Cape Peninsula
University of Technology

USE OF MACHINE LEARNING TECHNIQUES IN THE DETECTION OF WALL DEFECTS IN SOUTH AFRICAN HERITAGE STRUCTURES

by

KIERAN JURIES

A thesis submitted in fulfilment of the requirements for the degree
Master of Engineering: Civil Engineering
in the Faculty of Engineering and the Built Environment
at the

Cape Peninsula University of Technology

Supervisor: Dr Patrick Bukenya
Co-supervisor: Prof Kumar Pallav

Bellville
28 August 2024

Declaration

I, Kieran Juries declare that the contents of this thesis represent my own unaided work, and that the thesis has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

[illegible]

Signed

28th August 2024

Date

Abstract

South Africa has a storied past, with remnants of it remaining in heritage structures such as the Castle of Good Hope and Robben Island. These heritage structures preserve the country's past while also showing how it has progressed over time. Heritage structures play a role in the country's economy, contributing to both the tourism industry and job creation. Despite their importance, they are still prone to deterioration due to various factors and require monitoring to ensure their structural integrity. Currently, it is common practice to carry out monitoring through on site visual inspections. However, this has been found to produce inconsistent results while also incurring high costs due to the expertise required. This research therefore explored the use of machine learning techniques to classify wall defects in South African heritage structures.

For this, machine learning and deep learning algorithms were used to classify crack, spall and intact areas. These algorithms included decision trees, support vector machines, k-nearest neighbours, artificial neural networks and a pre-trained AlexNet convolutional neural network (CNN) model. Additionally, the research focused on the use of image processing to determine the extent and characteristics of crack damage.

Initially, a dataset of images made up of crack, spall and intact areas was collected from the Castle of Good Hope, and buildings on Robben Island and in Dal Josafat, Paarl. The Bag of Visual Words technique was then used to extract features from these images. Feature extraction was carried out using the speeded-up robust features (SURF) function, with k-means clustering subsequently carried out to create the visual vocabulary. Models were then trained to learn patterns and recognise the different damage types using the histogram of visual words obtained from the Bag of Visual Words as input. Models were then tested on a separate dataset where their accuracy, misclassification rate, precision and recall were obtained.

A different approach was used to train and test the pre-trained AlexNet CNN model. For this, a transfer learning strategy was applied whereby the CNN's output was modified. This was completed by adding a new specialised fully connected layer. The model's output layer weights were then fine-tuned during the training phase, after which testing was carried out to obtain the model's accuracy, misclassification rate, precision and recall.

When comparing the results of the different models, it was observed that the CNN model produced the best results in terms of accuracy, precision and recall, obtaining 97.40% for each metric. Furthermore, the model also achieved a misclassification rate of 2.60% which was the lowest among all models. The performance of the model can be attributed to the specialised convolutional and pooling layers that the model processes, as well as the fact that the model was pre-trained, where prior learned information could be leveraged for this classification task.

To determine the extent of crack damage in heritage structures, image processing techniques such as thresholding, binarisation, segmentation, skeletonisation and the Euclidean distance transform were used to extract measurements of the average crack width and the crack length. These techniques were used on a set of crack images collected from the Castle of Good Hope where the results obtained were then compared to the on site measurements. When comparing the average crack width measurements an average error of 10.73% was obtained. The comparison of crack length measurements was inconclusive as the image processing methodology produced high error rates which varied across all cracks. The error rates obtained can be attributed to various factors such as the image quality, limitations of the image processing techniques applied, sensitivity of the skeletonisation function and excessive noise in images.

Although results varied in terms of the extraction of crack characteristics using image processing, the machine learning and deep learning models produced results which show their potential to be used as resources in the structural health monitoring field. Furthermore, their integration into structural health monitoring provides an alternative tool for engineers and heritage experts which can be used in hand with non-destructive data, sensor data and even images to allow for better and more precise decisions in terms of the conservation, restoration and repair and maintenance of heritage structures.

Acknowledgements

I would like to express my appreciation and gratitude to the following people and organisations who contributed to the successful completion of my research.

Firstly, I would like to thank Dr Patrick Bukenya for taking me on as his student and guiding me throughout the process. This has by far been one of the most challenging experiences I have encountered academically, but he played a massive role in motivating me and helping me get through it. His experience and supervision also played a massive role in shaping this work while changing my outlook on engineering as a whole.

To Prof. Pallav Kumar, your co-supervision during this research and your efforts in evaluating my thesis and providing constructive feedback were greatly appreciated. Without your input, the research project would not be where it is today.

A special note of appreciation is extended to Virtual Consulting Engineers for the financial support provided and for accommodating my research needs, providing me with the necessary time off to complete this research.

To Ms M. Botha, I am truly grateful and appreciative for your aid in reviewing the literature and assisting in the proposal writing process. Your input was essential.

I am highly indebted to Ms A. Prinsloo for her generosity in supplying the research equipment essential for the collection of the dataset critical to this research. Her contribution considerably improved the quality and comprehensiveness of this research.

Most importantly, I owe an incredible debt of gratitude to my parents. Their endless love, support and constant encouragement have been my source of motivation throughout this academic journey. I am incredibly thankful for everything they have done for me; I owe them the world.

Each of you has left an unmistakable impression on this journey, and your contributions have been vital to its success. Thank you from the bottom of my heart.

Table of Contents

Page

List of Figures

List of Tables

Terms and concepts

Terms/Acronyms/Abbreviations	Definition/Explanation
Artificial neural network	A class of machine learning models inspired by the structure function of the human brain, consisting of interconnected nodes in layers
Bag of Visual Words	A technique used for image classification that represents images as histograms of visual words which involves feature extraction and vocabulary construction
Binary and multi-class classification	Classification tasks can be binary, where the goal is to distinguish between two classes, or multi-class, where the data points belong to one of several classes
Classification and regression	Classification aims to assign data points to predefined classes while regression predicts continuous values or quantities
Class imbalance	A situation where one class has fewer or higher samples than the other classes in a classification problem
Confusion matrix	A table used in classification tasks to visualise the performance of a machine learning model
Convolutional neural network (CNN)	A type of deep learning algorithm commonly used for image classification, feature extraction and image analysis
Cross-validation	A technique used to assess the performance of a model which involves dividing the training dataset into subsets of training and validation data where the model learns on the training data and is validated on the validation data subset
Damage classification	The process of categorising observed damage in structures
Damage typologies	Different categories or types of damage commonly observed in heritage structures
Data annotation	Also known as data labelling, the process of adding metadata or labels to the dataset
Data augmentation	A technique used to artificially increase the diversity of a dataset by applying transformations to it
Data pre-processing	Cleaning and preparing the dataset for machine learning, which includes processes like data augmentation or data normalisation
Decision trees	A machine learning algorithm that builds a tree-like model of decisions and their possible outcomes
Deep learning	A subset of machine learning that involves neural networks with multiple hidden layers
DSLR	Digital single lens reflex

FN	False negative
FP	False positive
Heritage structures	Historical or culturally significant buildings and structures
Image processing	The use of digital image analysis to detect and analyse defects in structures
K-nearest neighbours	A machine learning algorithm that assigns a data point to a class based on the majority class among its k-nearest neighbours in the feature space
Non-destructive testing	Techniques and methods used to evaluate the condition of structures without causing damage
Rectified linear unit (ReLU)	An activation function used in neural networks
SAHRA	South African heritage Resources Agency, responsible for preserving heritage sites in South Africa
Structural health monitoring (SHM)	The continuous or periodic assessment of the condition of structures to ensure their safety and longevity
Structural integrity	The ability of a structure to withstand loads without failing or showing major structural damage.
Supervised learning	A machine learning approach where the algorithm is trained on labelled data, and where input data is associated with known output values
Support vector machine (SVM)	A machine-learning algorithm that finds a hyperplane in a dimensional space to separate data points into classes
TN	True negative
TP	True positive
Transfer learning	A technique where a pre-trained model, trained on a a large dataset for a similar task, is adapted for a new task
Unsupervised learning	A machine learning approach where the algorithm discovers patterns and trends in data without labelled input data

Chapter 1 Introduction

Heritage structures drive economic development of any country through tourism and by maintaining cultural, traditional, and religious history. Due to their cultural and historical importance and the economic contribution heritage structures provide, it is important to preserve these important structures. Structural health monitoring forms an important part of preserving heritage structures by ensuring their structural integrity. It involves identifying areas of damage, classifying the observed damage, and providing a sustainable solution to restore structures. Researchers have utilised different machine learning and deep learning algorithms to assist in structural health monitoring tasks. Therefore, this research focuses on the comparison of machine learning algorithms' performance in classifying damage classes on heritage structures.

This chapter presents the background and motivation for this research, outlining the objectives while also emphasising the importance of the research. Furthermore, this chapter covers how the research aims to improve structural health monitoring technique efficiency, with the main goal of preserving and conserving cultural and historical heritage. A general overview of the methodology employed is also presented, along with the approach taken to achieve the research objectives.

1.1 Background and motivation

Heritage structures are existing structures of historical or cultural significance to society. These structures are distinct in their design, architecture, and construction. Heritage structures have unique characteristics that reflect a rich historical background, including historical religion, political history, aesthetic culture and even significant historical events (Loke, 2020:1). The Colosseum in Rome, the Leaning Tower of Pisa in Italy, Hagia Sophia in Turkey, the Castle of Good Hope in South Africa, and the Forbidden City in China are just a few examples of internationally renowned heritage buildings (Loke, 2020:1).

Heritage structures contribute significantly to the economy by providing key attractions for tourists, in addition to their historical and cultural interest. According to a survey completed by the World Travel and Tourism Council (2022), the travel and tourism sector significantly boosted South Africa's employment and gross domestic product (GDP). The data shows that the travel and tourism sector contributed 3.2% of the total economy (ZAR 195.2 billion) to GDP while contributing 7.3% of total jobs to employment (1.08 million) for 2021 (World Travel and Tourism Council, 2022). This shows an increase from the previous year, where 3.1% of the total economy was contributed to GDP while 7.0% of the total jobs were contributed to employment (World Travel and Tourism Council, 2022).

However, heritage structures deteriorate due to age, exposure to service loads, environmental loads and accidental actions. The service loads are generally any nominal load the structure is expected to support during its daily operation, while environmental loads are usually natural phenomena such as wind loads, seismic loads and even snow loads. These loads reduce strength and rigidity and cause damage to structural systems of heritage structures over time. Also, heritage structures exhibit surface defects that arise over time, including cracking, spalling, mould and staining. As heritage structures deteriorate, they become more vulnerable to compressive, tensile, bending, and buckling failures, and collapse. Examples of this type of failure occurred at the entrance to the Taj Mahal in India in 2018, where the minaret collapsed with no apparent warning (Mishra, 2021:227). A similar failure also occurred in Pavia, Italy, 1989, where a 14th-century bell tower collapsed (Mishra, 2021:227).

To ensure that heritage structures do not collapse, they must be preserved through repair and maintenance. Repair and maintenance activities are critical in preserving and ensuring that heritage structures perform effectively. The lack of maintenance results in the accelerated degradation of historic building materials, which results in a loss of cultural significance (Valero et al., 2019:1). According to Valero et al. (2019:1), surveys and reporting are essential to the repair and maintenance process. The most common is potentially a survey for a condition which serves as a basis for other procedures.

The structural integrity of heritage structures and their capacity to carry out their intended use over time must be regularly monitored and assessed for successful management. For this to be completed, structural health monitoring is carried out, which is the process of continuously or periodically assessing the condition of structures to ensure their safety and longevity. This is especially important when considering heritage structures, as the process combines damage detection and localisation, data analysis, risk management strategies and predictive maintenance to ensure the preservation of heritage structures. Furthermore, by actively evaluating the heritage structure's structural integrity and its response to applied loads and environmental factors that may accelerate its deterioration over time, one can ensure the longevity of the structure and protect its historical value.

Traditionally, visual inspection methods have been the most utilised approach in the structural health monitoring of heritage buildings. An example of this type of assessment includes the Conditional Assessment Report for Robben Island, where the condition of the various buildings on the heritage site was assessed (Charles Consult Consortium, 2018:3-4). Similarly, the Report on Existing Housing Condition and Assessment of Recommended Work on City-Owned Houses in Bo-Kaap provided a general condition assessment of different houses in the Bo-Kaap heritage area (Architects Associated, 2018:1). Visual inspections offer advantages, including their non-destructive and non-intrusive nature, allowing assessments without causing damage to the structure. These inspections facilitate real-time observations, allowing for immediate on site judgments and decisions. Additionally, visual inspections provide flexibility and accessibility, making them suitable for remote or challenging environments, often requiring minimal or no specialised equipment. However, visual inspections have several limitations, i.e., being time-consuming, error-prone, labour-intensive, costly and the production of inconsistent and variable results. This is because visual inspections depend on the technical skills and expertise of the inspector, which can result in a subjective assessment, human error, or even the misclassification of defects (Choi, 2020:2).

Furthermore, with the significant need to preserve heritage structures, there is an increased demand for resources to undertake proper monitoring. With the shortcomings of visual inspections, the effectiveness and efficiency of the inspection process need to be improved urgently. In this regard, image processing technology and machine learning algorithms offer a practical and powerful tool for detecting and analysing defects in heritage structures. By harnessing the capabilities of machine learning, the efficacy of monitoring heritage structures can be enhanced. Machine learning algorithms can be trained on large datasets of structural images, enabling them to learn and recognise any structural anomaly patterns. The integration of machine learning in image processing empowers heritage conservation by allowing more accurate, consistent, and data-driven insight, and this facilitates preservation efforts and timely maintenance for the long-term sustainability of heritage structures.

It is against this background that this research aims to compare the performance of different machine learning techniques in accurately detecting wall defects in South African heritage structures using image processing.

1.2 Research problem

Heritage structures are important in any country's economy due to the revenue they bring through tourism. Therefore, heritage structures must be structurally safe to continue serving their intended purpose. However, these important structures are ageing and deteriorating and need to be monitored regularly to ensure failure does not occur. Image processing techniques have been used to detect defects in structures such as bridges. There is little knowledge of the accuracy of the image processing techniques in detecting defects in heritage structures. This research focuses on applying machine learning-based image processing techniques in detecting defects in heritage structures and comparing the performance of the techniques.

1.3 Research question

What accuracy can be achieved when comparing abilities of various machine learning algorithms to detect wall defects in South African heritage structures using image processing?

1.4 Objectives

The preservation of heritage structures requires monitoring of their structural integrity.

To achieve this, the following objectives were met:

- To compare the performance of different machine learning algorithms, namely the convolutional neural network, artificial neural network, support vector machine, decision trees and k-nearest neighbour in detecting defects such as crack, spall and intact areas in heritage structures.
- To detect patterns and learn the distributions of extracted damage-sensitive features from an image database using machine learning methods.
- To determine the characteristics and extent of crack damage using image processing.

1.5 Outcomes

Through the completion of this study, the following outcomes were achieved:

- The performance and capability of various machine learning algorithms to detect and classify crack, spall, and intact areas were compared and assessed.
- Machine learning methods were used to accurately detect patterns and learn the distributions of damage-sensitive features from images.
- The characteristics and extent of crack damage were determined using image processing techniques.

1.6 Research context and significance

Machine learning algorithms play a crucial role in the structural health monitoring of heritage structures by providing a decentralised and autonomous approach to detecting and classifying defects in heritage buildings through image processing. Machine learning is essential because it enables algorithms to learn from data and improve their ability to detect and accurately classify different types of defects. By analysing large volumes of data, machine learning algorithms can detect signs of deterioration and classify damage that may be missed during visual inspections. The early detection capability is vital for preventing further damage, informing maintenance strategies, and preserving invaluable heritage structures for the future.

1.7 Delineation

The study was restricted to analysing image datasets collected from the Castle of Good Hope, Robben Island and Dal Josafat heritage sites in the Western Cape, South Africa. The study was also restricted to three categories of damage extent: crack, spall and intact areas/non-damage. The following restrictions were placed on the investigation:

- The accuracy and efficiency of five machine learning techniques were compared in this research.
- Only three categories of damage extent were considered, i.e., various crack patterns, spalling and intact areas.
- The methodology was restricted to existing damage detection algorithms and did not intend to develop a new defect detection algorithm.
- The methods intended for use were restricted to heritage structures constructed in the Western Cape, South Africa.

1.8 Methodology

Machine learning algorithms have gained significant attention in the structural health monitoring of structures. Many studies have concentrated on these methodologies to facilitate the monitoring process more effectively.

A quantitative research methodology was employed in the research design to accomplish the research objectives, incorporating machine learning and deep learning for data analysis. The analysis drew upon relevant literature on artificial intelligence, machine learning, deep learning and image processing, which served as a framework for the analytical techniques applied. The research centred around three case studies: the Castle of Good Hope, Robben Island, and Dal Josafat heritage sites. The primary goal of this study was to assess various machine learning approaches and compare their effectiveness in learning the patterns of damage-sensitive features, with a particular emphasis on accurately classifying different damage types. Additionally, the study also aimed to determine the characteristics and extent of crack damage using image processing techniques.

Initially, images depicting various wall defects like cracks, spalling and intact areas were collected from the selected heritage sites located in the Western Cape, South Africa. The data was then analysed, processed, and classified using MATLAB software. The research utilised existing machine learning and deep learning algorithms, including the support vector machine, decision tree, k-nearest neighbour, a pre-trained convolutional neural network (CNN) and artificial neural network for damage classification.

Furthermore, an additional dataset of images depicting crack damage collected from the Castle of Good Hope was then utilised to determine the extent of crack damage. This process involved the application of image processing techniques such as thresholding, binarisation, segmentation and skeletonisation, which were combined with image processing distance-based computations to determine the crack characteristics including the cracks’ length and average width.

1.9 Organisation of the thesis

The thesis is laid out as follows:

Chapter 1: Introduction: This chapter presents an overview of the research topic, offering essential background information and outlining the reason for the research. It stresses the significance of the research and highlights the objectives the study aims to attain.

Chapter 2: Literature review and theory: Chapter 2 evaluates structural health monitoring, machine learning, and deep learning literature. It explores the existing body of information in these areas, reviewing numerous studies, research papers, and scholarly publications. Furthermore, the chapter explores prior case studies that specifically concentrate on the application of image processing, machine learning, and deep learning approaches in the context of structural health monitoring for heritage structures.

Chapter 3: Research methodology: Chapter 3 provides an overview of the quantitative research methodology adopted in this research. This comprised an experimental research design comprised of data collection, pre-processing, image processing, model training, model testing and evaluation. The methodology made use of machine learning and a deep learning approach, which were used to classify defects in heritage structures. The machine learning approach utilised the Bag of Visual Words technique with the support vector machine, k-nearest neighbour, decision tree and artificial neural network machine learning algorithms. The deep learning methodology used a transfer learning approach in conjunction with a pre-trained AlexNet CNN model. Additionally, an overview of the image processing methodology utilised to determine the characteristics and extent of crack damage in walls is presented. The methodology was made up of a combination of advanced image processing techniques such as thresholding, binarisation, segmentation and skeletonisation, which were combined with distance-based computations to determine the extent of damage. This chapter provides an in-depth description of these approaches while the specific processes and steps used to carry out the research are discussed in depth.

Chapter 4: Results and discussion: This chapter discusses training and testing results of the machine learning algorithms in classifying defects such as cracks, spalling and identifying intact areas. Chapter 4 outlines each model's accuracy, misclassification rate, precision and recall in the defect classification task. A comparative analysis of the performance of the different machine-learning algorithms is also presented. Furthermore, an overview of the results in determining the extent of crack damage using image processing techniques is also presented. This chapter presents results obtained in accordance with the research question and objectives established at the beginning of the investigation. Furthermore, this chapter also focuses on the analysis and interpretation of the data obtained from the defect classification results and those obtained from the image processing methodology utilised to determine the extent of crack damage. This chapter seeks to provide an in-depth understanding of the research results and their implications for structural health monitoring.

Chapter 5: Conclusion and recommendations: A complete summary of the research findings is provided, including the key outcomes of the research. The analysis and interpretation of the data gathered from the research results are summarised, casting light on the major findings that address the research question and objectives. The implications of these findings are examined, highlighting their significance for the structural health monitoring and machine learning sectors. Moreover, recommendations for future research are presented based on the findings and limitations, with prospective areas of exploration and improvement indicated.

Chapter 2 Literature review and theory

This chapter presents an overview of the research topic, providing information and insight into structural health monitoring. The chapter covers the different methods of damage classification and the different machine learning and deep learning algorithms implemented to carry out the structural health monitoring of heritage structures. Image processing techniques are also discussed and methods employed to develop the computer vision-based detection approach are presented. An analysis of the literature and previous studies conducted in the field of machine learning and deep learning is also presented. These sections are important in understanding machine learning and deep learning algorithms that were used to identify wall defects in South African heritage structures.

2.1 Introduction

As heritage structures deteriorate over time, intervention in the form of structural health monitoring and repair and maintenance strategies is necessary to protect them and the cultural and historical value they hold. Heritage structures hold cultural and historical value and contribute to the economy, therefore monitoring them and preserving them becomes even more important. Research has focused on structural health monitoring and different machine learning techniques and their use to preserve heritage structures. This has been done to enhance the effectiveness of monitoring and assessment techniques using machine learning algorithms, leading to improved repair and maintenance approaches. This chapter provides a review of the literature and research based on current structural monitoring techniques as well as machine learning and deep learning algorithms used in this research.

Additionally, current image processing and machine learning methodologies implemented in the structural health monitoring field are presented.

2.2 Heritage structures

Heritage structures are buildings, monuments or sites with significant cultural, religious, political and historical value. Heritage structures represent important historical and religious periods or historical events, which makes them great assets (Loke, 2020:1). These structures contribute to the heritage and identity of different regions and communities.

Heritage structures often have unique and distinct architectural and physical features that represent the specific historical period and cultural influences. They are also built using construction materials such as lime-based mortars, limestone, sandstone and timber, which reflect that era's available resources and construction techniques.

The role that heritage structures play in countries' economies cannot be overlooked, as they promote economic development and growth through tourism and job creation. These structures often become iconic landmarks and tourist attractions, drawing visitors from around the world, with the Pyramids of Giza, the Great Wall of China and the Colosseum in Rome being prime examples. Furthermore, these structures serve as educational resources, providing valuable insights into political, religious and cultural history.

Over time, heritage structures are exposed to a combination of loads that are detrimental to their structural integrity and can lead to the degradation of the structure. Figure 2.1 shows the different typologies of damages in heritage structures in the Western Cape, South Africa.

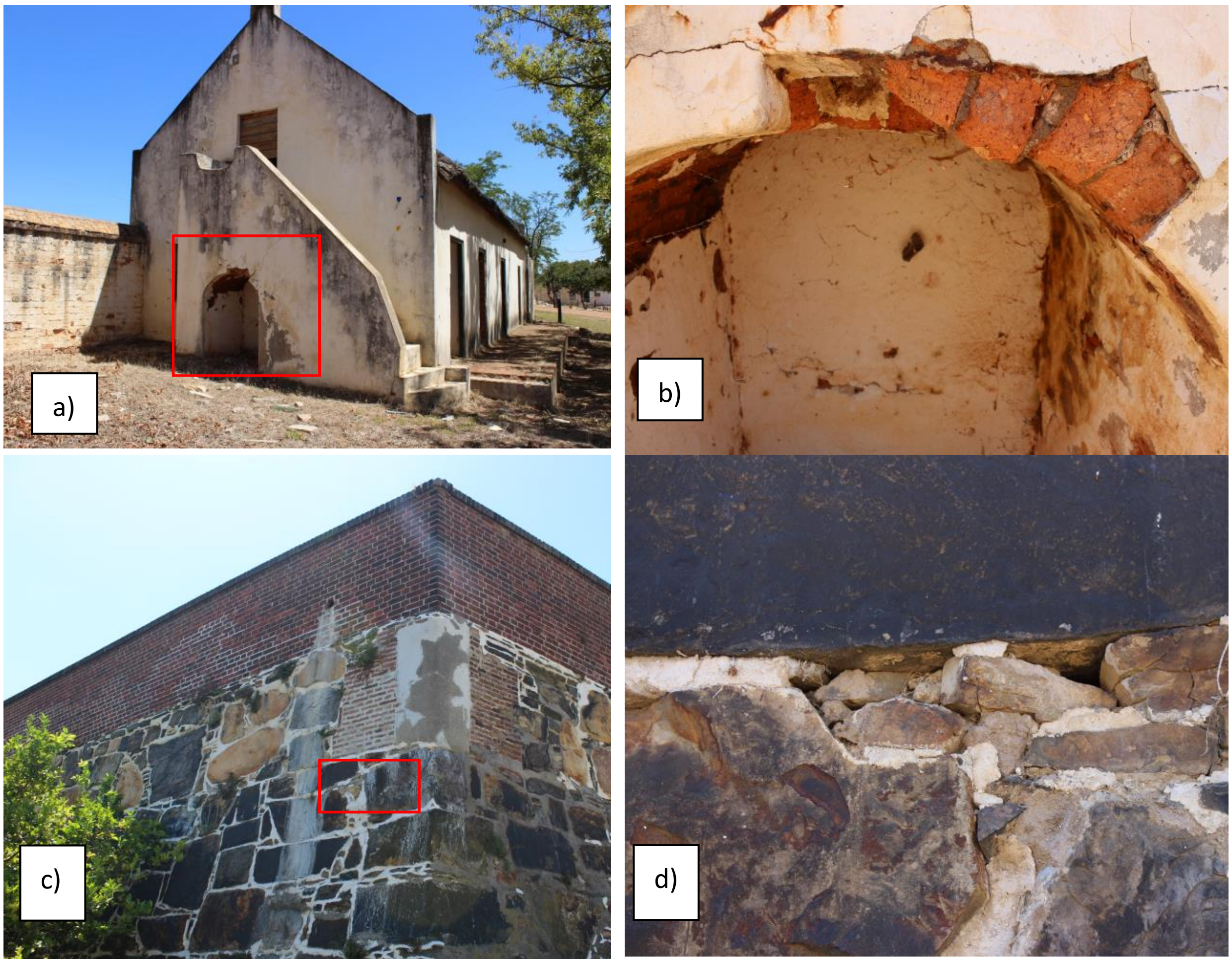


Figure 2.1 Different typologies of damage typical to heritage structures: (a) damage to the cottages at Dal Josafat Heritage Site, (b) a close-up view of the mortar loss, exposed brickwork, and some surface cracks at the cottage, (c) the mortar repair work at the Castle of Good Hope, d) visible damage at the mortar repair.

These damages include loss of mortar or plaster, exposed brickwork, cracks, incompatible mortar repair and the resulting damage.

According to Ćurčić et al. (2020:17), various methodologies and legislation have been put in place to protect and preserve heritage structures. These include:

- Transforming the structure to carry out new functions or restoring its original functions.
- Reconstruction of damaged or destroyed original areas whereby an attempt is made to replicate the original design or where new parts are implemented but are in alignment with the existing aesthetic and architectural environment.
- Repair and maintenance or preventative maintenance through the use of continual condition assessments and structural health monitoring.
- Dislocation, where an original part is moved or removed for protection in another area.
- Implementing a new construction or component into an existing historical structure.

- Protection and conservation in its original condition through the use of legislation.

Failure to properly maintain these irreplaceable heritage buildings can lead to frequent repairs and a loss of authenticity during restoration. This, in turn, results in a decline in tourism and consequent detrimental effects on the economy, making it imperative to prioritise their protection (Loke, 2020:7). The preservation and protection of heritage structures is a multifaceted undertaking that demands a deliberate and cautious approach. However, it is crucial for the long-term sustainability of cultural, historical and architectural heritage.

2.2.1 Heritage structures in South Africa

South Africa has a diverse history and is frequently described as a "rainbow" or multicultural nation. The cultures that influenced the South African culture include Dutch, British, Malay, and an array of African cultures, and this is reflected not only in its people but also in the heritage structures throughout the country. The features of these structures demonstrate the rich blend of cultures that has shaped South Africa's architecture.

The country has an abundance of heritage structures, some dating as far back as the 17th and 18th centuries. In South Africa, the declaration of heritage sites as national or provincial assets is determined by the South African Heritage Resources Agency (SAHRA), an agency of the Department of Arts and Culture. These sites are recognised for their historical value and typically serve as important tourist attractions. Some well-known heritage buildings around the country include the Union Buildings, the Castle of Good Hope, the Parliament Building, the Voortrekker Monument, the South African Astronomical Observatory, and Dal Josafat (South African Resource Agency, 2020).

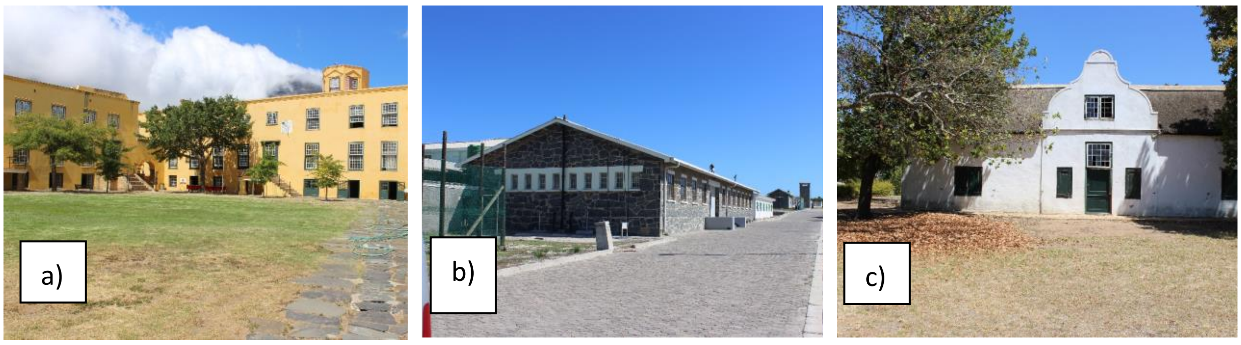


Figure 2.2 (a) The inner walls of the Castle of Good Hope, (b) the maximum-security prison on Robben Island and (c) Roggeland cottages in Dal Josafat.

Figure 2.2 shows some of the national heritage sites mentioned in Cape Town, South Africa. Figure 2.2 a) shows the Castle of Good Hope with the image taken from inside the castle walls, while Figure 2.2 b) shows the maximum-security prison located on Robben Island and Figure 2.2 c) provides one of the farmhouses on the Roggeland farm, which forms part of Dal Josafat.

According to the United Nations Educational, Scientific and Cultural Organisation (UNESCO) database, there are also many cultural heritage sites on the world heritage list. These include the Fossil Hominid Sites of South Africa, the Mapungubwe Cultural Landscape, the Richtersveld Cultural and Botanical Landscape, Robben Island, and the Khomani Cultural Landscape (United Nations Educational, Scientific and Cultural Organisation, 2021).

2.3 Overview of structural health monitoring

Structural health monitoring is the process of monitoring structures to identify changes that may suggest degradation or any form of damage. According to Turek (2007:9), the term "structural health monitoring" is defined as a technique for assessing the state of a structure using some form of quantity measurement on that structure. The condition could be anything related to the structure, such as loads, cracks, corrosion, etc. Strains, loads, acceleration, and other variables can be used as measured quantities and a primary tool to offer information on the condition of structures (Housner et al., 1997:943). The most important component of this concept is that there must be two primary components to a functional, structural health monitoring system: a measurement aspect, which involves deploying monitoring devices to collect data on parameters such as strains or environmental conditions; and the analysis aspect, which processes the collected data using statistical analysis and pattern recognition (Turek, 2007:9).

Mishra (2021:227-245) provided an in-depth account of various state-of-the-art structural health monitoring techniques and their applications on heritage structures. For example, Valero et al. (2019:1-14) measured the extent of material loss due to weathering and biological decay of ashlar walls of the Chapel Royal heritage structure in Stirling, Scotland. The process used a terrestrial scanner, point cloud data, and a logistic regression machine learning algorithm. Another case study included that of Sharma et al. (2019:347-359), where dust deposition was monitored on heritage monuments such as the Statue of Liberty, Taj Mahal and Leaning Tower of Pisa using an image processing and CNN methodology.

The use of non-destructive structural health monitoring techniques plays an important role when dealing with heritage structures not only because of their complexity, but also because of the historical and cultural significance they hold. As these structures deteriorate over time, damage such as spalling, cracks and efflorescence are reflected on the surface of the structure. Identifying and localising this damage quickly and efficiently is of the utmost importance so that proper maintenance and management of the historic structures can be carried out.

2.3.1 Causes of damage in heritage structures

According to Feilden (2003:92), numerous factors influence durability and affect the structural integrity of heritage structures by damaging their construction materials. These factors include climatic causes, biological processes, natural disasters and man-made causes. Feilden (2003:92) states that the deterioration results from chemical, physical, mechanical, and biological processes, which often occur simultaneously.

The deterioration of heritage structures is often further accelerated due to atmospheric pollution. Atmospheric pollution arises from substances or contaminants like sulphur oxides and nitrogen oxides. When these substances combine with water, they give rise to acidic solutions that can affect the strength of construction materials, when they come into contact (Loke, 2020:12).

Furthermore, Feilden (2003:93) suggests that climate and environmental conditions are the primary contributors to the deterioration of heritage structures. Sources of damage as a result of climatic and environmental conditions include:

- Thermal movement: according to Feilden (2003:93), heritage structures are often constructed with different materials, each with differing thermal expansion and contraction rates. Temperature fluctuations or changes can lead to stresses within the structure caused by cycles of expansion and contraction, which can cause cracks to develop and other forms of damage.

- Moisture and precipitation: the presence of excessive moisture in the form of rain or snow can accelerate the erosion of the materials which make up the heritage structure. It can also cause the corrosion of steel and promote the growth of mould and other botanicals.
- Freezing and thawing: in areas where temperatures below freezing are prevalent, water can penetrate cracks or crevices, and when the water freezes, it expands, breaking down materials over time.
- Sunlight and UV radiation: Feilden (2003:93-94) states that prolonged direct sunlight can lead to fading of the material, while in some cases, materials can become brittle and crack.
- Wind: strong winds, especially when combined with airborne debris, can physically wear down the exterior of a building, causing erosion and damage to surfaces.
- Salt attack or saline environments: structures constructed near coastlines are particularly vulnerable to saltwater exposure. This can cause the corrosion of steel and metal elements or break down construction materials through salt crystallisation cycles.

Table 2.1 below shows the factors contributing to damage in heritage structures, including climatic, biological, botanical, and human-induced causes, along with natural disasters or events, according to Feilden (2003:92).

Table 2.1 Causes of decay to cultural property (Fielden, 2003:92)

Climatic causes	**Biological and botanical causes**	**Natural disasters**
Seasonal temperature changes	Animals	Tectonics
Daily temperature changes	Birds	Earthquakes
Wind	Insects	Tidal waves
Precipitation of rain and snow	Trees and plants	Floods
Ice and frost	Fungi, moulds, lichens	Landslides
Ground water and moisture in soil dust	Bacteria	Avalanches
Dust		Volcanic eruptions
		Exceptional winds
		Wild fire
Man-made causes of decay		
Neglect of preventive conservation	Wars: ethnic cleansing	Enviromental pollution
Theft	Purposeful alterations	Water abstraction
Neglect of fire precautions	Encroachments	Vibrations
Neglect of security precautions	Fashion changes	Vandalism and arson

As a result of the processes presented in Table 2.1, the strength and durability of the construction materials that make up the heritage structure can deteriorate significantly. This can lead to a notable reduction in the structure's structural integrity, making it increasingly vulnerable to environmental factors or natural events.

According to Raszczuk and Karolak (2021:2), other sources of damage and threats faced by heritage structures include:

- Ground subsidence: this occurs when the ground substrate sinks, as seen in the Leaning Tower of Pisa and the Church of St. John in Gdansk (Burland et al., 2003; Topolnicki, 2001).
- Design and construction errors: errors made during the initial design and construction phase, such as in the Church of St. John in Gdansk case, resulted in the structure being constructed on weak founding soil (Topolnicki, 2001:1864).
- Unforeseen or additional loading: this includes unexpected factors like fires, explosions, and seismic activity, which can add additional stress to structures. Examples of these include seismic events which damaged the Hagia Sophia in Turkey, as well as the damage to the Notre-Dame de Paris Cathedral in France as a result of a fire (Ďurčíková and Vidholdová, 2021; Almac et al., 2014).

2.3.2 Conditional assessment of structures

Condition assessment of structures is a method used to establish the condition of the structural components of a building or heritage structure, to ensure structural integrity and avoid future failure (Noor et al., 2019:1). Noor et al. (2019:2) state that the goal of a conditional assessment is to evaluate the condition of key structural elements, such as the beams, columns, foundations or walls. Furthermore, it assists in quantifying defects and provides information such as the deterioration level of structural components. Additionally, the assessment provides data that can be utilised to create a budget that details maintenance and repair costs (Noor et al., 2019:2).

The conditional assessment usually includes reviewing documents such as engineering and architectural layouts. This is carried out to understand the initial design intent of the structure, gain knowledge about the structural system used, the intended load paths, and construction materials employed, and assess whether the structure has been subjected to any changes or alterations that may impact its performance or structural integrity. Other documentation, such as warranties and service contracts, are also reviewed (Noor et al., 2019:2). However, instances do exist where design information and layouts are not available, which is frequently seen in heritage structures. Furthermore, the condition assessment report should provide a detailed list of all structural and architectural defects, a severity rating for these defects, an overall rating of the building's condition, and a repair and maintenance methodology. This information is usually provided after the visual inspection is completed, non-destructive testing is performed, and an analysis of the structure is carried out. Examples of non-destructive tests that can be completed include utilising a rebound hammer or a concrete pull-out test to determine the concrete's compressive strength. Additionally, the ultrasonic pulse velocity test can be used to test the integrity and overall quality of the concrete and the concrete compressive strength.

The following steps should be carried out when completing a conditional assessment of heritage structures, according to the Guide for Structural Rehabilitation of Heritage Buildings (Santos, 2010:9):

- Gathering documented data and historical information about the building
- A detailed survey and visual inspection of the building
- Detailed assessment completing any in situ non-destructive tests such as concrete strength testing using the Schmidt hammer or rebound hammer
- Diagnosis of defects
- Structural safety assessment, including safety level, severity rating of defects and quantification of material strength
- Design of solutions for intervention
- Execution of intervention

Given that most heritage buildings have existed for many years and have either sustained significant damage or deteriorated, conditional assessments provide a form of intervention, ensuring the structure's safety and also serving as a basis for the preliminary estimate for any future repair and maintenance work.

2.3.3 Challenges of conditional assessments

Despite the need for conditional assessment of heritage structures, engineers face unique challenges when evaluating the structural integrity of heritage structures. According to Professional Engineers Ontario (2016:12-13), some of the challenges include:

- Some of the heritage buildings inspected predate the publication of design standards and provincial building codes;
- building materials used and construction methods applied during the construction of these buildings are not commonly used or referenced in standards and available texts; and
- due to the age of the structures, design information and layouts are not always readily available.

This guideline also states that inappropriate and inaccurate assessments have negatively impacted heritage structures, as they have caused incorrect repair strategies to be applied. According to Loke (2020:38), the primary emphasis in the restoration or repair of heritage structures is placed on preserving their authenticity, and this presents a considerable challenge when there is limited knowledge regarding the analysis of historic materials for restoration purposes. Furthermore, Loke (2020:38) highlights the importance of considering the material properties and not only the visual aspects. This is because improper repairs to historic materials such as mortar can harm the structure. This includes premature damage and accelerated deterioration of the original building materials. In many cases, the restoration of heritage buildings results in further damage, thereby increasing the expenses associated with future repairs (Loke, 2020:38).

Common instances encompass the restoration of heritage cementing materials by using Portland cement. Nevertheless, this approach frequently results in inefficacy. The ineffectiveness of Portland cement-based mortar in structural repairs, particularly in cases where historical lime-based mortars were originally employed, is rooted in the inherent constraints of Portland cement. These limitations include a tendency to crack and corrode the original fibres more rapidly, as Portland cement contains higher levels of soluble salts that cause salt crystallisation. Replacing the original material with Portland cement has been found to exacerbate the damage to the original fibres within the mortar (Marini et al., 2018:802-803). Loke (2020:39) states that this type of repair has occurred at the Robben Island heritage site, where buildings on the island have lost their historic appearance due to the repairs completed. Loke (2020:39) also states that the repairs included materials that are harmful to the overall fabric of the structure.

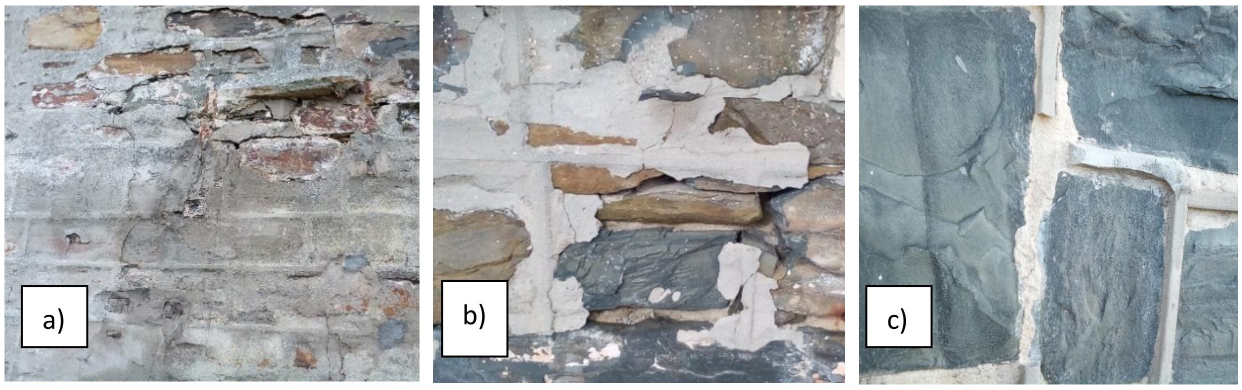


Figure 2.3 Failure in mortar repairs at Robben Island heritage site (Loke., 2020:48): (a) the deterioration of the newly applied mortar, (b) the incorrect application of the Portland cement mortar and (c) the failure of the mortar where it has separated from the original material.

Figure 2.3 illustrates the incorrect repair methodology implemented at the Robben Island heritage site, where the newly applied mortar is failing and separating from the original masonry and the original lime and shell mortar. According to Loke (2020:48), the Portland cement mortar is impermeable and has been applied on top of the original material. This has resulted in moisture ingress and movement between the materials, causing the Portland cement to fail and separate.

Another example includes Norway's Nidaros Cathedral heritage structure, constructed between the 11th and 14th centuries. The heritage structure has been restored for over a century, beginning in 1872. The restoration included the replacement of the soapstone walls, which started to erode and disintegrate. According to Aasly et al. (2017:1), despite the initial restoration effort, the process is still ongoing due to the continuous weathering of specific soapstone types that were incorrectly selected and utilised. According to Aasly et al. (2017:1), the enduring challenge arises from locating soapstone resources that meet the exacting quality standards demanded for restoration endeavours. This predicament has contributed to the protracted duration of restoration initiatives. The effective discovery of fresh soapstone resources is contingent upon a profound grasp of geological factors, extensive quarrying proficiency, and the mastery of stone carving experts.

An engineer with an accurate understanding and expertise in heritage structures must inspect the structure to ensure a practical assessment and the proper repair methodology.

2.3.4 Non-destructive testing

Non-destructive testing procedures are techniques used to determine material mechanical properties or properties of various components and systems without causing damage (Umar et al., 2015: 327). Non-destructive testing carried out on heritage structures primarily focuses on assessing the heritage structure's structural integrity. According to Umar et al. (2015: 327), when testing is carried out on new structures it is completed for quality control purposes or to confirm the quality and strength of the materials used. The non-destructive testing approach differs from destructive testing, which involves taking cores or samples that are then tested. This is completed to understand the material's behaviour and performance leading up to failure. Non-destructive testing can serve as a preliminary step before destructive testing is conducted.

