55%, F1-Score of 94.13%, precision value of 92.56% and recall value of 95.94%. This model has achieved its best performance using optimizer ADAM and a learning rate of 0.01. As discussed earlier, usually, an ADAM optimizer has a positive impact on model's performance better than SGD.

Table 5.4 Evaluation summary of PSP-Net based crack segmentation models' results depending on different optimizers and learning rate

Evaluation Metrics	Learning rate	PSP-Net *EfficientnetB3*		PSP-Net *InceptionV3*		PSP-Net *ResNet 50*		PSP-Net *VGG 19*	
		SGD	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***
IoU	0.1	81.56	85.39	81.59	84.43	74.56	80.82	72.39	78.49
	***0.01**	86.03	***89.55**	84.33	87.64	76	83.2	74.4	80.7
	0.0001	85.65	88.59	83.41	86.75	77.01	82.96	73.69	79.58
F1-score	0.1	89.95	91.22	88.76	91.38	83.74	83.85	83.94	83.12
	***0.01**	91.30	***94.13**	90.98	92.83	85.6	86	85.3	88.1
	0.0001	90.89	93.97	89.35	91.45	84.59	85.38	84.58	87.65
Precision	0.1	91.88	90.10	85.38	90.33	79.58	85.49	81.24	89.58
	***0.01**	93.07	***92.56**	89.67	91.74	82.2	88.6	83.4	92.2
	0.0001	92.03	92.45	88.96	91.57	82.59	87.96	82.87	91.97
Recall	0.1	90.39	91.74	91.24	90.84	82.38	88.23	81.64	85.38
	***0.01**	92.31	***95.94**	93.10	91.62	85.3	90.4	84.7	88.4
	0.0001	91.76	94.39	92.40	90.88	84.79	89.54	83.67	87.57

*Note: *Best performance*

Figure 5.8 presents the performance of the four backbone models against two optimizers with a learning rate of 0.01. Similar to U-Net crack segmentation, the ADAM optimizer has helped all models gain better performance than SGD. Also, among all the models, EfficientNetB3 has the best evaluation values.

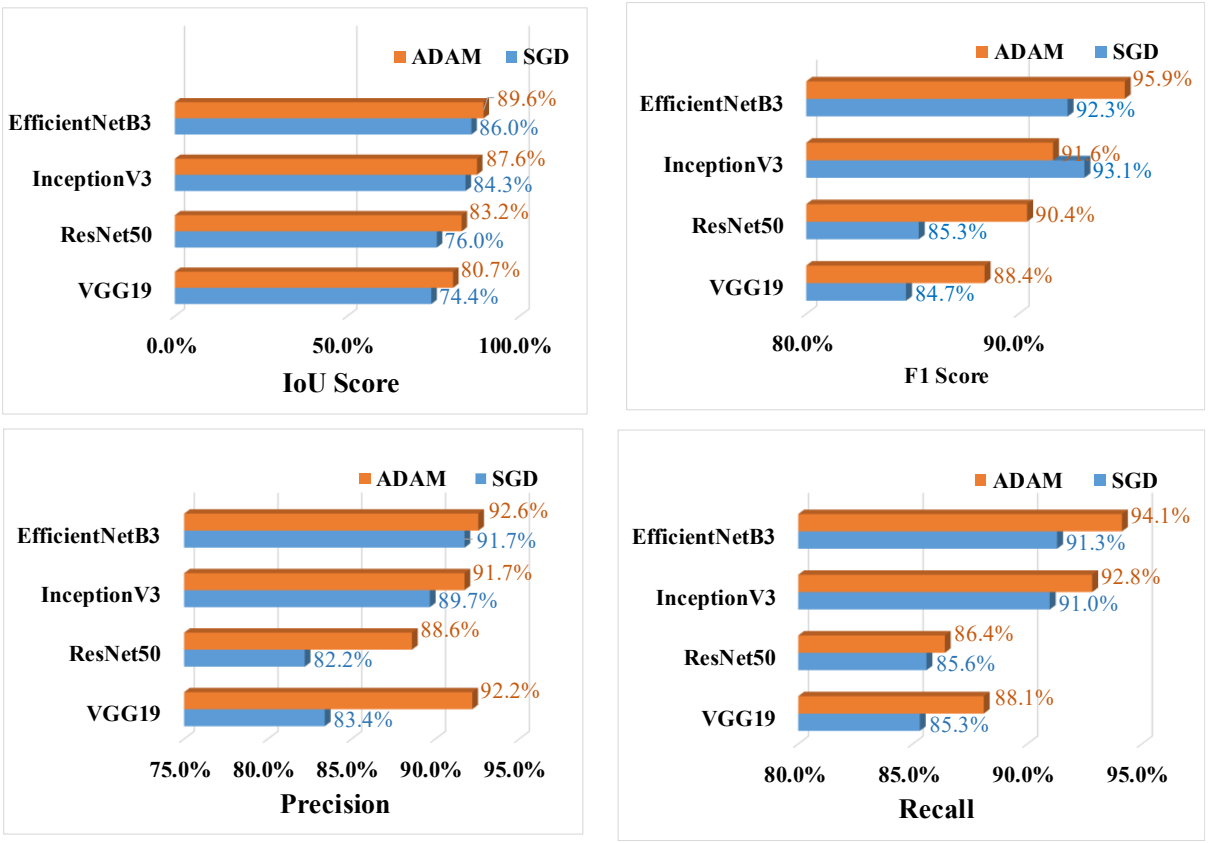


Figure 5.8: Crack segmentation: PSP-Net- comparison of the evaluation metrics based on SGD and ADAM optimizers and learning rate 0.01

Figure 5.9 illustrates the changes in evaluation metrics value over the training process for EfficienNetB3 based PSP-Net crack segmentation model. With the learning rate of 0.01 and ADAM optimizer, this model has achieved its optimized evaluation at epoch 40. Also, in the training process for all the evaluation metrics, IoU, F1-score, precision and recall, the model gained more than 95% performance.

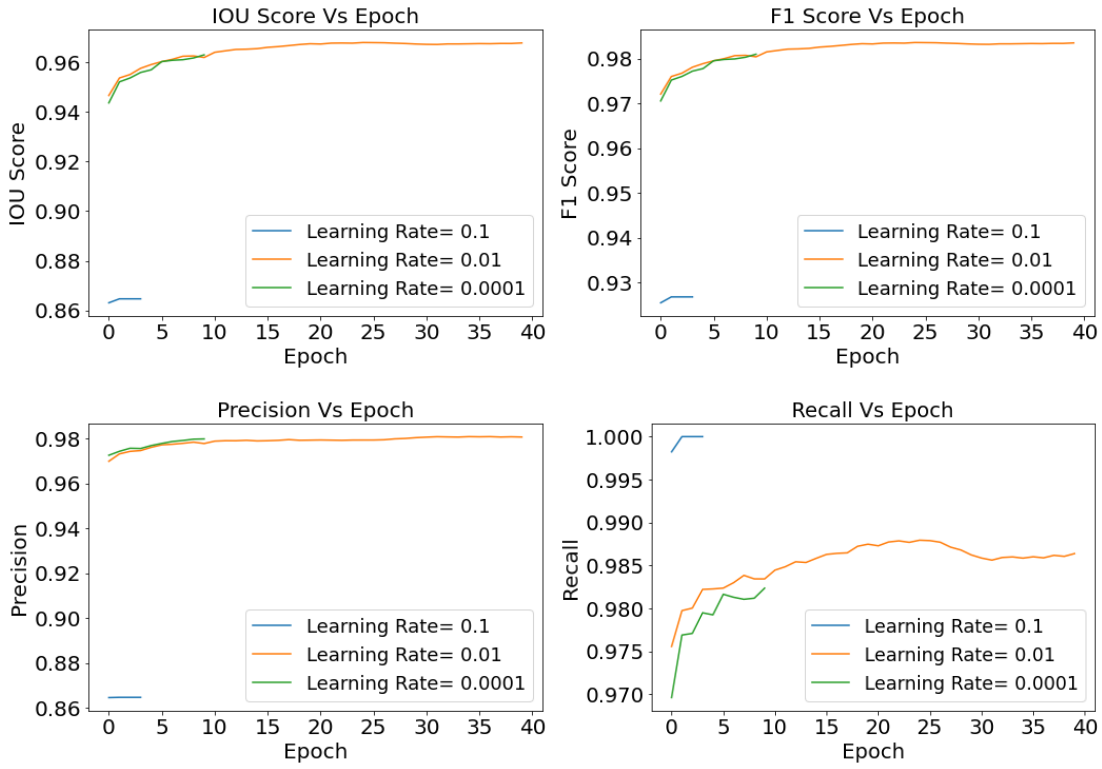


Figure 5.9: Crack segmentation: PSP-Net- comparison of the evaluation metrics based on ADAM optimizers and learning rate 0.1, 0.01 and 0.0001