According to Umar et al. (2015: 327), some popular methods of non-destructive testing include:

- The rebound hammer or Schmidt hammer test is conducted to assess the surface hardness of concrete.
- The half-cell electrical potential method is employed to determine the corrosion potential of reinforcing steel within concrete.
- The carbonation depth measurement test assesses the depth of moisture penetration in concrete and whether it has reached the reinforcing steel. This test helps evaluate whether reinforcing corrosion occurs within the concrete.
- The permeability test measures the water flow through concrete, providing valuable information about its permeability characteristics and potential issues related to water penetration.
- The penetration resistance or Windsor probe test is another method used to determine the surface hardness of concrete.
- The ultrasonic pulse velocity test is conducted to assess the compressive strength of concrete. This test measures the velocity of ultrasonic waves passing through the concrete, measuring its sound velocity and compressive strength.
- Ground penetrating radar or impulse radar testing is employed to determine the position of reinforcing steel within concrete. This non-destructive testing method utilises radar waves to locate and map the distribution of reinforcement within concrete.

Non-destructive testing techniques are popular in the heritage field as they cause minor to no damage to the unique historic materials. Non-destructive testing is preferred to destructive testing when working with heritage structures, as destructive testing requires multiple random samples to be taken and tested. This process is not only costly but can also cause damage to the historic building materials of the heritage structure. Furthermore, when different non-destructive testing techniques are used in conjunction, they can provide more accurate and useful information about the condition of the material tested and the condition of the structure as a whole.

2.3.5 Image processing in structural health monitoring

Visual inspections conducted by a professional are a common approach to structural health monitoring for damage detection. This approach offers several advantages, including its non-destructive nature, minimal equipment requirements, and adaptability to complex or challenging environments, allowing immediate on site judgements and decisions to be made. However, it is important to note that visual inspections are costly as they require a high level of professional expertise, and inspectors must possess a comprehensive understanding of the observed defects and the necessary repairs. This form of inspection may also be unreliable as serious damage may not be detected simply due to human error.

The introduction of machine learning provides a solution to some of the problems the visual inspection method faces. Machine learning forms a subset of artificial intelligence that enables a machine to carry out an intelligent task by learning from errors to become more adept over time. Pettit et al. (2021:729), states that by using statistical models that find patterns in data and drawing conclusions, machine learning adapts and can learn iteratively without human feedback. Deep learning is a subset of machine learning that employs different artificial neural network architectures to extract and analyse data features (Pettit et al., 2021:729). When implemented correctly, artificial intelligence, machine learning and deep learning can provide affordable and practical solutions that are superior to those obtained through conventional means.

Machine learning methods have been successfully integrated into engineering activities. Ye et al. (2016:3) state that two-dimensional structural displacement monitoring can take place by making use of image processing techniques such as edge detection algorithms and image sequencing to monitor the structural displacement of a target. Displacement monitoring was proposed by Feng et al. (2015:16558), who proposed a vision system for the measurement of non-contact structural displacement. The system used an advanced template matching machine learning algorithm in real time to measure displacement. The deformation of building structures was monitored using digital image processing by Henke et al. (2015:141). Close-range photogrammetry was used to measure the deflection occurring in bridges by Jauregui et al. (2003:212), with Yoneyama et al. (2007:34) completing similar tests, also measuring the deflection of a newly constructed steel girder bridge under load tests, using digital image correlation.

Two or more cameras can also obtain three-dimensional structural displacement. This allows two or more different image sequences to be obtained from different shooting angles. This enables the monitoring of the three-dimensional displacement of structures to be obtained using vision reconstruction techniques (Ye et al., 2016:4). This was carried out by Park et al. (2015:1), where three-dimensional displacements were monitored with the help of a high-speed motion capture system.

Vision-based systems were also used to obtain structural stress and strain. A method for assessing steel strain on reinforced concrete members was developed by Carmo et al. (2015:265), using surface measurement aided by photogrammetry and image processing. Another method to measure stress and strain was completed by Gales et al. (2012:476), where a digital image correlation method was introduced to measure prestressing steel strain and stresses during high-temperature tests.

The current advances in image processing technology have allowed surface defects to be analysed from imagery. A visual inspection technique for bridges was proposed by Yeum and Dyke (2015:759), whereby captured images were analysed through an automated process for detecting cracks near bolts. Furthermore, Liu et al. (2016:2) based a surface crack monitoring and assessment method on adaptive digital image processing.

Non-destructive digital classification techniques are primarily employed to diagnose buildings such as historic structures and classify damages. The scientific community has also shown much interest in using 3D digital models for cultural heritage structures. Much work has also been completed in attempting to automate the process of 3D modelling historic structures as they are useful for visualisation and monitoring of structures, digital archiving, as well as conservation and preservation purposes (Oses et al., 2014:1864).

Merono et al. (2015:2) evaluated an object-oriented classification technique by using photographs captured by a digital single-lens reflex camera. The images obtained were classified and integrated into a three-dimensional model created using terrestrial laser-scanned data for detecting and localising damaged stones in the Santa Marina Church, in Córdoba, Spain. Anton et al. (2019:2) followed a similar methodology instead of using terrestrial laser-scanned imagery to model the Basilica of the Baelo Claudio, in Tarifa, Spain. This was completed to justify the need for accurate heritage structural modelling rather than having a simplified approach when carrying out a finite element analysis approach.

Crespo et al. (2010:185) utilised laser scanning equipment and a digital camera to gather intensity data and colour information for defect detection at the Santo Domingo ruins in Galicia, Spain. Digital image processing was used to identify damage to granitic rock, a material used in several historic buildings.

Several unsupervised classification algorithms such as the Isodata, k-means, and fuzzy k-means algorithms were then used to analyse and classify data. The results of the study consisted of classified images depicting various types of damage affecting granite rock. Additionally, through post-analysis, thematic maps were generated, providing the damage's size and position.

A method of investigating architectural surfaces not often used is automated mapping. The work completed by Adamopolous (2021:40) uses a combination of reflectance imaging and supervised segmentation to automatically segment deterioration patterns in multi-spectral imagery. The spatial and semantic description of the deterioration patterns is facilitated by automated thematic mapping. The method is based on machine learning principles using various image classification techniques such as regression and decision trees. The Fort of Karababa in Euboea, Greece, was chosen as the historical structure for implementing and evaluating this proposed methodology.

2.3.6 Other techniques used in the structural health monitoring field

Other approaches applied in the structural health monitoring field include the virtual damage approach applied by Nguyen et al. (2019:1). This approach made use of data obtained from an undamaged, real-world structure where virtual data was generated for the damaged state. Nguyen et al. (2019:182) made use of a finite element method to calculate transmissibility functions from a simulation of vibration responses. These vibration responses were obtained from a bridge at various points while under the load of a moving truck. The functions obtained were used as input data for the training of an artificial neural network. The successful training allowed damage to be quantified and localised.

Tibaduiza et al. (2011:1-2) made use of a physical damage approach to simulate the damaged and undamaged state of a real-life structure. The study made use of a small-scale model and piezoelectric transducers which applied vibrational excitations to the model. Principal component analysis was applied to detect and localise damage. This was completed by applying its pattern recognition abilities, whereby a comparison could be made between the current state of the structure and a baseline model, where no damage was experienced. Subsequently, any changes or damage which had occurred could be determined.

Carrasco (2021:8-14) employed a novel image processing based method combining binarisation, segmentation, skeletonisation and k-means clustering to determine the width of cracks in digital images. The research centred around earth based construction material samples and compared the image processing results to those achieved through manual measurement. Through this comparison Carrasco (2021:13) found the physical crack measurements and those obtained through image processing to vary between 0.75mm and 2.25mm with accuracy dependent on image resolution.

2.4 Machine learning concepts

Machine Learning algorithms are becoming more popular as data becomes available due to significant information technology advancements. As a result, machine learning combines complex algorithms with conventional data analysis approaches to process large-scale data. Machine learning has been used to enhance and support a variety of fields such as engineering, medicine, and science (De Castro Mota, 2021:5).

2.4.1 Overview of machine learning methods

Machine learning algorithms allow us to solve issues and tasks which are very difficult to solve using conventional programmes designed and written by humans (Goodfellow et al., 2016:99). These algorithms analyse data, allowing features such as image, text, numerical, statistical and time-series features to be extracted from the data.

Machine learning algorithms can be categorised as either supervised or unsupervised, depending on the type of learning methods and datasets experienced during training. According to Tibaduiza et al. (2018:2), the supervised learning and unsupervised learning approaches can be described as follows:

- Supervised learning: according to Tibaduiza et al. (2018:2), in this method, the machine learning algorithm is trained to make predictions based on labelled data provided by the researcher. Tibaduiza et al. (2018:2) further state that the algorithm is provided with the input and expected output data. In this form, the input data represents the data features while the output provides the target variable or outcome. The algorithm can then be trained to find the patterns and relationships between data. The goal is to learn a mapping function that can effectively classify the output when given new data.
- Unsupervised learning: with this learning method, the machine is trained to find patterns and similarities in data without labelled data (Tibaduiza et al., 2018:3). The algorithm therefore finds patterns in the data without being explicitly told what to look for. In this way, unsupervised learning finds insights into data to create an efficient model.

Goodfellow et al. (2016:105) state that supervised learning uses labelled data to establish a relationship or function. The function can then be used for the mapping of new unlabelled data. Artificial neural networks, convolutional neural networks (CNNs), logistic regression, decision trees, support vector machines (SVMs) and k-nearest neighbours are examples of approaches that construct a model using a training dataset with known input and output values. Supervised learning methods need a sufficient quantity of labelled data to achieve good performance (Baladram et al., 2020:87). Supervised learning methods are therefore a means for hidden pattern detection in unlabelled data.

According to Baladram et al. (2020:88), supervised learning methods produce a predictor that accepts input data and outputs a prediction result against the input data. Two primary phases are incorporated into this procedure: learning and prediction. The system reads the input and target vectors and improves the rules during the learning phase. This process is repeated until the rules produced are of sufficient quality to achieve good prediction performance for a sustained period. The system predicts the properties or features of new data in the prediction phase.

A subset of machine learning is deep learning. Sarker (2021:2) states that deep learning refers to multiple layers of data processing which must occur for a data-driven model to be created. The deep learning process uses multi-layer neural networks to process data (Sarker, 2021:3).

According to Goodfellow et al. (2016:99-100), machine learning can address various problems; the most popular being classification and regression. These problems can be further broken down into categories such as binary and multi-class classification, scalar and vector regression, and multi-label classification.

2.4.2 Classification

Classification aims at determining and identifying which set of k categories an observation or input may belong to, where k is the number of possible outcomes. To carry out this process, a function of the form shown in Equation (2.1) must often be produced by the machine learning algorithm.

$$f: \mathbb{R}^n \rightarrow \{1, \dots, k\} \tag{2.1}$$

An input described by the vector x is assigned by the model to a category identified by a numeric code y, when:

$$y = f(x) \tag{2.2}$$

The algorithm is used for several applications, such as object recognition (De Castro Mota, 2021:6).

Challenges arise when every input is not provided to the algorithm. This is also known as classification with missing inputs. Instead of supplying a single classification function if some of the inputs are missing, the machine learning algorithm must learn a set of functions. Each of these sets of functions corresponds to classifying x with a different subset of its missing inputs. To effectively define a large set of functions like this, a probability distribution must be learnt over the various necessary variables (Goodfellow et al., 2016:100). According to De Castro Mota (2021:6), once this is complete, the classification task should be solved. This can be completed by marginalising the various missing variables. Furthermore, the 2^n different classification functions required for the different sets of missing inputs can be obtained when considering n input variables, where n is the number of occurrences. However, the system needs to learn a single function expressing the joint probability distribution.

2.4.3 Regression

Goodfellow et al. (2016:101) state that the algorithm is given some input value where a numerical value must be predicted in the regression process. The task can be solved when the machine learning algorithm produces the output function shown in Equation (2.3).

$$f: \mathbb{R}^n \rightarrow \mathbb{R}. \tag{2.3}$$

The function (2.3) is similar to the classification process with only the output format differs.

Typically, these are problems which can be solved using supervised learning. They can generally be classified into either classification or regression problems. When dealing with classification problems, a class of the input data can be predicted while regression predicts a numerical value (De Castro Mota, 2021:7).

Unsupervised learning is the process whereby patterns in relationship data points are identified. No output variables are to be predicted, with only input data previously known (De Castro Mota, 2021:7). According to Goodfellow et al. (2016:105), unsupervised learning algorithms can be used when learning the whole probability distribution that produced a dataset. This can be completed explicitly, for example density estimation. The process can also be carried out implicitly for tasks like denoising or synthesis. Some examples of unsupervised learning algorithms are autoencoders, principal component analysis, and independent component analysis. Unsupervised learning includes clustering as well such as hierarchical and k-means clustering.

According to De Castro Mota (2021:8), numerous examples of a random vector x are observed in unsupervised learning methods, and an attempt is made to learn the probability distribution $p(x)$ or various properties. The supervised learning method differs from unsupervised learning in that numerous examples of random vector x with a value or associated vector are observed, with y then predicted from x, through the estimation of $p(y|x)$.
In essence, in supervised learning, the target value y is provided by the researcher, who then shows the machine learning system how to achieve its target. Unsupervised learning analyses and clusters unlabelled data and discovers patterns without guidance or human intervention.

As the main objective of the research was the classification of different defects, a multi-class classification problem was presented. Sufficient input data was also provided with a well-defined output. Therefore, a supervised learning approach was implemented in this research, along with suitable machine algorithms that utilise supervised learning approaches.

2.4.4 Support vector machine

Cortes and Vapnick (1995:23-24) introduced SVMs as an algorithm used for two group classification problems. Cortes and Vapnick (1995:24) highlighted the performance of the machine learning approach in comparison to machine learning processes. A line function defines the model $\boldsymbol{w}^T\boldsymbol{x} + b$, in a similar manner to that of logistic regression, where $\boldsymbol{w}$ is the weight parameter, b is the bias parameter and $\boldsymbol{x}$ is a dimensional input vector. SVMs, unlike logistic regression, do not output probabilities but do produce a class identity. A positive or negative class is predicted by the support vector machine (SVM) when $\boldsymbol{w}^T\boldsymbol{x} + b$ is either positive or negative, respectively.

The kernel trick forms a vital innovation relating to SVMs. According to Goodfellow et al. (2016:141), dot products can be used to express the machine learning algorithm and the kernel trick involves observing this. The linear function SVMs use can be written as Equation (2.4).

$$\boldsymbol{w}^T\boldsymbol{x} + b = b + \sum_{i=1}^{m} \alpha_i\, \boldsymbol{x}^T\boldsymbol{x}^{(i)} \tag{2.4}$$

In this form, $\boldsymbol{\alpha}$ is a vector of coefficients while $\boldsymbol{x}^{(i)}$ is a training example. Furthermore, this allows the algorithm to be rewritten, replacing $\boldsymbol{x}$ with the output of the given feature function $\emptyset(\boldsymbol{x})$. The dot product can be replaced with a function $k\left(\boldsymbol{x},\boldsymbol{x}^{(i)}\right) = \emptyset(\boldsymbol{x}) \cdot \emptyset(\boldsymbol{x}^{(i)})$ referred to as a kernel. Goodfellow et al. (2016:141) state that the operator represents an inner product analogous to $\emptyset(\boldsymbol{x})^T\emptyset(\boldsymbol{x}^{(i)})$. The vector inner product may not be used for some feature spaces; for other dimensional spaces, other inner products must be used, such as those based on integration.

Predictions can be made when replacing dot products with kernel evaluations and being given by Equation (2.5).

$$f(\boldsymbol{x}) = b + \sum_{i} \alpha_i\, k(\boldsymbol{x},\boldsymbol{x}^{(i)}) \tag{2.5}$$

Goodfellow et al. (2016:141) state that applying the kernel-based function is equivalent to pre-processing by applying $\emptyset(\boldsymbol{x})$ to all inputs. This is followed by linear model learning in the newly transformed space.

The kernel trick is important for various reasons, such as providing models with the ability to learn even though they are nonlinear as a function of $\boldsymbol{x}$**,** using different techniques such as convex optimisation, where efficient convergence is assured. This is because only $\boldsymbol{\alpha}$ is optimised while $\emptyset$ is considered to be fixed.
SVMs are therefore capable of dealing with nonlinearly separable cases. Linear separability can be achieved, as illustrated in Figure 2.3 by applying the kernel trick (Hoang, 2018:3). This method allows the mapping of data points to a high dimensional feature space, resulting in linear separability.

The radial basis function is the most commonly used kernel. The kernel is also referred to as the Gaussian kernel. The kernel trick is also applied in other linear models with this category of algorithms referred to as kernel machines.

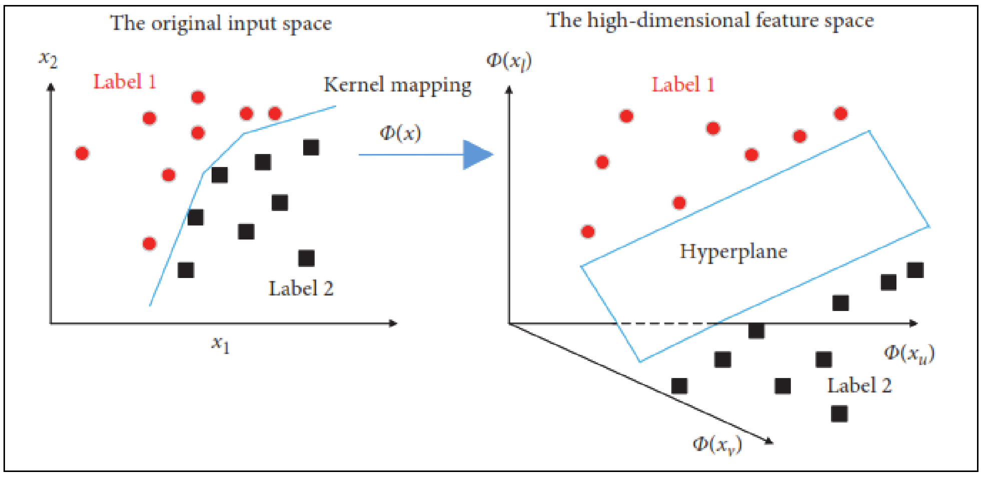


Figure 2.4 The learning phase of an SVM showing the applied kernel (Hoang, 2018:8).

In Figure 2.4, data is split into two classes, Label 1 and Label 2, and plotted on a plane. The SVM can then take the given data points and produce a hyperplane which separates the classes. The SVM optimises the decision boundary by producing a boundary where the maximum distance from the closest data of the classes is given. Data plotted or positioned on either side of the hyperplane can be categorised as either Label 1 or Label 2. The line, therefore, forms a boundary between the classes.

2.4.5 Decision trees

Another form of supervised learning is the decision tree. This decision tree helps to decide a certain problem and is extremely intuitive. Breiman et al. (1984:18-36;216-232), were early contributors and developed a classification and regression tree algorithm. Quinlan (1985:81), also presents another well-known method, the Iterative Dichotomiser 3 (ID3). According to Baladram et al. (2020:115), for predictions to be made, the decision tree uses a rooted tree.

Figure 2.5 is an example of the decision tree and shows how it is used to make predictions.

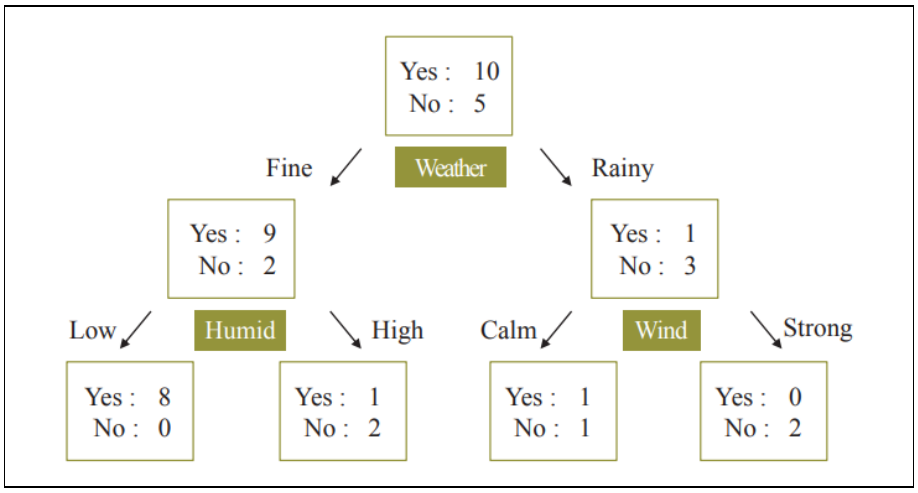


Figure 2.5 Decision tree example where the problem considered is going out for a picnic (Baladram et al., 2020:115).

In the tree structure in Figure 2.5, the box is called a node. Leaf nodes or leaves are the terminal nodes which do not have child nodes. Branches refer to the links between various nodes. The nodes present each have an occurrence, while each branch includes an option of an event or question, and lastly, a class label is presented by each leaf node. The path from the root to the leaves represents the classification rules.

Figure 2.5 helps to make a prediction using different information such as weather, humidity, and wind to determine whether conditions are suitable for a picnic. The boxes present the number of instances, with the top box showing a total of fifteen people, with ten having gone to the picnic while the remaining five did not go on the picnic. The left box in the middle layer shows a case where the weather was fine, where nine people went on a picnic while two did not. Furthermore, an attribute under the box presents a question, with the boxes further divided based on the question shown. The left box in the bottom layer shows that when humidity was low, eight people went on the picnic. Based on previous observations, you can decide whether to go on a picnic when the weather is fine and the humidity is low (Baladram et al., 2020:115).

Decision trees are therefore useful in solving both classification and regression-based problems. Furthermore, decision trees have a high level of interpretability in comparison to other machine learning methods such as SVMs and neural networks. Decision trees show relatively clear criteria as labels are obtained from instances, and it is easy to understand why the label is generated. This differs from typical black box methods like SVMs and neural networks, where it is difficult to determine how the predictor judges results (Baladram et al., 2020:116).

Principle of decision trees

The standard decision tree produces a binary value known as a label by using an instance of binary attributes. A question is stored in each internal node, while an answer is stored in each branch. Furthermore, starting at the root, a question is read, and an answer is selected from the branches, then a move can be made to the corresponding child node. This process is repeated until a label is obtained.

It is beneficial to first formalise a problem before generating a decision tree. For example, let X_k (where $k = 1, 2, \ldots . m$) be the set of attributes, while Y is the set of labels. Assume the labels and attributes are binary sets, for instance $X_1 = X_2 = \cdots = X_m = Y = \{0 , 1\}$. Baladram et al. (2020:116) state that a set of instances is the input for the decision tree as follows: $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)})\}$ where $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in X_1 \times X_2 \times \cdots \times X_m$ and $y^{(1)}, \ldots, y^{(n)} \in Y$.

A positive instance is produced when the label of an instance is one, while a negative instance is produced when the label of an instance is zero. When generating a decision tree, the main goal is therefore to produce a decision tree that can predict $y^{(i)}$ from $\mathbf{x}^{(i)} (i = 1, \ldots, m)$. Additionally, the decision tree should be able to work for instances where labels are unknown.

2.4.6 K-nearest neighbours

The k-nearest neighbour algorithm is a machine learning technique built on supervised learning principles. The algorithm is generally used for solving problems involving classification and regression. According to Mishra (2021:233), the algorithm is based on the principle that identical items will occur near each other. In other words, datasets which have similar attributes will be in proximity to each other.

One of the k-nearest neighbour algorithm's key characteristics is its ability to handle data without presuming a probability distribution or pattern in the dataset. However, good results can be obtained when prior knowledge about data distribution is combined.

This machine learning technique categorises an unlabelled dataset by using the majority label surrounding its closest k-neighbouring points. Metrics, including Euclidian, Manhattan, Cosine, Hamming, and weight, are used to determine the nearest neighbours. The algorithms' performance is determined by the metric that is used. In Equation (2.6) described below, the algorithm calculates the Euclidian nearest distance. This is then used to determine the number of nearest neighbours that are near the unclassified data point, for the various classes (Mishra, 2021:234).

$$ED(x, y) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \tag{2.6}$$

In Equation (2.6), the data point already classified is given by x_i, while the data point which still needs to be classified is given by y_i and lastly, the value N is the number of attributes. Mishra (2021:234) states that the algorithm is trained with the intention that k-nearest neighbours belong to the same class. However, a wider margin separates those that belong to different classes. Competitive results can be achieved when using this technique despite its simplicity. However, its performance is dependent on the k value used. The k value needs to be configured but is usually an odd number between 3 and 10, with this value indicating the number of nearest neighbours.

2.4.7 K-means clustering

According to Ghalyan et al. (2019:3902), clustering is a method of vector quantisation, with the main purpose ultimately being the partitioning of a dataset. Using clustering, the dataset is generally divided or grouped according to similarities or common attributes. This form of machine learning is carried out using unsupervised learning, where clustering can occur without labelling the dataset. According to

Ghalyan et al. (2019:3902), the k-means clustering method has proven to be one of the more efficient and effective forms of clustering out of all the clustering techniques.

In the k-means clustering technique, the dataset provided is split into a K number of clusters, where K is greater than n, and n is the number of features or dimensions in a dataset. Ghalyan et al. (2019:3902) state that k-means clustering can be carried out as follows:

$$J = \sum_{i=1}^{K} \sum_{x \in C_i} d(c_i, x)^2 \tag{2.7}$$

In Equation (2.7), the quality of clustering is represented by J where C_i is the set of data points that belong to the ith cluster, c_i is the centroid of the ith cluster, x gives the data point in the representation and the Euclidean distance between the data point x, and the centroid c_i is given by $d(c_i, x)$.

For all clusters in the dataset, the equation determines the total squared distances between each data point and the centroid. By determining the optimal values for the centroids and the clustering of the data points, the k-means technique can reduce this objective function.

Furthermore, C_i is given below in Equation (2.8):

$$c_i \triangleq \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{2.8}$$

According to Ghalyan et al. (2019:3903), in this form, the total amount of data points in the ith cluster is given by $|C_i|$, while the remaining components remain the same as per Equation (2.7). This calculation therefore provides the ith cluster mean value by dividing the sum of the ith cluster data points by the total number of cluster data points. The centroid is therefore used to calculate the distance between each data point and the cluster.

Using the k-means clustering technique means that up until convergence is reached, or when good separation is achieved between clusters, the centroids are iteratively updated depending on the assignment of data points to clusters.

The k-means clustering technique is the only unsupervised machine learning algorithm utilised for this research.

2.4.8 Artificial neural networks

Artificial neural networks are a data-driven supervised machine learning approach. They were influenced by simplified representations of the human brain's biological functions.

An artificial neural network consists of various nodes and layers, all of which are interconnected. The nodes, also known as artificial neurons, are separated into different layers such as the input (first layer), hidden (middle layer) and output layers (last layer). The layers are made up of weights and biases which can be modified depending on the input data provided for training. Mishra (2021:231-232) further states that artificial neural networks can solve a wide range of problems due to their ability to match patterns.

The number of control variables in the input data is equivalent to the total amount of neurons in the first layer. The input layer must be representative enough to model the structural phenomena. In the artificial neural network layer, the hidden nodes are used for processing and computing the signals' weighted sum from the input layers. The complexity of the problem being modelled will determine the number of hidden layers (Mishra, 2021:232).

According to Mishra (2021:232), the output signal is defined by Equation (2.9):

$$Oj = f\sum(w_{ij}l_i + b) \tag{2.9}$$

where the output of the model is given by Oj, the corresponding weight adjusted every epoch is given by w_{ij}, where l_i gives the input data supplied to the ith node with a term b, the bias term. To obtain the final output, the final summation is entered into the activation function f. To avoid overfitting data and to achieve the required accuracy, the architecture of the neural network structure is typically obtained through a trial-and-error method (Mishra, 2021:232).

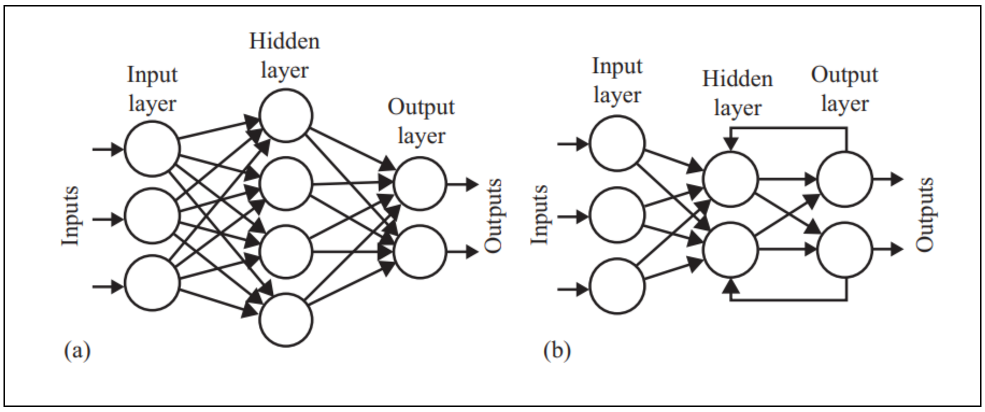


Figure 2.6 Typical feedforward neural network structure: (a) feedforward neural network, (b) recurrent neural network (Daneshtalab, 2020:10).

Artificial neural networks can generally be categorised as feedforward neural networks or recurrent neural networks (Daneshtalab, 2020:9). Figure 2.6 (a) illustrates the feedforward neural network structure where information moves from the nodes in the input layer to the nodes in the hidden layer and finally to nodes on the output layer, in a forward movement. Typically, in these structures, the output is produced as a static function of the input. Figure 2.6 (b) shows the recurrent neural network structure where information returns to the same node or to one in the previous layers. Recurrent networks understand time, enabling feedback and ultimately allowing them to understand and predict the behaviour of complex nonlinear systems (Daneshtalab, 2020:10).

Perceptron

A single layered neural network with a simple structure is known as a perceptron. A perceptron does not have a hidden layer, while the single layer it consists of only has one neuron (De Castro Mota, 2021:10). When input data and target data are linearly separated, this neural network converges to an optimal value. However, this process has several defects. One is the inability to learn non-linear relationships in the dataset provided. This is a huge issue, as linear separation of real-world data is not possible in most scenarios. Secondly, a considerable amount of time is required for convergence. Finally, a perceptron is susceptible to data noise and is not robust against it (De Castro Mota, 2021:10).

Multi-layer perceptron

Baladram et al. (2020:94), state that nonlinear relationships can be learnt by the multi-layer perceptron in the data provided. This is in contrast to the simpler perceptron model. When dealing with an artificial neural network, hidden layers are required when class patterns are not linearly separated.

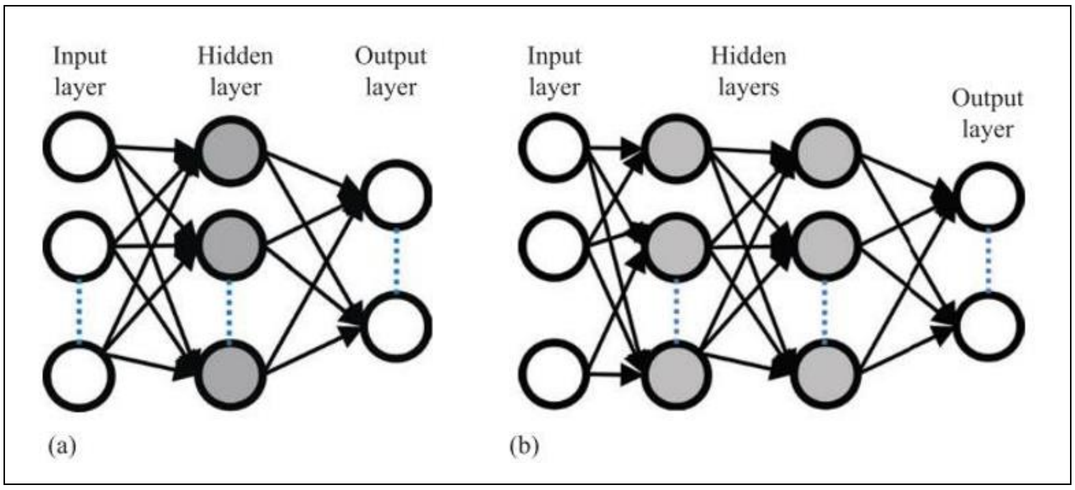


Figure 2.7 Typical multi-layer perceptron neural network structure: (a) one hidden layer, (b) two hidden layers (Daneshtalab, 2020:16).

Figure 2.7 illustrates two different multi-layer perceptron structures. Figure 2.7 (a) consists of one hidden layer while Figure 2.7 (b) has two hidden layers, with information and data sent from nodes in the input layer to the hidden layer and finally to the output layer. The addition of hidden layers provides the complexity necessary to create labels of a class pattern which are contained in a convex or non-convex shape depending on the number of hidden layers added.

Simple, multiple, and extended regressions

It is critical to understand the different regression forms as neural networks and the extended regression method have similar structures (De Castro Mota, 2021:10).

Simple regression is a form of machine learning technique where values of the one-dimensional input vector and target vector, x and y, must be fitted to the data on a line of best fit (De Castro Mota, 2021:10-11). This allows the novel input vector output prediction value to be given by y.

Equation (2.10) is obtained with simple regression analysis, using the data that is given:

$$y = f(x) = wx + b \tag{2.10}$$

Multiple regression is obtained if more than two variables are required. Considering an input vector given by $x = (x_1, x_2)^T$ and a target vector given by y, the goal is to predict the value y by determining an optimal regression plane of a novel input vector. De Castro Mota (2021:11) states that in a case like this, the multiple regression equation is shown as:

$$\begin{aligned} y &= f(x_1, x_2) \\ &= w_1, x_1 + w_2, x_2 + b \\ &= (w_1, w_1)(x_1, x_2)^T \\ &= \boldsymbol{w^T x} + b \end{aligned} \tag{2.11}$$

Where:
$\boldsymbol{w} = (w_1, w_2)^T$ – the weight parameters
$\boldsymbol{b} -$ bias term

By reading input data repeatedly and learning the features of the dataset provided, an attempt is made to determine the optimal values of w and b, in the multiple regression analysis. The two main differences between the simple and multiple regression methods are the amount of explanatory variables and their weight parameters. De Castro Mota (2021:11) says that one way to express the regression is expressed in Equation (2.12), when the multiple regression method is applied to an arbitrary number of dimensions:

$$y = f(x) = \boldsymbol{w^T}(\boldsymbol{Wx} + \boldsymbol{b}) + b \tag{2.12}$$

Where:

$\boldsymbol{w} - m$ x n matrix
$\boldsymbol{b} - m$ x 1 matrix (m– dimensional vector)
$\boldsymbol{w}, b - m$ x 1 and 1 x 1 parameter metrics

According to Baladram et al. (2020:96), in contrast to multiple regression, the input and output vector dimensions remain constant, regardless of the number of parameters.

Principle of neural network

The addition of the "activation function" $\emptyset$ is the major difference between a neural network and extended regression. The regression equation can be expressed in the neural network as in Equation (2.13):

$$y = f(x) = w^T \emptyset(\boldsymbol{wx} + \boldsymbol{b}) + b \tag{2.13}$$

Nonlinear data cannot be separated by simple and multiple extended regression approaches (Baladram et al., 2020:96). To replicate nonlinear functions and respond to data from the real world, neural networks require activation functions. Activation functions form one of the most important components of neural networks. The activation function determines if earlier signals must be transferred to the next neurons; and this serves as a neural cell activator (De Castro Mota, 2021:12).

A conceptual image of the multi-layer perceptron, which shows the feed-forward neural network process, can be seen in Figure 2.8. The multi-layer perceptron has an input, middle and output layer (Baladram et al., 2020:97).

The weight parameters W are given by the connecting lines, with the circles representing the neurons. The weight parameters given by the connecting lines must be optimised.

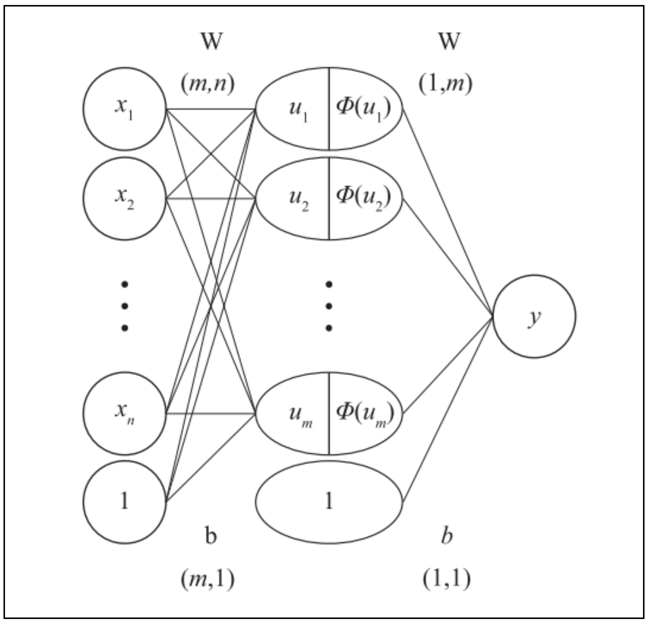


Figure 2.8 Typical structure of a multi-layer perceptron, according to Baladram et al. (2020:97).

Figure 2.8 shows that the process is made up of various elements. The first is the input vector x_1, which is transformed into the element u_j. An activation function Ø then activates u_j, after which the activated elements $\emptyset(u_j)$ are changed to y, which is a scalar value.

The recent interest in artificial intelligence can be attributed to the significant role of neural networks, as they are one of the more widely used machine learning techniques. Neural networks exist in different forms, such as convolutional neural networks and multi-layer perceptrons, with CNNs' primary application forming part of image processing studies.

2.4.9 Convolutional neural network

According to Baladram et al. (2020:103), CNNs are a more complex form of a neural network and are a type of deep learning model. CNNs are typically used for processes such as image analysis and object recognition.

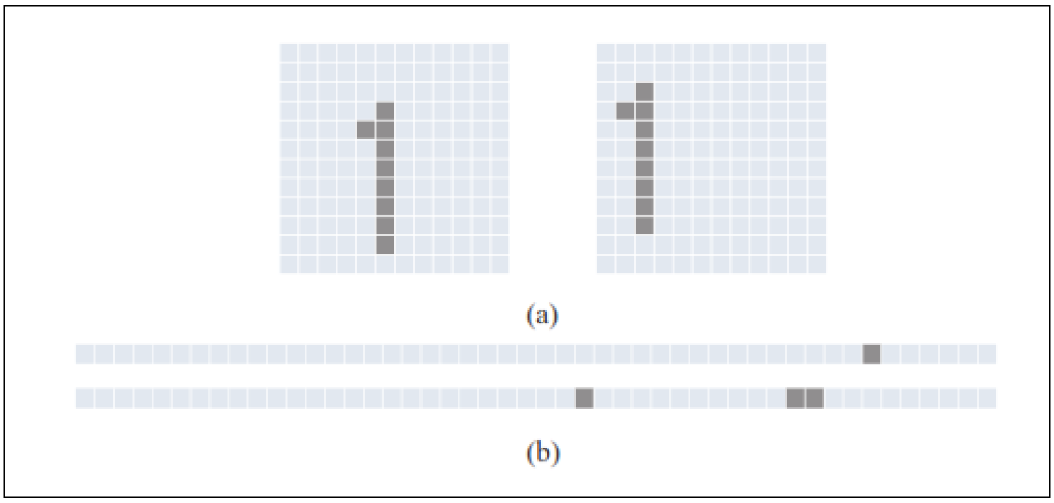


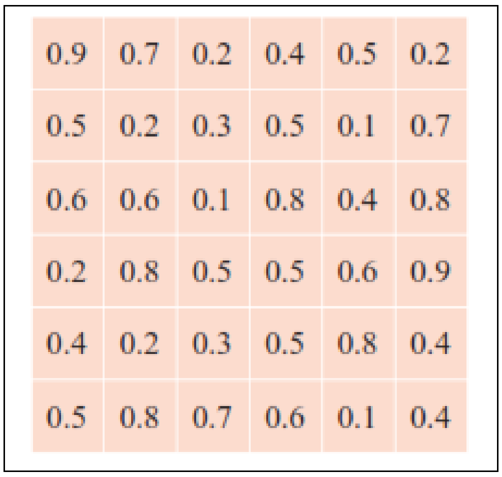
Figure 2.9 (a) Images containing 144 squares in a 12 x 12 matrix. Both images also contain the number 1. (b) The upper panel of the image matrix when flattened in a vector form. Only the first 48 squares are illustrated (Baladram et al., 2020:103).

Figure 2.9 is split into two images, with Figure 2.9 (a) illustrating a 12x12 matrix made up of 12 rows and columns. Figure 2.9 (b) shows the 12x12 matrix of Figure 2.9 (a) in a flattened 144-dimension vector, which is needed for the multi-layer perceptron to learn.

Figure 2.9 illustrates the categorisation process, in which both images in Figure 2.9 (a) are classified into the same class due to the presence of the number 1 in both images. In this task, identifying similar patterns in an image, with the pattern defined by the contrast between the number from both images and the image background forms a major challenge for machines. The gap in the number must also be neglected; this is another challenge posed to the machine, in addition to pattern recognition. In Figure 2.9 (a), in the first image, the number is positioned almost centrally in the image; while in the second image, it is located on the left. For the multi-layer perceptron to learn, the 12x12 image matrix must be flattened into a 144-dimension vector, as per Figure 2.9 (b). The images on the left and right in Figure 2.9 (a) correspond with the upper and lower vectors. The new vector can then be used as input data to the multi-layer perceptron. This is because the multi-layer perceptron can accept only vector-type data as input data. In the case presented, although the two images include the same number, the flattened vectors differ. This poses a difficult task for the multi-layer perceptron as there is a gap between the two images. Despite the fact that the multi-layer perceptron can accurately learn these types of images, this method is not efficient at learning this data (Baladram et al., 2020:103).

Convolutional neural network features

Baladram et al. (2020:103) state that CNNs have two distinct features: convolution and pooling. These processes help CNNs overcome various image processing difficulties such as pattern recognition and even a gap. The convolution process enhances the contrast between the original images, and this makes the detection of similar patterns easier for the machine. A filter, also known as a kernel, is used to perform this operation. The kernel's conceptual image is displayed in Figure 2.10.



0.9	0.7	0.2	0.4	0.5	0.2
0.5	0.2	0.3	0.5	0.1	0.7
0.6	0.6	0.1	0.8	0.4	0.8
0.2	0.8	0.5	0.5	0.6	0.9
0.4	0.2	0.3	0.5	0.8	0.4
0.5	0.8	0.7	0.6	0.1	0.4

Figure 2.10 Filter or kernel used during the convolution process (Baladram et al., 2020:103).

According to Figure 2.10, the filter is a matrix with various elements comprising numerical values. The filter's initial value is chosen at random. Convolution is performed using the filter, and the original input image using Equation (2.14):

$$u_{ij} = \sum_{p=1}^{F} \sum_{q=1}^{F} x_{i+p,j+q}\, f_{pq}, \tag{2.14}$$

In Equation (2.14), the value of the filtered cell (i, j) is given by u_{ij}; the value of cells of the filter, $\boldsymbol{F}$ is given by f_{pq} and the value of the cell (i, j) provided in the original image is given by x_{ij}. Figure 2.11 shows that the filter has contrast and according to Equation (2.14), the filtered image's cell value, also known as a map, is large if the contrast is similar to a portion of the original image. When the filtering process is used, any portion with similarities to the filter is emphasised, and those which are not similar are blurred. Distinctive contrast patterns can be extracted from images during the convolution process. The calculation of the convolution process is shown in Figure 2.11.

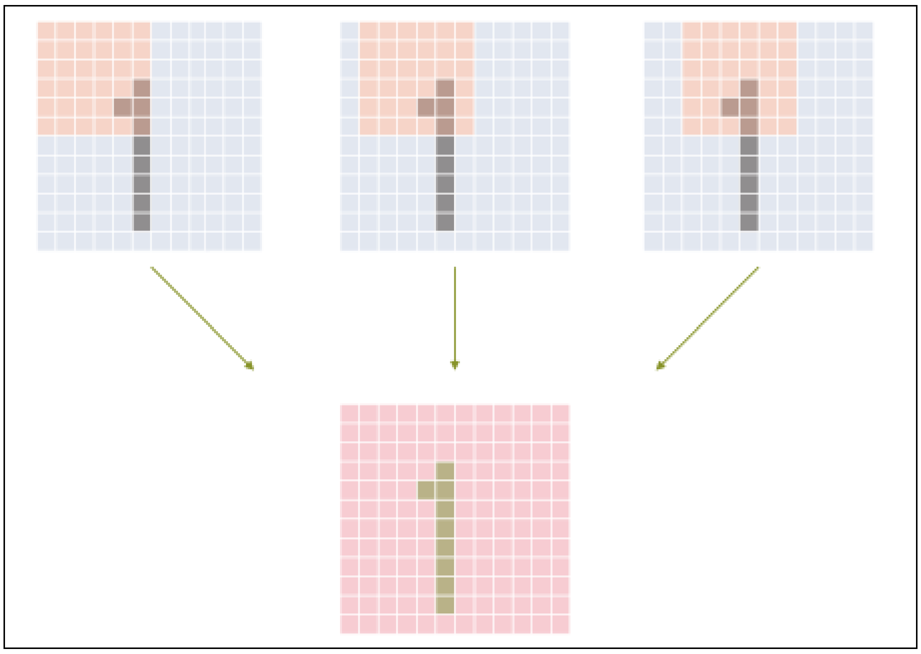


Figure 2.11 Computation of the convolution process illustrated in a conceptual image by Baladram et al. (2020:104). The input images are the upper blue images, with the filtered image or map indicated below in red. The filter is illustrated in orange.

Baladram et al. (2020:104) state that the convolution computation is completed first on the upper panel of the image on the left side. An action called a stride is carried out, as the filter is shifted to read other regions. The stride forms a hyperparameter as the size of the stride can be changed. A trial-and-error process is typically used to establish the size of the stride. An image known as a map is obtained after convolution has been completed; this resultant image is shown in the lower panel of Figure 2.11. Although the symbolic pattern “1” can be seen in the map obtained from the conceptual image, an actual resultant map obtained from the convolution process will not have a clear pattern. Additionally, the map size will be smaller than the original image, even though the image sizes are the same in Figure 2.11. The zero-padding technique is used to prevent the reduction in size, where the technique fills the outer portion of the original image matrix with zeroes to create an image which has the same size (Baladram et al., 2020:104).

The gap in the input image produces an inferior effect; the pooling process is used to cancel this out (Baladram et al., 2020:104). Figure 2.12 shows the pooling process, with maps produced by the convolution process illustrated in the upper red images. The range of the pooling calculation is shown by the green area which overlaps the map. The bottom pixel of Figure 2.12 shows a novel matrix comprising components extracted by the pooling process. The process is carried out and repeated until a final matrix is obtained. This is achieved when pooling is completed on all regions of the map. Baladram et al. (2020:104) state that the information is given in a range through the pooling calculation; and it can therefore cancel the inferior effect out.

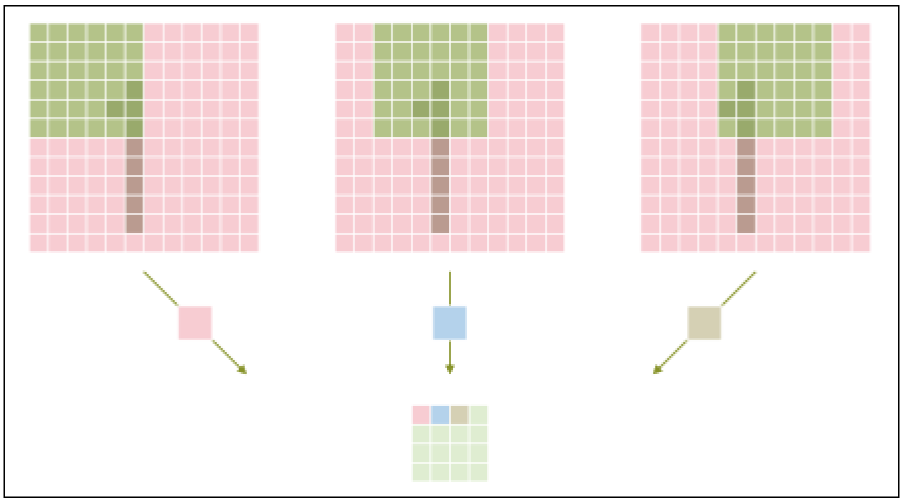


Figure 2.12 The pooling process is shown in a conceptual image by Baladram et al. (2020:105). The upper red images illustrate the mapped images obtained from convolution. The pooling process is shown by the green range, with the bottom image of the resultant matrix after pooling has been completed.

Therefore, CNNs are very efficient in detecting image patterns and absorbing the image gaps through their distinct convolution and pooling processes.

2.4.10 Machine learning for damage detection in heritage structures

According to Mishra (2021:238), due to the current advances in machine learning, studies are shifting to applications where computer vision-based techniques, in conjunction with terrestrial laser scanning equipment (De Paiva et al., 2018:558; Aicardi et al., 2018:258; Alkadri et al., 2022:3) are used for the classification and quantification of damages in heritage structures. Bassier et al. (2017:25) made use of a laser scanning and photogrammetry process for the modelling of historical structures while making use of an SVM approach for the classification of structural components. For the realistic portrayal of the wall surfaces of the Chapel Royal in Scotland, Valero et al. (2019:1,4) combined logistic regression with laser scanning point cloud data for defect classification to determine the repair needs. Valero et al. (2018:29-30) completed research using a supervised learning approach with various machine learning techniques to detect discolouration and material loss in ashlar walls.

Mishra (2021:238) also states that image processing can be applied for the autonomous detection of damage pattern pathologies along with surface defects using images of the structure. Research has advanced to such an extent that images can be captured, and the damage identification process carried out using a mobile phone. This was completed in a recent study by Palma (2019:551-552), where a mobile phone CNN application was developed and used for damage detection on the Imperial Fora in Rome.

Several other studies have been completed using a deep-learning CNN architecture for damage detection. This includes a study by Wang et al. (2018:3), where four damage categories were located and identified for a city wall in China. The four damage classes used for training and evaluation were intact, crack, efflorescence, and spalling images. This research successfully combined a CNN approach with a sliding window algorithm. Zou et al. (2019:2) also used a CNN approach to count the number of intact, impaired and missing components in the Forbidden City, situated in Beijing, China. The objective of the research was to reduce the amount of time spent carrying out in situ visual inspections. This study was completed using the Faster R-CNN algorithm. Furthermore, a study by Chaiyasarn et al. (2018:240-241) made use of different combinations of CNN. These combinations included a CNN and an SVM combination and a CNN and random forest combination. This was completed to compare the performance of the various combinations of machine learning algorithms for crack detection and classification in heritage structures.

Various studies have also implemented the YOLO network, a form of CNN. These include studies by Ma et al. (2022:4), where the network was used to detect cracks in ancient timber architecture and also work by Idjaton et al. (2022:302-303), where damage detection was carried out in the limestone walls of the Château de Chaumont in the Loire Valley, France.

Hatir et al. (2020:2) completed a study to assess the various types of weathering frequently seen in heritage structures in Konya, Turkey. This study used deep learning methods, using an artificial neural network to detect various forms of weathering (scaling, flaking, cracking, differential erosion, black crust, efflorescence). Other research was also completed based on stone monument weathering, with Heidari et al. (2017:29) comparing in situ weathering conditions to weathering in a controlled laboratory environment. Research by Rymarczyk et al. (2021:4;21-22) used various statistical and machine learning methods to detect dampness in heritage building walls. The case study included different historical buildings in Partisan Hill, Wroclaw, Poland. Statistical and machine learning methods such as total variation, least-angle regression, elastic net, and artificial neural networks were used for this research.

Bayesian learning was used by Ierimonti et al. (2021:2-3) to identify the probability of damage occurring in Consoli Palace in Umbria, Italy. This study utilised a metamodeling process to create a surrogate model of the structure. The model considers damage-sensitive mechanical properties and can reproduce the structural dynamic behaviour.

Lastly, other research on heritage structures includes work completed by Shalunts et al. (2011:280-289), where various Romanesque, Gothic, and Renaissance/Baroque architectural styles of building facades were categorised. This was completed using the scale-invariant feature transform (SIFT) as a feature extractor while a Bag of Visual Words methodology was carried out to classify the data.

From the studies and research completed, it is clear that various structural health monitoring tasks can be completed using machine learning techniques. Furthermore, machine learning can be successfully implemented to complete inspection tasks efficiently to preserve and conserve heritage structures.

2.5 Chapter summary and critical review

The literature review establishes that a reasonable amount of research has been completed on fundamental areas of machine learning and image processing techniques for classification and object recognition, providing a much-needed platform for the research at hand.

The various machine learning models were reviewed, with descriptions and explanations provided on traditional machine learning algorithms like SVMs and deep learning artificial neural networks such as CNN. This was necessary to help describe the machine learning aspects and assist with the fundamentals of pattern recognition and defect classification going forward in the study. Furthermore, several case studies were also presented that made use of machine learning defect detection methodology for structural health monitoring applications in heritage structures. Table 2.2 below provides a summary of the case studies that were reviewed.

Table 2.2 Review of the literature on machine learning methods utilised in defect detection, as well as other relevant applications for heritage structures.

Source	Machine learning method	Application
Bassier et al. (2017)	Support vector machine	An automated system for categorizing structural
Valero et al. (2018)	Logistic regression with laser scanning point cloud data	Automated segmentation of masonry and mortar areas in digitized rubble stone construction for repair and
Valero et al. (2019)	Logistic regression multi-class classification algorithm	Detection of discoloration and material loss in ashlar walls.
Palma (2019)	Convolutional neural network	Automatic image recognition and localization techniques for sites using a mobile phone application
Wang et al. (2018)	Convolutional neural network approach with a sliding window algorithm	Damage detection and evaluation of four different damage categories
Zou et al. (2019)	Faster region-based convolutional neural network	Detection of the number of intact, impaired and missing components in buildings
Chaiyasarn et al. (2018	Convolutional neural network, support vector machine and random forest	Crack detection and damage classification
Ma et al. (2022)	Convolutional neural network (YOLO)	Crack detection in ancient timber architecture
Idjaton et al. (2022)	Convolutional neural network (YOLO)	Damage detection in the limestone walls
Hatir et al. (2020)	Artificial neural network	Assessment of different weathering types observed in
Heidari et al. (2017)	Fuzzy interference system	Comparison of situ weathering with weathering in a
Rymarczyk et al. (2021)	Total variation, least-angle regression, elastic net and artificial neural networks	Detection of dampness and moisture in the walls
Ierimonti et al. (2021)	Bayesian learning	Study to determine the probability of damage
Carnimeo et al. (2015)	Artificial neural network	Monitoring heritage structures for early vulnerability warnings
Shalunts et al. (2011)	Scale invariant feature transform and Bag of Visual Words	Categorizing architechtural facades of Gothic, Romanesque and Baroque Buildings
Sharma et al. (2018)	Convolutional neural network with image processing techniques	Detection of dust deposition

Despite the advances of machine learning in structural health monitoring, these damage detection and classification methods still face various challenges. According to Mishra (2021:240), erroneous results may be obtained when considering image processing techniques on heritage structures with differing typologies and characteristics. This means a model trained to predict surface defects, dust deposits or weathering effects in one heritage structure may not be robust enough to be able to do so in another.

Another challenge image processing algorithms face is their inadaptability when implemented under luminance changes. The results of image processing algorithms tend to be poor when considering images of noisy features caused by factors such as luminance changes. An option to improve results would be to implement denoising techniques. However, this is a manual and laborious process, as parameters in the denoising process must be adjusted for each image (Choi, 2020:110-111).

Furthermore, a technique employed by several researchers is the retraining of prebuilt models. Choi (2020:111) argued that creating a model that is dedicated to the specific task is a better approach as it greatly reduces the number of misclassifications. However, developing a model from scratch poses a problem because it requires a lot of computational power and time to train a model from scratch.

The training dataset also plays a role in the classification abilities of the machine learning algorithm. If the dataset provided for training is insufficient or does not fully represent the sample, poor classification results can be expected. Additionally, images with complex features should be used for training to assist in simulating a real-world situation. Although current research shows how effective machine learning and deep learning can be, most of these models were trained using a small training dataset, which is likely to reflect only a portion of real problems. Therefore, building a robust classifier requires a large dataset, which has issues as it is laborious and time-consuming to annotate and build a large dataset of images.

The recent trend and increase in the use of neural networks play a significant role in the overall increase in the use of artificial neural networks. Different types of neural networks exist such as multi-layer perceptron and CNNs, with the major focus of this study on CNNs due to their production of better results, according to the literature. The different techniques such as classification and regression are also reviewed to assist in fully understanding the artificial neural network capabilities. This study will also focus on the more traditional SVM because of its statistical pattern recognition ability. Understanding the characteristics and properties of these machine learning methods plays a significant role in understanding how object recognition and pattern recognition can be carried out.

In conclusion, the sections making up the literature review form the basis for this study. Understanding these sections is of the utmost importance as they provide the foundation for the study and assist in deriving a defect detection methodology for wall defects in heritage structures using machine learning techniques and image processing techniques.

Chapter 3 Methodology

3.1 Introduction

In situ visual inspection methods have gained popularity as a means of monitoring heritage buildings. Such inspections are routinely conducted at globally renowned heritage structures, including the Notre Dame de Paris Cathedral in France and the Hagia Sophia in Turkey. The objectives of such inspections are to safeguard structural integrity and enhance understanding of structural attributes. The introduction of image processing and machine learning and their integration into structural health monitoring represents a novel approach for accurately assessing and classifying defects in heritage buildings. Hence this research aimed to assess and compare the performance of different machine learning algorithms in detecting and classifying these defects. Additionally, the research aimed to determine the extent of crack damage in heritage structures using image processing methods.

This chapter describes the methodology used in comparing the performance of different machine learning techniques in detecting wall defects in South African heritage structures. Furthermore, the chapter also explains the image processing methodology utilised in determining the extent and characteristics of crack damage in these structures. To carry out this task, traditional machine learning algorithms like SVMs, decision trees and k-nearest neighbours coupled with deep learning techniques such as CNNs were trained to detect patterns, ultimately for object classification accurately. The training of the machine learning algorithms required a database of images to be constructed, whereby images were gathered at the Castle of Good Hope, Robben Island and Dal Josafat heritage sites in the Western Cape.

Furthermore, to determine the extent of crack damage in heritage structures, an image processing methodology made up of advanced techniques such as thresholding, binarisation, segmentation and skeletonisation coupled with distance-based computations were used to extract crack characteristics, such as crack length and average crack width, on a dataset collected from the Castle of Good Hope. Methods used for evaluating the algorithms' performance, and determining the damage's extent are described in this chapter.

3.2 Research design

The research design utilised a quantitative methodology whereby image processing, machine learning and deep learning were used to analyse data. The research focused on three case studies, namely the Castle of Good Hope, Robben Island and Dal Josafat heritage sites. Data were gathered from these sites in the form of photographs to create a database for the training, testing, and validation of the machine learning algorithms. Images were captured using a standard handheld digital single-lens reflex camera. The captured images were then put into the MATLAB Machine Learning and Deep Learning Toolbox, specifically the Classification Learner Application and Deep Network Designer, that allowed for the training, testing and validation of the different machine learning classifiers such as SVMs, decision trees, k-nearest neighbours, artificial neural networks and CNNs. Additionally, a secondary dataset made up of crack images collected from the Castle of Good Hope were put into the Image Processing and Computer Vision toolbox in MATLAB. This was completed to carry out various image processing procedures which were then combined with distance-based computations to determine the characteristics and extent of crack damage.

3.3 Research methodology

The main objective of this research was to compare the performance of different machine learning approaches and their capacity to learn the distribution of damage-sensitive features and more importantly, their ability to accurately classify the damage into different damage classes. This process required images containing various wall defects such as cracks and spalling as well as intact areas, gathered from the heritage sites identified in section 3.2. After image data was gathered, the data could be analysed, pre-processed and the classification process carried out. This research only made use of existing machine learning and deep learning algorithms such as SVMs, decision trees, k-nearest neighbours and a pre-trained CNN for the classification of damage. Figure 3.1 summarises the general workflow that was utilised to carry out the research.

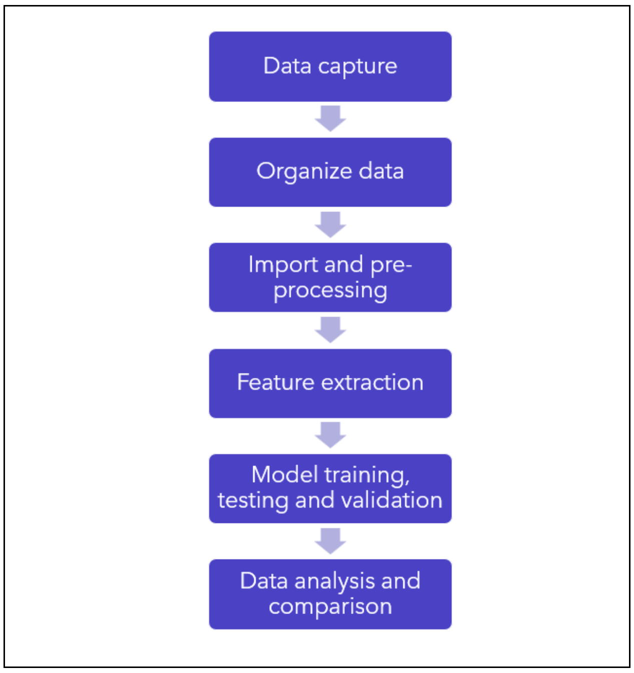

Data capture
Organize data
Import and pre-processing
Feature extraction
Model training, testing and validation
Data analysis and comparison

Figure 3.1 Summary of the general workflow carried out for the research.

Upon the completion of the training, testing, and validation of the model, an evaluation and comparison of the different models' performance was carried out. One of the primary metrics used for the comparison was accuracy, which measures the overall correctness of the models' predictions. A higher accuracy indicates a more reliable and effective classification of the damage categories. Other important metrics considered to provide a more complete and rounded assessment included the misclassification rate, precision and recall metrics.

A confusion matrix was generated to acquire a better visual interpretation of the model's performance. The confusion matrix illustrates the number of correct and incorrect predictions for each damage class, enabling a detailed analysis of the models' performance across different classes. Additionally, the receiver operating characteristic (ROC) curve was plotted, which indicates the trade-off between the true positive rate and the false positive rate. This curve aids in assessing the models' performance and determining an optimal decision-making threshold.

Overall, the combination of accuracy, misclassification rate, precision, recall, confusion matrix, and ROC curve provided an evaluation of the different machine learning models' performance in classifying different types of damage.

3.4 Case studies

The research was restricted to heritage structures located in Western Cape, South Africa. The level of heritage declaration was not specified. Therefore, sites that were provincially or nationally declared heritage sites by the South African Heritage Resources Agency (SAHRA) could be used. However, only sites that were nationally declared heritage sites by SAHRA were used for this research.

As the main aim of the research was defect classification, only structures that showed suitable defects, such as cracks and spalling, were selected. This was necessary as a significant amount of image data, consisting of the images of the defects, had to be collected for a database to be constructed. These data were required to train, test, and validate the machine learning models.

Another requirement for selection was the need for unrestricted access. This was to ensure that there would be no disruptions or hindrances when gathering data and completing the research.

The Castle of Good Hope, Robben Island and Dal Josafat heritage sites met the requirements for selection and were therefore chosen as case studies for this research.

3.4.1 The Castle of Good Hope

The Castle of Good Hope shown in Figure 3.2 is the oldest colonial structure currently standing in South Africa. The Dutch East India Company constructed the building between 1666 and 1679, going through different phases of construction during this period. SAHRA has classified the Castle of Good Hope as a National Heritage Site (Loke, 2020:44).

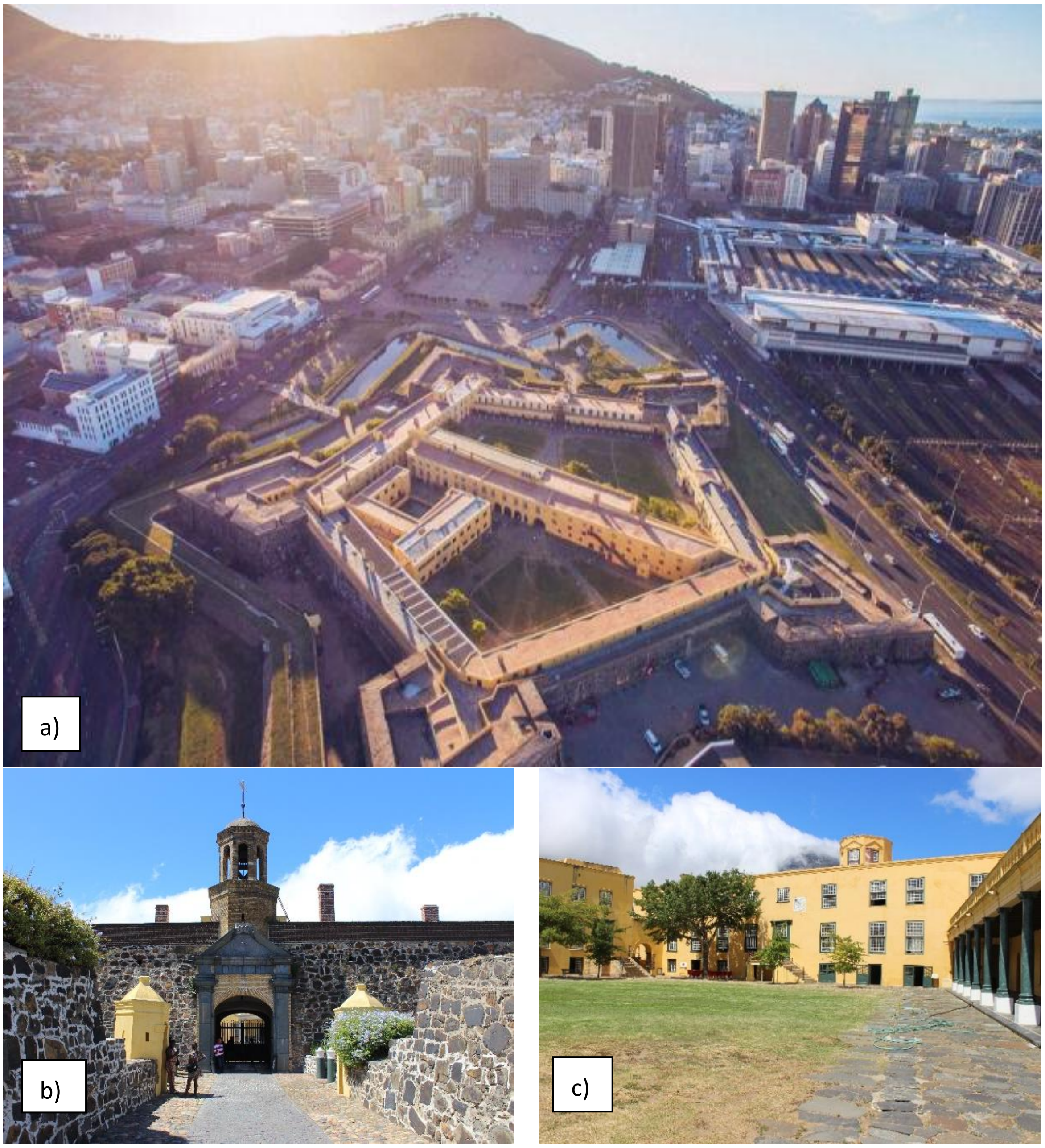


Figure 3.2 (a) Aerial view of the Castle of Good Hope (Castle of Good Hope, 2018). (b) Entrance to the Castle of Good Hope. (c) Typical building structure inside the castle walls.

The structure is composed of stone and block masonry and is constructed in a pentagonal shape. Currently, the Castle of Good Hope serves as a tourist destination, housing the Castle military museum as well as the Iziko William Fehr Collection of historical paintings and ceramics.

According to Loke (2020:46), the Castle of Good Hope has experienced deterioration threatening its serviceable use. The deterioration had progressed to such an extent that the Castle of Good Hope faced either demolition or a full restoration, with the latter option being selected (Gilbert, 1994:22-23).

3.4.2 Robben Island

Robben Island is designated as a National Heritage Site, according to SAHRA (Loke, 2018:46). Robben Island is situated in Table Bay, Western Cape, South Africa. Figure 3.3 shows the maximum-security prison which forms part of the Robben Island Museum. According to Davids and Blacky (2019:1-2), the island went through various phases of uses, e.g., during the 16th century when the Dutch East India Company used it as a fresh supply station. It was used as a hospital in various periods through the 17th and 20th centuries. However, in 1936, during World War II, the islands' main function was military use, becoming a maximum security prison for political prisoners in 1959 (Davids and Blacky, 2019:2). Robben Island is rich in history and is internationally renowned as it was used as a prison for notable leaders such as Nelson Mandela and Walter Sisulu.

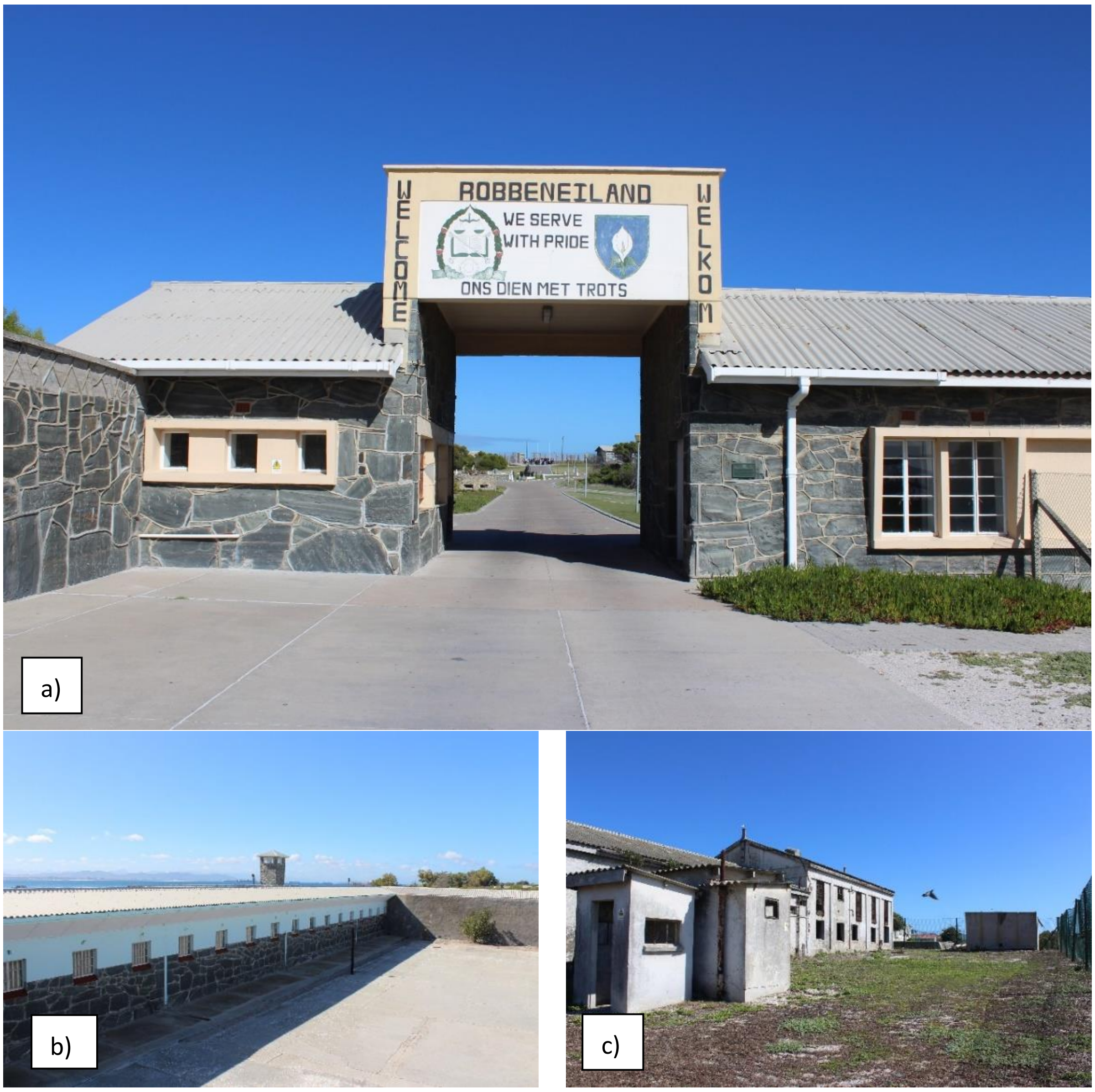


Figure 3.3 (a) Entrance to Robben Island Heritage Site, (b) maximum security prison and (c) old power station.

Like the Castle of Good Hope, Robben Island Museum is constructed using stone masonry (Loke, 2018:47).

According to Robben Island (2013), the island’s heritage is fragile due to the ongoing processes of deterioration. Furthermore, its proximity to the ocean brings further environmental challenges that exacerbate deterioration. The deterioration faced by the island’s structures and the defects they exhibited made Robben Island a suitable option for this research.

3.4.3 Dal Josafat

Dal Josafat heritage site is located in Paarl, Western Cape, South Africa. The Dal Josafat Farm is made up of the Goede Rust, Non-Pareille and Roggeland farms (Gertenbach, 2019:1-2).

The Dal Josafat farm consists of buildings that have distinct Cape Dutch or Cape Vernacular architecture, as shown in Figure 3.4. Gertenbach (2019:1-2) states that from as early as 1690, French Huguenots and their successors owned farms and homesteads. Later, the farms became home to the founding members and leaders of the Afrikaans Language Movement.

On 15 February 1985, the historic buildings on the Dal Josafat farm were collectively designated as a National Monument in accordance with the Government Gazette No. 9588 (Gertenbach, 2019:3). The site is currently a Grade 1 National Heritage Site, according to the SAHRA Heritage Properties Manager Zaida Allie (Allie, 2022). This is the highest level of significance and importance assigned to heritage structures in South Africa. This grading indicates that the site possesses exceptional historical, cultural, and architectural value that requires the utmost protection.

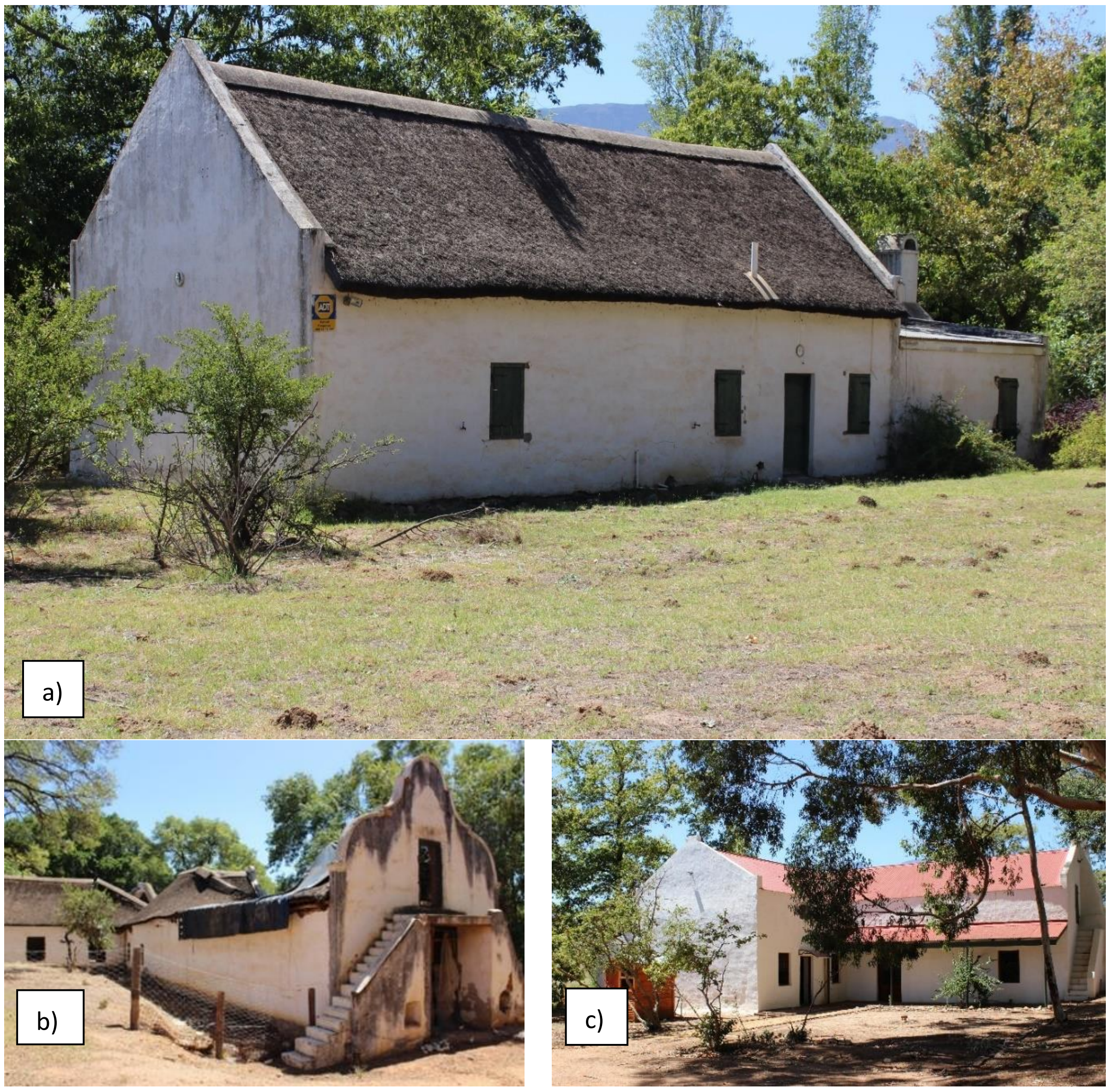


Figure 3.4 Residential homestead buildings located on the (a) Roggeland, (b) Non Pareille and (c) Goede Rust farms, Dal Josafat.

According to SAHRA (2018), the buildings are constructed using brick and plaster. Zaida Allie stated in a recorded interview that some structures are currently occupied through rental, while others are vacant (Allie, 2022). The buildings' continued deterioration has jeopardised their continued usage, and some have reached a point where they are no longer usable (Allie, 2022). Therefore, the Dal Josafat Heritage Site formed another suitable site for the research.

3.5 Material and methods

3.5.1 Equipment

Figure (3.5) below shows the standard digital single lens reflex camera (DSLR) that was utilised to take images of the defects in the selected case studies. The DSLR is an 18-megapixel camera with a dual-pixel autofocus function, allowing highly detailed pictures to be captured with low noise. Captured images consisted of a maximum resolution of 5184 x 3456 colour pixels.

A standardised laptop computer was used to process images, train, test and validate the various machine learning algorithms.

Figure 3.5 Canon EOS 700D DSLR camera.

3.5.2 Data acquisition

The first stage in the process was the construction of an image database consisting of photographs of the wall defects. This formed an important aspect of the research as the image database is key to the performance of the machine learning algorithms. This is because extracting the defect's geometric properties and features plays an important role when training machine learning algorithms. Furthermore, Kim and Cho (2018:3) state that the variety of images generally determines the performance of the classifier during the training process. Generally, those constrained to a certain environment will show poor classification when images outside of that environment must be classified. Therefore, the basis of good machine learning research is a large dataset.

Images were taken approximately 1m to 2m from the area of interest, focusing on the specific damage type. This ensured that the defect occupied a significant portion of the frame and allowed for finer details to be captured in high resolution. This was completed to ensure that image data collected was optimal and allowed for the best possible model training and performance. For this research, the dataset comprised defects categorised into the following damage classes: cracks, intact areas, and spalling. Figure 3.6 shows a sample of the three different defect classes.

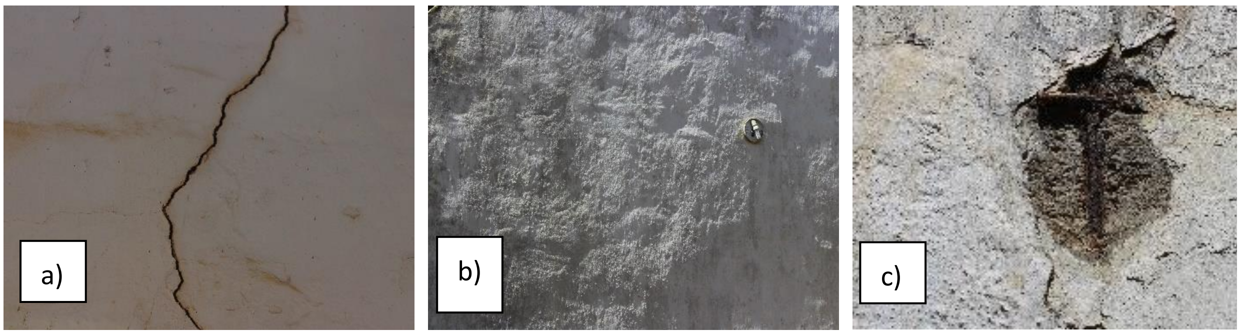


Figure 3.6 Sample of the damage classes captured at the various heritage sites. (a) crack, (b) intact areas, and (c) spalling.

A total of 636 images were gathered from all the visited heritage sites. This included 100 images from the Castle of Good Hope, 250 images from Robben Island and 276 from Dal Josafat. Table 3.1 provides a breakdown of the images collected at each heritage site.

Table 3.1 Images collected from each heritage site according to the crack, spalling and intact areas classes.

Data Collection	The Castle of Good Hope	Robben Island	Dal Josafat	Total
Crack	58	122	134	314
Spalling	4	83	42	129
Intact	38	45	100	183
Total	100	250	276	626

3.6 Data pre-processing

3.6.1 Data annotation

The data annotation, also known as data labelling, involved identifying and labelling the various images according to their defects. The labels, therefore, provided information and context regarding what defect was present in each image for the machine learning algorithms to learn from it. Highlighting features by labelling assists the machine learning models to identify the defect type so that the necessary patterns can be extracted and learned. This is essential for feature extraction, training and the classification process, which is carried out at a later stage.

For the annotation of images, a process was followed similar to Murphy (2020:19), where images were stored and labelled in a tree directory structure, as shown in Figure 3.7. The task involved dividing images into multiple subfolders, where each folder contained images from the specific damage class.

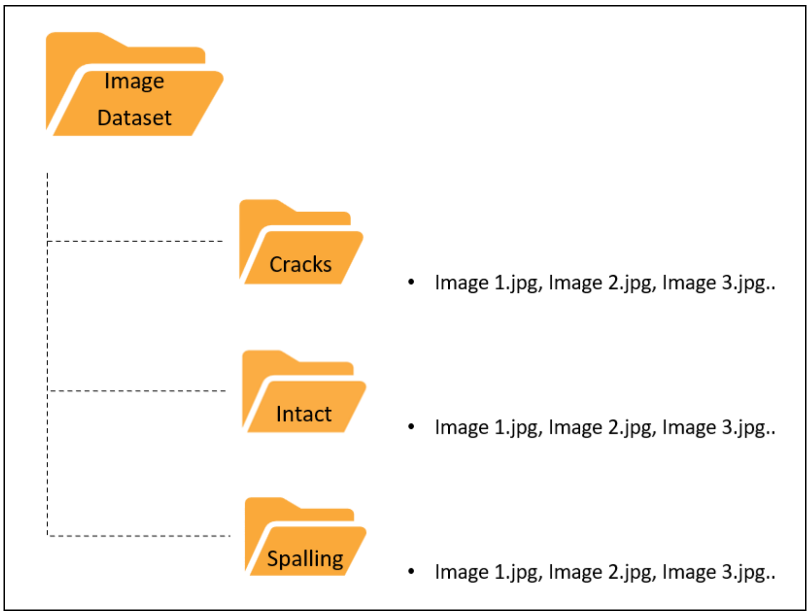


Image Dataset
Cracks
• Image 1.jpg, Image 2.jpg, Image 3.jpg..
Intact
• Image 1.jpg, Image 2.jpg, Image 3.jpg..
Spalling
• Image 1.jpg, Image 2.jpg, Image 3.jpg..

Figure 3.7 Image dataset annotation and typical tree directory structure.

Image data was therefore annotated and subdivided into three main damage classes/folders: crack, intact and spalling damage. Images were organised into the damage classes accordingly to automatically derive labels from the tree directory naming convention when integrated into the MATLAB computing program.

3.6.2 Image pre-processing

To prepare the labelled data for implementation in the MATLAB platform, pre-processing was carried out. This included data augmentation, resizing and image normalisation.

Data augmentation

Data augmentation is a crucial step when carrying out model training as it is a recognised and successful technique used to prevent overfitting of the model. Additionally, it provides a method to increase the sample size of the under-sampled damage classes to prevent the class imbalance problem. Training on an imbalanced dataset could result in the model learning too many features from the over-sampled class.

Data augmentation involves modifying and transforming image data using popular techniques such as mirroring, rotating, colour modification, and even cropping of images to enhance training and provide a more robust model.

For this research, colour modification, cropping and rotation were applied to the image dataset to provide a balanced dataset. Figure 3.8 depicts some of the image data augmentation techniques commonly used in image classification tasks, such as cropping, colour modification, rotations and flipping of images.

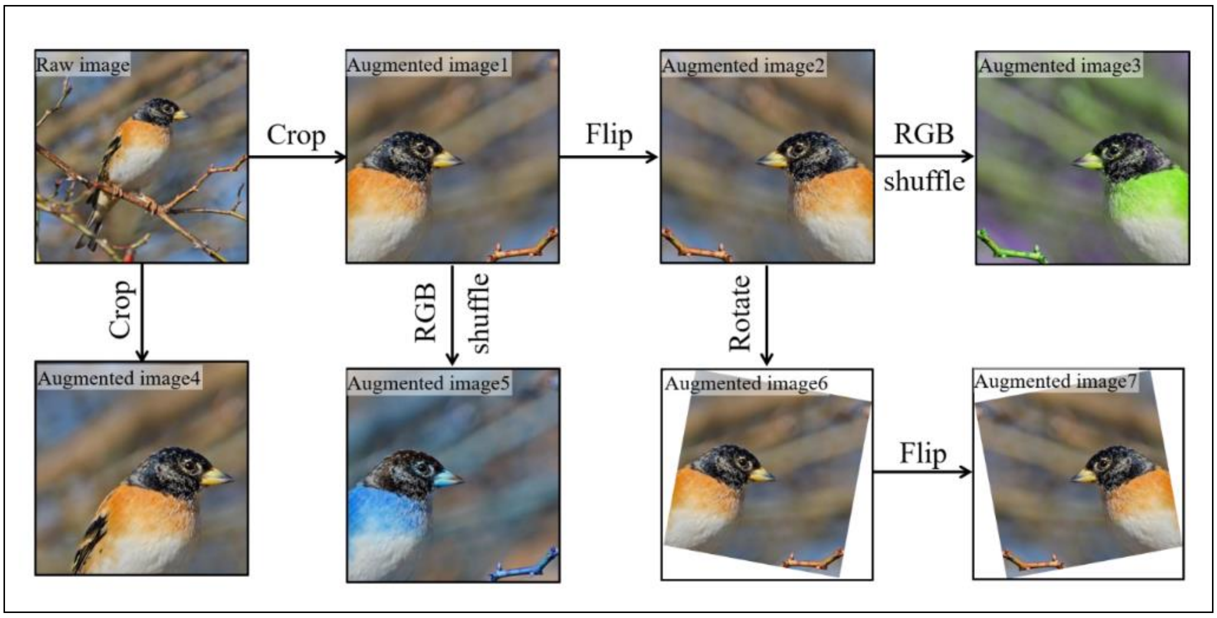


Figure 3.8 Data augmentation techniques (Choi, 2020:14).