After analyzing both the U-Net and PSP-Net models, it is evident that the EfficientNetB3 backbone model has achieved the best performance in the crack semantic segmentation model. Also, the ADAM optimizer has shown superiority over other optimizers. A learning rate of 0.01 has proven to be the most efficient learning rate for weight update in each training step. According to Table 5.3 and Table 5.4, both U-Net and PSP-Net model has obtained quite a similar performance in crack segmentation with the EfficientNetB3 model, which glorifies the statement that encoder-decoder models have higher accuracy in crack semantic segmentation.

5.7 Result and Discussion: Spalling Segmentation

This section is also divided into two sub-sections where 5.7.1 discusses the results of the spalling segmentation model using the U-Net encoder-decoder model, and sub-section 5.7.2 shows the output of the PSP-net model. Four backbone models are implemented in both cases, and their results are summarized in those sub-sections.

5.7.1 Spalling Segmentation: U-Net model

For spalling segmentation U-Net model is analyzed with four different backbone models- VGG-19, ResNet50, InceptionV3 and EfficientNetb3. These models are also evaluated for different learning rates, i.e., 0.1, 0.01, 0.0001 and two types of optimizers, the same as crack segmentation models.

Table 5.5 presents the performance of the U-Net models based on four backbone models, three learning rates and two optimizers. After analyzing all twenty-four models, the InceptionV3 based U-Net model has the best achievement on spalling segmentation. According to the InceptionV3 model's architecture, the height and depth of the trainable layers are very deep, which can eventually prompt the model's performance. Upon introducing the model for the first time by Szegedy et. al. (2015), this model was considered one of the noteworthy CNN architectures for better performance with image processing. InceptionV3 has reached its optimized situation with the implementation of the ADAM optimizer and a learning rate of 0.01. This model can predict the spalling segmentation on the images with an IoU value of 81.30%, F1-score of 89.43%, the precision value of 86.40% and recall value of 92.87%. As discussed in the earlier section, ADAM and a learning rate of 0.01 are expected to help the model reach the optimized convergence point.

Table 5.5 Evaluation summary of U-Net based spalling segmentation models' results depending on different optimizers and learning rate

Evaluation Metrics	Learning rate	U-Net *EfficientnetB3*		U-Net *InceptionV3*		U-Net *ResNet 50*		U-Net *VGG 19*	
		SGD	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***
IoU	0.1	72.33	50.30	72.03	74.69	63.66	70.63	65.38	56.10
	***0.01**	77.16	78.10	77.31	***81.30**	80.78	74.68	74.98	81.28
	0.0001	40.01	79.53	36.37	79.90	81.86	42.19	72.32	81.46
F1-score	0.1	83.16	49.05	83.03	84.88	76.37	81.93	78.24	69.55
	***0.01**	86.73	87.30	86.75	***89.43**	89.13	84.95	85.10	89.42
	0.0001	55.39	88.30	52.35	88.52	89.79	58.22	83.24	89.56
Precision	0.1	88.28	77	85.33	87.45	95.66	88.49	79.21	56.20
	***0.01**	87.70	88.58	87.20	***86.40**	82.02	84.86	85.94	88.38
	0.0001	59.65	85.65	56.86	86.07	86.64	58.43	88.32	83.73
Recall	0.1	79.66	51.24	82.20	82.87	65.53	77.48	78.03	66.34
	***0.01**	85.93	86.27	86.57	***92.87**	97.92	83.34	82.61	93.31
	0.0001	56.00	91.24	51.86	91.29	93.22	62.08	79.37	96.48

*Note: *Best performance*

Figure 5.10 compares evaluation metrics for U-Net models for spalling detection in the context of two different optimization functions. As per the graphical representation, most models gained efficient prediction performance with the help of an ADAM optimizer, except for the ResNet50 model. This summarization clearly shows that InceptionV3 is the best performed model.

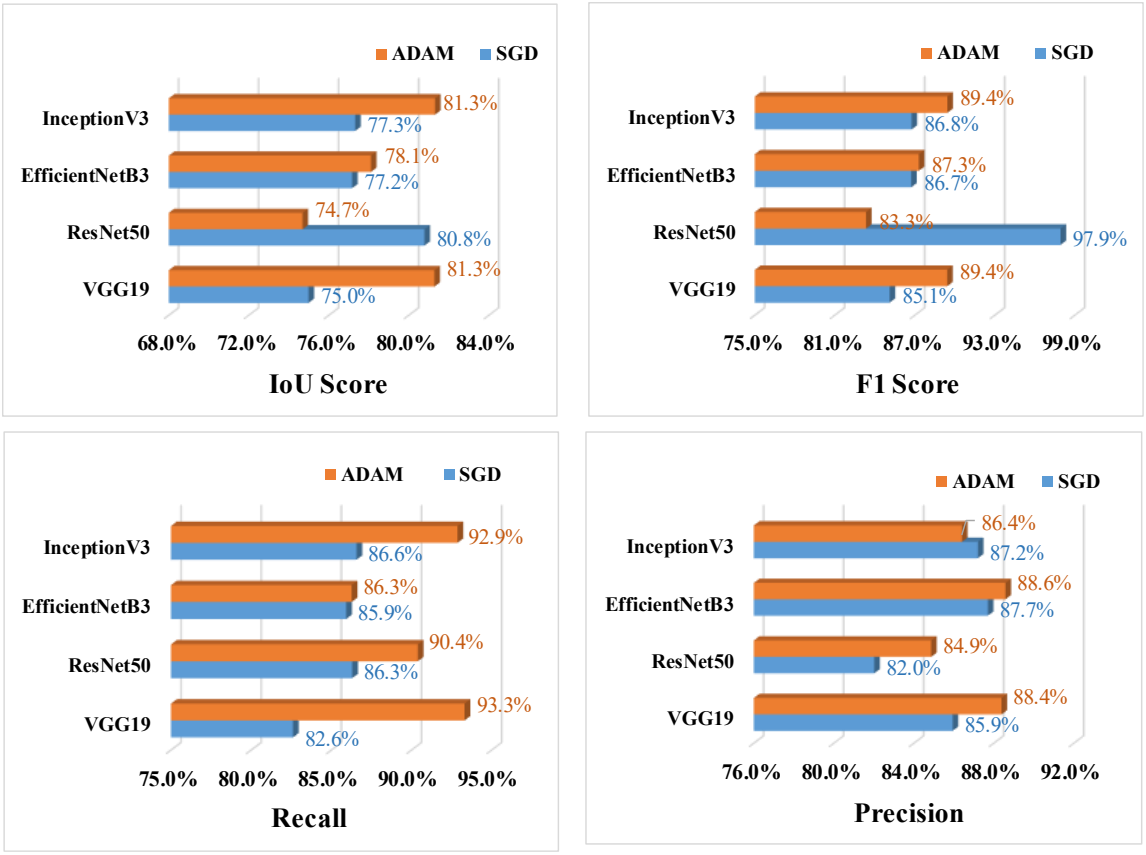


Figure 5.10: Spalling segmentation**:** U-Net- comparison of the evaluation metrics based on SGD and ADAM optimizers and learning rate 0.01

Figure 5.11 shows the training accuracy of evaluation metrics over three different learning rates. It is prominent from these graphs that the model training outputs remained in straight lines over the epochs, meaning the model has reached its' convergence point too fast in the training process. One of the possible reasons behind this scenario can be that when preparing the ground truth mask of the original dataset, this study used a generalized annotation system, which eventually affected the exact outlining of the spalling area. However, even though this study has used a generalized annotation system, the final results for spalling predictions are satisfactory.