Image resizing

Another form of image pre-processing in computer vision includes resizing images. This formed another crucial tool in training the deep learning network, as these networks train more quickly on smaller images. Additionally, pretrained models such as the AlexNet neural network are only compatible with input images of a certain size. Image data had to be resized to the required 227 x 227 pixel size.

Image normalisation

According to Choi (2020:15), different techniques can be used to normalise image data, such as z-score normalisation, logarithmic normalisation, histogram equalisation or feature scaling. These techniques include scaling or transforming pixel values and changing the pixel intensity range to bring data to a standardised or common scale, which are standard practice. This assists with the training process, allowing it to be carried out more stably.

For this research, the feature scaling technique, called min-max normalisation, was carried out to normalise data. This technique scaled the pixel values to a range between 0 and 1. This was completed by subtracting the minimum value of the image from each pixel value and then dividing it by the difference between the minimum and maximum values. This was the preferred method as it preserves the original distribution of data, can normalise noisy data, and can be implemented relatively easily.

3.7 Defect classification approaches

The research aimed at categorising three types of damage. The research therefore falls into the multi-class classification category of machine learning. This research employed machine learning algorithms such as CNN, artificial neural network, SVM, k-nearest neighbour, decision trees, and deep learning techniques for classification purposes.

The defect classification approach comprised two methodologies: machine learning and deep learning. The machine learning approach used the Bag of Visual Words technique to extract features for training and subsequent classification. In contrast, the deep learning approach adopted a transfer learning approach using the AlexNet CNN model. Both methodologies employed supervised learning.

Supervised learning was applied due to its suitability for solving multi-class classification problems and the ability to train models using labelled data. According to Baladram et al. (2020:87), a supervised approach is appropriate when ample input data is available and the target variable or outcome is well-defined.

The subsequent section details the machine learning and deep learning approaches, providing a comprehensive overview of the methodologies applied.

3.8 Machine learning-based classification

The machine learning-based classification methodology adopted follows a similar process to that outlined by Kim et al. (2018:1-14) and Ghalyan et al. (2019:3899-3913). In this process, the Bag of Visual Words language-based machine learning technique was used for feature extraction and later combined with the machine learning algorithms, to accurately classify different damage categories.

The Bag of Visual Words algorithm is a technique used to describe the image content by presenting it as a histogram of visual words. The technique is founded on "visual vocabulary" or the concept thereof. The concept uses functions such as speeded-up robust features (SURF) to collect salient visual features from the images (Kim et al., 2018:2). Kim et al. (2018:2) state that the popularity of the SURF technique was gained from its efficiency in terms of computational time and performance, and its ability to efficiently extract features or interest points. Once the features have been collected, the “visual words” or “visual vocabulary” are grouped using k-means clustering techniques.

According to Ghalyan et al. (2019:3903), each image can be represented as a Bag of Visual Words, when the visual vocabulary has been compiled. Ghalyan et al. (2019:3903) state that a histogram is created based on the frequency with which a visual word is counted in the image. The histogram represents the image content, in terms of the visual words that comprise its various features and the distribution of those visual words in the histogram.

3.8.1 Bag of Visual Words

The research methodology used the Bag of Visual Words technique for the defect classification of the three different damage classes. According to Ghalyan et al. (2019:3903) and Kim et al. (2018:3), three stages make up the Bag of Visual Words technique: feature extraction, vocabulary construction, and classification.

Feature extraction

The first stage of the process is feature extraction, and this is completed to identify distinct characteristics and various points of interest in the images. Kim et al. (2018:3) state that extracting features is integral to object identification and image processing. The process uses the image as an input and produces features as descriptors as an output of the extracted images (Ghalyan et al., 2019:3903).

Feature extraction was carried out using the SURF technique. This differs from Ghalyan et al. (2019:3899-3913) who extracted image features using scale-invariant feature transform (SIFT). SURF was selected as

Kim et al. (2018:3) state that the technique has a high performance at an efficient computational cost and is a specialised tool designed to extract unique image features. Furthermore, the feature extraction process can be broken down into two processes: point detection as well as point description.

- Point detection: according to Kim et al. (2018:3), the Hessian matrix is used to assess and detect any localised changes surrounding pixels to obtain points of interest such as edges or corners.
- Point description: once a point of interest has been detected, a calculation is completed to determine Haar wavelet responses inside a circular region, with each point given an orientation based on the responses (Kim et al., 2018:3). To resolve any image rotations, a square region is then generated along the orientation which was previously obtained. Kim et al. (2018:3) state that the Haar wavelet responses are then used in the horizontal and vertical directions to produce a feature vector with 64 elements. According to Ghalyan et al. (2019:3903), different methods can be used to estimate descriptors, such as the previously mentioned Haar wavelets and histograms of oriented gradients (HOG).

Vocabulary construction

The second stage in the Bag of Visual Words process is the vocabulary construction. Visual words are created using the previously extracted feature vectors in this stage. The visual words highlight characteristics such as colour, shape and even surface texture while acting as a representative image segment (Kim et al., 2018:3).

To effectively carry out training and to handle the large numbers of images during training, it is essential that the various distinguishing features of the crack, spalling and intact images are identified. Kim et al. (2018:3) state that this is because images and image datasets contain a large number of different points of interest along with their accompanying feature vectors. Clustering techniques such as k-means clustering and fuzzy c-means are carried out to identify representative clusters. The visual words that are obtained from the clustering process are then bundled together, and the collection of the visual words forms the visual vocabulary better known as the Bag of Visual Words. According to Ghalyan et al. (2019:3903), k-means clustering is commonly used in conjunction with the Bag of Visual Words technique due to its efficiency and simplicity, and therefore, it was selected to carry out clustering in this research.

Classification

The final stage in the process is the classification stage. The SVM, decision trees, k-nearest neighbour and artificial neural network machine learning algorithms were used for this process. Classification followed a similar methodology to that set out by Kim et al. (2018:1-14) and Ghalyan et al. (2019:3899-3913), who only used SVMs to categorise the presence of crack defects in concrete and metallic materials respectively. Kim et al. (2018:3) advise that when carrying out training to produce a predictive classifier and more than one image set is used, it is preferable for a visual vocabulary to be created first from the entire image dataset using clustering. A histogram of the visual words can then be created by determining the number of descriptors or points of interest in the different clusters and calculating the rate at which each visual word occurs. The SVM, decision trees and k-nearest neighbour algorithms are then used to construct a predictive classifier using the histograms as an input.

3.8.2 Implementation

The methodology carried out by Ghalyan (2019:3899-3913) was used to implement the Bag of Visual Words algorithm. The process involved four main phases, including the detection and description of points of interest for the labelled dataset, clustering and construction of the Bag of Visual Words, and the application of the multi-class classifier models, utilising the extracted features from the Bag of Visual Words algorithm. Furthermore, all processes were carried out using the MATLAB platform.

The different processes were essentially carried out to increase the accuracy of the various classifiers while reducing the computational power required. Csurka et al. (2004:3) state that descriptors obtained in the initial stages should have enough information to make predictions. However, they must also be resistant to changes in lighting, transformations in images or any other variations that are not relevant when classifying or when carrying out the categorisation task. Furthermore, the second step generated vocabulary should be broad and be able to differentiate between any changes in the images. However, the vocabulary should not be too large as this would allow changes such as noise to be distinguished by the algorithm, as stated by Csurka et al. (2004:3).

The last stages in the process involved using the extracted features to construct the histogram and the development of the predictive classifier. The various models had to undergo two essential stages to develop the different predictive classifiers: training and testing. The training involved feeding the machine learning algorithm labelled data to create a predictive model capable of distinguishing between the different classes, using a statistical decision procedure (Csurka et al., 2003:7). The trained models obtained from this process were then used to classify the other images into the various damage classes. The classification performance was then obtained for each model by determining the accuracy, precision and recall of the different machine learning algorithms.

Loading image dataset

The first step involved preparing and loading the image dataset with the three different damage classes into MATLAB. The dataset was loaded into MATLAB Statistics and Machine Learning Toolbox using an image datastore.

The image datastore assists in managing all the data and images. Furthermore, it allows a large dataset to be used efficiently, as it does not load all the images into memory, as this can be very computationally intensive and slow. This was completed using the image datastore function.

Once completed, images of the specific damage classes were present in the image data store. Furthermore, the image labels were obtained directly from the folder names which were used to create the various image sets. These labels are essential for the feature extraction phase, training and, ultimately, the classification of image defects.

Constructing training and testing datasets

The next step involved setting up the training dataset. Each damage class must have an equal split of images when carrying out this task. This is essentially to avoid a class imbalance problem, which could result in the model learning more features from the over-presented damage class.

To avoid this, techniques such as under-sampling or over-sampling can be used. When using under-sampling, the number of images in the over-represented class is reduced to balance and match the other classes. When over-sampling is utilised, the opposite is completed, and the under-sampled class is increased using data augmentation.

Data augmentation was completed in the spalling and intact areas classes to increase their sample size. This was due to the high number of images obtained for the crack defect class. Data augmentation was completed by rotating and modifying the colour of images in the under-sampled classes to increase their sample size and match the number of images in the crack defect class. Once augmentation was completed, each damage class had a total of 387 images. This brought the full dataset size to 1161 images.

Data was then split into the training, validation and testing datasets. For this, 65% of the data was used for training, 15% was used for validation, and the remaining 20% was used for testing and evaluating the pre-trained model. To avoid bias, this task was completed at random.

Once data was partitioned into the training and testing sets, it was ready for the training phase.

Vocabulary construction and training

Images were then converted into a visual vocabulary using the Bag of Visual Words algorithm. The feature extraction process was carried out using the SURF approach, while k-means clustering was used for the reduction of feature vectors while also carrying out quantisation of the feature space. The visual words obtained from this process were then used for training the different models. This was completed to ensure that the predictive classifiers could categorise images according to their defects.

The process involved utilising the bag of features function to extract features from the training dataset. This resulted in the generation of a bag of feature objects. Additionally, a histogram was constructed to represent the image, allowing for the encoded counting of visual word frequencies in the image.

Certain algorithm parameters were adjusted to best suit the dataset for the bag of features function to work best. These parameters included:

- Point selection: this parameter determines where the algorithm detects different SURF features. The options provided included a grid method and a detector method. For this, the grid method was used, and this allowed features to be extracted throughout the images at prespecified positions. This allowed information from the image to be collected uniformly throughout the image. The size of the grid was altered using the GridStep command, changing the default step size to 38 pixels.
- Patch size: the SURF function determines features using the gradient value of pixels surrounding a point (the pre-specified point using the grid method). The size of the surrounding size, called patch size, is determined using the BlockWidth parameter. The SURF function allows the scales for descriptor extraction to be adjusted. This therefore proved advantageous as crack and spall damage appeared in varying sizes throughout images. For this process, two patch sizes of 98 pixels and 128 pixels were used.

For this research, the bag of features function was adjusted to accommodate the point selection, with two patch sizes of 98 and 128 selected.

Next, a matrix of predictor features was generated by utilising the encode function on the training image datastore, which contained the bag of features object.

The resulting matrix represented each image as a row and each feature as a column. The matrix was transformed into a table format to facilitate its use in the Classification Learner application. Variable names were assigned to each feature column to ensure consistency during testing. Additionally, class labels were appended to the table for further analysis.

The Classification Learner application was utilised to facilitate the training of machine learning algorithms. A new session was opened in the Classification Learner application and the previously generated table, consisting of the images and predictor features, was selected as the dataset for training.

For the validation of the models, a cross-validation approach was used. The cross-validation approach involves dividing the training dataset into several subsets and training the model multiple times on those different subsets. Validation is an essential step in training when developing machine learning models. Validation was used to verify the trained parameter and improve the model's classification capabilities. Furthermore, this process reduces the risk of overfitting the model.

The SVM, k-nearest neighbour and decision trees machine learning algorithms were then selected for training, using the predictor features extracted by the bag of features function. The Classification Learner application allowed for the training of multiple machine learning models to be carried out simultaneously.

The Classification Learner application provided a user-friendly interface and visual display for the results of the training of models. Furthermore, it provided the confusion matrix for training results in the visual display.

Once the model classification accuracy was suitable, the test dataset could be implemented to determine the final accuracy.

Testing

In the final stage, the test data set aside initially was evaluated and classified using the trained model. This was completed in MATLAB using a prepared set of predictor features, again utilising the bag of features and the test data set aside.

Next, a matrix of predictor features was generated for the test data, which was subsequently transformed into a table format along with the corresponding labels. This table was prepared specifically for the Classification Learner application, mirroring a similar process undertaken during the training phase.

Subsequently, the generated table was imported into the Classification Learner application to test the performance of various trained models. The table served as the input for the test data, allowing the models to classify the test dataset. The Classification Learner application provided visual test results and presented a confusion matrix and Receiver Operating Characteristics (ROC) Curve to visualise the performance of different models.

3.9 Deep learning-based classification

Deep learning, a subfield of machine learning, is based on artificial neural networks and the various hierarchical layers which make up these neural networks. The hierarchical layers carry out many complex processes and can learn complex data representations.

In this research, image data was used as the input to the deep-learning CNN model. The deep learning model then performed feature extraction on images, systematically carrying out feature extraction on images, moving from one layer to the next, with the input for each layer obtained from the output of the preceding layer.

For this research, a supervised transfer learning approach was applied to the AlexNet CNN model. Transfer learning methodologies are typically utilised in deep learning applications. This process uses a pre-trained model as the starting point for the research. Feature extraction and fine-tuning are then carried out on a pre-trained network, which was previously trained to complete similar tasks, thus reducing the training curve instead of training a network from scratch with randomised weights and parameters (Joubert, 2020:49-50). The transfer learning, feature extraction and fine-tuning processes can be described as follows:

- Transfer learning: this process uses pre-trained models and alters the network output by removing the fully connected layers. Joubert (2020:49) states that a fully connected layer using the correct number of classes is added to the model. Joubert (2020:49) further states that network training is completed on a dataset where the last layer’s weights are updated. This process forces the model to utilise the pre-existing features to classify the new classes. This process was carried out for the development of neural networks.
- Feature extraction: the pre-trained CNN model consists of multiple fully connected layers and convolutional layers. The feature extraction process is carried out using the global image features, by replacing the existing fully connected layers with new layers capable of classifying images into various classes. The convolutional base weights and biases were kept the same, and this was used for feature extraction of the new image dataset. The features obtained were then able to learn the fully connected layers’ new parameters, allowing for the classification of defects. According to Choi (2020:12), each layer in a network can carry out feature extraction, which can be completed in the presence or absence of learnable parameters.
- Fine-tuning: the fine-tuning process is carried out by training the convolutional base top layers with the last layers or fully connected layers. This is completed to help generalise new data.

3.9.1 AlexNet Convolutional Neural Network

The AlexNet CNN model differs from traditional machine learning models as its hierarchical architecture generally consists of five convolutional layers, some of which are followed by max-pooling layers and three fully connected layers. As such, the AlexNet model provides better feature extraction ability than traditional CNN models due to its deeper architecture. To improve the model's performance, the model has both max pooling and rectified linear units (ReLUs), which are applied between the five convolutional layers. To carry out classification, a softmax activation function is applied to output after convolution, as well as the three fully connected layers (Zhu et al., 2018: 218). This allows the model to predict the class of the image, such as crack, spalling and intact images, accurately and efficiently. Furthermore, a response normalisation layer is also present after the first two convolution layers in the model.

Each of the different functions is responsible for different operations when creating or training the network and can be described as follows:

- Pooling: the pooling layer carries out a downsampling of the feature map representation produced as an output of convolution by summarising its features and reducing dimensionality. As a result, processes that follow are carried out on summarised features instead of precisely positioned features, creating a more robust model, and reducing overfitting and the number of parameters of the model. In this research, the pooling layer carried out max pooling, where the main features were extracted from the original feature map (Zhu et al., 2018:219).
- Rectified linear units: according to Zhu et al. (2018:219), the AlexNet model utilises a ReLU output function whereas traditional CNN models utilise tanh or sigmoid functions. Using the tanh or sigmoid functions to calculate gradient descent using backpropagation can produce a large and complex calculation, depending on the data size and model complexity. However, the ReLU function reduces the calculation and essentially speeds up the training of the model while reducing the chance of overfitting the model.
- Softmax activation function: The AlexNet neural network employs the softmax activation function in the output layer to convert the raw outputs of the preceding layers into a probability distribution over the various potential classes. It uses a vector of random real values, or the log-odds function for the potential categories, as an input, and these values are then normalised into a probability distribution. The different scores are transformed into a probability, which reflects the possibility of the input belonging to a certain class.

According to Kim (2018:3), when comparing feature extraction between the CNN model and the Bag of Visual Words technique, CNNs differ as they classify using the global features of images. In contrast, the Bag of Visual Words technique uses local features in the form of clustered visual words. Where global features are used, features are described or obtained using the entire image, while local features are obtained from image patches or subsets of images. The two methods offer diverse information about images at a computational level. Murphy (2020:27) states that the AlexNet neural network can learn from complex and feature-rich images over a wide range.

CNN models generally require large, annotated datasets, resulting in complex training of the model over a long period of time. Employing a transfer learning approach with the AlexNet neural network provides an alternative method of computer vision, which assists in alleviating the training time required. The AlexNet CNN model and a transfer learning methodology allow a new neural network to be created and trained using an existing pre-trained network as a starting point, where the pre-trained network weights are updated during training. This significantly decreases the training time required, while providing a model capable of accurately predicting different damage classes.

3.9.2 Implementation

A methodology similar to Murphy (2020:1-47) was applied to implement the CNN, utilising the AlexNet neural network and a transfer learning approach. Like the machine learning approach, the implementation of the CNN model required different phases to be completed, such as:

- Data preparation: as a comparison was being carried out between the various algorithms and the neural network, the same dataset utilised for the machine learning approach was used for the AlexNet CNN model. As such, the dataset was already prepared, labelled, and split into the various damage classes and was ready for implementation into the MATLAB platform.

- Data pre-processing: images were resized to a compatible input size that could be used with the AlexNet CNN. For this, the image dataset had to be resized to an input size of 227 x 227 x 3.
- Feature extraction: the feature extraction process utilised the different convolution layers. The input image was passed through the convolutional layers, and an output was obtained from the final convolution layer. A feature representation of the input image was then constructed using this output, essentially a group of high-level features that captured the localised patterns in the input image.
- Fine-tuning: fine-tuning the network involved updating the final layers of the model so that the specified classification task could be carried out. For this, only the weight of the final layer was updated while the weights of the preceding layers were frozen.
- Evaluation: like the machine learning approach, the performance of the trained AlexNet model was evaluated on the validation and test datasets. A confusion matrix was then used to visualise the model's performance.

In general, carrying out the various phases in a transfer learning approach enables one to use information previously obtained from a large dataset to enhance the model's performance. The different phases involved in the transfer methodology all work together and enhance the model's ability to learn the necessary features from the input data, increasing the model's accuracy.

Loading image dataset

Like the machine learning approach, the first step involved preparing and loading the image dataset into MATLAB split into three different damage classes. The dataset was loaded into the platform, using an image datastore.

This was completed to construct a labelled image datastore, specifying the root path and folder path, including subfolders, while using the subfolder names to label images. After doing this, the image datastore now had images related to the distinct damage categories present.

Constructing training and testing datasets

The next step was to construct the training, validation, and test datasets. Since the dataset was already balanced during the implementation of the machine learning methodology, the dataset could be directly partitioned into the respective sets without any further adjustments.

Data was then partitioned into the training, validation and testing datasets. To do this, 65% of the data was used for training, 15% was used for validation, and the remaining 20% was used for testing and evaluating the pre-trained model. Furthermore, the dataset was split up randomly to avoid bias.

Developing the model and transfer learning

The network was imported into the Deep Network Designer tool in the MATLAB application to develop and define the AlexNet neural network. From there, the AlexNet model could be modified to suit the classification task at hand.

In its current form, the output class of the AlexNet neural network has one thousand classes. However, these classes are not necessary for the defect classification task. Therefore, the AlexNet neural network's final layers could be adapted and replaced to suit the required defect classification task.

For this, it was necessary to replace the final three layers, which included the fully connected layer, the softmax layer and the output classification layer. This was completed using the interactive drag and drop display. The fully connected layer was then given an output size of three to match the number of damage classes of the data. Once completed, the network was analysed to ensure it was capable of carrying out the classification task, to ensure it could learn from the new dataset, and to determine if there were any faults in the network.

No faults or errors were obtained from the analysis. The modified network could then either be exported from the Deep Network Designer tool to the workspace, or the actual network code could be generated. Alternatively, another option was available, and this was to complete the training in the network in the Deep Network Designer tool. This was the preferred method, and the training data was selected and imported from the image datastore in the workspace into the Deep Network Designer tool. Furthermore, an option was provided to import the validation dataset, which was selected as 15% of the dataset.

A call was then made to the training tab and the network training was carried out. Carrying this out in the Deep Network Designer automatically normalises training images, resizing them to 227 x 227 pixel size. Additionally, a visual representation is provided of the training progress. Once training was finalised and satisfactory training accuracy was achieved, the network was exported to the MATLAB workspace for the testing phase.

Testing

In the testing stage, the remaining test data set aside was used as input to the trained model. However, the test dataset was first resized to the necessary 227 x 227 pixel size, as the AlexNet neural network required.

An augmented image datastore of the test dataset was created with images resized to the necessary 227 x 227 size. This was completed to avoid overriding the original test set.

The classification ability of the AlexNet model was tested on the augmented image test set. The augmented test dataset was input into the AlexNet neural network for classification, which generated the predicted labels as a vector. These predicted labels could then be compared to the true labels to evaluate the model's performance.

The predict function was used to obtain classification probabilities for each damage class, while the classify function was used, which returned a matrix of the predicted scores for the various images in the augmented test datastore.

The final stage of the defect classification methodology entailed acquiring the model's test accuracy, misclassification rate, precision and recall. A confusion matrix and ROC curve were also generated to visualise the model's performance. The resulting confusion matrix and ROC curve were essential for assessing the ultimate network performance.

3.10 Extent of damage using image processing

An additional objective of this research was to determine the extent of cracks in heritage structures using an image processing approach. This process required images containing only crack defects, gathered from the Castle of Good Hope heritage site in Cape Town, Western Cape, South Africa. Figure 3.9 summarises the general workflow carried out to determine the extent of crack damage.

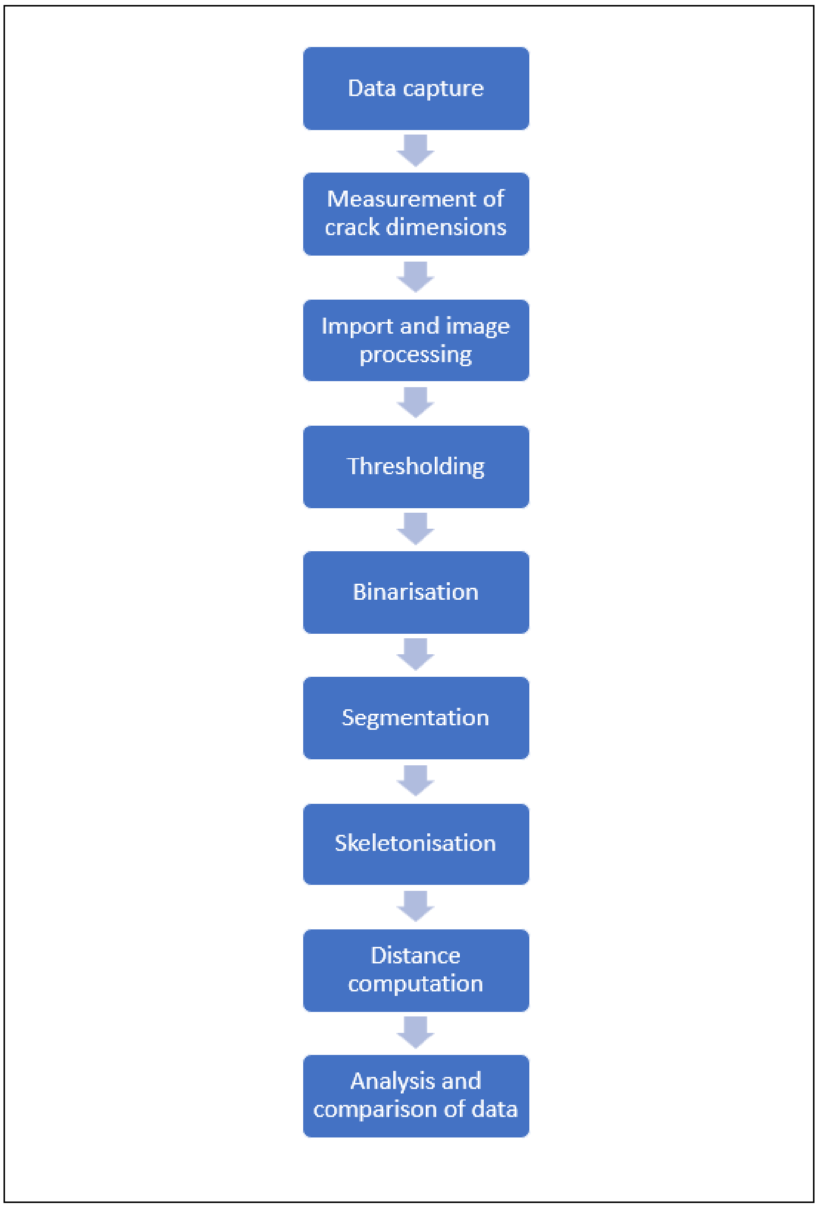

Data capture
Measurement of crack dimensions
Import and image processing
Thresholding
Binarisation
Segmentation
Skeletonisation
Distance computation
Analysis and comparison of data

Figure 3.9 Summary of the general workflow carried out to determine the extent of crack damage.

The dataset was made up of ten crack defect images, with the physical dimensions of each crack measured on site using a tape measure. Once the measurements were obtained, an image of the crack was captured. This process was followed for ten cracks, which differed in shape and size.

The image processing procedure was carried out for each image. For this, a methodology similar to Carrasco (2021:4) was followed, where thresholding, binarisation, segmentation, and skeletonisation were used. Carrasco (2021:1-18) combined these techniques with k-means clustering to extract crack characteristics. However, the methodology used in this research differed as the image processing techniques were combined with distance-based Euclidean distance transform computation to determine crack characteristics such as linear crack length and average crack width.

The image processing methodology was then assessed using the percentage error rate, which is the percentage difference between the expected value and the actual value obtained. The error rate computation was based on a comparison of the direct physical measurement and the measurement obtained through image processing, and this provided a way to evaluate the image processing methodology's performance in accurately extracting crack characteristics to determine the extent of damage.

3.10.1 Data acquisition and physical measurement

To determine the extent of damage, an image database consisting of photographs of the crack defects had to be collected. Ten images containing the crack defect were therefore collected from the Castle of Good Hope heritage site using a standard handheld DSLR camera. The captured images comprised 5184 x 3456 colour pixels. As in the machine and deep learning-based methodology, images were taken approximately 1m to 2m from the area of interest while focusing on the defect, in this instance, the crack. Figure 3.10 shows a sample of the crack defects which were collected.

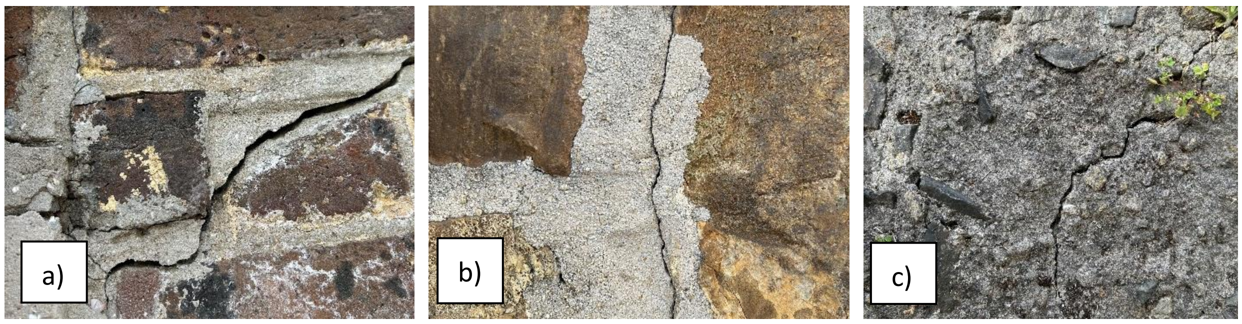


Figure 3.10 Sample of the crack images collected at the Castle of Good Hope.

The physical dimensions of the crack were then measured using a tape measure so that a comparison could be conducted between the actual physical measurements and those values obtained through image processing. For this, the length of the cracks was measured along with the average crack width. The length was measured from the starting point of the crack to the end point as shown in Figure 3.11 (a). Measurements were taken across the width of the crack at three positions to obtain the average crack width, which is shown in Figure 3.11 (b). An average value was then obtained using the three crack width measurements taken.

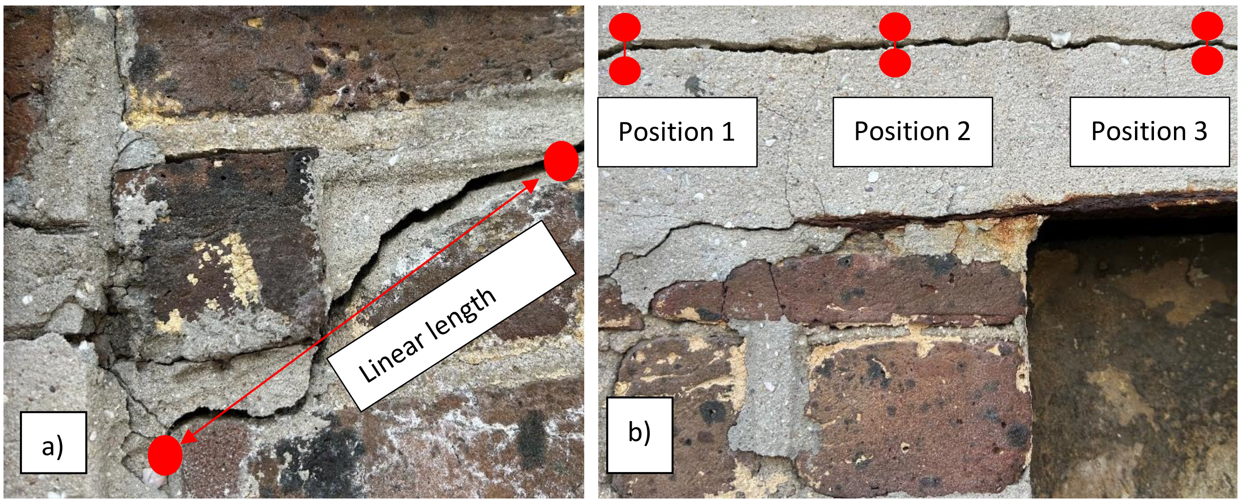


Figure 3.11 (a) The crack length measurement and (b) the crack width measurement.

This was completed for all the cracks which formed part of the dataset. Image processing was carried out once the full dataset was in place and all the physical dimensions were obtained.

3.10.2 Image processing-based measurement

For this research, the images underwent various stages of image processing. This included the conversion of images to greyscale, thresholding, binarisation, segmentation and skeletonisation. These processes were combined with Euclidean distance-based computation to obtain the average crack width and length.

Greyscale conversion

The first step in the image processing process was converting the input images or image dataset to a greyscale format. This is completed in the MATLAB Image Processing and Computer Vision toolbox by using the weighted sum of colour channels to produce a single intensity value for each pixel. The pixel values which result create a new single-channel image. This process simplifies the image, reducing data and highlighting structural details and variations, making the image easier to analyse. The conversion of a colour image (a) to grey scale format (b) is illustrated in Figure 3.12.

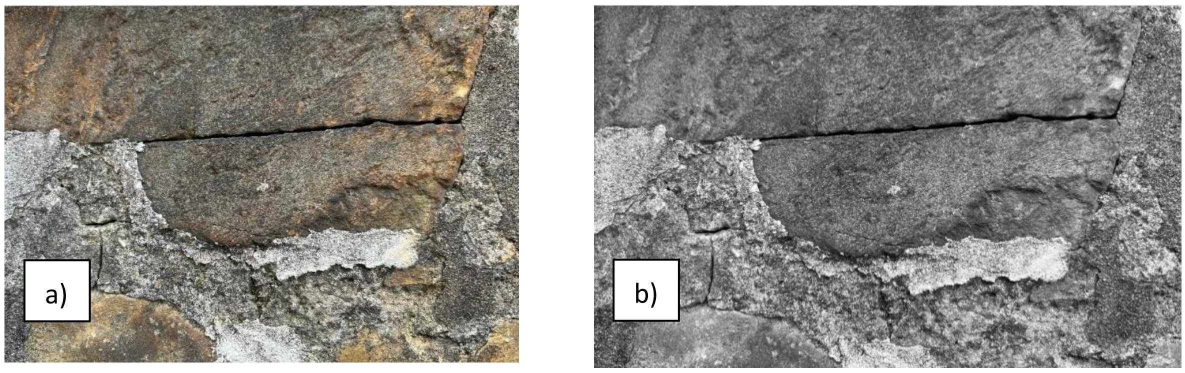


Figure 3.12 Conversion of a colour image to grey scale format: (a) original colour image and (b) grey scale image.

Thresholding, binarisation and segmentation

This stage of image processing involves thresholding, binarisation and segmentation. Thresholding involved dividing the grey scale image into regions based on their pixel intensity values. This process was carried out using global thresholding where a single threshold was applied to the entire image. The main objective was to highlight the areas of interest. This was completed by categorising pixels into various groups according to their intensities, and these groups included those below the threshold and those equal to or above the threshold. Those above the threshold were given a value, in this instance black, while those equal to or above the threshold were given another value (white), resulting in a segmented binary image. Furthermore, to ensure a complete representation of the crack structure, any holes in the binary were filled, with the largest connected component retained using area filtering.

Figure 3.13 illustrates the conversion of the grey scale image to a binarised image through the segmentation process.

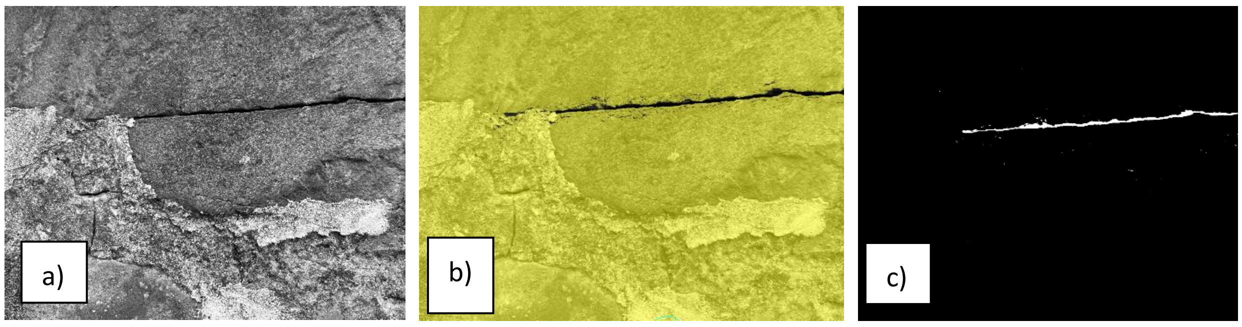


Figure 3.13 The conversion of the grey scale image to a binarised image: (a) the grey scale image where (b) a background mask is applied through thresholding to segment the region of interest and (c) the final binarised image.

Skeletonisation

Using the binary image obtained through segmentation the bwskel function was applied and skeletonisation was carried out on the image. This process reduced the shape of the structure to produce a one-pixel wide representation, while ensuring the crack topology was maintained.

Figure 3.14 shows the one-pixel wide "skeleton" in red overlaid on the binarised image.

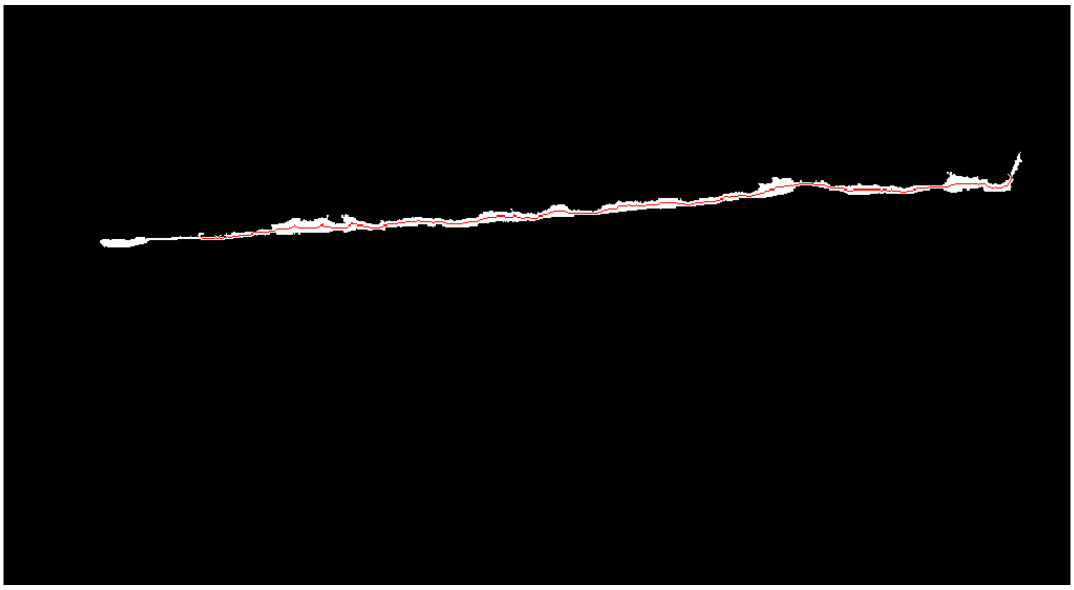


Figure 3.14 Skeletonisation carried out on the binarised image.

Distance-based computation

The Euclidean Distance Transform was applied to the binary image to quantify the length and average width of the cracks. This created a distance map where each pixel value represented the Euclidean distance to the nearest background pixel. Distances along the skeleton were then extracted from the distance map, with measurements taken from the centre of the skeleton to the background; the average crack width was then computed as a mean of the distances.

To determine the length of the cracks, the endpoints of the skeleton first had to be identified. This was completed using the bwmorph function. The Euclidean distance was then used to determine the straight-line distance between the two endpoints.

The final computation provided both the average crack width and length in pixel measurements, which were then converted to millimetres based on the original image's dimensions.

The final stage of the methodology entailed determining the error percentage achieved between the actual physically measured values and those obtained through image processing. The error percentage indicated how great a difference there was between the measured values and those obtained through image processing, and this was essential for assessing the performance of the image processing methodology.

3.11 Presentation of results

A performance metric was created to quantify the performance of the model. The model's predictions demonstrate its level of performance when using new data. One metric used to evaluate a classification models' performance is accuracy. The accuracy metric can be determined by dividing the number of accurate predictions by the overall number of predictions and computing the proportion of correct predictions. This can be seen in Equation (3.1).

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{3.1}$$

The misclassification rate is another often-used metric for evaluating a model. The proportion of inaccurate predictions was calculated as the misclassification rate. This is given by Equation (3.2).

$$Misclassification\,rate = \frac{FP + FN}{TN + TP + FN + FP} \tag{3.2}$$

Another metric used is precision, which is the model's accuracy in correctly predicting positive values, seen in Equation (3.3).

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

Recall, also known as sensitivity, is another metric that measures the model's strength in predicting positive outcomes (Kulkarni et al., 2020:87). Equation (3.4) shows the recall calculation.

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

In these calculations, TP indicates that both the actual and predicted values are positive; TN signifies that both the actual and predicted values are negative. FP shows that the actual value is negative, although the model predicted value is positive, and FN indicates that while the actual is positive, a negative predicted value was obtained (Kulkarni et al., 2020:86-87).

In addition, since the research presented a multi-class classification problem, precision and recall values were obtained for each damage class. A macro averaging strategy was utilised to derive a comprehensive precision and recall value. This process considered the precision and recall values of each class and calculated the average by dividing it by the number of damage classes. The resulting average value was then used as the overall precision and recall values. This approach allowed for a comprehensive evaluation of the model's performance across all damage classes.

The model's performance can be compared by using the accuracy, misclassification, precision and recall metrics. However, a confusion matrix was used for a more thorough analysis. The confusion matrix displays the number of observations for the actual and predicted class combinations.

3.11.1 Confusion matrix

Table 3.2 shows a typical confusion matrix with a table structure for classification, enabling one to visualise the algorithm's performance. Each column of the matrix shows the actual value, whereas each row of the matrix represents a prediction. True positive (TP), true negative (TN), false positive (FP), and false negative (FN) are the four cells in the output matrix (FN).

Table 3.2 Typical classification confusion matrix (Kulkarni et al., 2020:88).

		Predicted	
Actual	**Negative**	TN	FP
	Positive	FN	TP

A confusion matrix provides a tabular representation that displays the count of observations for each combination of the true class and the predicted class in a classification problem. It allows for a visual breakdown of the model's predictions, showing how many instances were correctly classified (true positives and true negatives) and how many were misclassified (false positives and false negatives).

Furthermore, the confusion matrix provides a comprehensive overview of the distribution of predictions and helps identify patterns of correct and incorrect classifications. This information is valuable for evaluating the model's effectiveness and gaining insights into specific areas where it may struggle or excel.

3.11.2 Receiver operating characteristics curve

The Receiver operating characteristic curve is another method of evaluating the classifier's performance. The false positive rate (FPR) is drawn on the x-axis against a y-axis consisting of the true positive rate (TPR). A threshold ranging from 0 to 1 is then defined for classification to be performed by the classifier. The FPR and TPR are plotted against each other for every value, as shown in Figure 3.15.

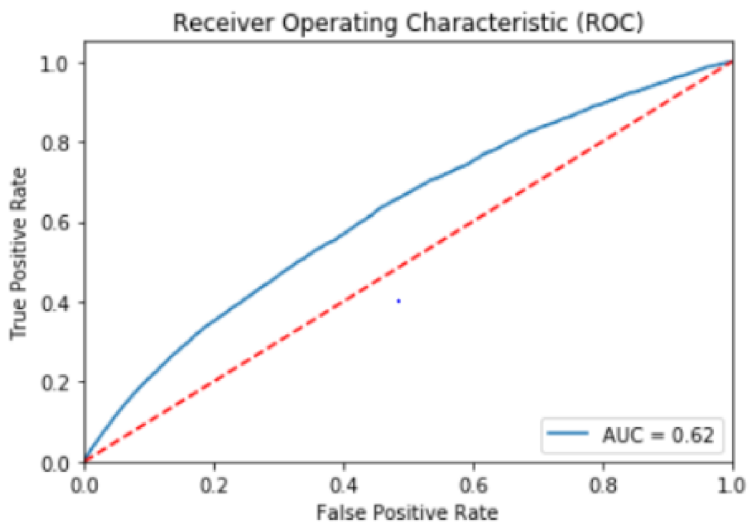


Figure 3.15 Typical ROC curve (Kulkarni et al., 2020:89).