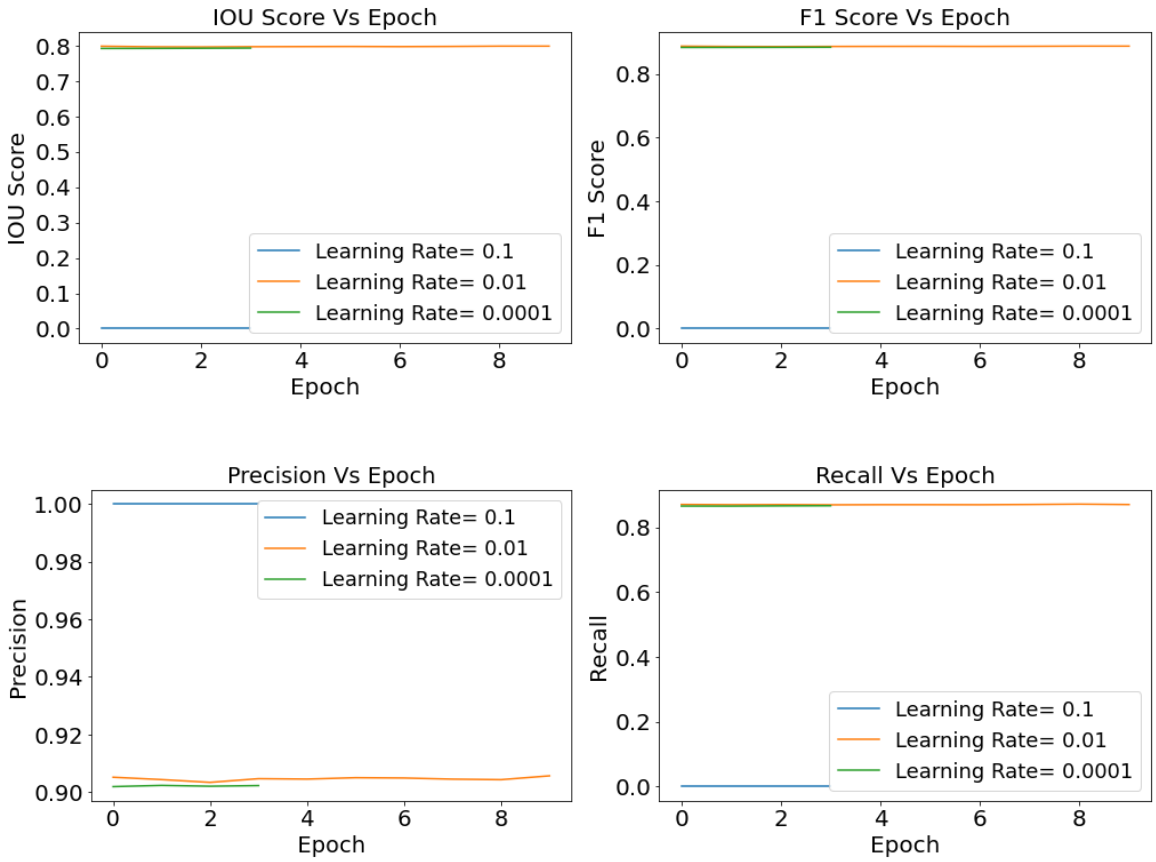


Figure 5.11: Spalling segmentation: U-Net InceptionV3- comparison of the evaluation metrics based on ADAM optimizers and learning rate 0.1, 0.01 and 0.0001

5.7.2 Spalling Segmentation: PSP-Net model

Similar to U-Net based spalling segmentation, PSP-Net has used four backbone models, two optimizers, and three learning rates to compare models' output and finalize the best prediction model. Table 5.6 shows that PSP-Net has obtained its best performance with the backbone model EfficientNetB3, once aging, proving the superiority of the EfficientNetB3 model over the other models. Also, similar to all the previous cases ADAM and a learning rate of 0.01 has exhibited the most affirmative impact for spalling segmentation prediction. Moreover, this model has the best IoU, F1-score, precision and recall values of 75.30%, 85.28%, 86.88% and 84.41, respectively.

Table 5.6 Evaluation summary of PSP-Net based spalling segmentation models' results depending on different optimizers and learning rate

Evaluation Metrics	Learning rate	PSP-Net ***EfficientnetB3***		PSP-Net ***InceptionV3***		PSP-Net ***ResNet 50***		PSP-Net ***VGG 19***	
		SGD	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***	***SGD***	***ADAM***
IoU	0.1	68.34	69.68	62.59	61.24	59.32	71.84	58.48	64.38
	***0.01**	70.78	***75.30**	66.89	65.09	60.66	74.67	63.94	68.76
	0.0001	70.43	74.31	63.91	64.91	61.24	73.72	61.38	65.10
F1-score	0.1	79.75	79.47	71.34	73.80	69.47	80.46	72.40	76.61
	***0.01**	82.25	***85.28**	79.10	77.51	73.60	84.84	77.13	80.63
	0.0001	81.86	84.83	74.49	76.42	71.56	82.35	76.39	80.54
Precision	0.1	81.05	80.25	76.29	75.39	72.38	85.66	77.05	85.98
	***0.01**	84.10	***86.88**	81.64	77.43	78.25	88.45	79.28	86.9
	0.0001	85.63	85.44	79.50	78.23	76.77	87.30	78.98	87.20
Recall	0.1	80.88	81.40	70.94	77.86	69.58	79.66	73.44	74.23
	***0.01**	82.08	***84.41**	78.90	81.87	72.97	82.31	75.89	76.28
	0.0001	81.37	84.34	77.48	81.36	72.45	81.98	74.09	75.20

*Note: *Best performance*

Based on Figure 5.12, all the PSP-Net models have obtained their optimized performance with the ADAM optimizer agreeing with the statement that ADAM is one of the best optimizers for the segmentation process. EfficeintNetB3 model has the best accuracy with ADAM, followed by InceptionV3, ResNet50 and VGG-19. Figure 5.13 represents the variation of different evaluation metrics of EfficientNetB3 based PSP-Net over the training process for different learning rates. It indicates that with a learning rate of 0.01 model has the most epochs and best numerical

performance. Both these graphs are presented to support the statement that ADAM is the best optimizer and a learning rate of 0.01 is the most effective learning step for this model.