Good classification is represented in the ROC curves in the top left corner, while poor classification is shown in the bottom right corner. A diagonal line is then plotted, and this represents random guessing. If the ROC curve falls anywhere below the diagonal line, its classification is deemed poorer than random guessing. If the classifier manages to reach the top left corner, it is deemed good (Kulkarni et al., 2020:88-89).

3.11.3 Image processing error rate

To assess the precision of the image processing technique in measuring the crack characteristics, the research quantified the performance using percentage of error. This metric quantified the difference between values gathered from physical measurements obtained on site and those obtained from image processing methodology.

This was computed for both the average crack width and length utilising Equation (3.5), which, according to Yapuncich (2018:141) is commonly used as an accuracy metric.

$$(\%)Error = \frac{Observed\ value - Predicted\ value}{Predicted\ value} \text{ x } 100 \tag{3.5}$$

Equation (3.5) calculates the percentage difference between the on site measurements, which was taken as the benchmark, and the values obtained through the complex image processing method.

Analysing the percent error figures yielded valuable information regarding the effectiveness of the image processing methods. A smaller percent error implied a more precise alignment of the measurements, indicating greater precision. Alternatively, a greater percentage indicated a larger deviation, highlighting places where the image processing may have difficulties or require improvement.

The analysis of the error percentage contributes to a full evaluation of the performance of the image processing methodology in assessing the crack characteristics and extent of damage.

3.12 Chapter summary

Heritage structures are important structures that need to be monitored regularly to ensure that if any damage is detected, remedial actions can be done on time. Machine learning and deep learning methods for image processing are methodologies that can be used to detect and classify damage. In this chapter, machine learning algorithms like SVMs, decision trees and k-nearest neighbours coupled with deep learning techniques such as CNNs were presented. Three case studies, namely Castle of Good Hope, Robben Island and Dal Josafat heritage sites in the Western Cape were described. The procedure of collecting data in the form of images to ascertain the performance of selected algorithms in detecting damage of heritage structures was also presented in this chapter.

Furthermore, chapter 3 described the performance metrics, i.e., accuracy, misclassification rate, precision, recall, confusion matrix and image processing rate. These metrics evaluated the performance of the machine learning algorithm. Chapter 4 discusses the results obtained from the research.

Chapter 4 Results and discussion

4.1 Introduction

This chapter presents the results of the research and investigation into the performance comparison of machine learning and deep learning algorithms in classifying damage in heritage structures in the Western Cape. The analysis of results includes an assessment of the model's classification performance on an image dataset containing defect classes such as crack, spalling, and intact areas. The research was based on case studies carried out at the Castle of Good Hope, Robben Island, and Dal Josafat heritage sites. The performance was based on metrics, i.e., accuracy, misclassification rate, precision and recall. Additionally, the results measuring the extent of damage using image processing methods are outlined.

Chapter 4 discusses the research findings, centred around the performance comparison of the machine learning and deep learning algorithms. The discussion involves analysing and evaluating the model's effectiveness in classifying the dataset of crack, spalling, and intact areas. It reaches deeper into the implications of these findings while also examining the performance of the models.

4.2 Image dataset

As mentioned in Chapter 3, the initial image and data captured at the Castle of Good Hope, Robben Island, and Dal Josafat heritage sites provided 626 defect images. This dataset comprised 314 crack images, 129 spalling images and 183 images of intact areas (see Table 3.1). Figures 4.1a, b and c show images of crack, spalling and intact areas.

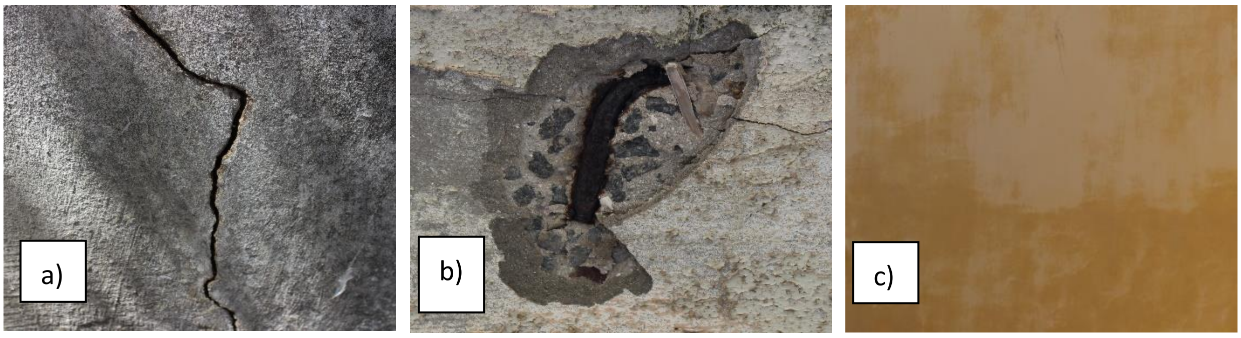


Figure 4.1 Image dataset sample of (a) crack, (b) spalling and (c) intact areas.

The uneven number of images across the crack, spalling, and intact areas categories provided a challenge as class imbalance emerged. Specifically, the spalling and intact classes had a smaller number of images compared to the crack class. Class imbalance is a common issue in machine learning and can impact the performance of models, especially in multi-class classification tasks such as defect classification in heritage structures. Class imbalance can introduce bias during the training process, where the model becomes more biased towards the majority class (in this case, the crack class) and may struggle to properly distinguish and learn the characteristics of the minority classes (spalling and intact). Bias can lead to a model being overly focused on classifying the majority class while neglecting the minority classes. Data augmentation was carried out to address the class imbalance problem.

4.2.1 Data augmentation

Choi (2020: 14), states that data augmentation is a technique that can increase the size of the training dataset by modifying or applying transformations to the dataset. The dataset becomes balanced by creating additional augmented images for the underrepresented classes. By balancing the dataset, the model can learn equally from each class.

The augmentation techniques implemented included some of those used by Choi (2020: 14), such as rotations and colour modification. These techniques help introduce diversity into the training data, which assists in creating a robust model. Rotating the images at various angles and modifying their colours helped the model better understand different orientations and lighting conditions that may be encountered in real-world scenarios.

However, although data augmentation can reduce the chance of model overfitting, it can still occur if the augmentations are carried out excessively. Choi (2020: 28), states that overfitting occurs when the model becomes too specialised in learning the training data and fails to generalise well to unseen data. Excessive data augmentation can result in too many variations of the same image, which may cause the model to memorise these augmentations rather than learn the necessary patterns. As a result, the model's performance might be worse on new, unseen data.

To avoid this, balancing and adding diversity to the training data was necessary but excessive augmentations had to be prevented. Therefore, augmentation techniques and the extent of the augmentation were carefully considered. Furthermore, the model's performance on the validation set during training was monitored, a technique advised by Baladram et al. (2020: 120), to ensure the augmentation carried out on the dataset was sufficient, avoiding model overfitting. For this research, augmentations were only applied to balance the different damage classes of the image dataset were kept to a minimum, while cross-validation was applied at the training stage to further ensure that overfitting did not occur. Data augmentation was also carried out to ensure that the minimum requirements for simpler image classification tasks were satisfied. For these tasks generally hundreds of images per class are required to allow for optimal model training and performance.

Data augmentation was completed by rotating images through 90 degrees and modifying the colours of various images at random. This provided a total of 1161 images in the dataset. Figure 4.2 shows the 90-degree rotation and colour modification augmentation techniques that balanced the image dataset.

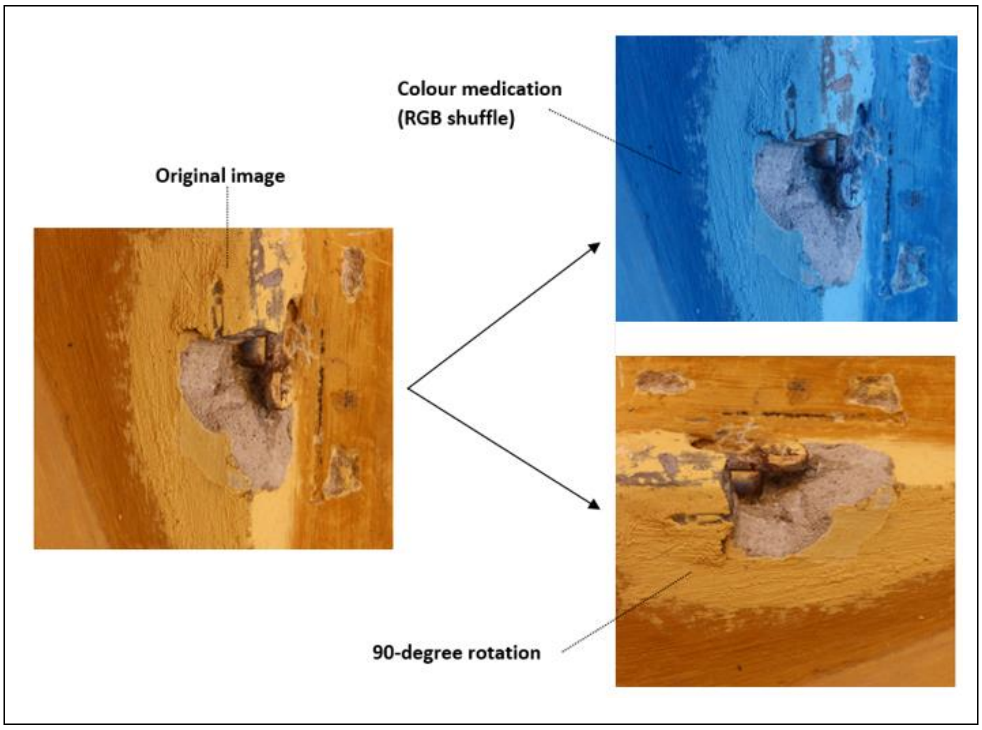


Figure 4.2 Typical data augmentation techniques carried out on the images collected from the Castle of Good Hope.

The outcome of the data augmentation process is presented in Table 4.1 below. Initially, the smallest image class, the spalling image category, underwent colour modification and a 90-degree rotation for all images. In the intact class, colour modification was applied to all images, while the rotation was applied to only 21 images. Finally, in the crack class, which initially had the largest number of images, 73 additional images were generated by performing a 90-degree rotation.

Table 4.1 Images gathered from each heritage site according to the crack, spalling and intact areas classes.

Data Collection	Original dataset	Rotation	Colour modification	Total
Crack	314	73	0	387
Spalling	129	129	129	387
Intact	183	21	183	387
Total	626	223	312	1161

Figure 4.3 shows a sample of 9 images of the final augmented image dataset used during the classification process. Image 1 to 5 provides examples of the spalling damage used in the dataset. However, Image 2 and Image 3 differ as these images were augmented, with a colour adjustment made. Additionally, Image 5 also underwent augmentation and was rotated by 90 degrees. Image 6 and Image 7 show an example of the intact areas. However, Image 6 was rotated by 90 degrees. Lastly, Image 8 and Image 9 show examples of crack images used.

The final dataset was made up of 1161 images, obtained after the data augmentation process, of which 65% was used for training data, 15% for validation data and 20% for test data for the classification of images.

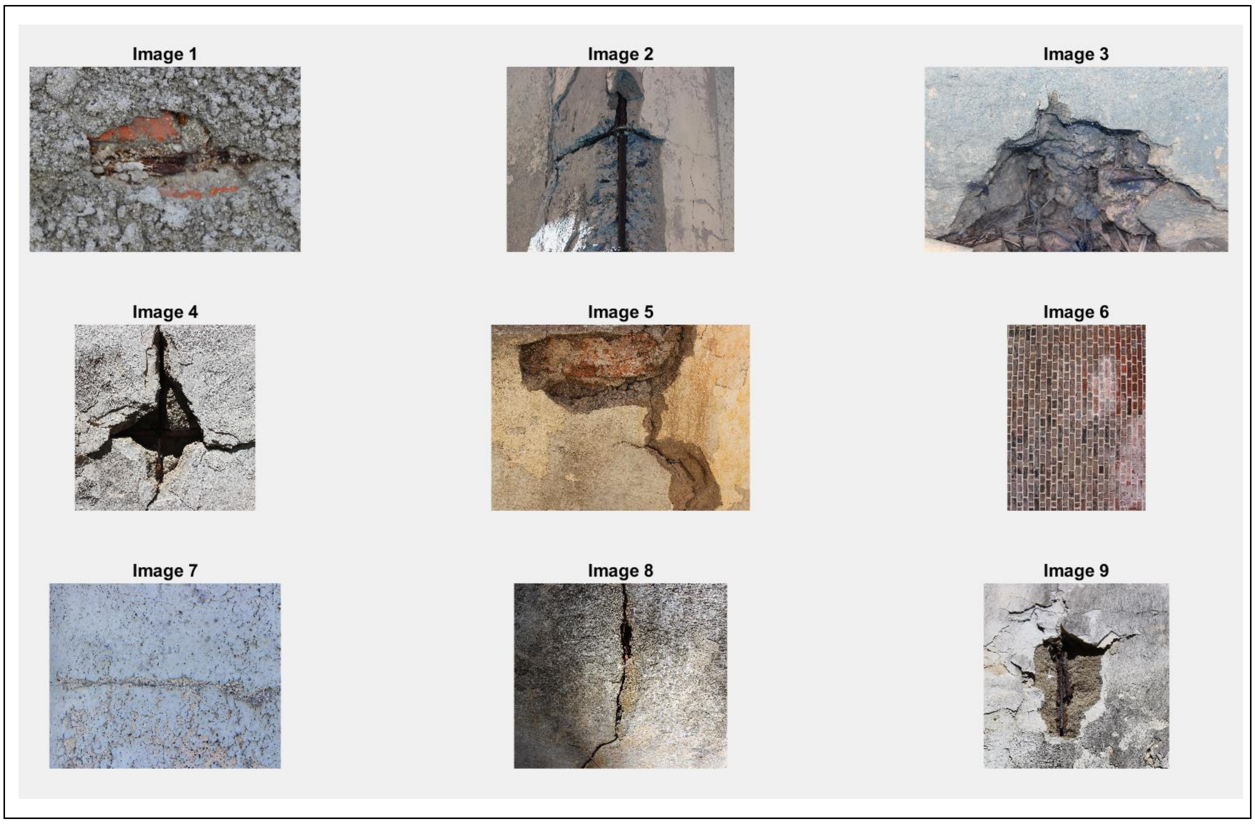


Figure 4.3 Sample of the augmented image dataset used for the classification of images.

4.3 Classification of damage classes

To evaluate the effectiveness of the different machine learning algorithms and the deep learning model, their test prediction accuracy and other metrics such as misclassification rate, precision, and recall were assessed.

This study centred on two main methodologies shown in Figure 4.4: (a) the Bag of Words technique employed with the traditional machine learning models and (b) the deep learning CNN approach.

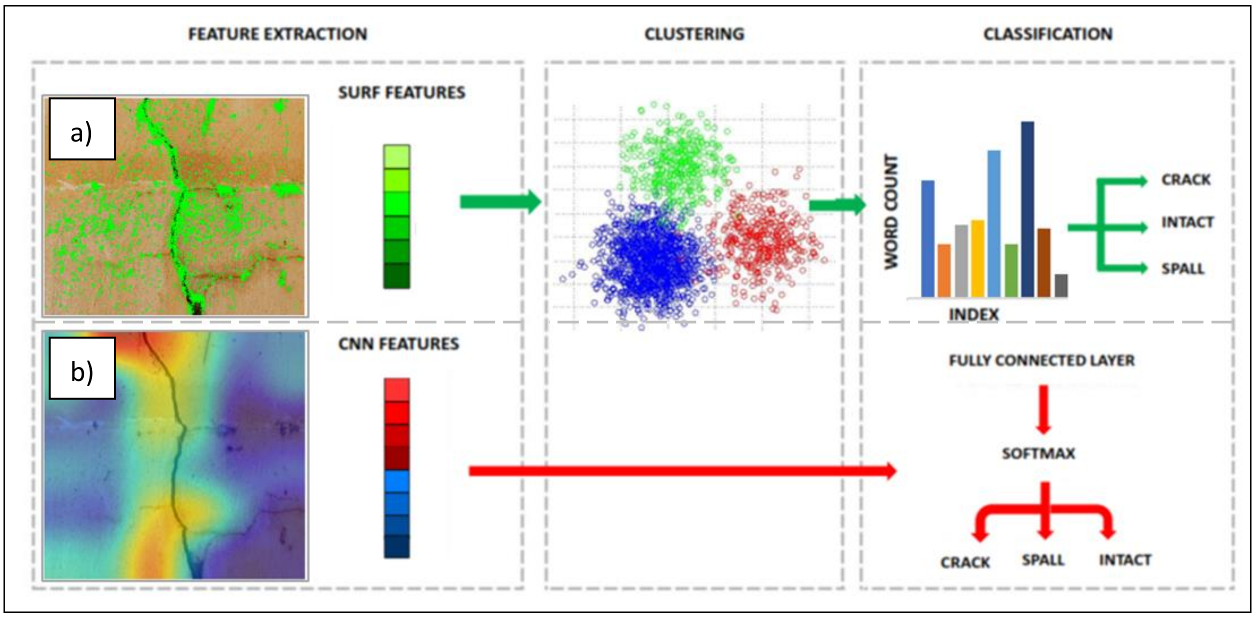


Figure 4.4 (a) The Bag of Visual Words using SURF feature extraction and (b) the CNN feature extraction process for the classification of defects.

4.3.1 Traditional machine learning algorithm-based classification

The performance of machine learning models, whether conventional or deep learning, heavily relies on the quality of the features extracted from the input data. Training algorithms on complex features would allow for a robust model capable of classification in a real-world situation (Choi, 2020: 111; Kim et al., 2020:2). In the context of defect classification in heritage structures, the choice of the feature extraction technique plays a critical role in determining the model's ability to capture the unique patterns and characteristics of different defects. For the traditional machine learning models, feature extraction was carried out using the Bag of Visual Words approach.

Bag of Visual Words approach

The Bag of Visual Words methodology was utilised for the traditional machine learning models. This involved a methodology similar to that of Ghalyan et al. (2019: 3899-3913) and Kim et al. (2018:1-14). Using the Bag of Visual Words technique allowed for extracting local features from images, which were then clustered to create a "visual vocabulary". In this case, the SURF function extracted local features from the images.

The step by step process of the Bag of Visual Words technique for defect classification is illustrated in Figure 4.5. The process starts with extracting SURF features from the images (Figure 4.5a), followed by clustering these extracted features to form a histogram (Figure 4.5b). This histogram is a crucial component in classifying defects using the Bag of Visual Words approach.

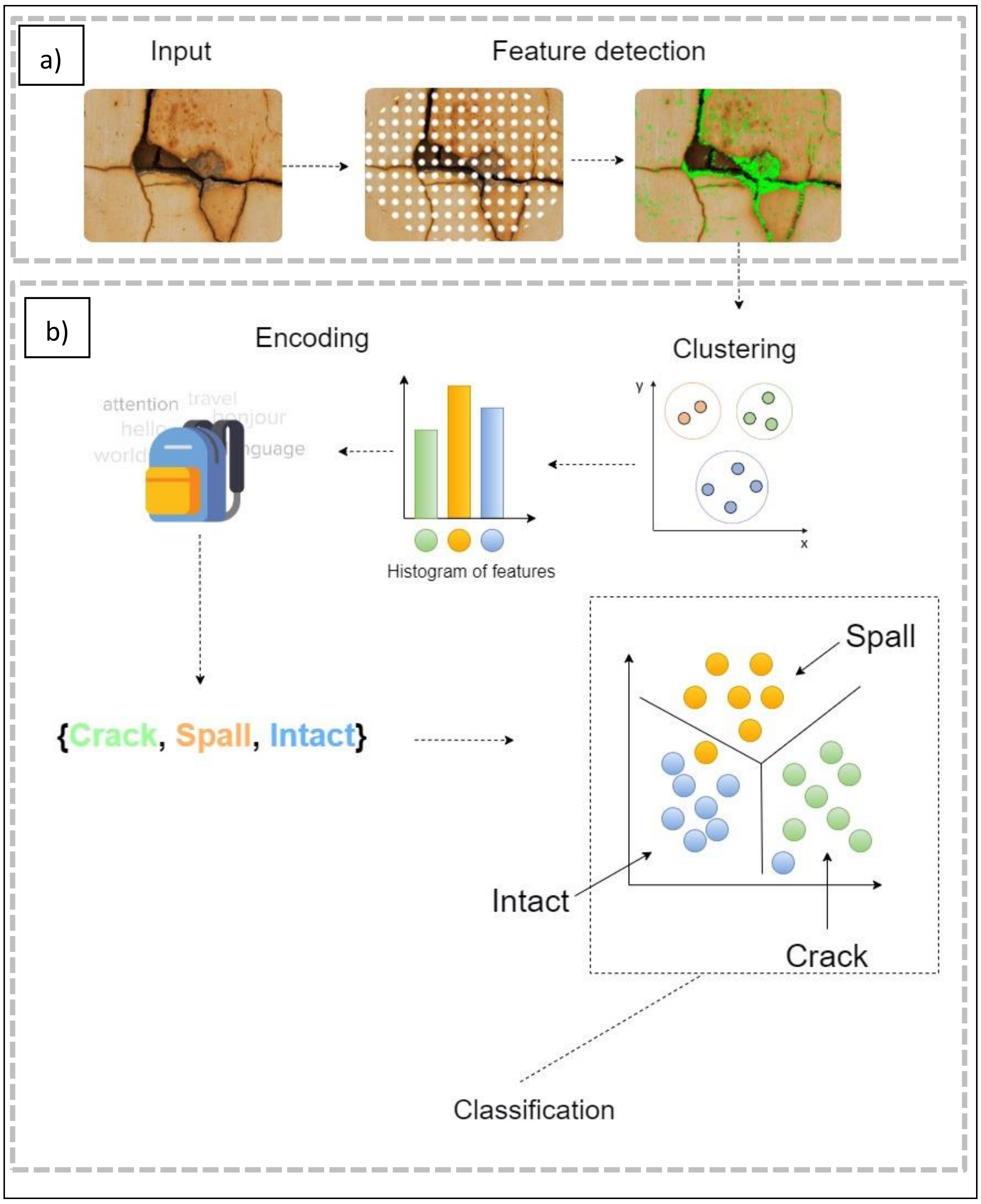


Figure 4.5 Bag of Visual Words feature extraction and a traditional machine learning classification process.

Figure 4.6 below shows the features extracted using the SURF function and Bag of Visual Words methodology. The input images are shown in Figure 4.6 (a), (b) and (c). Features are then extracted, and this is visible as the green spots in Figure 4.6 (d), (e) and (f). These features play a big role in the defect classification process. The extracted features are used to characterise the patterns and structures present in the images, aiding in the classification task.

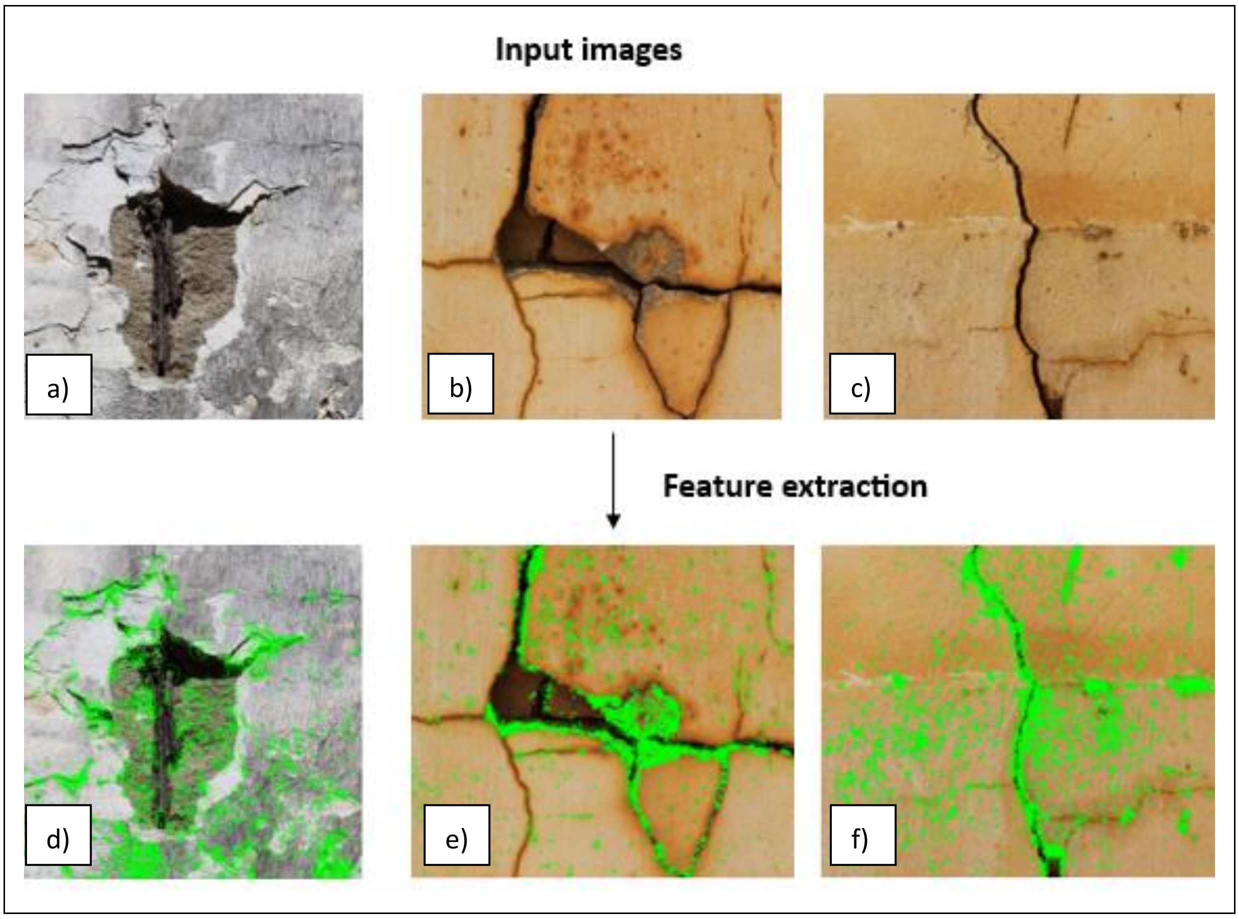


Figure 4.6 Bag of Visual Words feature detection and extraction using the Bag of Visual Words approach and the SURF function.

The following sub-sections describe the training and testing results of the decision trees, SVMs, KNNs and artificial neural network algorithms for the defect classification task. These sections highlight the model's performance in training and testing and also describe their effectiveness and reliability in addressing the defect classification problem.

4.3.2 Decision tree model results

Training

Training was completed using the fine, medium, coarse, and optimisable decision tree algorithms. Table 4.2 shows that the optimisable decision tree performed better during training, achieving an accuracy of 68.28%, closely followed by the fine decision tree with an accuracy of 66.77%, the coarse decision tree with an accuracy of 65.59% and the medium decision tree with the lowest accuracy of 64.62%.

Table 4.2 Defect classification accuracy achieved by the decision tree models during training.

Model Number	Model Type	Type	Accuracy % (Training)
1	Tree	Fine Tree	66.77
2	Tree	Medium Tree	64.62
3	Tree	Coarse Tree	65.59
4	Tree	Optimisable Tree	68.28

Figure 4.7 presents the optimisable decision tree confusion matrix that illustrates the classification outcomes during training for the three damage classes, which were used to determine the training accuracy.

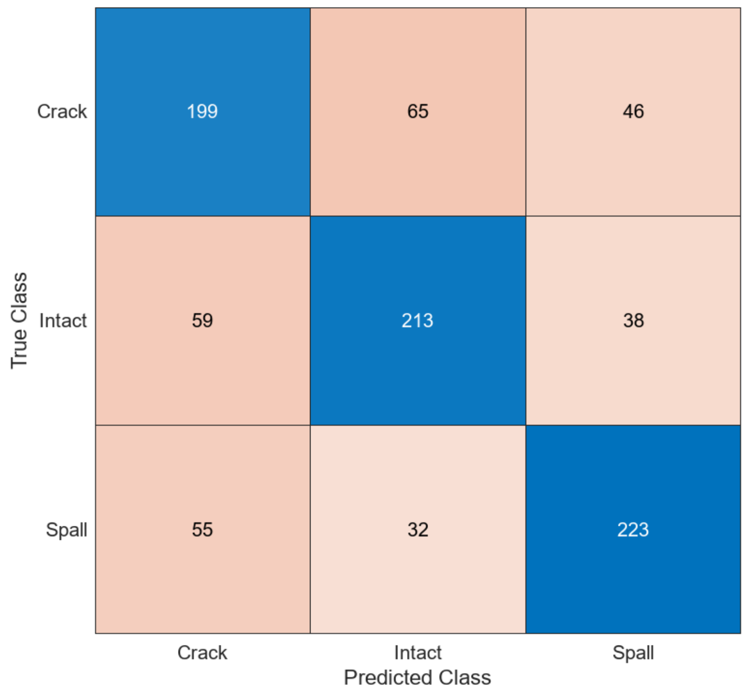


Figure 4.7 Optimisable decision tree confusion matrix obtained during training.

By analysing the optimisable decision tree confusion matrix during training, the following was observed:

- For the crack class, out of a total of 310 instances, 199 were correctly classified as crack (true positives), 65 instances were incorrectly classified as intact, and 46 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of a total of 310 instances, 213 were correctly classified as intact (true positives), 59 instances were incorrectly classified as crack, and 38 instances were misclassified as spalling (false negatives).
- Out of the 310 predictions for the spall class, 223 were correctly classified as spalling (true positives), 55 instances were incorrectly classified as crack, and 32 instances were misclassified as intact (false negatives).

The final training accuracy was determined by extracting the true positive and negative values. Using Equation (3.1), the number of accurate predictions (635) was divided by the total number of predictions (930), using the values obtained from the confusion matrix, providing a training accuracy of 68.28%. The accuracy was the highest amongst the decision tree models at training.

ROC curves generated for the optimisable decision tree during training are shown in Figure 4.8.

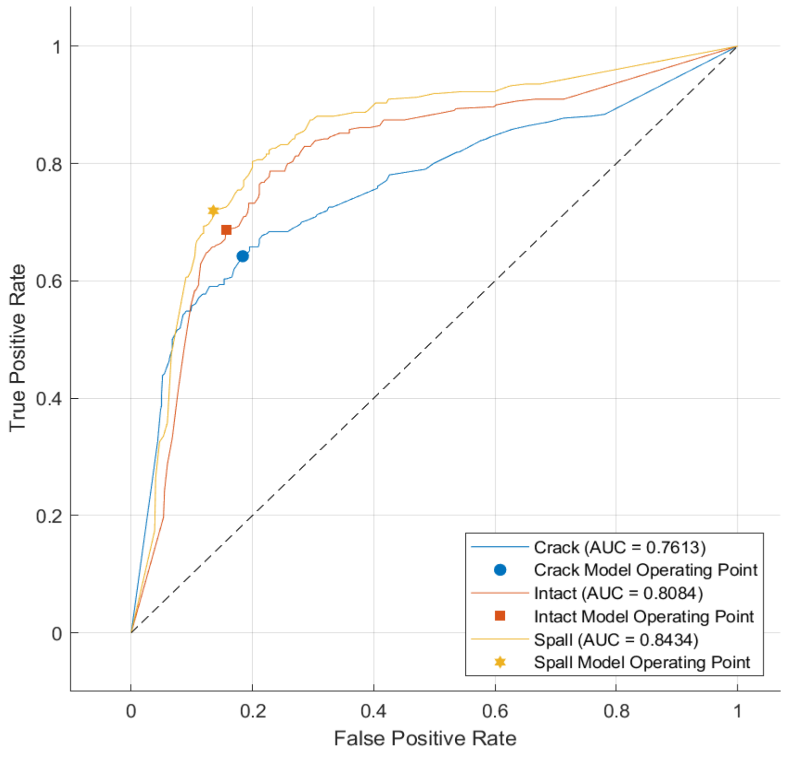


Figure 4.8 ROC curves for the optimisable decision tree during training.

The ROC curves depicted in Figure 4.8 for each defect class illustrate the relationship between the true and false positive rates across the various damage classes during training. The area under the curve (AUC) for each ROC curve quantitatively measures the model's overall performance. The AUC value ranges from 0.5 to 1, with a higher value indicating better classification performance.

According to Mandrekar (2010: 1316), an AUC of 0.5 suggests that the model is performing at random, while an AUC value of 0.7 to 0.8 indicates an acceptable or fair level of classification, between 0.8 and 0.9 indicates excellent performance, and those above 0.9 are considered to show outstanding model performance.

In Figure 4.8, the AUC values for each damage class are provided. The crack class achieved an AUC of 0.7613, the intact class achieved 0.8084, and the spalling class obtained an AUC of 0.8434. These values show that the model is achieving classification at an acceptable level for cracks while spalling and intact area classifications are considered excellent.

Testing

Testing was completed using the trained fine, medium, coarse, and optimisable decision tree algorithms. Table 4.3 presents the accuracy achieved by each decision tree model during testing.

Table 4.3 Defect classification accuracy achieved by the decision tree models during testing.

Model Number	Model Type	Type	Accuracy % (Testing)
1	Tree	Fine Tree	76.62
2	Tree	Medium Tree	72.73
3	Tree	Coarse Tree	73.59
4	Tree	Optimisable Tree	71.86

According to Table 4.3, the fine decision tree performed best, with an accuracy of 76.62%. This was followed by the coarse decision tree with the second highest accuracy of 73.59%, the medium decision tree with an accuracy of 72.73% and lastly the optimisable tree with an accuracy of 71.86%.

These results show the performance of the fine decision tree in terms of test accuracy. Despite having the highest accuracy during training, the optimisable decision tree ranked last in terms of test accuracy among the decision tree models. Generally, all models showed an increase in accuracy from training to testing.

Figure 4.9 shows the fine decision tree confusion matrix, and illustrates the classification outcomes for the different damage classes during testing. These values were used to determine the final accuracy of the model during testing.

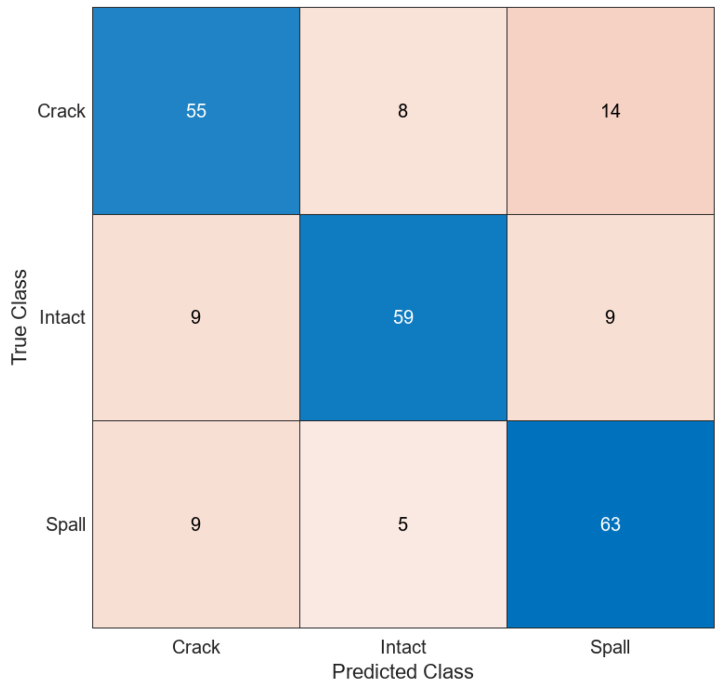


Figure 4.9 Fine decision tree confusion matrix during testing.

By analysing the fine decision tree confusion matrix during testing, the following was observed:

- For the crack class, out of a total of 77 instances, 55 were correctly classified as crack (true positives), 8 instances were incorrectly classified as intact, and 14 instances were incorrectly classified as spalling (false negatives).

- In the intact class, out of the 77 predictions made, 59 were correctly classified as intact (true positives), 9 instances were incorrectly classified as crack, and 9 instances were misclassified as spalling (false negatives).
- Out of the 77 predictions for the spall class, 63 were correctly classified as spalling (true positives), 9 instances were incorrectly classified as crack, and 5 instances were misclassified as intact (false negatives).

The final test accuracy was calculated, where the number of accurate predictions (177) was divided by the total number of predictions (231), providing a test accuracy of 76.62%. Equation (4.1) below provides the calculation.

$$\begin{aligned} Accuracy &= \frac{55+59+63}{231} \times 100 \\ &= 76.62\% \end{aligned} \tag{4.1}$$

To further evaluate the fine decision tree model, ROC curves were generated and are shown in Figure 4.10.

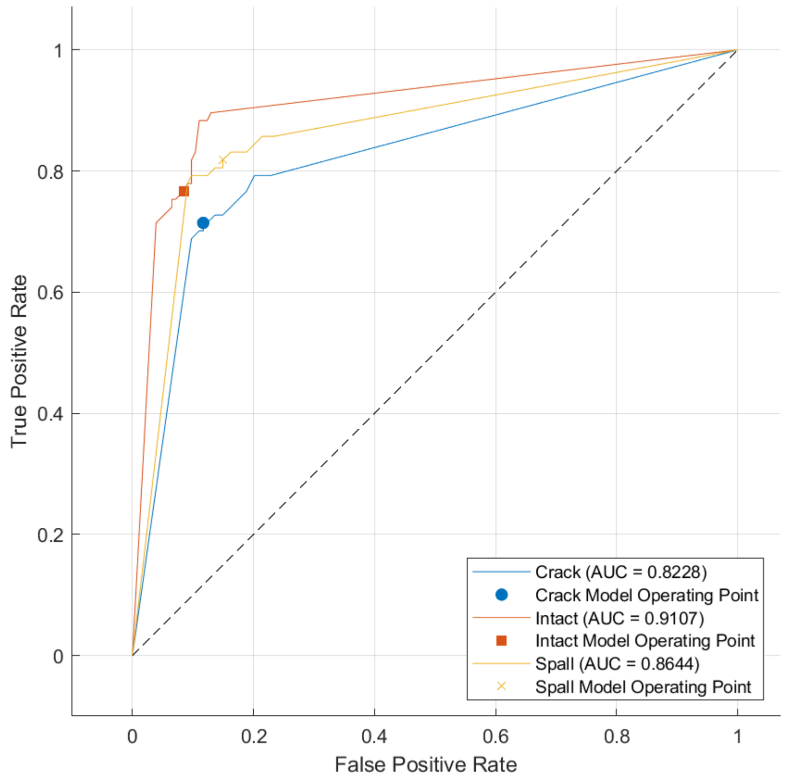


Figure 4.10 ROC curves for the fine decision tree model’s performance during testing.

Figure 4.10 shows the relationship between the true and false positive rates across the various damage classes during testing. The shape and steepness of the curves provide an idea of the model's overall performance. The ROC curves of the fine decision tree show consistent upward trends, indicating high true positive rates and low false positive rates. Furthermore, the curves rise steeply and approach the top-left corner of the axis, indicating a good classification performance. Additionally, the curves indicate the model's classification performance, with the intact classification performance proving to be the most effective as it is closest to the y-axis and rises almost vertically upwards towards the top left corner of the axis. The performance is closely followed by the classification of the spall class and then the classification of the crack class. The performance is further reflected in the AUC values obtained for each class.

The AUC values for each damage class provided show that the crack class obtained an AUC of 0.8228, the intact class obtained an AUC of 0.9107, and the spalling class obtained an AUC of 0.8644. These values show that the model achieves classification at an excellent level for cracks and spalling while achieving outstanding performance for the classification of intact areas (Mandrekar, 2010: 1316).

Other metrics such as misclassification rate, precision and recall were also reviewed during testing to further compare model performance. The misclassification rate indicated the percentage of incorrectly classified instances, where a lower misclassification rate indicates a better overall performance. The precision represents the percentage of correctly identified positive instances, measuring the accuracy of positive predictions. Lastly, the recall indicated the percentage of correctly identified instances from the positive class. Higher values for precision and recall indicate better performance. The misclassification rate, precision and recall were calculated according to Equation (3.2), Equation (3.3) and Equation (3.4) respectively.

Table 4.4 presents the misclassification rate, precision and recall of the different decision tree models at the testing phase.

Table 4.4 Defect classification misclassification rate, precision and recall of the decision tree models during testing.

Model Number	Model Type	Type	Misclassification rate %	Precision %	Recall %
1	Tree	Fine Tree	23.38	76.85	76.62
2	Tree	Medium Tree	27.27	73.54	72.73
3	Tree	Coarse Tree	26.41	73.79	73.59
4	Tree	Optimisable Tree	28.14	72.19	71.86

Based on Table 4.4, the fine decision tree's performance stood out compared to the other decision tree models as it achieved the lowest misclassification rate and the highest precision and recall scores.

The misclassification rate was determined to be 23.38% and was determined by measuring the proportion of misclassified instances (54) relative to the total number of predictions (231). Equation (4.2) below provides the calculation for the misclassification rate.

$$\begin{aligned} Misclassification\ rate &= \frac{8+14+9+9+9+5}{231} \times 100 \\ &= 23.38\% \end{aligned} \tag{4.2}$$

Precision and recall were calculated individually for each damage class due to the nature of the multi-class classification task involving three damage classes.

To determine the precision, the crack class provided 55 true positive predictions and 18 false positive predictions; the intact class produced 59 true positive predictions and 13 false positive predictions; and the spall class produced 63 true positive predictions and 23 false positive predictions. A precision rate of

75.34% was obtained for the crack class, 81.94% was obtained for the intact class, and 73.25% was obtained for the spall class. This produced an overall precision rate of 76.85% for the fine tree decision tree model. This means that when the model predicted an instance as positive, it was accurate 76.85% of the time. The calculation of the model precision is shown in Equation (4.3) below.

$$
\begin{aligned}
Precision\ crack\ class &= \frac{55}{9+9+55} \times 100 \\
&= 75.34\% \\[1em]
Precision\ intact\ class &= \frac{59}{8+5+59} \times 100 \\
&= 81.94\% \\[1em]
Precision\ spall\ class &= \frac{63}{14+9+63} \times 100 \\
&= 73.26\% \\[1em]
Average\ precision &= \frac{75.34+81.94+73.26}{3} \\
&= 76.85\%
\end{aligned}
\tag{4.3}
$$

The precision values for each class are presented in Figure 4.11 below.