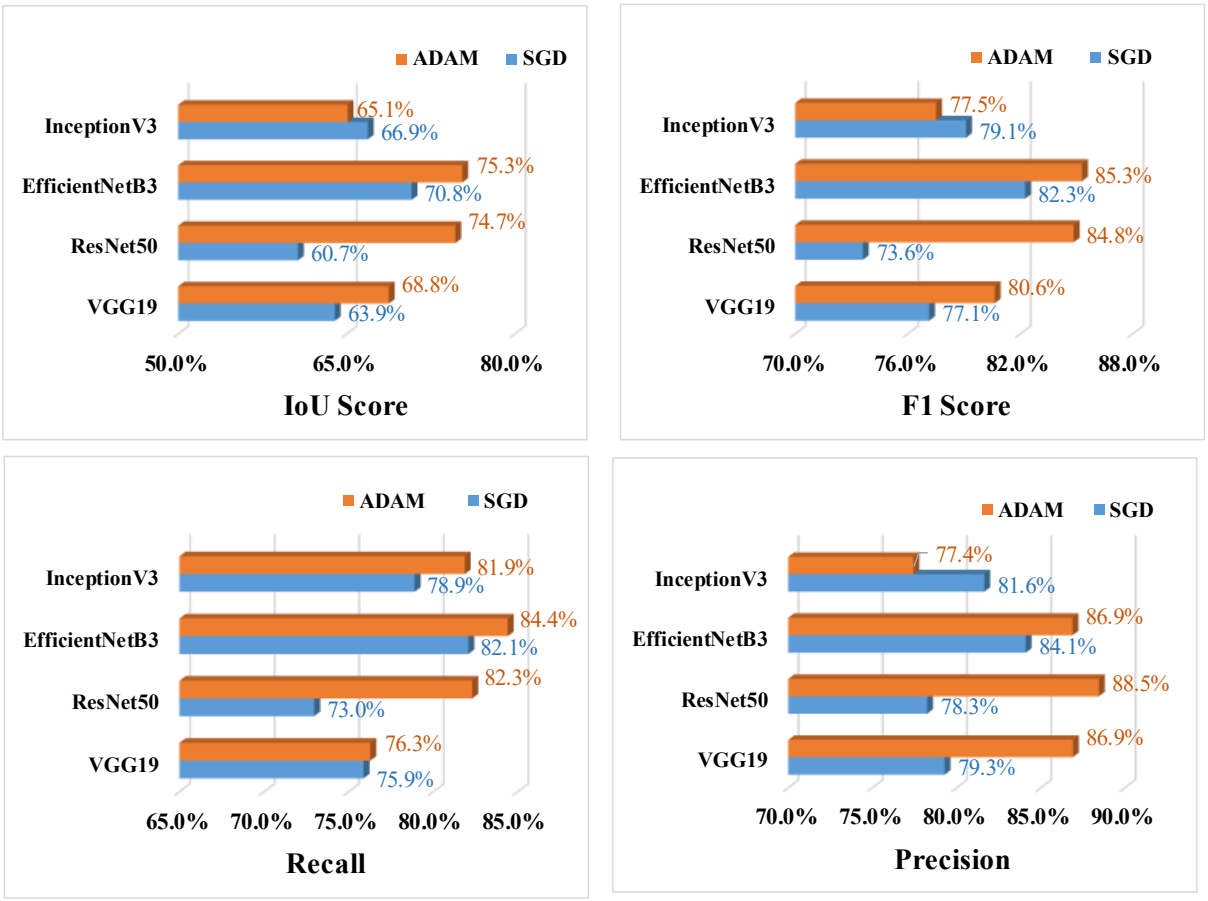


Figure 5.12: Spalling segmentation: PSP-Net- comparison of the evaluation metrics based on SGD and ADAM optimizers and learning rate 0.01

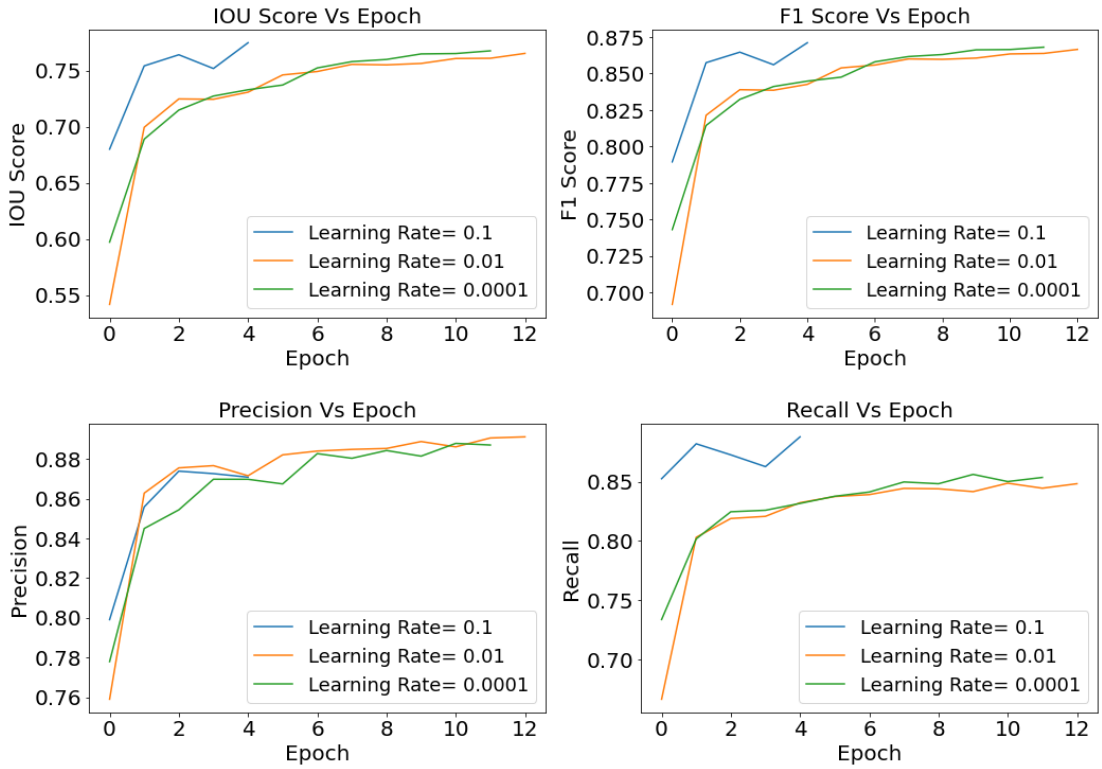


Figure 5.13: Spalling segmentation: **P**SP-Net EfficientNetB3- comparison of the evaluation metrics based on ADAM optimizers and learning rate 0.1, 0.01 and 0.0001

After analyzing both the U-Net and PSP-Net models, it can be concluded that InceptionV3 based U-Net has performed better than any other models for spalling segmentation. On the other hand, EfficientNetB3 based PSP-Net has the best second performance in spalling segmentation with very close evaluation metrics values to InceptonV3 based U-Net model. This eventually justifies that encoder-decoder models are highly profound at spalling segmentation.

5.8 Crack and Spalling Segmentation Prediction

Figure 5.14 illustrates some of the sample images of crack prediction done by the best performed model for both U-Net and PSP-Net models separately. The first column shows the original images, and the second column shows the ground truth mask dataset. The rest of the two columns exemplifies the prediction done by EfficientNetB3 based, U-Net model and PSP-net model, respectively. From the graphs, it is prominent that the crack area prediction quality is relatively similar for both models, with a slightly better performance with U-Net.

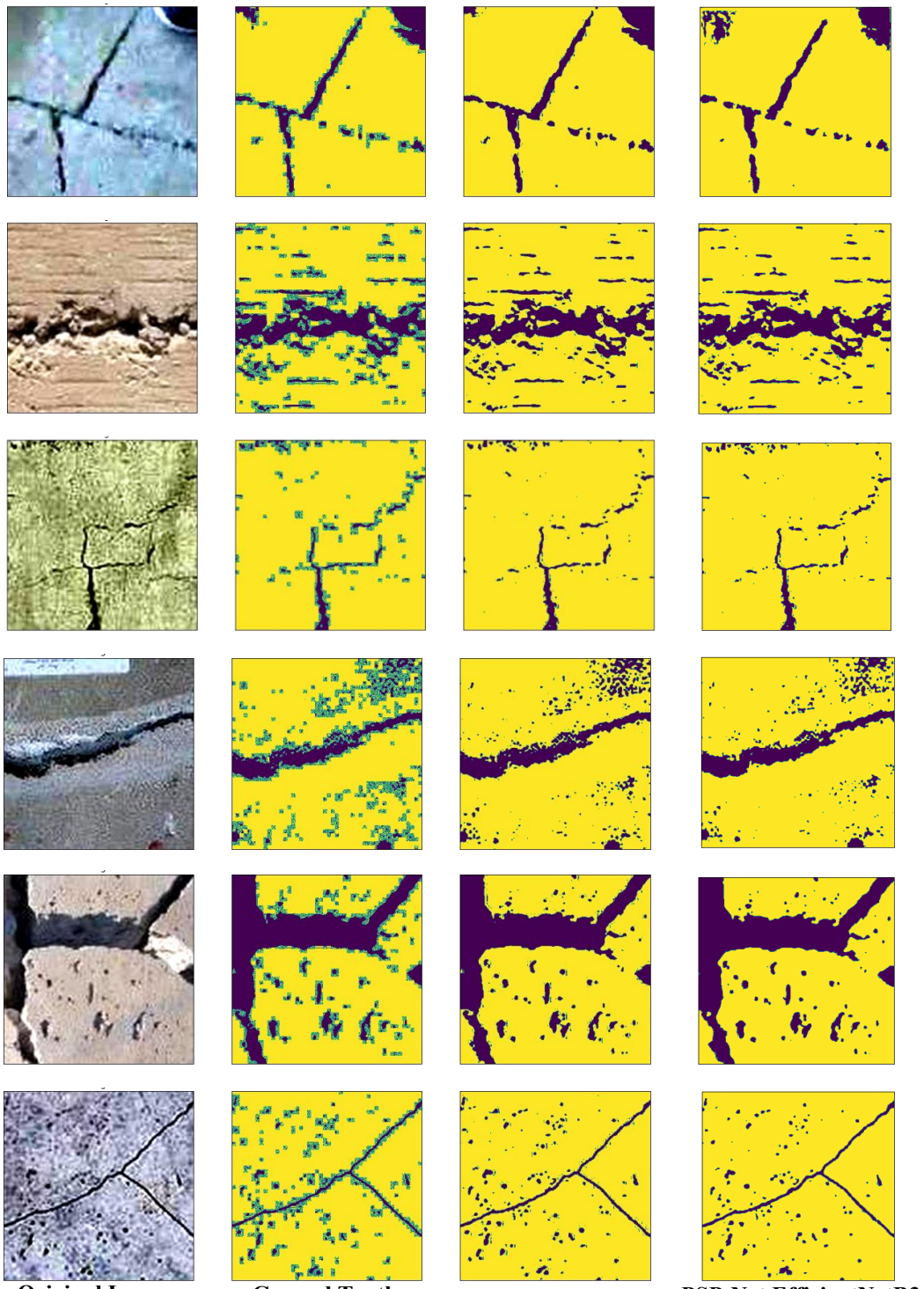


Figure 5.14: Semantic segmentation of crack detection using best two models

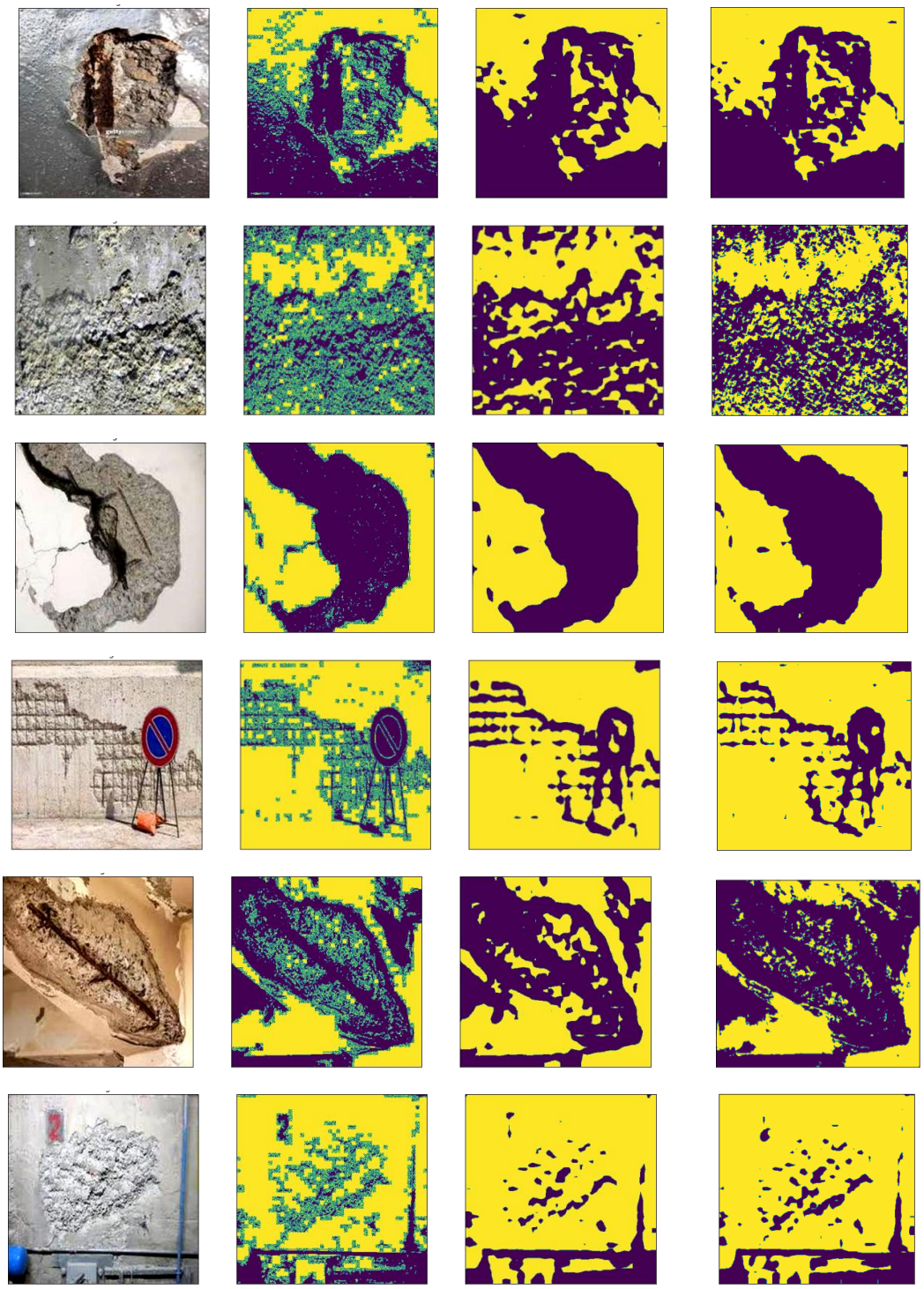


Figure 5.15: Semantic segmentation of spalling detection using best two models

5.9 Summary

This chapter explores the potentiality of two different encoder-decoder models, i.e., U-Net and PSP-Net, as CNN-segmentation to predict defects area on the images. Later, these two models are analyzed with four different backbone models, i.e., VGG-19, ResNet50, Inception and EfficientNetB3. The semantic segmentation process focuses on two goals: a) conducting CNN-segmentation for crack and spalling with U-Net and PSP-Net models and b) training the models with different learnable hyper-parameters types to obtain the best output from the CNN segmentation models. This section has used a dataset of crack and spalling consisting of 4082 and 1100 images, respectively. After finalizing the dataset, the semantic segmentation models are analyzed for different types of hyper-parameters and evaluated based on their prediction potentiality. The highlights of this study can be summarized as follows:

a) This study has performed a detailed sensitivity analysis of semantic segmentation for both crack and spalling cases by combining different learnable and optimization parameters, i.e., encoder backbone model, optimizers and learning rates with two different encoder-decoder model- U-Net and PSP-Net. As a result, SGD and ADAM are the two optimization functions applied in CNN-segmentation algorithms while implementing three different learning rates 0.1, 0.01 and 0.0001.

b) In the case of crack segmentation with U-Net, EfficientNetB3 based model have outranked all the other segmentation models with IoU, F1-score, precision and recall of 91.98%, 95.66%, 94.73% and 97.59%, respectively. In this case, optimizer ADAM and learning rate of 0.01 has enlightened the learning process of the CNN segmentation model towards an optimized convergence

c) After the crack segmentation analysis with PSP-Net, the comparative analysis has revealed that EfficientNetB3 backend PSP-Net has the significant potential for crack area segmentation. Furthermore, similar to U-Net analysis, this model obtained the best execution with ADAM and a learning rate of 0.01.

d) For spalling segmentation InceptionV3 based U-net model has gained the highest rank for spalling area detection based on the evaluation metrics' numerical values- IoU value of 81.30%, F1-score of 89.43%, precision value of 86.40% and recall value of 92.87%. However, with PSP-Net EfficienNetB3 functioned as the best model. In both these cases,

ADAM and a learning rate of 0.01 assisted the models in reaching their utmost output accuracy.

e) Even though U-Net has the upper hand in the crack and spalling segmentation, it is also to mention that PSP-Net models also have obtained a performance adjacent to U-Net models, which certainly glorifies the overall architecture of encoder-decoder models.

f) In all the cases, ADAM is found to be the best optimizer which supports the statement of previous studies. Also, with a learning rate of 0.01, the training process has achieved its' optimized output without losing much information.