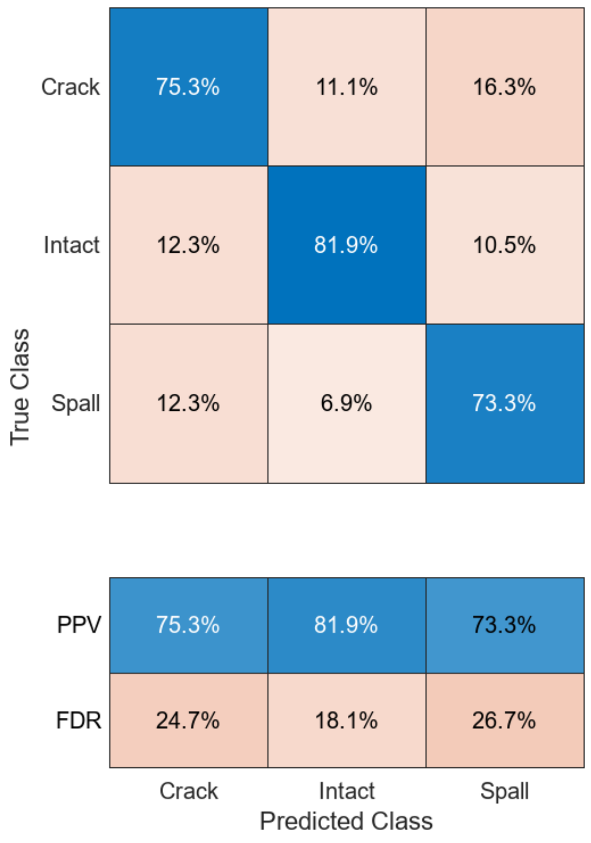


Figure 4.11 The precision or positive predictive value (PPV) and false discovery rate (FDR) achieved for each damage class, using the fine decision tree model.

Figure 4.11 also provides the false discovery rate for each class, indicating the proportion of positive predictions that are false positives. The crack, intact, and spall classes obtained false discovery rates of 24.7%, 18.1%, and 26.7%, respectively. A higher false discovery rate was observed for the spall class, while lower rates were observed for the crack and intact classes.

To determine the recall, the crack class provided 55 true positive predictions and 22 false negative predictions; the intact class provided 59 true positive predictions and 13 false negative predictions; and the spall class produced 63 true positive predictions and 23 false negative predictions. This produced a recall score of 71.42% for the crack class, 76.62% for the intact class and 81.81% for the spall class. An overall recall rate of 76.62% was obtained for the fine decision tree model, indicating the model's ability to identify instances of each class correctly and the model's ability to detect positive instances while producing a relatively low rate of false negatives. Equation (4.4) below provides the calculation used to obtain the overall recall.

$$\begin{aligned}
Recall\ crack\ class &= \frac{55}{55+8+14} \times 100 \\
&= 71.43\% \\
Recall\ intact\ class &= \frac{59}{9+59+9} \times 100 \\
&= 76.62\% \\
Recall\ spall\ class &= \frac{63}{9+5+63} \times 100 \\
&= 81.82\% \\
Average\ recall &= \frac{71.43+76.62+81.82}{3} \\
&= 76.62\%
\end{aligned} \tag{4.4}$$

Figure 4.12 below presents the recall values for each class.

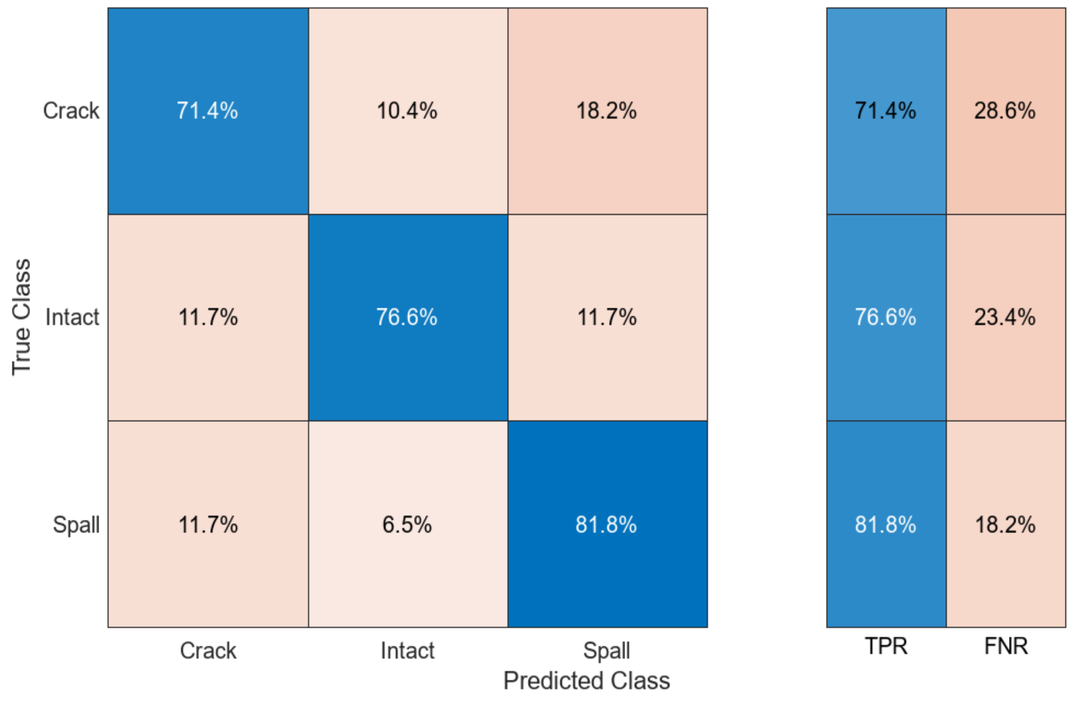


Figure 4.12 The recall or true positive rate (TPR) and false negative rate (FNR) achieved for each damage class, using the fine decision tree model.

The false negative rates achieved for the crack, intact and spall classes were 28.6%, 23.4% and 18.2% respectively. Furthermore, a lower false negative rate shows that this model is better at identifying the positive instances, while a higher false negative rate indicates that the model is missing several positive instances.

By analysing the precision and recall values for each damage class individually, it was observed that the prediction of the crack class achieved a precision of 75.3% and a recall of 71.4%. This shows that the model can correctly identify positive instances for the crack class, but some instances were misclassified as positive. The precision achieved for the prediction of the crack class indicates that of the crack predictions, 75.3% belonged to the crack class. The recall rate of 71.4% achieved for this class indicates that the model captured 71.4% of all the crack instances.

For the prediction of the spall class a precision of 73.3% and a recall of 81.8% were obtained, indicating that the model could correctly classify positive instances for this class, but there were also some false positive and false negative predictions. The precision rate of 73.3% indicates that the model correctly classified 73.3% of all the spall predictions. The recall rate of 81.8% indicates that the model captured 81.85% of all the spall instances.

For the prediction of the intact class, a precision of 81.9% and a recall of 76.6% was achieved. This indicates that the model had a high accuracy in correctly identifying positive instances for the intact class. However, again, some false positives and negatives were present. The precision rate of 81.9% achieved for this class indicates that the model accurately classified 81.9% of all the intact class predictions. Furthermore, the recall rate of 76.6% shows that the model captured 76.6% of all the intact instances.

Classification

Figure 4.13 shows an image obtained from the Castle of Good Hope and the defect classification results achieved by the fine decision tree model during testing. The visual representation displays both the true or actual label of the image and the label predicted by the model, along with the associated confidence score. In this case, the image belonged to the crack defect class, and the model accurately classified it as crack damage with a confidence score of 1.00 (100%).

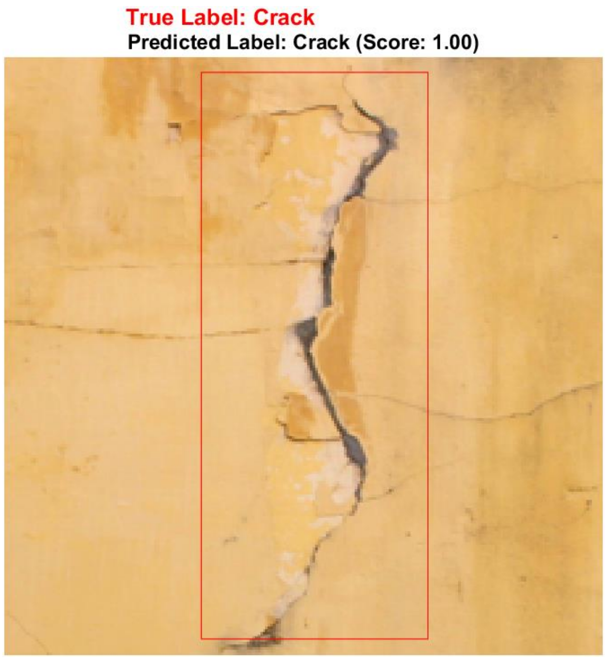


Figure 4.13 Image classification carried out by the fine decision tree model during testing on a crack image obtained from the Castle of Good Hope.

4.3.3 Support vector machine model results

Training

Training was carried out using the seven different SVM models. These were the linear, quadratic, cubic, fine Gaussian, medium Gaussian, coarse Gaussian and optimisable SVM models. Table 4.5 presents the accuracy achieved by each SVM model during training.

Table 4.5 Defect classification accuracy achieved by the SVM models during training.

Model Number	Model Type	Type	Accuracy % (Training)
5	SVM	Linear SVM	77.31
6	SVM	Quadratic SVM	85.38
7	SVM	Cubic SVM	84.95
8	SVM	Fine Gaussian SVM	66.02
9	SVM	Medium Gaussian SVM	80.00
10	SVM	Coarse Gaussian SVM	63.33
11	SVM	Optimisable SVM	85.48

Table 4.5 shows that the optimisable SVM model performed best during training, achieving an accuracy of 85.48%, followed by the quadratic SVM model with 85.38% and the cubic SVM model with 84.95%. The medium Gaussian, linear, fine Gaussian and coarse Gaussian SVM models performed less accurately during training, achieving accuracy results of 80.00%, 77.31%, 66.02% and 63.33%, respectively.

Figure 4.14 shows the optimisable SVM model confusion matrix and this provides the classification outcomes during training for the three damage classes. These were used to determine the accuracy during training.

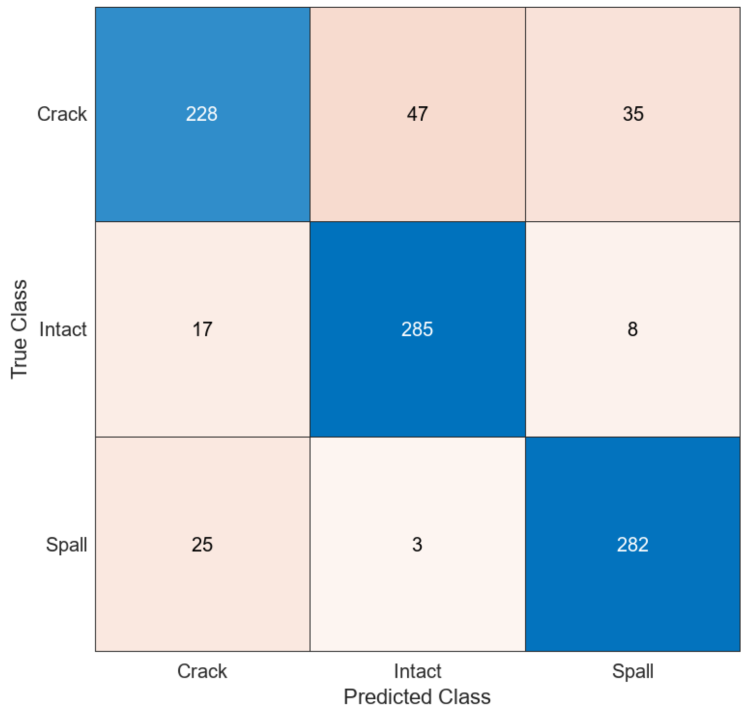


Figure 4.14 Optimisable SVM model confusion matrix during training.

By analysing the optimisable SVM model confusion matrix at training, the following was observed:

- For the crack class, out of a total of 310 instances, 228 were correctly classified as cracks (true positives), 47 instances were incorrectly classified as intact, and 35 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of a total of 310 instances, 285 were correctly classified as intact (true positives), 17 instances were incorrectly classified as cracks, and 8 instances were misclassified as spalling (false negatives).
- Out of the 310 predictions for the spall class, 282 were correctly classified as spalling (true positives), 25 instances were incorrectly classified as cracks, and 3 instances were misclassified as intact (false negatives).

The training accuracy was determined by extracting the true positive and negative values. The number of accurate predictions (795) was divided by the total number of predictions (930), and this provided a training accuracy of 85.48%.

ROC curves were generated for the optimisable SVM model during training as shown in Figure 4.15.

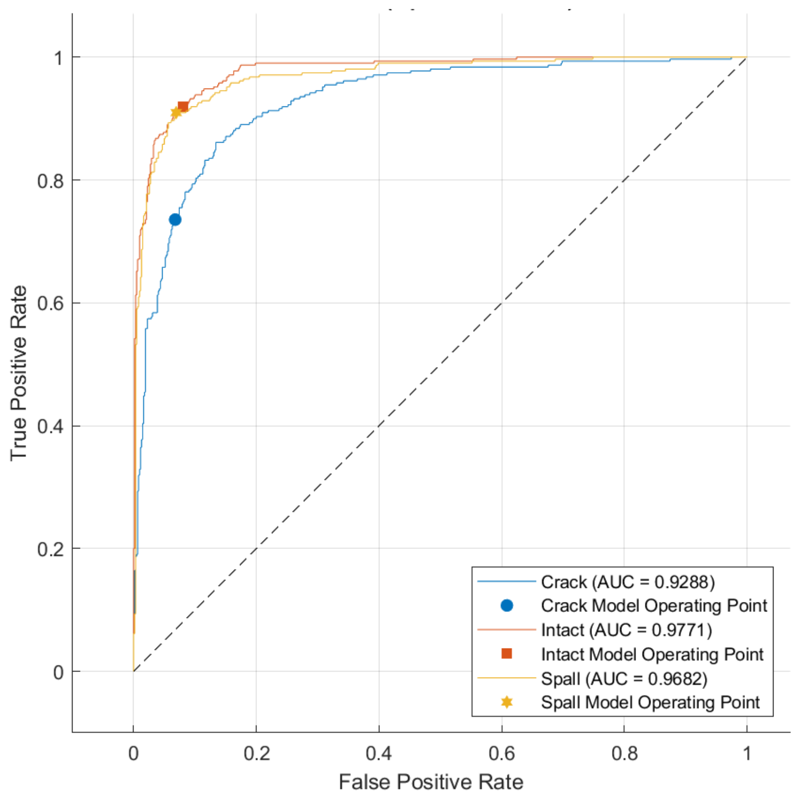


Figure 4.15 ROC curves for the Optimisable SVM model during training.

Figure 4.15 illustrates the relationship between the true and false positive rates across the various damage classes during training for the optimisable SVM. Figure 4.15 also provides the AUC values for each damage class. For the classification of cracks, an AUC value of 0.9288 was achieved, for the intact class an AUC value of 0.9771 was obtained, while an AUC value of 0.9682 was obtained for the classification of spall damage. Using the ROC curve and AUC rating system provided by Mandrekar (2010: 1316), the optimisable SVM model produced an outstanding performance for the classification of cracks, spalling and intact areas.

Furthermore, the ROC curves for the performance in classifying intact and spall classes exhibit consistent upward trends, rising almost uniformly together, indicating high true positive rates and low false positive rates for both classes. The curves both rise steeply and approach the top-left corner of the axis together, indicating a good classification performance. Additionally, the curves maintain close proximity to the top left corner of the axis, further highlighting the model's ability to classify instances from different damage classes accurately. The performance in classifying the crack class proved to be slightly less effective than the other two classes as the curve deviates away from the y-axis and does not reach the same point/height towards the upper left top corner of the axis as the intact and spall curves.

Testing

Testing was completed using the trained linear, quadratic, cubic, fine Gaussian, medium Gaussian, coarse Gaussian and optimisable SVM models. Table 4.6 presents the accuracy achieved by the different SVM models during testing.

Table 4.6 Defect classification accuracy achieved by the SVM models during testing.

Model Number	Model Type	Type	Accuracy % (Testing)
5	SVM	Linear SVM	74.46
6	SVM	Quadratic SVM	86.58
7	SVM	Cubic SVM	88.31
8	SVM	Fine Gaussian SVM	72.29
9	SVM	Medium Gaussian SVM	75.76
10	SVM	Coarse Gaussian SVM	67.97
11	SVM	Optimisable SVM	85.28

Table 4.6 shows that the cubic SVM model performed best, achieving an accuracy of 88.31% during testing. This was followed by the quadratic SVM model with 86.58% and the optimisable SVM model with 85.28%. The medium Gaussian, linear, fine Gaussian and coarse Gaussian SVM models performed less accurately during testing, achieving final accuracies of 75.76%, 74.46%, 73.39% and 67.97%, respectively.

These results highlight the accuracy of the cubic SVM model during testing, with its accuracy increasing from 84.95% in training to 88.31% during testing. Furthermore, the results also show the drop in accuracy of the optimisable SVM model from 85.48% in training to 85.28% in testing. However, this was not the only SVM model which showed a drop in performance from training to testing, with the linear and medium Gaussian SVM models also showing a lower accuracy in testing. The other SVM models, such as the quadratic, fine and coarse Gaussian models all had improved accuracy from training to testing.

Figure 4.16 shows the cubic SVM model confusion matrix and illustrates the classification outcomes for the different damage classes during testing.

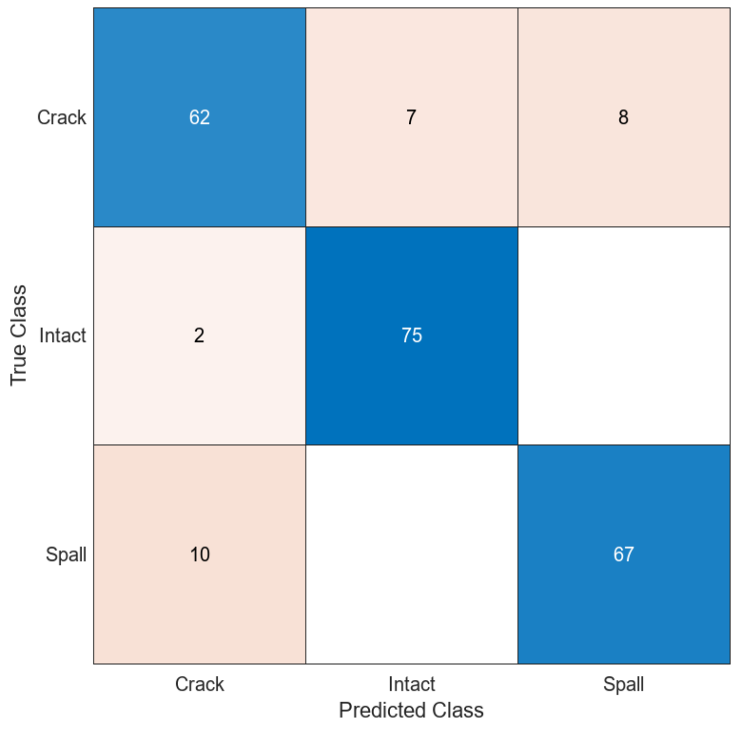


Figure 4.16 Cubic SVM model confusion matrix during testing.

By analysing the cubic SVM model confusion matrix during testing, the following was observed:

- For the crack class, out of a total of 77 instances, 62 were correctly classified as cracks (true positives), 7 instances were incorrectly classified as intact, and 8 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of the 77 predictions made, 75 were correctly classified as intact (true positives), 2 instances were incorrectly classified as cracks, and 0 instances were misclassified as spalling (false negatives).
- Out of the 77 predictions for the spall class, 67 were correctly classified as spalling (true positives), 10 instances were incorrectly classified as cracks, and 0 instances were misclassified as intact (false negatives).

By extracting the true positive and negative values, the number of accurate predictions (204) was divided by the total number of predictions (231), providing a final test accuracy of 88.31%.

To further evaluate the cubic SVM model, ROC curves were generated and are shown in Figure 4.17.

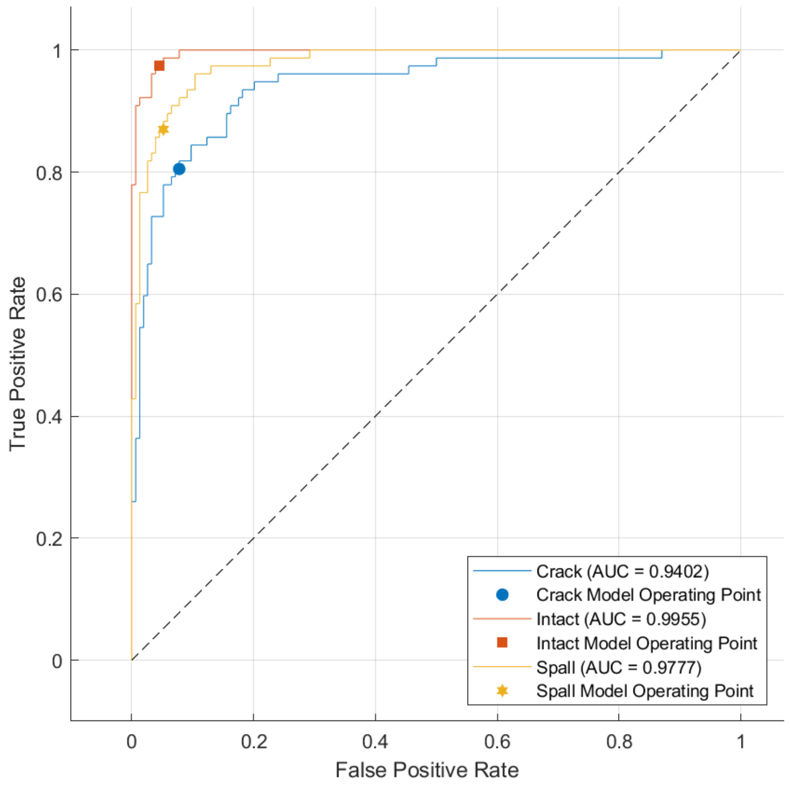


Figure 4.17 ROC curves for the cubic SVM model's performance during testing.

Figure 4.17 shows the ROC curves generated for each damage class when classifying with the cubic SVM model. Similar to the ROC curves obtained during training, the cubic SVM curves show consistent upward trends (almost vertical), indicating a very high true positive rate and a very low false positive rate. All three curves rise steeply, almost vertically upwards and approach the top-left corner of the axis, indicating an outstanding classification performance. Furthermore, for the classification of the intact class, the performance is almost perfect, with the curve reaching the highest point out of all three classes, this is followed by the performance in classifying the spall damage and then the performance in crack classification. The model, therefore, demonstrates a favourable balance between precision and recall, with a high true positive rate and a low false positive rate.

Figure 4.17 displays the individual AUC values for each damage class. The crack class achieved an AUC of 0.9402, the intact class achieved an AUC of 0.9955, and the spalling class obtained an AUC of 0.9777. These values achieved are very high and show the capability of the model to classify positive and negative instances within each class accurately. Furthermore, the AUC for each damage class is very close to 1, indicating an almost perfect model performance.

The misclassification rate, precision and recall were reviewed during testing to further compare model performance. The performance achieved for these metrics is presented in Table 4.7.

Table 4.7 Defect classification misclassification rate, precision and recall of the SVM models during testing.

Model Number	**Model Type**	**Type**	**Misclassification rate %**	**Precision %**	**Recall %**
5	SVM	Linear SVM	25.54	75.07	74.46
6	SVM	Quadratic SVM	13.42	86.45	86.58
7	SVM	Cubic SVM	11.69	88.19	88.31
8	SVM	Fine Gaussian SVM	27.71	83.77	72.29
9	SVM	Medium Gaussian SVM	24.24	77.28	75.76
10	SVM	Coarse Gaussian SVM	32.03	68.99	67.97
11	SVM	Optimisable SVM	14.72	85.02	85.28

Based on Table 4.7, the cubic SVM model again produced the best results when compared to the other SVM models. The cubic SVM achieved the lowest misclassification rate and the highest precision and recall scores.

The misclassification rate was obtained using the proportion of misclassified instances (27) relative to the total number of instances or predictions (231). A misclassification rate of 11.69% was therefore obtained and this shows that 11.69% of the predictions were classified incorrectly. Furthermore, the low misclassification rate obtained indicates the model's better overall performance.

The model precision was determined to be 88.19% while the recall was determined to be 88.31%. Figure 4.18 displays the precision scores for each class, with the crack class achieving a precision of 83.8%, the intact class achieving 91.5%, and the spall class achieving 89.3%. The overall precision for the model was determined 88.19% by calculating the mean of these precision values. This means that when the model predicted an instance as positive, it was accurate 88.19% of the time. This indicates that the model could correctly identify positive instances for each damage class, with a relatively low rate of false positives.

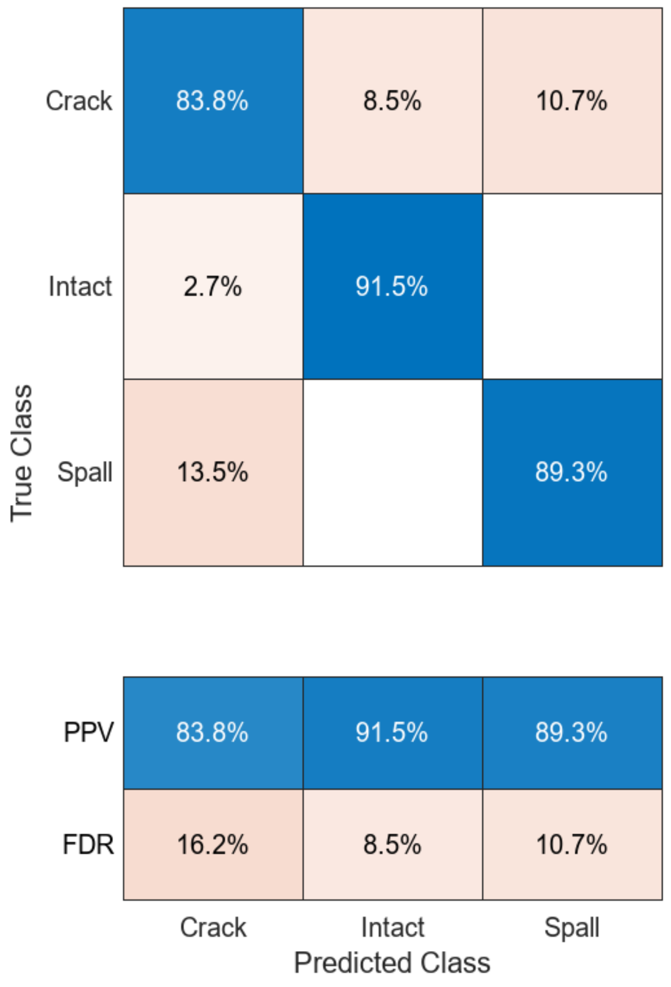


Figure 4.18 The precision or positive predictive value (PPV) and false discovery rate (FDR) achieved for each damage class, using the cubic SVM model.

Figure 4.18 also provides the false discovery rate for each class. The crack, intact, and spall classes obtained false discovery rates of 16.2%, 8.5%, and 10.7%, respectively. A higher false discovery rate was observed for the crack class, while lower rates were observed for the spall and intact classes.

The recall values and false negative rate achieved for each damage class are presented in Figure 4.19.

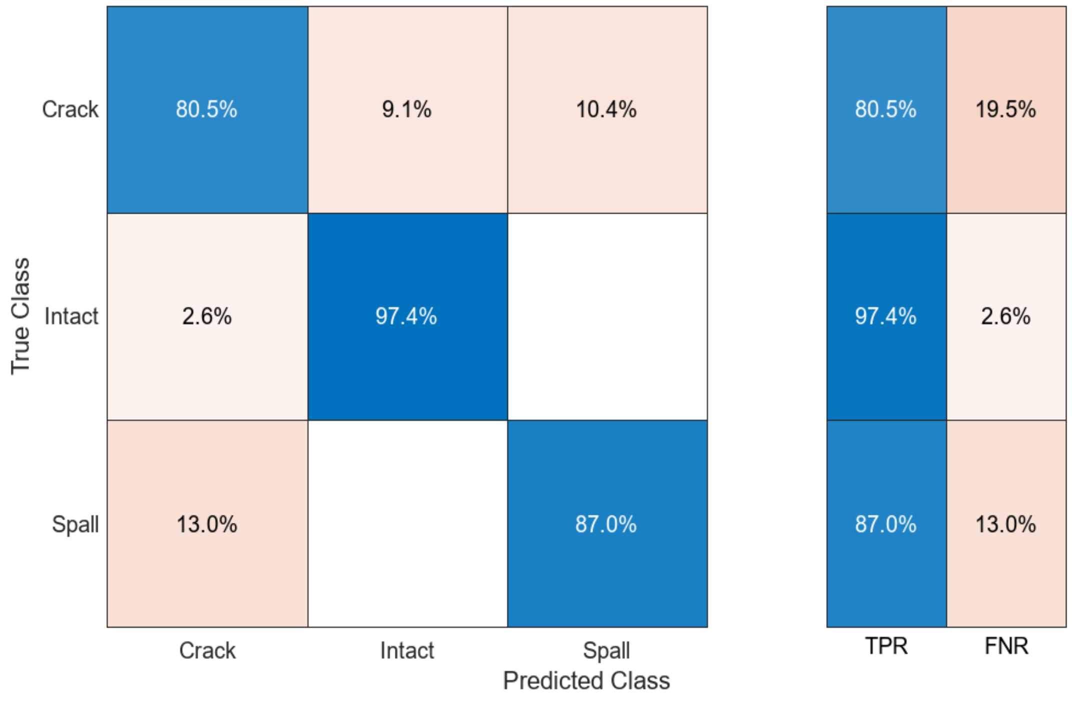


Figure 4.19 The recall or true positive rate (TPR) and false negative rate (FNR) achieved for each damage class, using the cubic SVM model.

Recall rates of 80.5%, 97.4%, and 87.0% were obtained for the classification of crack, intact and spall damage classes. The average or overall recall for the model was calculated to be 88.31%.

By analysing the precision and recall values for each damage class individually, it was observed that a precision of 83.8% and a recall of 80.5% was obtained for the prediction of the crack class. This shows that the model had a relatively high accuracy in correctly identifying positive instances for the crack class, where 83.8% of the crack predictions were part of the crack class. The model also had a high recall of 80.5% for the prediction of the crack class, indicating that the model captured 83.8% of all the crack instances.

The prediction of the spall class obtained a precision of 89.3% and a recall of 87.0%, indicating that the model could accurately classify positive instances for the spall class at a very high rate (89.3%), where 89.3% of the spall predictions belonged to the spall class. The model also had a high recall for this class (87.0%), indicating that it captured 87.0% of all spall instances.

The model’s prediction of the intact class proved even better, achieving an extremely high precision of 91.5% and a recall of 97.4%. This indicates that the model performed very well in correctly identifying positive instances for the intact class, where 91.5% of the class predictions were accurately predicted. Furthermore, the recall achieved for this class showed that the model captured 97.4% of all the intact instances.

Overall, the results demonstrate that the cubic SVM model exhibited competitive performance regarding the misclassification rate, precision, and recall metrics. The model showed a low rate of misclassification, a good ability to identify positive instances, and a balanced trade-off between false positives and false negatives.

Classification

Figure 4.20 shows an image obtained at Robben Island and the defect classification results achieved by the cubic SVM model during testing. The image displays the true label of the image, the model's predicted label, along with the associated confidence score. For this image, the image belonged to the spall defect class, with the model accurately classifying it as spall damage with a confidence score of 0.50 (50%).

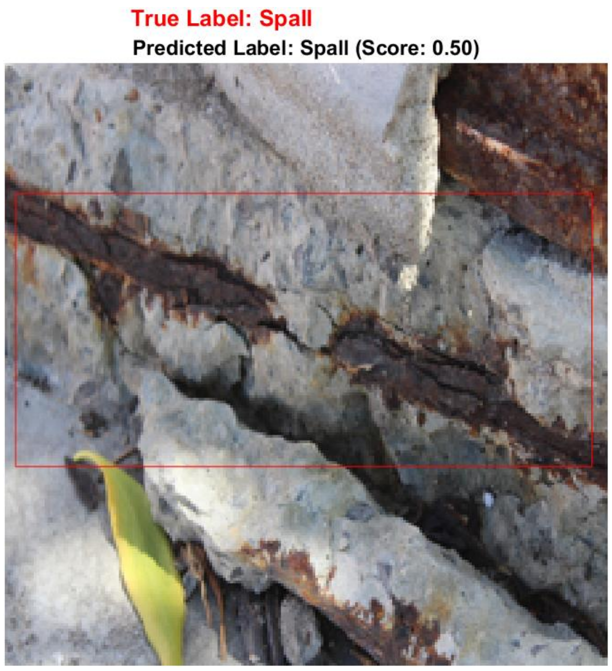


Figure 4.20 Image classification carried out by the cubic SVM model during testing on a spall image obtained from Robben Island.

4.3.4 K-nearest neighbour model results

Training

For the k-nearest neighbour models, training was carried out using seven different k-nearest neighbour models. These were the fine, medium, coarse, cosine, cubic, weighted and optimisable K-nearest neighbour models. Table 4.8 presents the accuracy achieved by each k-nearest neighbour model during training.

Table 4.8 Defect classification accuracy achieved by the k-nearest neighbour models during training.

Model Number	Model Type	Type	Accuracy % (Training)
12	KNN	Fine KNN	76.34
13	KNN	Medium KNN	67.63
14	KNN	Coarse KNN	56.02
15	KNN	Cosine KNN	66.13
16	KNN	Cubic KNN	67.63
17	KNN	Weighted KNN	77.10
18	KNN	Optimisable KNN	77.53

Table 4.8 shows that the optimisable k-nearest neighbour model performed best during training achieving an accuracy of 77.53%. Other models which performed well include the weighted and fine k-nearest neighbour models with accuracies of 77.10% and 76.34% respectively. The other models, such as the cubic and medium k-nearest neighbour models, both obtained accuracies of 67.63%, while the cosine and coarse k-nearest neighbour models achieved accuracies of 66.13% and 56.02%.

Figure 4.21 presents the optimisable k-nearest neighbour model confusion matrix that shows the classification outcomes during training. These were used to determine the training accuracy.

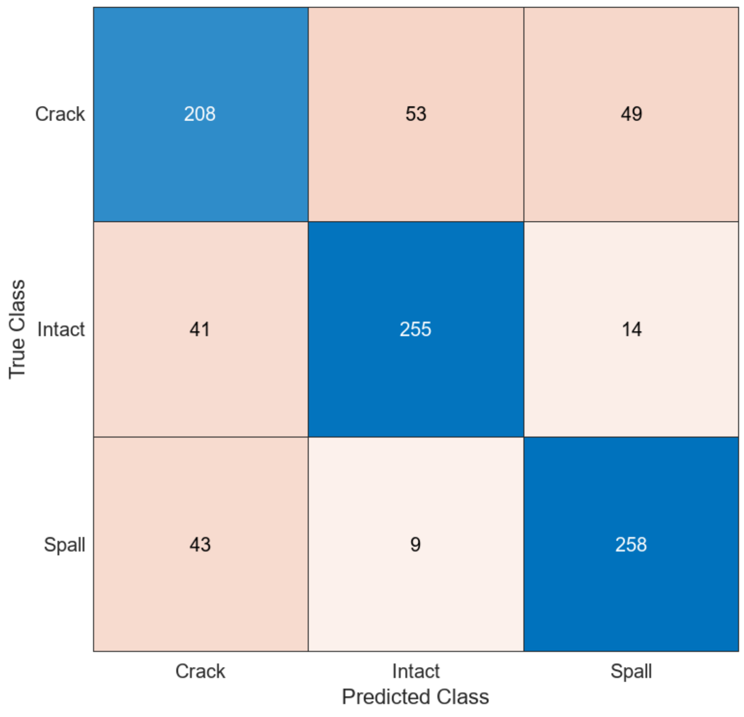


Figure 4.21 Optimisable k-nearest neighbour model confusion matrix during training.

By analysing the optimisable k-nearest neighbour model confusion matrix during training, the following was observed:

- For the crack class, out of a total of 310 instances, 208 were correctly classified as cracks (true positives), 53 instances were incorrectly classified as intact, and 49 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of a total of 310 instances, 255 were correctly classified as intact (true positives), 41 instances were incorrectly classified as cracks, and 14 instances were misclassified as spalling (false negatives).
- Out of the 310 predictions for the spall class, 258 were correctly classified as spalling (true positives), 43 instances were incorrectly classified as cracks, and 9 instances were misclassified as intact (false negatives).

The number of accurate predictions extracted from the confusion matrix (721) was divided by the total number of predictions (930), and this provided a final training accuracy of 77.53% for the optimisable k-nearest neighbour model.

ROC curves were generated for the optimisable k-nearest neighbour model at training, and they are shown in Figure 4.22.

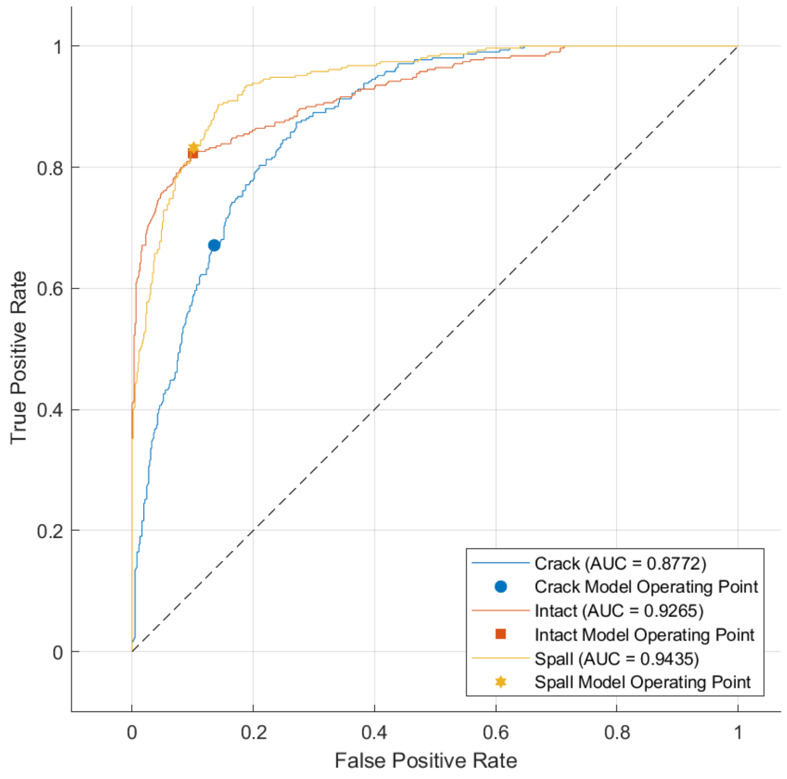


Figure 4.22 ROC curves for the Optimisable k-nearest neighbour model during training.

Figure 4.22 illustrates the relationship between the true and false positive rates across the various damage classes during training for the optimisable k-nearest neighbour model. The ROC curves for the classification of the intact and spall class rise steeply, almost vertically and consistently upward towards the top left corner of the axis to a point where it then tapers away from the y-axis. The classification of the crack class was less effective as the curve deviates slightly, falling away from the y-axis as it rises and fails to reach the same height as the spall and intact curves.

The performance of the model is further highlighted in the AUC values obtained. For the classification of cracks, an AUC value of 0.8772 was achieved, for the intact class, an AUC value of 0.9265 was obtained, while an AUC value of 0.9435 was obtained for the classification of spall damage. Using the ROC curve and AUC rating system provided by Mandrekar (2010: 1316), the optimisable k-nearest neighbour model produced an outstanding performance for the classification of spalling and intact areas, while it showed an excellent performance for the crack class.

Testing

Testing was completed using the trained fine, medium, coarse, cosine, cubic, weighted and optimisable k-nearest neighbour models. Table 4.9 presents the accuracy achieved by the different k-nearest neighbour models during testing.

Table 4.9 Defect classification accuracy achieved by the k-nearest neighbour models during testing.

Model Number	Model Type	Type	Accuracy % (Testing)
12	KNN	Fine KNN	82.25
13	KNN	Medium KNN	67.97
14	KNN	Coarse KNN	60.17
15	KNN	Cosine KNN	66.23
16	KNN	Cubic KNN	68.40
17	KNN	Weighted KNN	78.35
18	KNN	Optimisable KNN	76.19

Table 4.9 shows that the fine k-nearest neighbour model performed best, achieving an accuracy of 82.25% during testing. This was followed by the weighted k-nearest neighbour model with the second highest accuracy of 78.35% and the optimisable k-nearest neighbour model with an accuracy of 76.19%. The cubic, medium, cosine and coarse k-nearest neighbour models performed less accurately during testing, achieving final accuracies of 68.40%, 67.97%, 66.23% and 60.17% respectively.

The results show the accuracy of the fine k-nearest neighbour model during testing, with the model showing an increase in accuracy from 76.34% in training to 82.25% during testing. Furthermore, the results also show a drop in the accuracy of the optimisable k-nearest neighbour model from 77.53% in training to 76.19% in testing. All other k-nearest neighbour models showed an improvement in accuracy from training to testing.

Figure 4.23 shows the fine k-nearest neighbour model confusion matrix and illustrates the classification outcomes for the damage classes during testing.

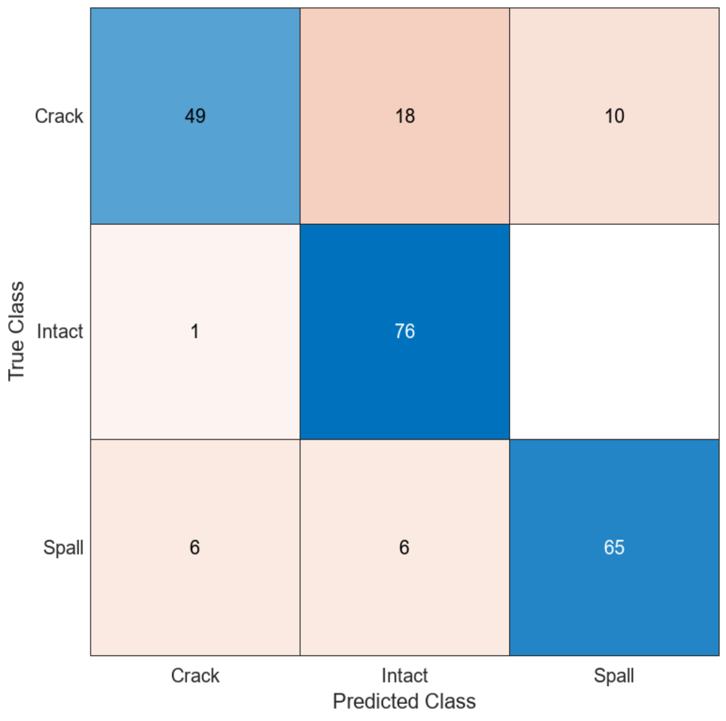


Figure 4.23 Fine k-nearest neighbour model confusion matrix during testing.

By analysing the fine k-nearest neighbour model confusion matrix during testing, the following was observed:

- For the crack class, out of a total of 77 instances, 49 were correctly classified as cracks (true positives), 18 instances were incorrectly classified as intact, and 10 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of the 77 predictions made, 76 were correctly classified as intact (true positives), 1 instance was incorrectly classified as cracks, and 0 instances were misclassified as spalling (false negatives).
- Out of the 77 predictions for the spall class, 65 were correctly classified as spalling (true positives), 6 instances were incorrectly classified as cracks, and 6 instances were misclassified as intact (false negatives).

By extracting the true positive and negative values, the number of accurate predictions (190 accurate predictions) was divided by the total number of predictions (231 predictions), providing a final test accuracy of 82.25%.

To further evaluate the fine k-nearest neighbour model, ROC curves were generated and are shown in Figure 4.24.