Chapter 6 Conclusion and Future Works

6.1 Introduction

This thesis planned on implementing deep learning methods for automatic damage detection on concrete surface. First it presented a comprehensive summary of the previous researches conducted to explore the compute vision technologies in damage identification and detection. On implementation stage this study used various CNN models in context of transfer learning methods to decide on the best performed model for multiple defects identification and segmentation. Also an extensive sensitivity analysis for hyper-parameters were performed to track the optimal condition for the successful automatic damage detection model.

6.2 Core Contributions

The outcomes of this research work are expected to expedite future research toward optimizing the CNN models to develop an automatic damage detection process with real-world application. Also, this thesis has worked on multiple damage detection- concrete crack and concrete spalling. To achieve the ultimate goal this study has focused working on few concepts which supposedly improves the model performance significantly and the core contributions of this study are as follows:

In chapter 2, a detailed literature review of previous developments in computer-vision based damage detection are presented and the findings of this chapter have helped shaping this study. To summarize-

- The accuracy of any trained CNN model highly depends on the size and quality of image dataset.
- Transfer learning can aid in initializing a CNN model analysis and reach the optimized training condition by reducing the model's computational cost.
- Deep learning based models are competent of multiple object detection at a time and can be utilized for several damage detections of structures.

Based on these findings this study has developed a work procedure which are discussed in next chapters.

In chapter 3, this study has discussed the base methodology of a CNN model and specified the CNN models used for transfer learning-

- Selection of various CNN models based on their learning depth and optimized learning process.
- Identified the appropriate evaluation metrics to evaluate the trained CNN model and understand the performance of the model at prediction stage.

This study has worked in two steps- first classifying the damage types and finally segmentation of damage area. In chapter 4 a detailed research method and CNN model outputs for defects classification is presented-

- Created a comparative dataset collecting images from various resources, such as- original damage inspection reports, open-source online resources to imitate the real structural site conditions. To the author's best knowledge, this is one of the largest datasets of both concrete defects without applying any image augmentation process
- Conducted multiple sensitivity analysis for different hyper-parameters- learning rate (0.0001, 0.001, 0.1) and optimization function (SGD and RMSprop).
- Implemented multiple CNN models, such as- VGG-19, ResNet50, MobileNetV2, Xception and InceptionV3 to make a comparison of the models and decide on the best performed model.

In chapter 5, this study introduced the semantic segmentation method for defects area recognition-

- Implemented two different types of Encoder-Decoder model- U-Net and PSP-Net, and analyzed these models for four different base CNN models- VGG-19, ResNet50, InceptionV3 and EfficienNetB3.
- Conducted multiple sensitivity analysis for different hyper-parameters- learning rate (0.0001, 0.01, 0.1) and optimization function (SGD and ADAM).

6.3 Conclusions

6.3.1 Defects Classification

For defects classification this study had two goals- a) conduct CNN classification for multi-class defects- crack and spalling, and b) train the model with different types and values of hyper-parameters to obtain the best output from the CNN classifiers. To achieve the first goal, this study has collected a dataset of 4080 crack images and 1100 spalling images. To summarize the output of this defects classification models-

- A total of thirty models are evaluated combining the learning rates (0.0001, 0.001, 0.1) and optimization functions (SGD, RMSprop) with five different CNN models (VGG-19, ResNet50, MobileNetV2, Xception and InceptionV3).
- InceptionV3 model has outranked the other models with accuracy, precision and recall of 91%, 83% and 100%, respectively. One possible reason behind the InceptionV3 model functioning better than other models is that the model has the highest layers of depth for learning, which facilitates the model to gain better performance. VGG19 has the least prospect with defects identification.
- With the help of confusion matrix this study has found that IncpetionV3 has made the least false prediction with crack identification. Also in case of spalling identification IncpetionV3 has labelled all the spalling cases correctly.
- Among three learning rate 0.0001, 0.001 and 0.1 with learning rate 0.001 all the CNN models has achieved the best performance which establishes the idea that a low learning rate does not always confirm better performance with CNN models, some cases increase the computation cost of the learning process without improving the models' prediction proficiency.
- In case of optimization functions SGD has assisted the CNN modes to achieve the better performance, which proves the statement that SGD has better stability and generalization capacity than other adaptive optimization methods (i.e., RMSprop).

6.3.2 Defects Segmentation

In case of segmentation, this study has engaged semantic segmentation process because of its competency with differentiating various the objects and locate each object individually. Also, the

semantic segmentation is done for both the defects- crack and spalling separately. This study has explored two types of Encode-Decoder models, i.e., U-Net and PSP-Net, as these models have previously proved too have better segmentation accuracy. For segmentation purpose this study has used the same dataset from defects classification stage. The highlights of segmentation task are summarized as follows-

- This study has performed a detailed sensitivity analysis of semantic segmentation for both crack and spalling cases by combining different learnable parameters, i.e., encoder backbone model (VGG-19, ResNet50, Inception and EfficientNetB3), optimizers (SGD and ADAM) and learning rates (0.0001, 0.01,0.1) with two different encoder-decoder model- U-Net and PSP-Net.
- In case of crack segmentation, U-Net with EfficientNetB3 backbone model has outranked all the other segmentation models with IoU, F1-score, precision and recall of 91.98%, 95.66%, 94.73% and 97.59%, respectively. Among all the PSP-Net models the best performance is achieved by EfficientNetB3 backend PSP-Net, which indicates that EfficientNetB3 has performed better than other backbone models.
- For spalling segmentation InceptionV3 based U-net model has gained the highest rank for spalling area detection based on the evaluation metrics' numerical values- IoU value of 81.30%, F1-score of 89.43%, precision value of 86.40% and recall value of 92.87%.
- Even though U-Net has the upper hand in both the crack and spalling segmentation, it is prominent that PSP-Net models also have obtained a performance close to U-Net models, which certainly glorifies the overall architecture of encoder-decoder models.
- In all the cases, ADAM is found to be the best optimizer which supports the statement of Choi et. al. (2019) that with the best-tuned model adaptive gradient methods always perform better than the gradient descent optimizers. Also, with a learning rate of 0.01, the training process has achieved its' optimized output without losing much information

6.4 Limitations and Recommendations for Future Works

A real-word DL-based application requires a lot of proper resources and proficiency in defects assessment. Nevertheless, there are many limitations that need to be addressed before implicating the CNN trained model for defects detection. This study has worked with some of the current

limitations and gained a satisfactory performance in both defects classification and defects segmentation cases. However, some challenges are still need a lot of attention and substantially requires further studies, such as-

- One of the biggest challenge for this research was to collect images with complex background implicating the original structural condition. Previously many researches has worked on concrete crack identification and collected images of concrete crack surfaces. So, it was easier to get open-sourced crack dataset but it was difficult to get any open-sourced dataset for any other type of defects. This study has built a spalling dataset of 1100 images from scratch but unfortunately still the dataset size is quite smaller than the crack dataset (almost ¼ of crack dataset).

Based on the analysis performed in this study, there are few areas that has future scopes to improve the automatic defects detection process.
First, it has been observed that in this study the crack and spalling dataset has an imbalance in size. Also the images from original inspection reports were very limited and was forced to depend on online resources. In future the project working with DL based defects detection need more collaboration with industrial partners to collect a large amount of diverse images. Also, the images from dataset need to be annotated individually as the generalized annotation process can not always identify the specific area of the defects.

Secondly, the future studies can take advantage of DL's multiple object detection proficiency and create model capable to identify multiple defects at a time from both images and videos. This study has worked with images of two types of defects because of the limited availability of resources of other type of defects. Once an adequate amount of dataset is developed it is possible to identify various type of defects from a single image or a video clip.

Finally, to make the models' proficiency easily accessible for engineers and other stack holders it is necessary to build an digital application system which is competent enough to identify various defects of the structures and in advanced level can also quantify the defects.