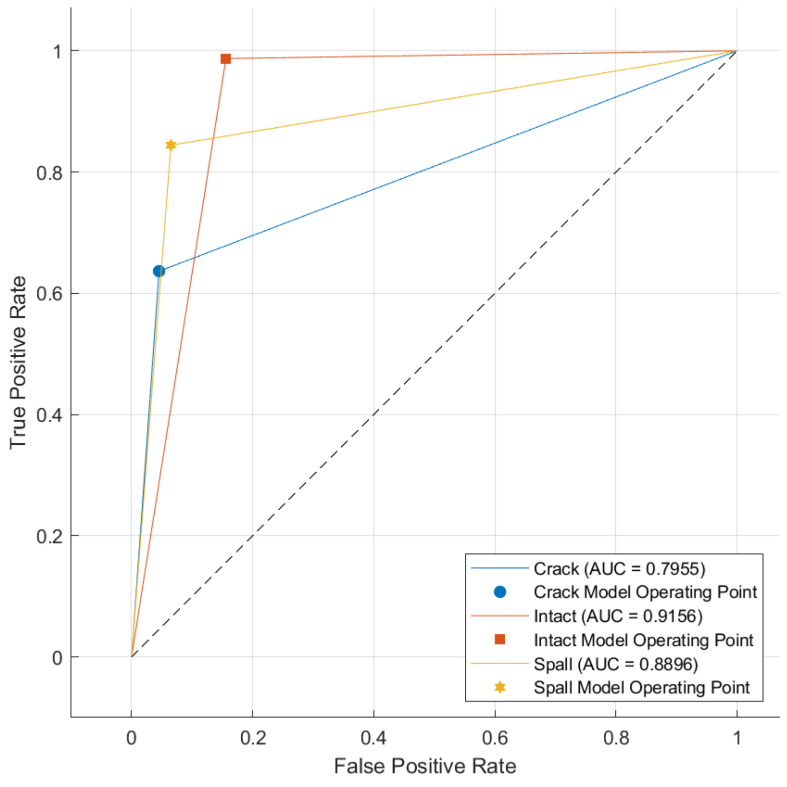


Figure 4.24 ROC curves for the fine k-nearest neighbour model's performance during testing.

Figure 4.24 shows the relationship between the true and false positive rates across the damage classes during testing for the fine k-nearest neighbour model. Furthermore, the ROC curves for the performance in classifying the different damage classes varied. Initially, all the curves rise steeply, in close proximity to the y-axis. However, the performance then changes across the different damage classes. The performance in classifying the intact class rises to the highest point followed by the prediction of the spall class and then the crack class. The prediction of the intact class therefore provides the best performance rising to the highest point, indicating superior classification performance, while maintaining a proximity to the upper corner of the y-axis. There is a slight decline in performance for the prediction of the spall and crack class with the prediction of the crack class being the weakest among the three. Despite this, the model still produces a fair overall performance.

For the classification of cracks an AUC value of 0.7955 was obtained; for the intact class an AUC value of 0.9156 was obtained; while an AUC value of 0.8896 was obtained for the classification of spall damage. Using the ROC curve and AUC rating system provided by Mandrekar (2010: 1316), the fine k-nearest neighbour model produced a fair performance for the classification cracks, an excellent performance for spalling classification and an outstanding performance for the classification of intact areas.

The misclassification rate, precision and recall were reviewed during testing to further compare model performance. The performance achieved for these metrics is presented in Table 4.10.

Table 4.10 Defect classification misclassification rate, precision, and recall of the k-nearest neighbour models during testing.

Model Number	Model Type	Type	Misclassification rate %	Precision %	Recall %
12	KNN	Fine KNN	17.75	83.39	82.25
13	KNN	Medium KNN	32.03	68.83	67.97
14	KNN	Coarse KNN	39.83	61.34	60.17
15	KNN	Cosine KNN	33.77	66.93	66.23
16	KNN	Cubic KNN	31.60	69.25	68.40
17	KNN	Weighted KNN	21.65	78.10	78.35
18	KNN	Optimisable KNN	23.81	76.31	76.19

Based on Table 4.10, the fine k-nearest neighbour model produced the best results compared to the other k-nearest neighbour models. The fine k-nearest neighbour model achieved the lowest misclassification rate while achieving the highest precision and recall scores.

The misclassification was determined using the proportion of misclassified instances (41) relative to the total number of predictions (231). A misclassification rate of 17.75% was obtained, indicating that approximately 17.75% of the predictions were misclassified.

The precision was calculated to be 83.39%. The recall was determined to be 82.25% using the confusion matrix in Figure 4.23.

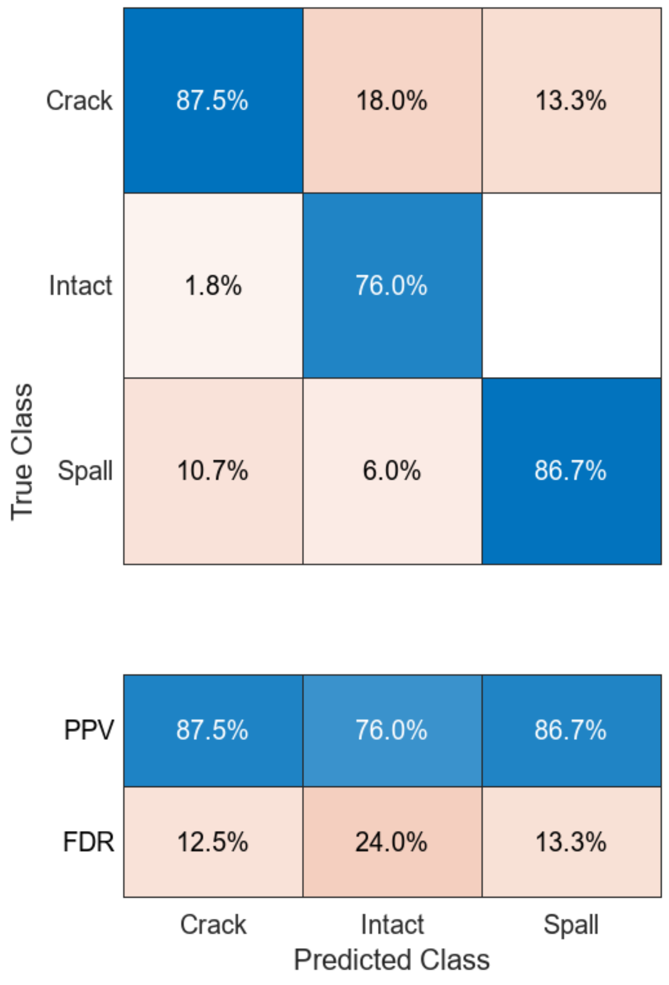


Figure 4.25 The precision or positive predictive value (PPV) and false discovery rate (FDR) achieved for each damage class, using the fine k-nearest neighbour model.

Figure 4.25 shows the precision values for each class, with the prediction of the crack class achieving a precision of 87.5%, the prediction of the intact class achieving 76.0%, and the spall class prediction achieving 86.7%. The mean precision for the model was determined to be 83.39%. This indicated that for every positive prediction by the model, it was accurate 83.39% of the time. This indicates that the model could correctly identify positive instances for each damage class, with a low rate of false positives.

Figure 4.25 also provides the false discovery rate for each class. The crack, intact, and spall classes obtained false discovery rates of 12.5%, 24.0%, and 13.3% respectively. A higher false discovery rate was observed for the intact class, while much lower rates were observed for the spall and crack classes.

The recall values and false negative rate achieved for each damage class are presented in Figure 4.26.

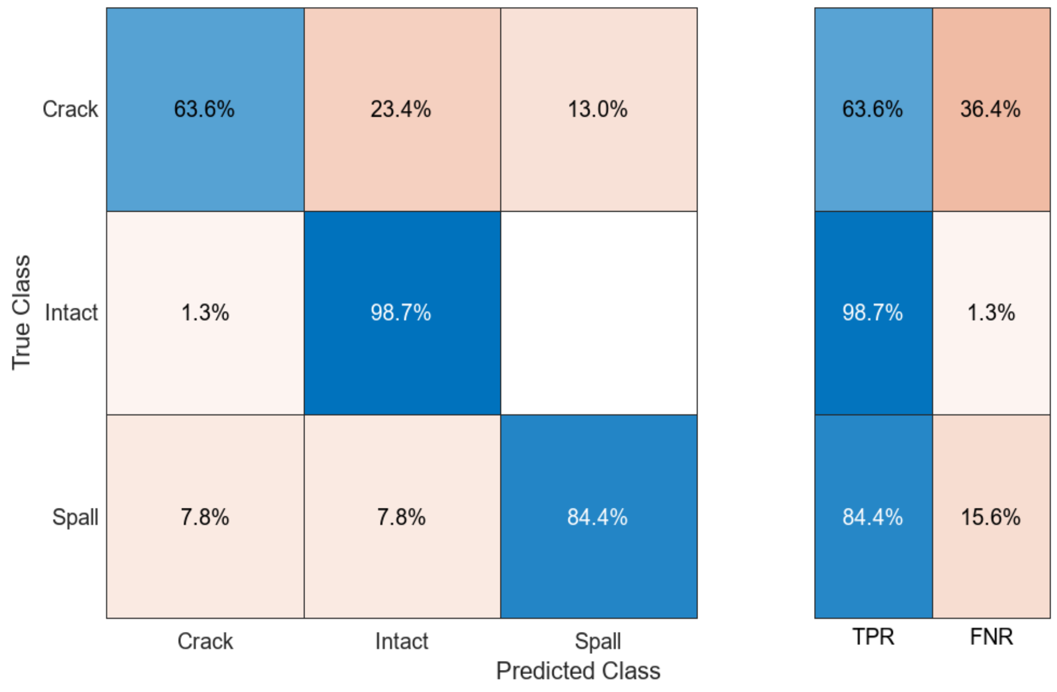


Figure 4.26 The recall or true positive rate (TPR) and false negative rate (FNR) achieved for each damage class, using the fine k-nearest neighbour model.

Recall rates of 63.6%, 98.7%, and 84.4% were obtained for the crack, intact and spall damage classes. A mean recall value of 82.25% was obtained from the recall rates of each class.

By analysing the precision and recall values for each damage class individually, it was observed that for the prediction of the crack class, a precision of 87.5% and a recall of 63.6% was obtained. This shows that the model had a high accuracy in correctly identifying positive instances for the crack class (87.5%), with a low rate of false positive predictions. Alternatively, the model had a fairly low recall of 63.6% for the prediction of cracks, indicating that the model captured 63.3% of all the actual crack instances.

For the prediction of the spall class, a precision of 86.7% and a recall of 84.4% was obtained. This shows that the model had a high accuracy in predicting true positives in the spall class (86.7%). Furthermore, the model also had a high recall rate (84.4%) for the prediction of the spall class, indicating that the model captured 84.4% of all the spall instances, which suggests good performance in identifying spalling.

Lastly, the intact class achieved a precision score of 76.0% and a recall of 98.7%. This indicates that the model performed very well in accurately identifying positive instances for the intact class, where 76.0% of the intact predictions were accurately predicted as part of the intact class. The model also achieved a very high recall for intact predictions (98.7%), indicating that the model captured 98.7% of all intact instances. This suggests that the model is very effective in finding and capturing intact areas with a low number of false negatives.

Classification

Figure 4.27 shows an image obtained at Dal Josafat and the defect classification results achieved by the fine k-nearest neighbour model during testing. This figure shows the true label of the image, the model's predicted label, along with the confidence score. This image belonged to the spall defect class, with the model accurately classifying it as spall damage with a confidence score of 1.00 (100%).

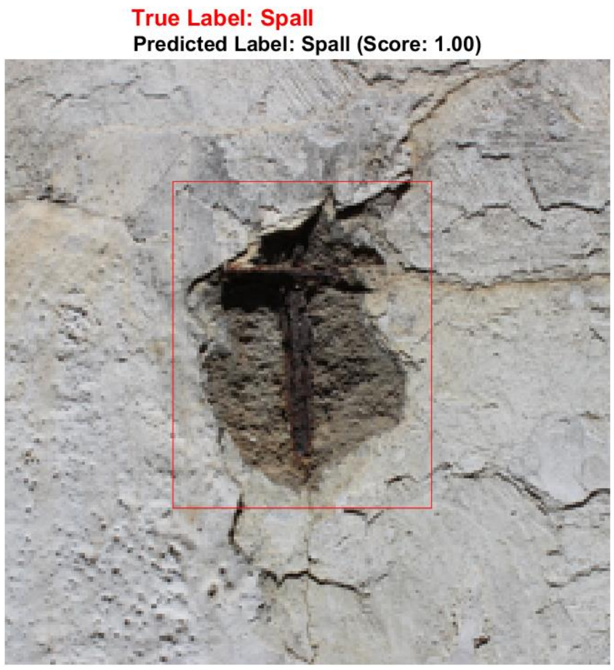


Figure 4.27 Image classification carried out by the fine k-nearest neighbour model during testing on a spall image obtained from Dal Josafat.

4.3.5 Artificial neural network model results

Training

For the artificial neural network models, training was carried out using six different neural network models. These were the narrow, medium, wide, bi-layered, tri-layered and optimisable artificial neural network models. Table 4.11 presents the accuracy achieved by each artificial neural network model during training.

Table 4.11 Defect classification accuracy achieved by the artificial neural network models during training.

Model Number	Model Type	Type	Accuracy % (Training)
19	Neural Network	Narrow Neural Network	82.15
20	Neural Network	Medium Neural Network	85.59
21	Neural Network	Wide Neural Network	85.70
22	Neural Network	Bi-layered Neural Network	80.65
23	Neural Network	Tri-layered Neural Network	81.08
24	Neural Network	Optimisable Neural Network	82.90

Table 4.11 shows that the wide artificial neural network model performed best during training, achieving an accuracy of 85.70%. All other artificial neural network models performed well, with their performance closely following that of the wide artificial neural network. This included the medium, optimisable,

narrow, tri-layered and bi-layered artificial neural network models with training accuracies of 85.59%, 82.90%, 82.15%, 81.08% and 80.65%.

Figure 4.28 shows the wide artificial neural network model confusion matrix and classification results for training.

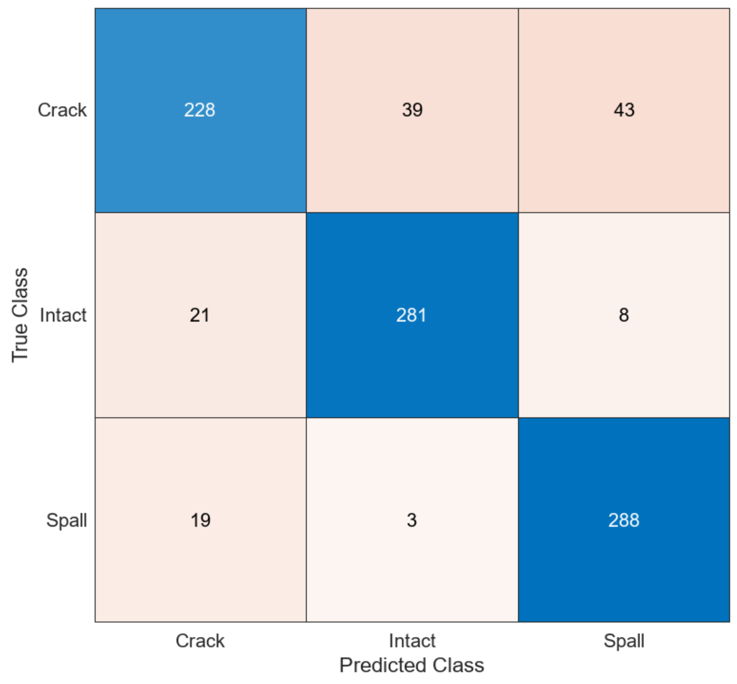


Figure 4.28 Wide artificial neural network model confusion matrix during training.

By analysing the wide artificial neural network model confusion matrix during training, the following was observed:

- For the crack class, out of a total of 310 instances, 228 were correctly classified as cracks (true positives), 39 instances were incorrectly classified as intact, and 43 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of a total of 310 instances, 281 were correctly classified as intact (true positives), 21 instances were incorrectly classified as cracks, and 8 instances were misclassified as spalling (false negatives).
- Out of the 310 predictions for the spall class, 288 were correctly classified as spalling (true positives), 19 instances were incorrectly classified as cracks, and 3 instances were misclassified as intact (false negatives).

The number of accurate predictions were extracted from the confusion matrix (797) and was divided by the total number of predictions (930), which provided a training accuracy of 85.70%.

ROC curves were generated for the wide artificial neural network model under training, which are shown in Figure 4.29.

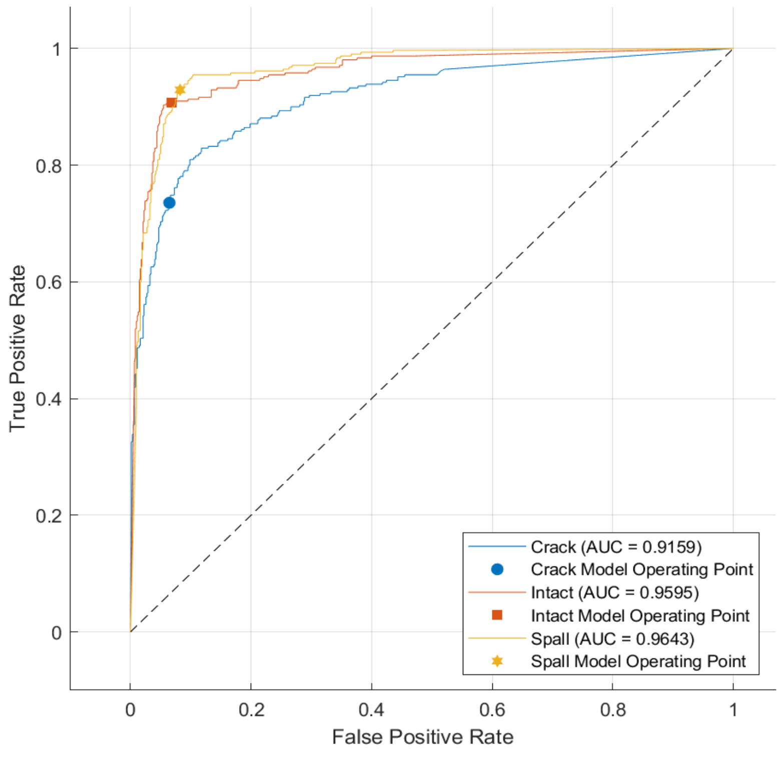


Figure 4.29 ROC curves for the wide artificial neural network model during training.

Figure 4.29 shows the performance of the model in classifying the different classes. The ROC curves for the classification of the intact and spall class rise steeply upward towards the top left corner of the axis, showing the high true positive rates and low false positive rates. The classification of the crack class was slightly less effective as the curve tapers off, falling away from the y-axis. Furthermore, the prediction of the crack damage class does not reach the same height/point as the other two classes, showing slightly more false positive predictions.

The performance at the training stage was further evaluated using the AUC values obtained. For the classification of cracks an AUC value of 0.9159 was achieved, for the intact class an AUC value of 0.9595 was obtained, while an AUC value of 0.9643 was obtained for the classification of spall damage. This shows that the wide artificial neural network produced an outstanding performance in classifying all damage classes.

Testing

Testing was completed using the trained narrow, medium, wide, bi-layered, tri-layered and optimisable artificial neural network models. The accuracy achieved by the different artificial neural network models during testing is presented in Table 4.12.

Table 4.12 Defect classification accuracy achieved by the artificial neural network models during testing.

Model Number	Model Type	Type	Accuracy % (Testing)
19	Neural Network	Narrow Neural Network	84.42
20	Neural Network	Medium Neural Network	87.45
21	Neural Network	Wide Neural Network	85.28
22	Neural Network	Bi-layered Neural Network	82.25
23	Neural Network	Tri-layered Neural Network	83.12
24	Neural Network	Optimisable Neural Network	80.95

According to Table 4.12, the medium artificial neural network model performed better, achieving an accuracy of 87.45% during testing. Some of the other neural network models showed improvement in accuracy from training to testing. These models included the narrow artificial neural network increasing from 82.15% to 84.42%; the medium artificial neural network increasing from 85.59% to 87.45%; the bi-layered artificial neural network which improved from 80.65% to 82.25%; and the tri-layered artificial neural network which improved from 81.08% to 83.12%. Other models showed a decrease in accuracy, such as the wide artificial neural network decreasing from 85.70% to 85.28% and the optimisable artificial neural network which decreased from 82.90% to 80.95%.

These results show that the medium artificial neural network model performed best during testing, showing an increase in accuracy from 85.59% in training to 87.45% during testing.

Figure 4.30 shows the medium artificial neural network model confusion matrix and illustrates the classification outcomes for the different damage classes during testing.

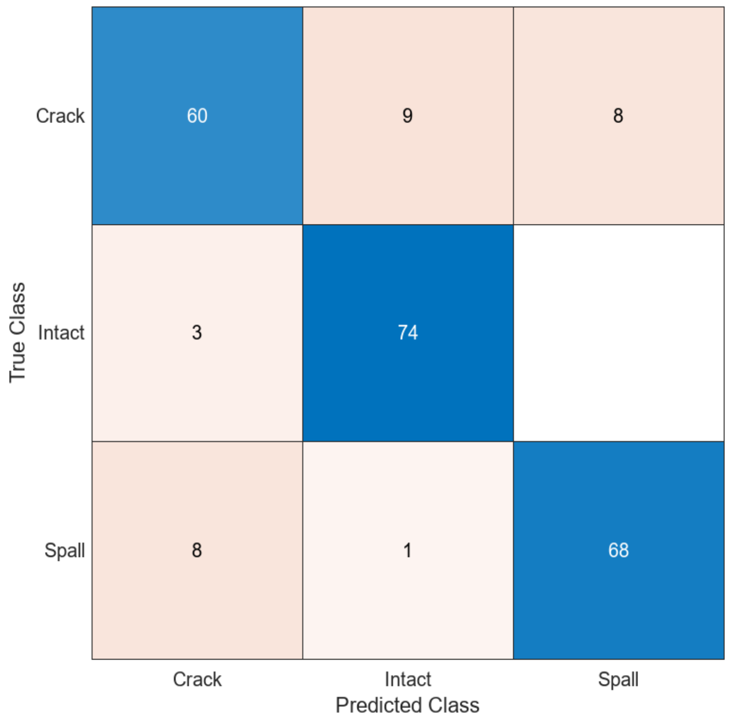


Figure 4.30 Medium artificial neural network model confusion matrix during testing.

By analysing the medium artificial neural network model confusion matrix during testing, the following was observed:

- For the crack class, out of a total of 77 instances, 60 were correctly classified as cracks (true positives), 9 instances were incorrectly classified as intact, and 8 instances were incorrectly classified as spalling (false negatives).
- In the intact class, out of the 77 predictions made, 74 were correctly classified as intact (true positives), 3 instances were incorrectly classified as cracks, and 0 instances were misclassified as spalling (false negatives).
- Out of the 77 predictions for the spall class, 68 were correctly classified as spalling (true positives), 8 instances were incorrectly classified as cracks, and 1 instance was misclassified as intact (false negatives).

By extracting the true positive and negative values, the number of accurate predictions (202) was divided by the total number of predictions (231), providing a final test accuracy of 87.45%.

To further evaluate the medium artificial neural network model, ROC curves were generated and are shown in Figure 4.31.

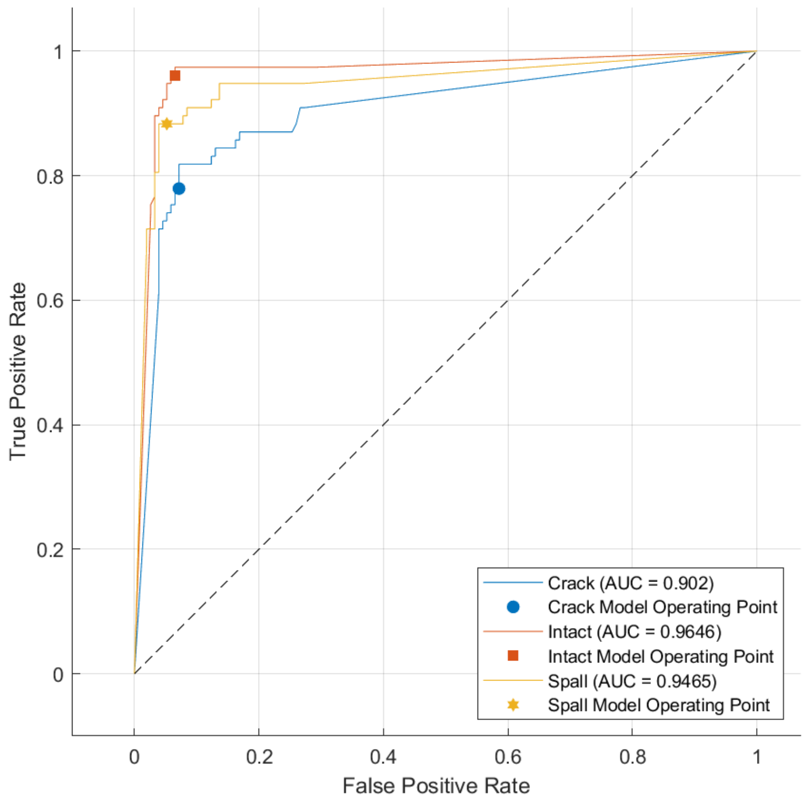


Figure 4.31 ROC curves for the medium artificial neural network model's performance during testing.

Figure 4.31 shows the ROC curves generated for each damage class when classifying damage with the medium artificial neural network model. The medium artificial neural network model ROC curves show consistent upward trends (almost vertical), indicating a very high true positive rate and a very low false positive rate. The curves also rise steeply and vertically upwards, approaching the upper left corner of the axis, highlighting how well the model performed in classifying damage.

Figure 4.31 shows the AUC values for the prediction of the different damage classes. The crack class achieved an AUC of 0.902, the intact class achieved an AUC of 0.9646, and the spalling class obtained an AUC of 0.9465. The high AUC values achieved for the prediction of the different damage classes are indicative of the model's performance and its ability to distinguish different damage patterns. Furthermore, the AUC values for each damage class are close to 1, indicating an outstanding and almost perfect model performance.

The misclassification rate, precision and recall were reviewed during testing to further compare model performance. The performance achieved for these metrics are presented in Table 4.13.

Table 4.13 Defect classification misclassification rate, precision and recall of the artificial neural network models during testing.

Model Number	Model Type	Type	Misclassification rate %	Precision %	Recall %
19	Neural Network	Narrow Neural Network	15.58	84.28	84.42
20	Neural Network	Medium Neural Network	12.55	87.36	87.45
21	Neural Network	Wide Neural Network	14.72	85.21	85.28
22	Neural Network	Bi-layered Neural Network	17.75	82.00	82.25
23	Neural Network	Tri-layered Neural Network	16.88	83.28	83.12
24	Neural Network	Optimisable Neural Network	19.05	80.70	80.95

Based on Table 4.13, the medium artificial neural network model produced the best results compared to the other artificial neural network models. It achieved the lowest misclassification rate while achieving the highest precision and recall scores.

The misclassification rate was calculated using the proportion of misclassified instances (29) relative to the total number of instances or predictions (231). A misclassification rate of 12.55% was obtained, indicating that approximately 12.55% of the predictions were misclassified.

The precision was calculated to be 87.36%, and the recall was calculated to be 87.45% using the confusion matrix in Figure 4.30.

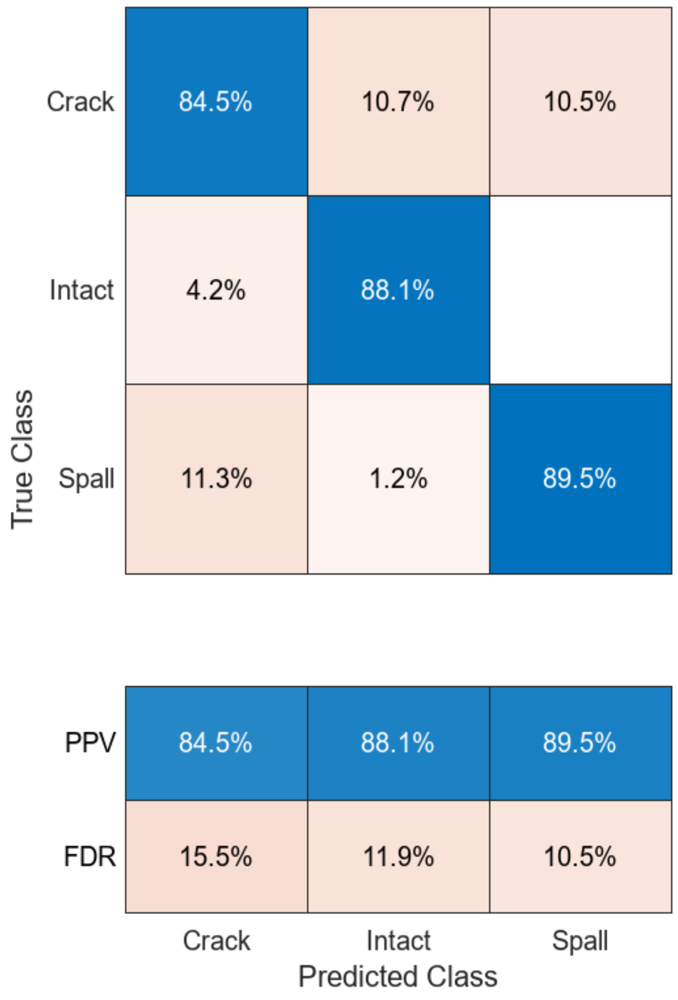


Figure 4.32 The precision or positive predictive value (PPV) and false discovery rate (FDR) achieved for each damage class, using the medium artificial neural network model.

Figure 4.32 shows the precision values for each class, with the prediction of the crack class achieving a precision of 84.5%; the prediction of the intact class achieving 88.1%; and the spall class prediction achieving 89.5%. The mean precision for the model was calculated to be 87.35%. This indicated that for every positive prediction by the model, it was accurate 87.35% of the time.

Figure 4.32 also provides the false discovery rate for each class, indicating the proportion of positive predictions that are false positives. The crack, intact, and spall classes obtained false discovery rates of 15.5%, 11.9%, and 10.5%, respectively. A higher false discovery rate was observed for the crack class, while rates were observed for the spall and intact classes.

The recall values and false negative rate achieved for each damage class are presented in Figure 4.33.

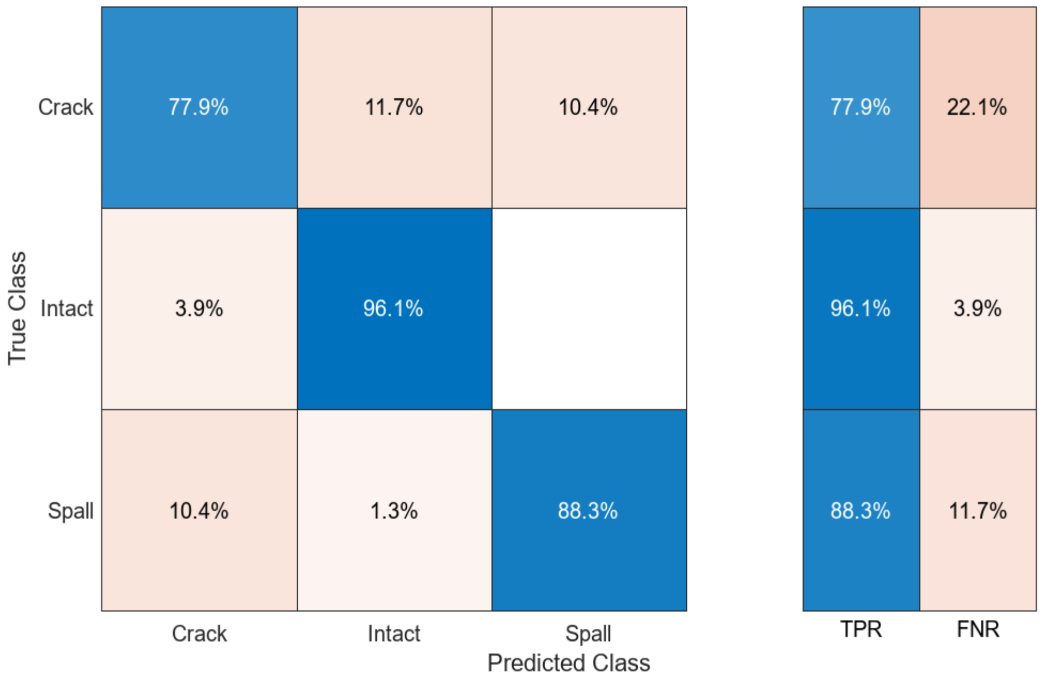


Figure 4.33 The recall or true positive rate (TPR) and false negative rate (FNR) achieved for each damage class, using the medium artificial neural network model.

Recall rates of 77.9%, 96.1%, and 88.3% were obtained for the crack, intact and spall damage classes respectively. The mean recall for the model using these values was calculated to be 87.44%.

By analysing the precision and recall values for each damage class individually, it was observed that for the prediction of the crack class, a precision of 84.5% and a recall of 77.9% was achieved. This suggests that the model had a high accuracy in correctly identifying positive instances for the crack class (84.5%), with a low rate of false positive predictions. Alternatively, the model had a fair recall score of 77.9% for the prediction of cracks, indicating that the model captured 77.9% of all the actual crack instances.

For the prediction of the spall class, a precision of 89.5% and a recall of 88.3% was obtained. This indicates that the model showed high accuracy in predicting true positives in the spall class (89.5%). Furthermore, the model also had a high recall rate (88.3%) for the prediction of the spall class, indicating that the model captured 88.3% of all the spall instances, which suggests good performance in identifying the spall instances.

Lastly, the intact class achieved a precision score of 88.1% and a recall of 96.1%. This indicates that the model performed very well in accurately identifying positive instances for the intact class, where 88.1% of the intact predictions were accurately predicted as part of the intact class. The model also achieved a very high recall for intact predictions (96.1%), indicating that the model captured 96.1% of all intact instances. This demonstrates that the model is very effective in finding and capturing intact instances with a low number of false negatives.

Classification

Figure 4.34 shows an image taken at Dal Josafat and the defect classification results achieved by the medium artificial neural network model during testing. The image displays the true label of the image, the model's predicted label, along with the associated confidence score. For this image, the image belonged to the crack defect class, with the model accurately classifying it as crack damage with a confidence score of 1.00 (100%).

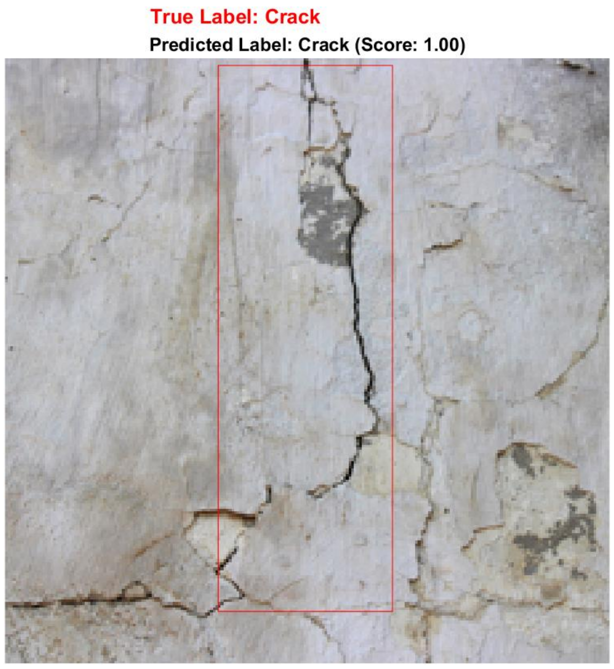


Figure 4.34 Image classification carried out by the medium artificial neural network model during testing on a crack image obtained from Dal Josafat.

4.3.6 Deep learning based classification

The deep learning CNN approach utilised the pre-trained AlexNet model to extract features directly from the input images for the classification of the different damage classes. The pre-trained AlexNet CNN takes a data-driven feature extraction and learning approach, whereby the model can automatically learn hierarchical features from input data and their local invariance (Joubert, 2020: 32). According to Joubert (2020: 32), these are advantages to the CNN model as it can extract features and edges regardless of their position in the image. This is because CNNs consist of multiple layers, including convolutional layers and pooling layers, that are designed for image processing and progressively learn complex features of the dataset.

Furthermore, the pre-trained AlexNet CNN has already been trained on a large dataset in the transfer learning approach. This provides a significant advantage as the model has already learned a rich set of features from a diverse range of images. This transfer learning approach allows the model to leverage the knowledge gained from the pre-training task and apply it to the new defect classification task (Joubert, 2020: 49).

The CNN deep learning approach is shown in Figure 4.35. Features are extracted from the input images and passed through various layers of the deep learning architecture. These layers include the convolutional layer, fully connected layers, and the softmax function, ultimately leading to the classification of defects.

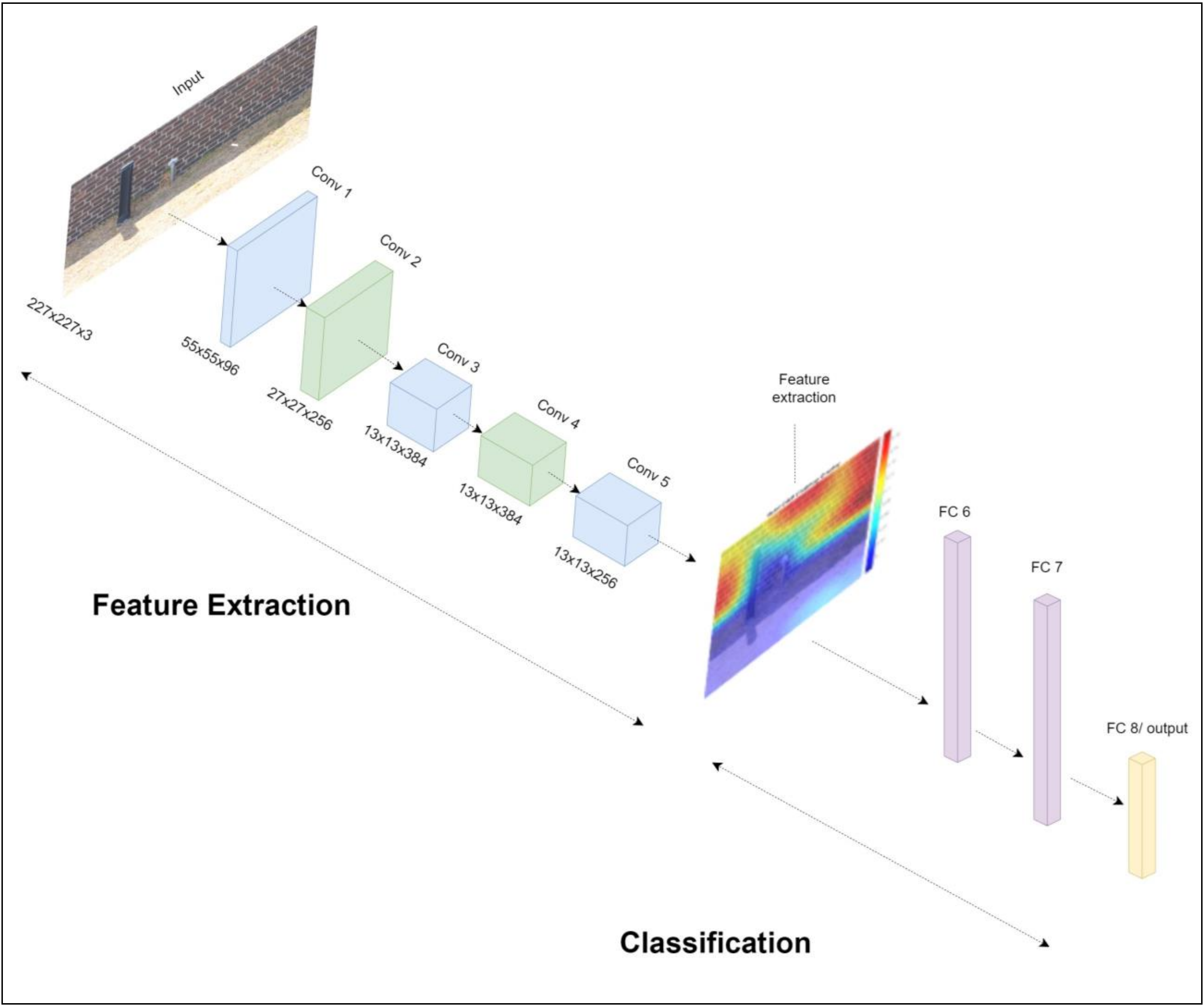


Figure 4.35 CNN feature extraction and classification process, shown through its different layers.

Figure 4.36 illustrates the feature extraction process performed by the convolutional layers. The input images are shown in Figure 4.36 (a), (b) and (c). Features are then extracted, and this is shown by the heatmap of the extracted features displayed and superimposed on the original images in Figure 4.36 (d), (e) and (f). Gradient-weighted class activation mapping (Grad-CAM) was used to understand which parts of the input image were essential in predicting a specific class. Much like the Bag of Words technique, these extracted features characterise the distinctive patterns found in the images and assist in the classification process.

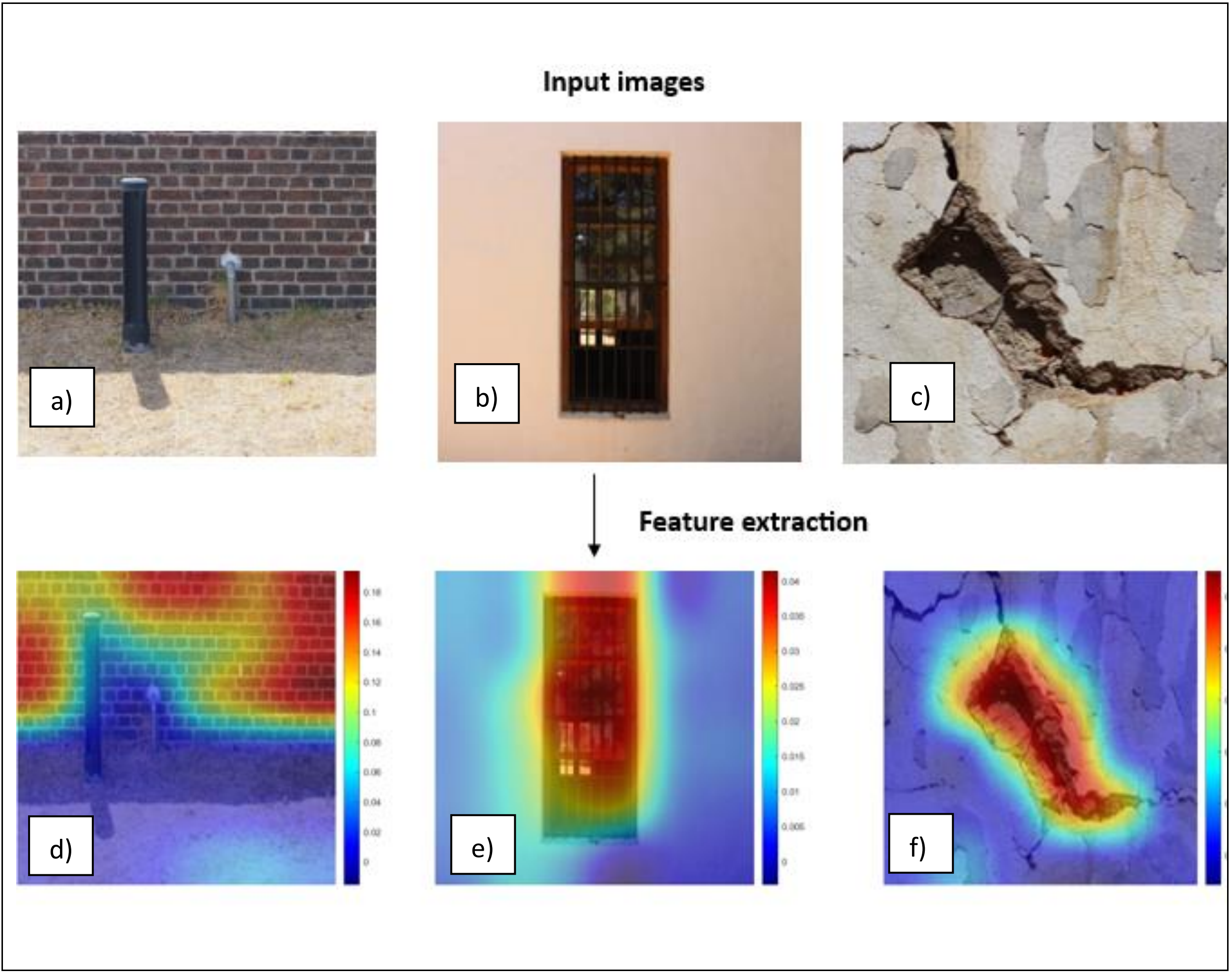


Figure 4.36 Input images and Grad-CAM heatmap showing regions of interest for feature extraction.