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.

Agarwal, M., Rajak, A., & Shrivastava, A. K. (2021). *Assessment of optimizers impact on image recognition with convolutional neural network to adversarial datasets.* Paper presented at the Journal of Physics: Conference Series.

Anai, S., Yabuki, N., & Fukuda, T. (2019). Comparison of Deep Learning Model Precision for Detecting Concrete Deterioration Types from Digital Images. In *Computing in Civil Engineering 2019: Smart Cities, Sustainability, and Resilience* (pp. 196-203): American Society of Civil Engineers Reston, VA.

Argyris, C., Chowdhury, S., Zabel, V., & Papadimitriou, C. (2018). Bayesian optimal sensor placement for crack identification in structures using strain measurements. *Structural Control and Health Monitoring, 25*(5), e2137.

Azimi, M., Eslamlou, A. D., & Pekcan, G. (2020). Data-driven structural health monitoring and damage detection through deep learning: State-of-the-art review. *Sensors, 20*(10), 2778.

Bengio, Y. (2012). *Deep learning of representations for unsupervised and transfer learning.* Paper presented at the Proceedings of ICML workshop on unsupervised and transfer learning.

Caruana, R. (1994). Learning many related tasks at the same time with backpropagation. *Advances in neural information processing systems, 7*.

Cha, Y.-J., You, K., & Choi, W. (2016). Vision-based detection of loosened bolts using the Hough transform and support vector machines. *Automation in Construction, 71*, 181-188.

Cha, Y. J., Choi, W., & Büyüköztürk, O. (2017). Deep learning-based crack damage detection using convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering, 32*(5), 361-378.

Cha, Y. J., Choi, W., Suh, G., Mahmoudkhani, S., & Büyüköztürk, O. (2018). Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. *Computer-Aided Civil and Infrastructure Engineering, 33*(9), 731-747.

Chen, F. C., Jahanshahi, M. R., Wu, R. T., & Joffe, C. (2017). A texture-based video processing methodology using Bayesian data fusion for autonomous crack detection on metallic surfaces. *Computer-Aided Civil and Infrastructure Engineering, 32*(4), 271-287.

Chen, K., Yadav, A., Khan, A., Meng, Y., & Zhu, K. (2019). Improved crack detection and recognition based on convolutional neural network. *Modelling and simulation in engineering, 2019*.

Chen, P.-H., Shen, H.-K., Lei, C.-Y., & Chang, L.-M. (2012). Support-vector-machine-based method for automated steel bridge rust assessment. *Automation in Construction, 23*, 9-19.

Cheng, H., Shi, X., & Glazier, C. (2003). Real-time image thresholding based on sample space reduction and interpolation approach. *Journal of computing in civil engineering, 17*(4), 264-272.

Choi, W. (2020). Deep learning implemented structural defect detection on digital images.

Choi, W., & Cha, Y.-J. (2019). SDDNet: Real-time crack segmentation. *IEEE Transactions on Industrial Electronics, 67*(9), 8016-8025.

Chollet, F. (2017). *Xception: Deep learning with depthwise separable convolutions.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Choudhary, G. K., & Dey, S. (2012). *Crack detection in concrete surfaces using image processing, fuzzy logic, and neural networks.* Paper presented at the 2012 IEEE fifth international conference on advanced computational intelligence (ICACI).

Ciregan, D., Meier, U., & Schmidhuber, J. (2012). *Multi-column deep neural networks for image classification.* Paper presented at the 2012 IEEE conference on computer vision and pattern recognition.

Dai, B., Gu, C., Zhao, E., Zhu, K., Cao, W., & Qin, X. (2019). Improved online sequential extreme learning machine for identifying crack behavior in concrete dam. *Advances in Structural Engineering, 22*(2), 402-412.

Dai Wenyuan, Y. Q., Guirong, X., & Yong, Y. (2007). *Boosting for transfer learning.* Paper presented at the Proceedings of the 24th International Conference on Machine Learning, Corvallis, USA.

Dais, D., Bal, I. E., Smyrou, E., & Sarhosis, V. (2021). Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning. *Automation in Construction, 125*, 103606.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). *Imagenet: A large-scale hierarchical image database.* Paper presented at the 2009 IEEE conference on computer vision and pattern recognition.

Deng, J., Lu, Y., & Lee, V. C. S. (2020). Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering, 35*(4), 373-388.

Duarte, D., Nex, F., Kerle, N., & Vosselman, G. (2018). Multi-resolution feature fusion for image classification of building damages with convolutional neural networks. *Remote sensing, 10*(10), 1636.

Dunphy, K., Sadhu, A., & Wang, J. (2022). Multiclass damage detection in concrete structures using a transfer learning-based generative adversarial networks. Structural Control and Health Monitoring, e3079.

Feng, C., Zhang, H., Wang, S., Li, Y., Wang, H., & Yan, F. (2019). Structural damage detection using deep convolutional neural network and transfer learning. *KSCE Journal of Civil Engineering, 23*(10), 4493-4502.

Flah, M., Suleiman, A. R., & Nehdi, M. L. (2020). Classification and quantification of cracks in concrete structures using deep learning image-based techniques. *Cement and Concrete Composites, 114*, 103781.

Fujita, Y., & Hamamoto, Y. (2011). A robust automatic crack detection method from noisy concrete surfaces. *Machine Vision and Applications, 22*(2), 245-254.

Fujita, Y., Mitani, Y., & Hamamoto, Y. (2006). *A method for crack detection on a concrete structure.* Paper presented at the 18th International Conference on Pattern Recognition (ICPR'06).

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., & Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*.

German, S., Brilakis, I., & DesRoches, R. (2012). Rapid entropy-based detection and properties measurement of concrete spalling with machine vision for post-earthquake safety assessments. *Advanced Engineering Informatics, 26*(4), 846-858.

Ghosh Mondal, T., Jahanshahi, M. R., Wu, R. T., & Wu, Z. Y. (2020). Deep learning-based multi-class damage detection for autonomous post-disaster reconnaissance. *Structural Control and Health Monitoring, 27*(4), e2507.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*: MIT press.

Hardt, M., Recht, B., & Singer, Y. (2016). *Train faster, generalize better: Stability of stochastic gradient descent.* Paper presented at the International conference on machine learning.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Huyan, J., Li, W., Tighe, S., Xu, Z., & Zhai, J. (2020). CrackU-net: A novel deep convolutional neural network for pixelwise pavement crack detection. *Structural Control and Health Monitoring, 27*(8), e2551.

Ioffe, S., & Szegedy, C. (2015). *Batch normalization: Accelerating deep network training by reducing internal covariate shift.* Paper presented at the International conference on machine learning.

Iyer, S., & Sinha, S. K. (2006). Segmentation of pipe images for crack detection in buried sewers. *Computer-Aided Civil and Infrastructure Engineering, 21*(6), 395-410.

Jahanshahi, M. R., Masri, S. F., Padgett, C. W., & Sukhatme, G. S. (2013). An innovative methodology for detection and quantification of cracks through incorporation of depth perception. *Machine Vision and Applications, 24*(2), 227-241.

Kang, D. H., & Cha, Y.-J. (2021). Efficient attention-based deep encoder and decoder for automatic crack segmentation. *Structural Health Monitoring*, 14759217211053776.

Kim, H., Ahn, E., Shin, M., & Sim, S.-H. (2019). Crack and noncrack classification from concrete surface images using machine learning. *Structural Health Monitoring, 18*(3), 725-738.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koziarski, M., & Cyganek, B. (2017). Image recognition with deep neural networks in presence of noise–dealing with and taking advantage of distortions. *Integrated Computer-Aided Engineering, 24*(4), 337-349.

Kumar, A., Sarkar, S., & Pradhan, C. (2020). Malaria disease detection using cnn technique with sgd, rmsprop and adam optimizers. In *Deep learning techniques for biomedical and health informatics* (pp. 211-230): Springer.

Learning, D. (2016). Ian Goodfellow, Yoshua Bengio, Aaron Courville. *The reference book for deep learning models*.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., & Jackel, L. (1989). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems, 2*.

Li, G., Zhao, X., Du, K., Ru, F., & Zhang, Y. (2017). Recognition and evaluation of bridge cracks with modified active contour model and greedy search-based support vector machine. *Automation in Construction, 78*, 51-61.

Li, S., & Zhao, X. (2018). *Convolutional neural networks-based crack detection for real concrete surface.* Paper presented at the Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018.

Li, S., & Zhao, X. (2020). Automatic crack detection and measurement of concrete structure using convolutional encoder-decoder network. *IEEE Access, 8*, 134602-134618.

Li, S., Zhao, X., & Zhou, G. (2019). Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network. *Computer-Aided Civil and Infrastructure Engineering, 34*(7), 616-634.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). *Focal loss for dense object detection.* Paper presented at the Proceedings of the IEEE international conference on computer vision.

Lin, Y. z., Nie, Z. h., & Ma, H. w. (2017). Structural damage detection with automatic feature-extraction through deep learning. *Computer-Aided Civil and Infrastructure Engineering, 32*(12), 1025-1046.

Liu, J., Yang, X., Lau, S., Wang, X., Luo, S., Lee, V. C. S., & Ding, L. (2020). Automated pavement crack detection and segmentation based on two-step convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering, 35*(11), 1291-1305.

Miao, Z., Ji, X., Okazaki, T., & Takahashi, N. (2021). Pixel-level multicategory detection of visible seismic damage of reinforced concrete components. *Computer-Aided Civil and Infrastructure Engineering, 36*(5), 620-637.

Mohtasham Khani, M., Vahidnia, S., Ghasemzadeh, L., Ozturk, Y. E., Yuvalaklioglu, M., Akin, S., & Ure, N. K. (2020). Deep-learning-based crack detection with applications for the structural health monitoring of gas turbines. *Structural Health Monitoring, 19*(5), 1440-1452.

Na, W., & Tao, W. (2012). *Proximal support vector machine based pavement image classification.* Paper presented at the 2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI).

Nahata, D., Mulchandani, H. K., Bansal, S., & Muthukumar, G. (2019). Post-earthquake assessment of buildings using deep learning. *arXiv preprint arXiv:1907.07877*.

Nguyen, H.-N., Kam, T.-Y., & Cheng, P.-Y. (2014). An automatic approach for accurate edge detection of concrete crack utilizing 2D geometric features of crack. *Journal of Signal Processing Systems, 77*(3), 221-240.

Nhat-Duc, H., Nguyen, Q.-L., & Tran, V.-D. (2018). Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network. *Automation in Construction, 94*, 203-213.

Nishikawa, T., Yoshida, J., Sugiyama, T., & Fujino, Y. (2012). Concrete crack detection by multiple sequential image filtering. *Computer-Aided Civil and Infrastructure Engineering, 27*(1), 29-47.

O'Byrne, M., Ghosh, B., Schoefs, F., & Pakrashi, V. (2014). Regionally enhanced multiphase segmentation technique for damaged surfaces. *Computer-Aided Civil and Infrastructure Engineering, 29*(9), 644-658.

Ouma, Y. O., & Hahn, M. (2017). Pothole detection on asphalt pavements from 2D-colour pothole images using fuzzy c-means clustering and morphological reconstruction. *Automation in Construction, 83*, 196-211.

Pan, X., & Yang, T. (2020). Postdisaster image-based damage detection and repair cost estimation of reinforced concrete buildings using dual convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering, 35*(5), 495-510.

Perez, H., & Tah, J. H. (2021). Deep learning smartphone application for real-time detection of defects in buildings. *Structural Control and Health Monitoring, 28*(7), e2751.

Poojary, R., & Pai, A. (2019). *Comparative study of model optimization techniques in fine-tuned CNN models.* Paper presented at the 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA).

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-net: Convolutional networks for biomedical image segmentation.* Paper presented at the International Conference on Medical image computing and computer-assisted intervention.

Rudin, L. I., Osher, S., & Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena, 60*(1-4), 259-268.

Sifre, L., & Mallat, S. (2014). Rigid-motion scattering for texture classification. *arXiv preprint arXiv:1403.1687*.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, Z., Liu, Y., Song, R., Chen, Z., Yang, J., Zhang, C., & Jiang, Q. (2018). A sparsity-based stochastic pooling mechanism for deep convolutional neural networks. *Neural Networks, 105*, 340-345.

Sony, S., Dunphy, K., Sadhu, A., & Capretz, M. (2021). A systematic review of convolutional neural network-based structural condition assessment techniques. *Engineering Structures, 226*, 111347.

Sun, H., Burton, H. V., & Huang, H. (2021). Machine learning applications for building structural design and performance assessment: State-of-the-art review. *Journal of Building Engineering, 33*, 101816.

Szegedy, C., Liu, W., & Jia, Y. Going deeper with convolutions 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) June 2015Boston. *MA, USA1–9, 10*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). *Going deeper with convolutions.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Tan, M., & Le, Q. (2019). *Efficientnet: Rethinking model scaling for convolutional neural networks.* Paper presented at the International conference on machine learning.

Verma, P., Tripathi, V., & Pant, B. (2021). Comparison of different optimizers implemented on the deep learning architectures for COVID-19 classification. *Materials Today: Proceedings, 46*, 11098-11102.

Wang, J. J., Liu, Y. F., Nie, X., & Mo, Y. (2022). Deep convolutional neural networks for semantic segmentation of cracks. *Structural Control and Health Monitoring, 29*(1), e2850.

Wang, N., Zhao, Q., Li, S., Zhao, X., & Zhao, P. (2018). Damage classification for masonry historic structures using convolutional neural networks based on still images. *Computer-Aided Civil and Infrastructure Engineering, 33*(12), 1073-1089.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., & Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. *Advances in neural information processing systems, 30*.

Wu, L., Lin, X., Chen, Z., Lin, P., & Cheng, S. (2021). Surface crack detection based on image stitching and transfer learning with pretrained convolutional neural network. *Structural Control and Health Monitoring, 28*(8), e2766.

Xu, Y., Wei, S., Bao, Y., & Li, H. (2019). Automatic seismic damage identification of reinforced concrete columns from images by a region-based deep convolutional neural network. *Structural Control and Health Monitoring, 26*(3), e2313.

Xue, Y., & Li, Y. (2018). A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects. *Computer-Aided Civil and Infrastructure Engineering, 33*(8), 638-654.

Yamaguchi, T., Nakamura, S., Saegusa, R., & Hashimoto, S. (2008). Image-based crack detection for real concrete surfaces. *IEEJ Transactions on Electrical and Electronic Engineering, 3*(1), 128-135.

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into imaging, 9*(4), 611-629.

Yang, C., Chen, J., Li, Z., & Huang, Y. (2021). Structural crack detection and recognition based on deep learning. *Applied sciences, 11*(6), 2868.

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.

Ye, X.-W., Jin, T., & Chen, P.-Y. (2019). Structural crack detection using deep learning–based fully convolutional networks. *Advances in Structural Engineering, 22*(16), 3412-3419.

Yeung, M., Sala, E., Schönlieb, C.-B., & Rundo, L. (2022). Unified Focal loss: Generalising Dice and cross entropy-based losses to handle class imbalanced medical image segmentation. *Computerized Medical Imaging and Graphics, 95*, 102026.

Ying, L., & Salari, E. (2010). Beamlet transform-based technique for pavement crack detection and classification. *Computer-Aided Civil and Infrastructure Engineering, 25*(8), 572-580.

Yu, F., Koltun, V., & Funkhouser, T. (2017). *Dilated residual networks.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Zeiler, M. D., & Fergus, R. (2013). Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*.

Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). *Pyramid scene parsing network.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Zhao, Z.-Q., Zheng, P., Xu, S.-t., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems, 30*(11), 3212-3232.

Zou, Q., Zhang, Z., Li, Q., Qi, X., Wang, Q., & Wang, S. (2018). Deepcrack: Learning hierarchical convolutional features for crack detection. *IEEE Transactions on Image Processing, 28*(3), 1498-1512.