4.3.7 AlexNet convolutional neural network model results

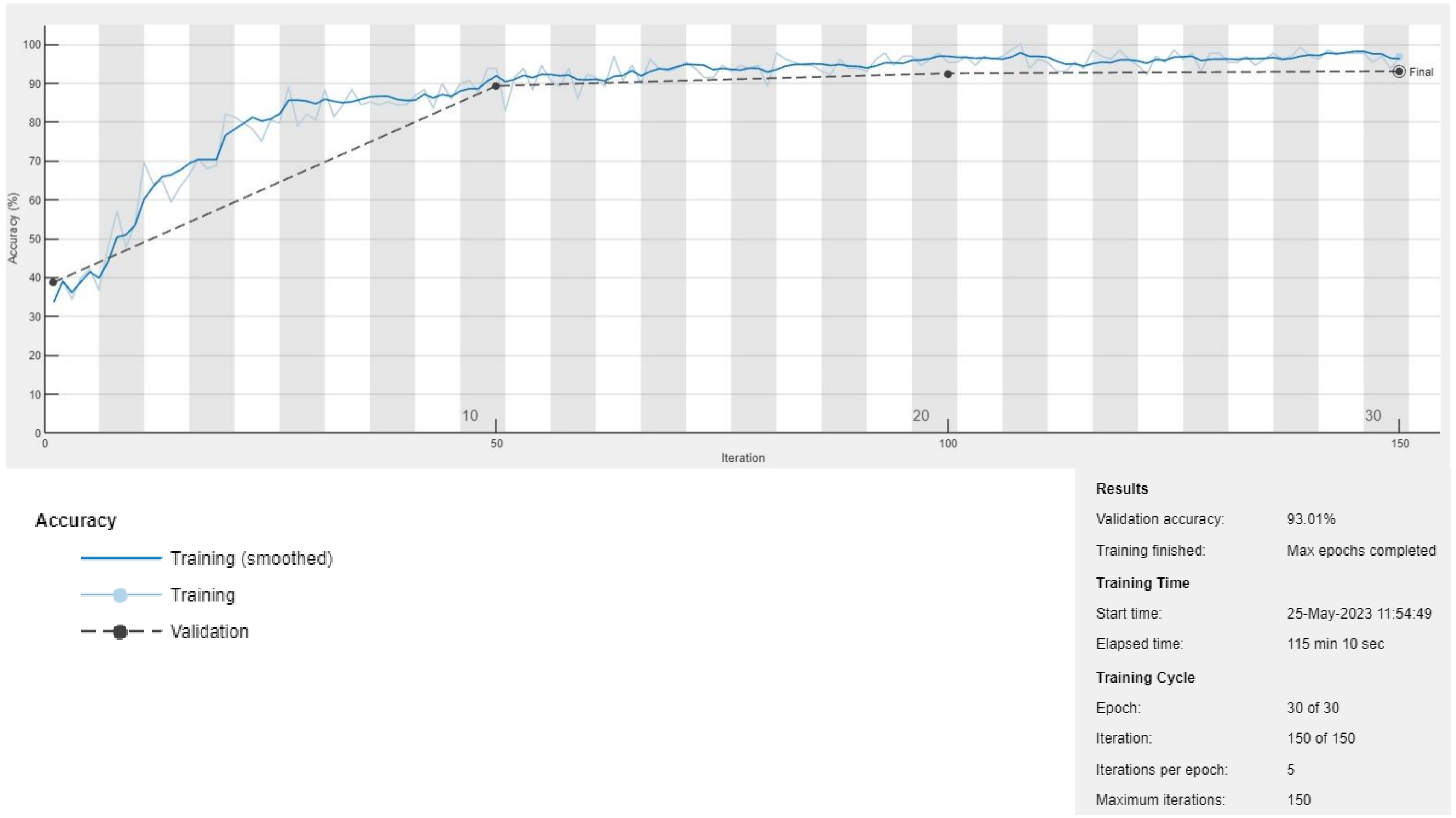
Training

Training was carried out on the pre-trained AlexNet CNN model with the results of the training shown in Table 4.14 below.

Table 4.14 Defect classification accuracy achieved by the pre-trained AlexNet model during training.

Model Number	Model Type	Type	Accuracy % (Training)
25	CNN	AlexNet CNN	93.01

Table 4.14 shows that the AlexNet CNN model achieved an accuracy of 93.01% during training. Furthermore, the training progress and accuracy are illustrated in Figure 4.37, showing how the model's accuracy improves as the model undergoes training.



Results

Validation accuracy:	93.01%
Training finished:	Max epochs completed

Training Time

Start time:	25-May-2023 11:54:49
Elapsed time:	115 min 10 sec

Training Cycle

Epoch:	30 of 30
Iteration:	150 of 150
Iterations per epoch:	5
Maximum iterations:	150

Figure 4.37 AlexNet CNN model progress and accuracy achieved during training.

Testing

Testing was completed using the trained AlexNet CNN model with the test accuracy displayed in Table 4.15.

Table 4.15 Defect classification accuracy achieved by the AlexNet CNN model during testing.

Model Number	Model Type	Type	Accuracy % (Testing)
25	CNN	AlexNet CNN	97.40

Table 4.15 shows that the AlexNet CNN model achieved an accuracy of 97.40% during testing. The table also highlights the performance and accuracy of the model during testing, with the model showing an increase in accuracy from 93.01% in training to 97.40% during testing.

Figure 4.38 shows the AlexNet CNN model confusion matrix and illustrates the classification outcomes for the different damage classes during testing.

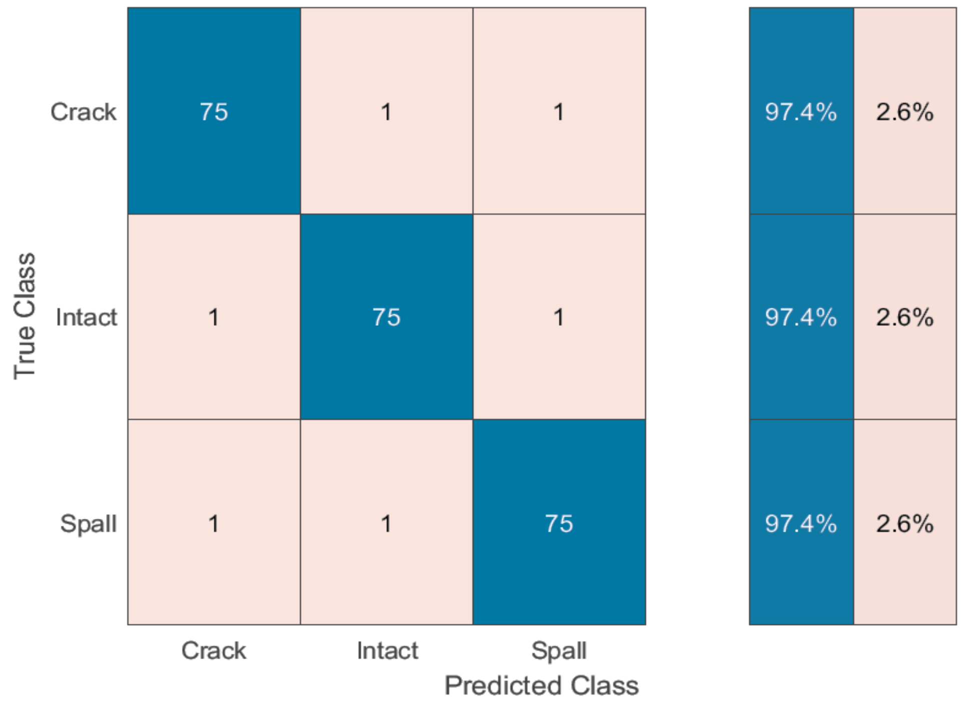


Figure 4.38 AlexNet CNN model confusion matrix during testing.

By analysing the AlexNet CNN model confusion matrix during testing, the following was observed:

- In the crack class, there were 75 instances correctly identified as cracks (true positives), 1 instance was incorrectly classified as intact, and another as spalling (false negatives).
- Similarly, the model correctly predicted 75 instances as intact (true positives) for the intact class. However, it misclassified 1 instance as a crack and 1 instance as spalling (false negatives).

- Regarding the spall class, the model accurately classified 75 instances as spalling (true positives), but it incorrectly labelled 1 instance as a crack and 1 instance as intact (false negatives).

The number of accurate predictions (225) was divided by the total number of predictions (231), using the values obtained from the confusion matrix, providing an accuracy of 97.40%.

Additionally, the ROC curves generated for the AlexNet CNN model are shown in Figure 4.39.

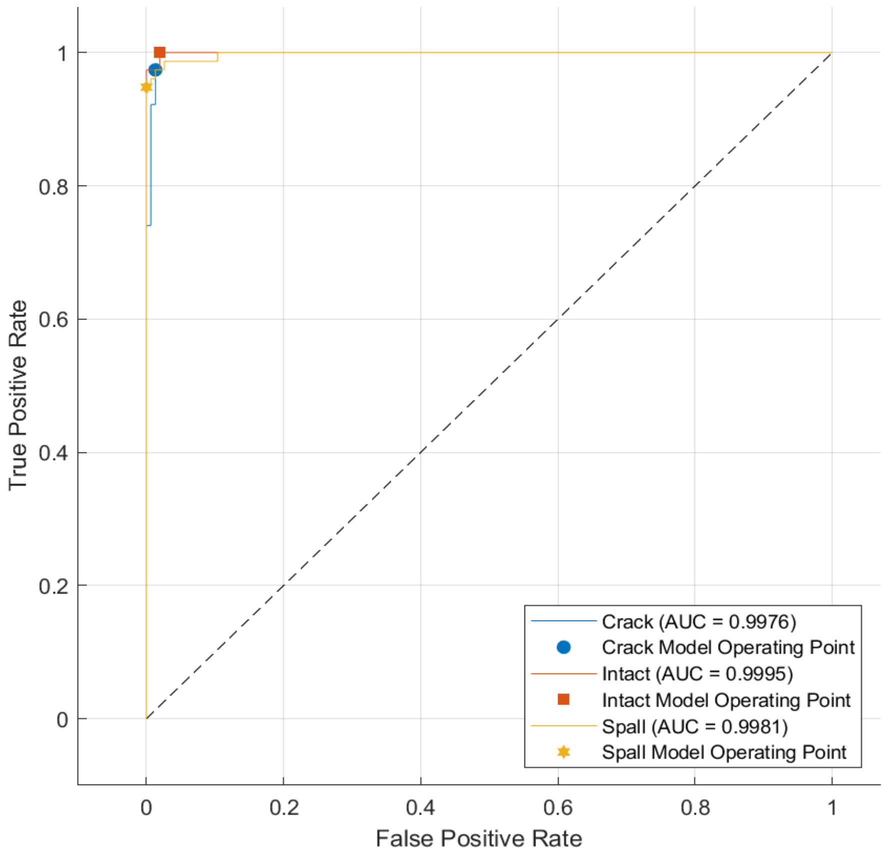


Figure 4.39 ROC curves for the AlexNet CNN model’s performance during testing.

The ROC curves depicted in Figure 4.39 all show a consistent upward trend, indicating a high true positive rate alongside a low false positive rate. The pre-trained AlexNet CNN model displays curves that rise rapidly and approach the top-left corner, suggesting better classification capabilities. The closeness to the top left corner signifies a higher true positive rate and a lower false positive rate, while also indicating the optimal threshold for achieving a balance between precision and recall. The steepness of the curve further indicates a more precise and accurate model.

In Figure 4.39, the AUC values for each damage class are provided. The crack class achieved an AUC of 0.9976, the intact class achieved 0.9995, and the spalling class obtained an AUC of 0.9981. These values demonstrate the model's high performance in effectively distinguishing between positive and negative instances and accurately identifying different damage classes. According to Mandrekar (2010: 1316), the model's AUC values signify outstanding performance in classifying damage.

Overall, the ROC curve and AUC values demonstrate the strong classification abilities of the CNN model for the crack, intact, and spalling classes. The model shows high and low true positive rates, indicating accurate and reliable classification performance.

The misclassification rate, precision and recall were reviewed during testing to further compare model performance. The performance achieved for these metrics are presented in Table 4.16.

Table 4.16 Defect classification misclassification rate, precision and recall of the AlexNet CNN model during testing.

Model Number	Model Type	Type	Misclassification rate %	Precision %	Recall %
25	CNN	AlexNet CNN	2.60	97.40	97.40

The misclassification rate was calculated using the number of misclassified instances (6) and the total number of predictions (231) and was calculated to be 2.6%.

The precision was calculated using Figure 4.38. It was determined that each class provided 75 true positive predictions and 2 false positive predictions. A precision rate of 97.4% was obtained for each class as well as for the overall model precision, indicating how highly reliable the model predictions are.

Similarly, recall was calculated using Figure 4.38, where each class again obtained 75 true positive predictions and 2 false negative predictions. The recall rate for each class and the overall model was also calculated as 97.4%, indicating the model's ability to identify instances of each class correctly.

Classification

Figure 4.40 shows an image obtained from Robben Island and the defect classification results achieved by the AlexNet CNN model during testing. The image displays the true label of the image along with the model’s predicted label and the associated confidence score. This image belonged to the crack defect class, with the model accurately classifying it as crack damage with a confidence score of 1.00 (100%).

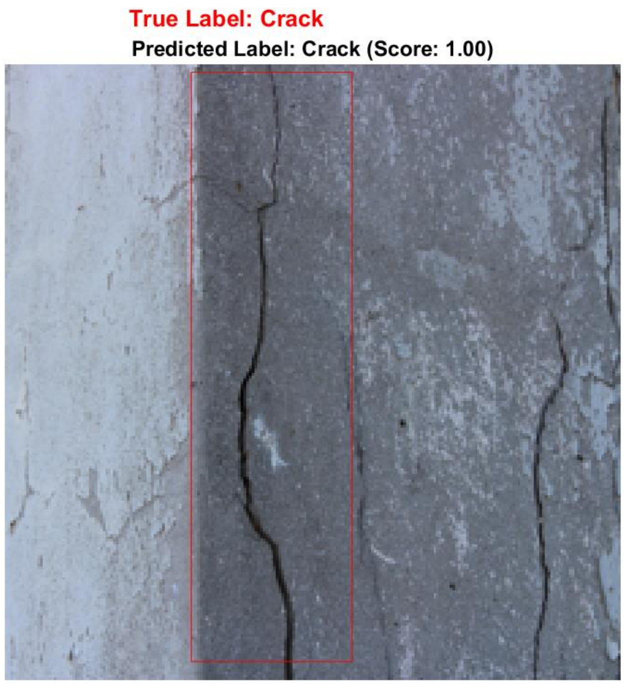


Figure 4.40 Image classification out by the AlexNet CNN model during testing on a crack image obtained from Robben Island.

4.4 Comparison of model performance

The results presented show the performance of the different machine learning and deep learning methods in classifying categories of crack and spalling damage, and intact areas.

The comparison of model performance centred around the Bag of Visual Words technique used with the conventional machine learning algorithms, and the transfer learning strategy used pre-trained AlexNet CNN model.

The Bag of Visual Words technique combined both the SURF function and k-means clustering when extracting features. These features were then input into the conventional machine learning models. This differed from the transfer learning strategy used with the CNN model, which extracted features directly from images.

4.4.1 Model accuracy

The machine learning and deep learning models were assessed in terms of accuracy at the testing phase to determine the best performing model.

Among the decision tree-based models, the fine tree achieved the best accuracy of 76.62%, which was followed by the medium tree at 72.73%, the coarse tree at 73.59%, and the optimisable tree at 71.86%.

In the SVM category, the cubic SVM performed obtained the highest accuracy of 88.31%. This was followed by both the quadratic SVM and optimisable SVM models which produced accuracies of 86.58% and 85.28% respectively. The linear SVM produced an accuracy of 74.46%, whereas the fine Gaussian SVM, medium Gaussian SVM, and coarse Gaussian SVM produced accuracies of 72.29%, 75.76%, and 67.97%.

For the k-nearest neighbours models, the fine k-nearest neighbour obtained the highest accuracy of 82.25%. The other k-nearest neighbour models all showed lower accuracies, ranging from 60.17% to 78.35%.

In the artificial neural network category, the medium artificial neural network performed best, obtaining the highest accuracy of 87.45%. This was followed by the narrow artificial neural network at 84.42%, the wide neural network at 85.28%, the bi-layered neural network at 82.25%, the tri-layered neural network at 83.12%, and the optimisable neural network at 80.95%.

Out of all the machine learning methods, the cubic SVM obtained the highest accuracy of 88.31%. The coarse k-nearest neighbour model obtained the lowest accuracy out of all the models, obtaining an accuracy of 60.17%.

Overall, the pre-trained AlexNet CNN model obtained the highest accuracy among all models where an accuracy of 97.40% was obtained.

4.4.2 Model misclassification rate, precision and recall

The different models' misclassification rate, precision and recall were also assessed during testing to further assist in comparing their performance.

In the category of decision tree models, the fine tree produced the best performance with a misclassification rate of 23.38%, precision of 76.85%, and recall of 76.62%.

In the SVM category, the cubic SVM obtained the best results, achieving a misclassification rate of 11.69%, precision of 88.19%, and recall of 88.31%.

Among the k-nearest neighbour models the fine k-nearest neighbour model obtained the best results achieving a misclassification rate of 17.75%, precision of 83.39%, and recall of 82.25%.

Among the artificial neural networks, the medium neural network performed best with a misclassification rate of 12.55%, precision of 87.36%, and recall of 87.45%.

Once again the cubic SVM produced the best results among the traditional machine learning models. However, among all models CNN showed the best results, obtaining a misclassification rate of 2.60%, precision of 97.40%, and recall of 97.40%. This shows the CNN model's ability to accurately classify defects surpassing all models in terms of the different metrics.

4.4.3 Classification results

The results obtained show the capacity of both deep learning and machine learning methodologies in classifying different types of defects in heritage structures. The AlexNet CNN model produced the best results in terms of all metrics, obtaining scores of 97.40% for accuracy, precision and recall while obtaining the lowest misclassification rate of 2.60%. The high scores obtained for the precision and recall also indicate that the model can correctly identify positive instances while minimising predictions of false positives and false negatives.

When carrying out the classification task using the traditional machine learning algorithms, the cubic SCM produced the best results, obtaining an accuracy score of 88.31%. Additionally, the model's high scores for both precision and recall showed its ability to effectively classify different defect types.

These results show that both machine learning and deep learning approaches can carry out the classification task and can enhance the accuracy and precision with which defect classification is carried out in heritage structures.

The effectiveness of the models depended on several factors, such as the feature extraction capability of the methodologies applied, which affected the models' ability to capture patterns in the image data, and the complexity and capacity of the different models where deeper and more complex models showed better accuracy. Another factor which affected model performance was the dataset size and its complexity. Collectively, these factors contributed to the results of the models and are discussed in the subsequent sections.

4.4.4 Feature extraction

The quality of features extracted by models played a huge role in how effectively the models performed. When models are trained on complex and diverse ranges of features, it assists in creating a model that is robust and able to generalise well to unseen data, which enhances its ability to accurately classify a wide range of defects (Choi, 2020: 111; Kim et al., 2020:2). Feature extraction is therefore important, influencing the model's ability to identify patterns and characteristics associated with defects. For the traditional machine learning models, feature extraction was carried out using the Bag of Visual Words technique.

Bag of Visual Words approach

This approach followed a similar methodology to that of Ghalyan et al. (2019: 3899-3913) and Kim et al. (2018:1-14). In this approach, the Bag of Visual Words technique was used to extract local features from images using the SURF function. These features were then clustered using k-means clustering to create the "visual vocabulary".

Several factors influenced the Bag of Visual Words. These included the parameters set out for clustering, which were the number of clusters (visual words) used. Another factor was the type of feature extraction selected. The selection of parameters such as the 500-word visual vocabulary and the grid step feature extraction method might provide better computational efficiency (quicker feature extraction) and improved classification results. Finding the best combination of parameters was important as the Bag of Visual Words is sensitive to these parameters. The best combination also allows for optimal training and ultimately testing results of the machine learning models.

Al Chanti and Caplier (2018: 1-2) also identified factors which may affect the performance of the Bag of Visual Words. This included the number of key points detected by the model, where too many key points will affect the computational efficiency. Furthermore, k-means clustering may be poor as multiple local features may be extracted and associated with the same word. Al Chanti and Caplier (2018: 1-2) also identified the weighting scheme of the Bag of Visual Words as a possible issue as it treats all visual words equally and disregards the importance of certain words.

For this research, the performance of Bag of Visual Words technique proved successful in identifying features in image data, which is reflected in the high accuracy, precision and recall of the best performing models in each category. However, the technique extracted 12 826 232 features from the dataset and was computationally expensive. This was to the extent that the MATLAB cloud-based platform had to be accessed to complete the feature extraction and k-means clustering.

Transfer learning approach

A transfer learning approach was utilised with the pre-trained AlexNet CNN. Unlike the Bag of Words approach the CNN uses a data-driven feature extraction process whereby the CNN automatically learns hierarchical features from image data. Joubert (2020:32) states that CNN models are able to extract various patterns, features and edges from image data regardless of their position in the image. This is a result of the CNN architecture which has specialised convolution and pooling layers capable of completing the task. These layers allow the CNN to interactively extract features from the dataset, as explained in Section 2.4.9.

Another benefit of the transfer learning approach is that the AlexNet CNN has already undergone training on an extensive dataset. The model has therefore acquired features from an array of images. Joubert (2020:49) states that the model can use this previously gained knowledge and apply it to the new defect classification task. This is completed through fine-tuning the new dataset.

Through the CNN's architecture, it is able to extract more descriptive features and patterns in comparison with the Bag of Visual Words technique. Furthermore, the combination of transfer learning with the pre-trained CNN allows better performance, especially if the provided dataset is small, where sufficient data is not available to train the model from scratch.

4.4.5 Model complexity and capacity

The difference in performance between machine learning algorithms and deep learning models can also be attributed to how computationally powerful and complex they are.

Although artificial neural networks and deep learning CNN models are more complex than conventional machine learning algorithms such as decision trees, k-nearest neighbour and SVMs, these simpler models still showed good performance in this classification task. However, these models were all outperformed by the more complex deep learning CNN.

The CNN complexity is due to a combination of its architecture and its learnable parameters which are sometimes several million (Choi, 2020: 12). Due to this, the CNN model has a greater capacity to learn features and patterns. Joubert (2020:32) states that the CNN's ability to automatically extract patterns through its multiple convolutional and pooling layers allows the model to model relationships between pixels. This allows the model to distinguish different features for accurate classification.

Furthermore, the architecture of the model allows it to learn textures, shapes and fine details more efficiently than traditional models. This is important when dealing with heritage structures which have a variety of different textures due to the construction materials used.

4.4.6 Dataset size

Another factor in the model performance is the size of the dataset used. When using a larger dataset it provides a number of advantages as it provides a wider range of features and patterns that can be extracted. It also provides a more representative and diverse sample for the model to learn, which creates a more robust model that can generalise better to unseen data. In terms of heritage structures, a larger dataset would mean a variety of cracks, spalling textures and intact images which would allow the model to learn an array of features (Choi, 2020:109).

As the AlexNet CNN is pre-trained on the ImageNet dataset, which contains millions of images, it has an advantage over conventional machine learning models trained from scratch. Joubert (2020:94) states that this gives the CNN model a good foundation where previously learned features can be used for the new classification task. This also means the model requires less training data to adapt to the new classification task. Transfer learning therefore plays a huge role in the good performance of the CNN model as the machine learning models used had to be trained from scratch on a smaller dataset.

4.4.7 Defect class ambiguity and image variability

In this research, various challenges were presented that were mainly due to class ambiguity and image variability. These factors affect the performance of the models as described below.

- Class ambiguity: in this research, only crack, spalling and intact surfaces were considered. This was a challenge as the spalling and crack class sometimes presented similar features. This was especially the case for the spalling class as cracks were observed in these images as well. This could potentially lead to misclassifications, resulting in false positive or false negative classification. If a well-defined boundary is not set it may pose difficulties for models and they may not be able to distinguish different defects. Data augmentation can assist with this as it can introduce variations to data and additional training samples where the model can learn patterns more effectively.
- Image variability: Choi (2020:110-111) states that images can vary as a result of lighting, viewing angles, and resolutions. Lighting may cause different contrasts, shadows and highlights which affects the image's appearance. Varying image resolution affects the model, in that a lower resolution may cause images to lose finer details, impacting the models ability to extract features and patterns. For this image, pre-processing can be used to assist in improving contrasts and image quality while normalisation can be used to allow a consistent pixel input into the model. Choi (2020: 110) also highlighted the effects of noise and irrelevant objects in the image background as a factor which affects model performance. This can be in the form of shadows, lighting, edges of brick work and mortar joints and even simple markings on walls which can be misinterpreted by the model. Denoising of images during pre-processing can assist in improving image quality to ameliorate this problem.

Addressing these challenges was important for the development of a reliable model. Using a combination of pre-processing techniques and quality control when data was collected was important in ensuring the best model performance during model testing.

4.5 Extent of damage using image processing

To assess the extent of damage, a methodology was used whereby image processing techniques such as segmentation or binarisation, thresholding and skeletonisation were used, with the main aim of determining characteristics of the damage. The damage characteristics being measured included the length and average width of the cracks. Due to the structure of cracks, an approach was used to combine image processing with distance-based computations. The methodology binarised images to isolate their shape and structure. Subsequently, quantitative metrics were extracted to quantify their length and width.

For this work, a separate dataset made up of ten crack images obtained at the Castle of Good Hope was used. The length and average width of the cracks were measured physically on site, and images of the cracks were captured. To enhance the precision of the methodology, a tape measure was incorporated along the sides of the images, allowing the measurement of the crack dimensions through both direct measurement and later image processing methods. Figure 4.41 shows one of the images that were captured, showing the approach utilised to carry out the crack analysis.

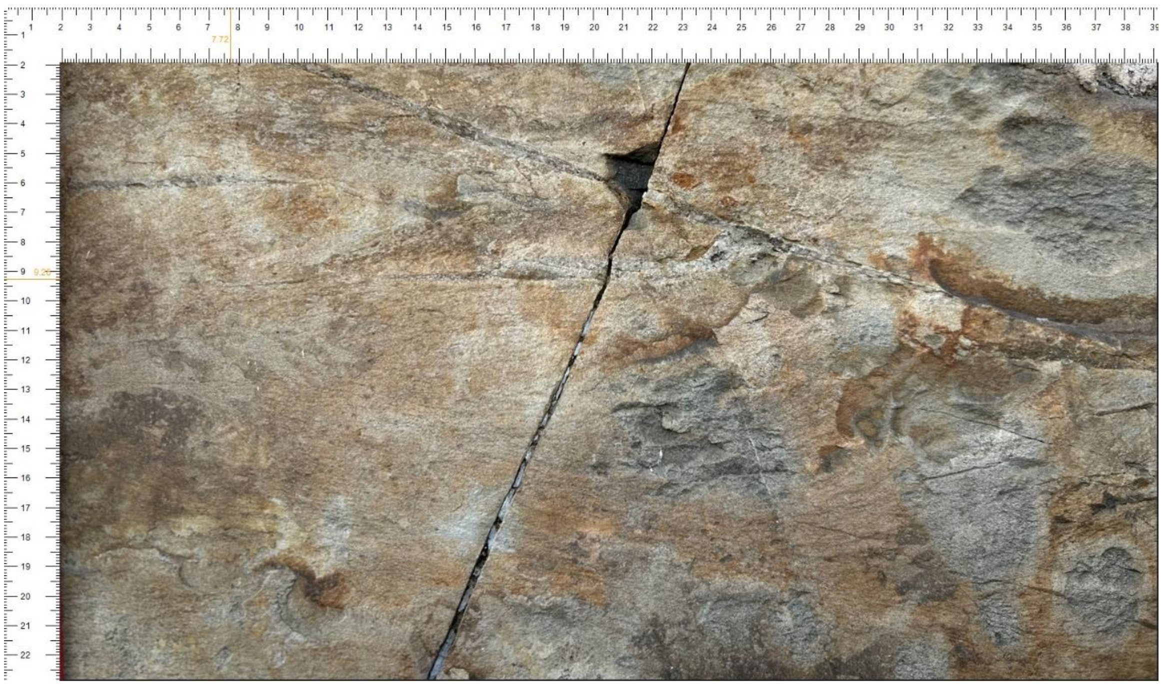


Figure 4.41 Crack image used to determine the extent of damage using image processing.

The full sample of images is shown in Figure 4.42.

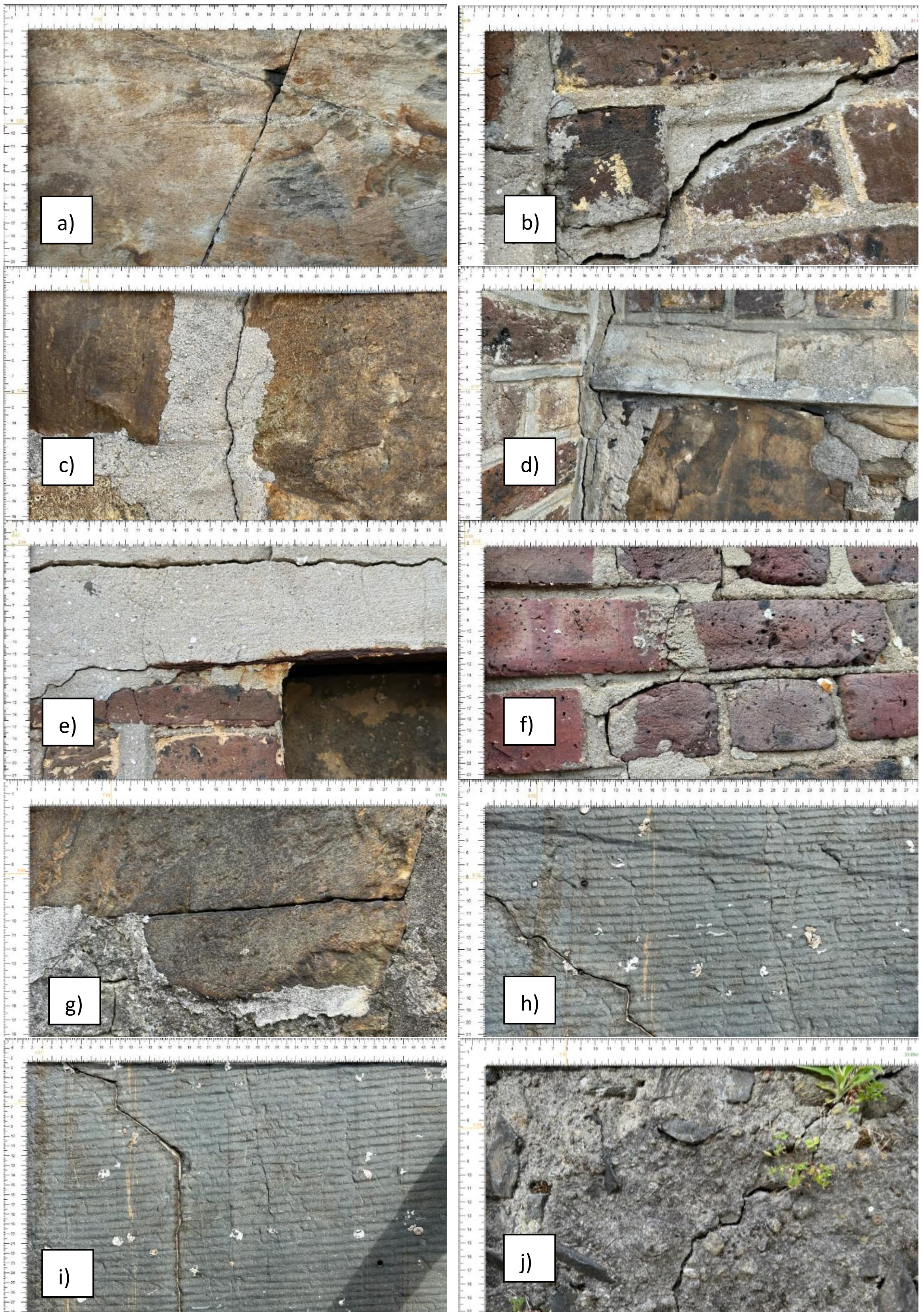


Figure 4.42 Sample of images (a) to (j) used to determine the extent of damage using image processing.

The length of the cracks was obtained by measuring directly from the start point to the end point of the crack, as shown in Figure 4.43.

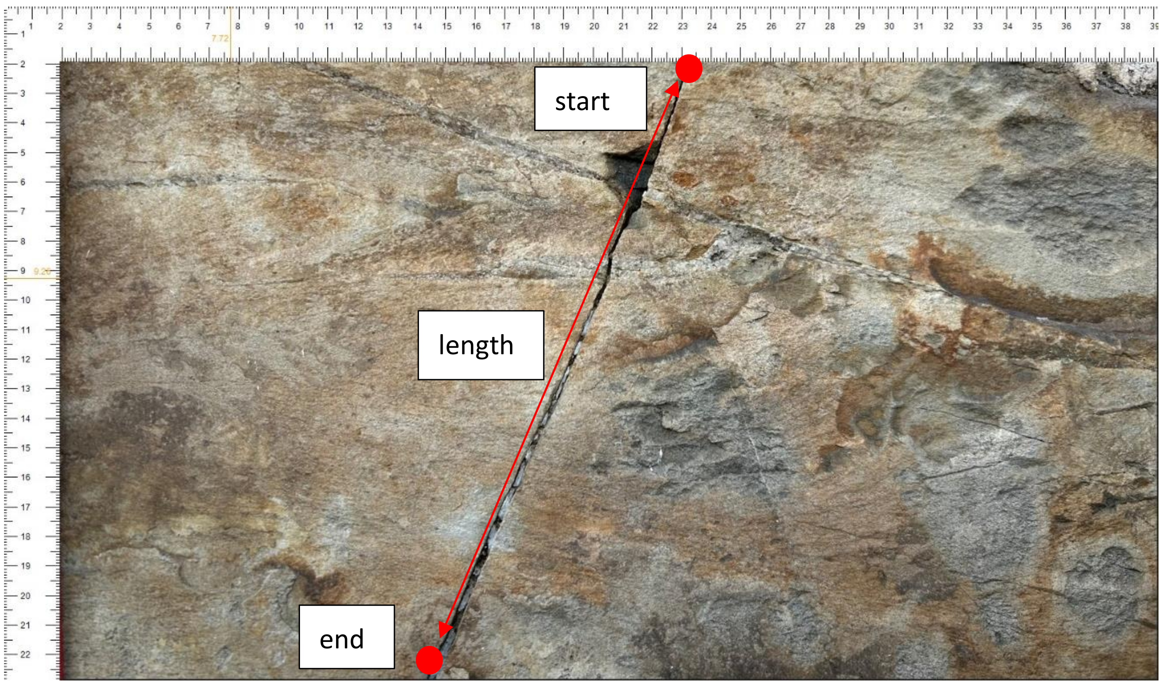


Figure 4.43 Length measurements.

The average crack width was obtained by measuring across the width of the crack at three different points, as shown in Figure 4.44.

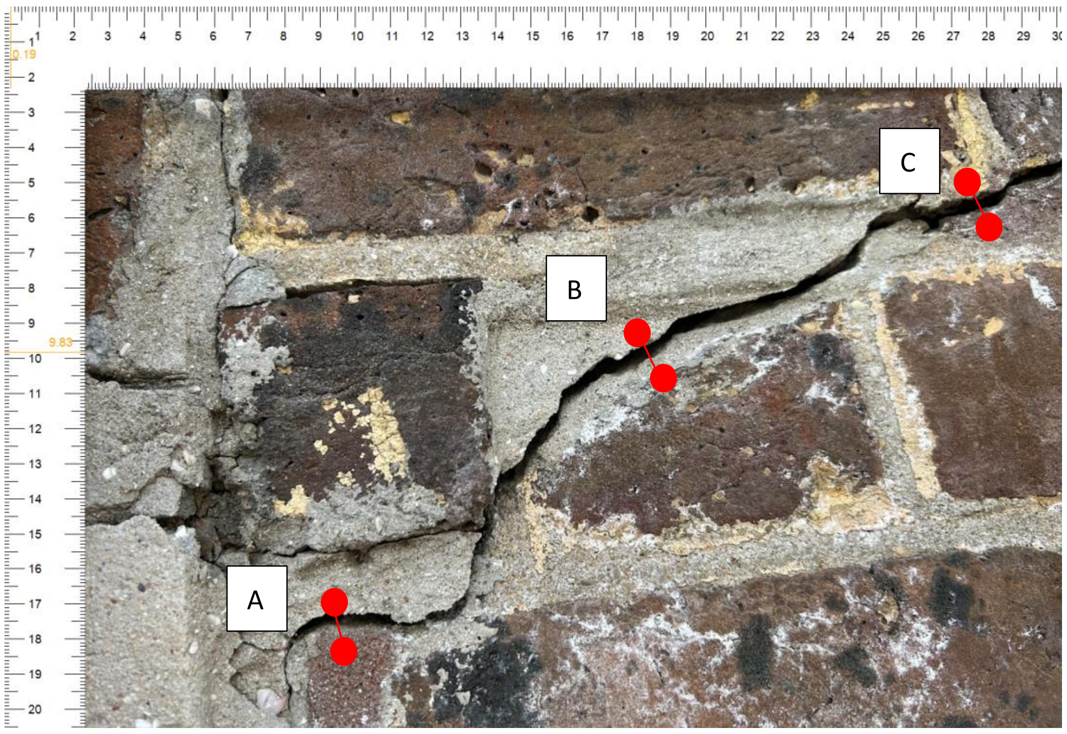


Figure 4.44 Measurements taken at point “A”, point “B” and point "C" to obtain the average crack width.

Table 4.17 shows the values obtained for the length and average width physical measurements.

Table 4.17 Average width and length measurements obtained for the crack image dataset.

Crack	Average Width (mm)	Linear length (mm)
a	3	25
b	3.5	29
c	1	20
d	2	26
e	2.5	43
f	2	26
g	2	21
h	2.5	30
i	3	30
j	3	27

4.5.1 Image processing

Once the dataset was obtained, the images underwent image processing. This included the conversion of images to a grey scale format, after which segmentation was carried out using thresholding.

Figure 4.45 shows the input image, converted to a grey scale format, was then segmented or binarised using thresholding to create a binary image.

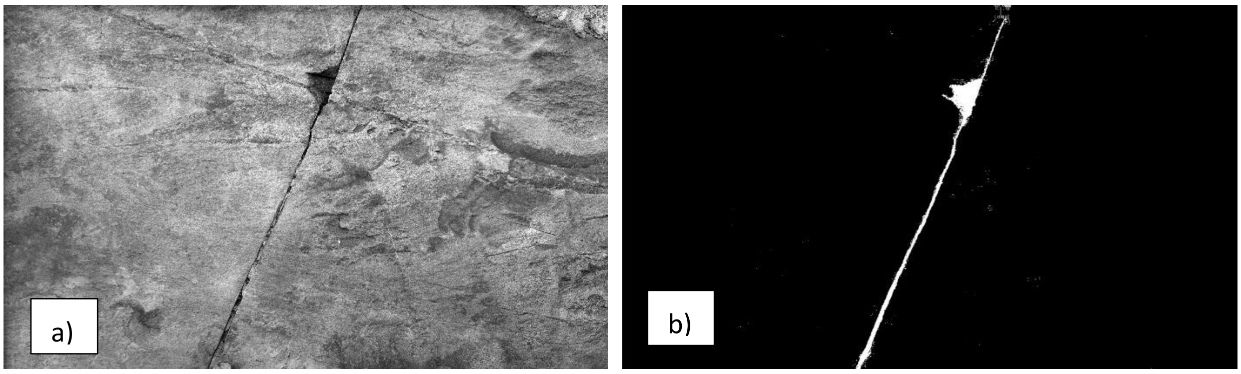


Figure 4.45 Image segmentation and thresholding used on the (a) grey scale image to create a (b) binary image.

Skeletonisation was then carried out on the binary image to reduce the shape to one pixel wide while maintaining the topology. This was then overlayed on the binary image. Figure 4.46 below shows the overlay of the red skeletonisation on the binary image.

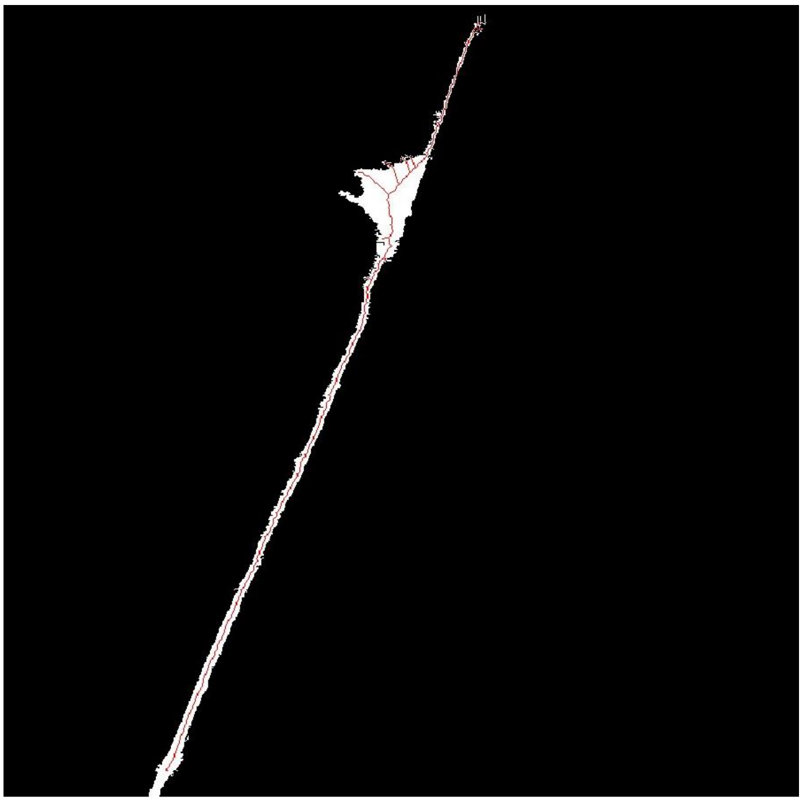


Figure 4.46 Skeletonisation overlay on the binary crack image.

The Euclidean distance transform was then applied to the binary image to obtain the distance map, where distances along the skeleton image were extracted and the average crack width calculated.

The endpoints of the skeleton were then identified, and the straight line distance was calculated using the Euclidean distance formula, which measured the geometric distance between the identified endpoints in the skeletonised image.

As the image dimensions were specified in pixels, the resolution per millimetre was calculated using the measured physical dimensions. Figure 4.47 shows the final quantitative measurement achieved through image processing for the first crack image.

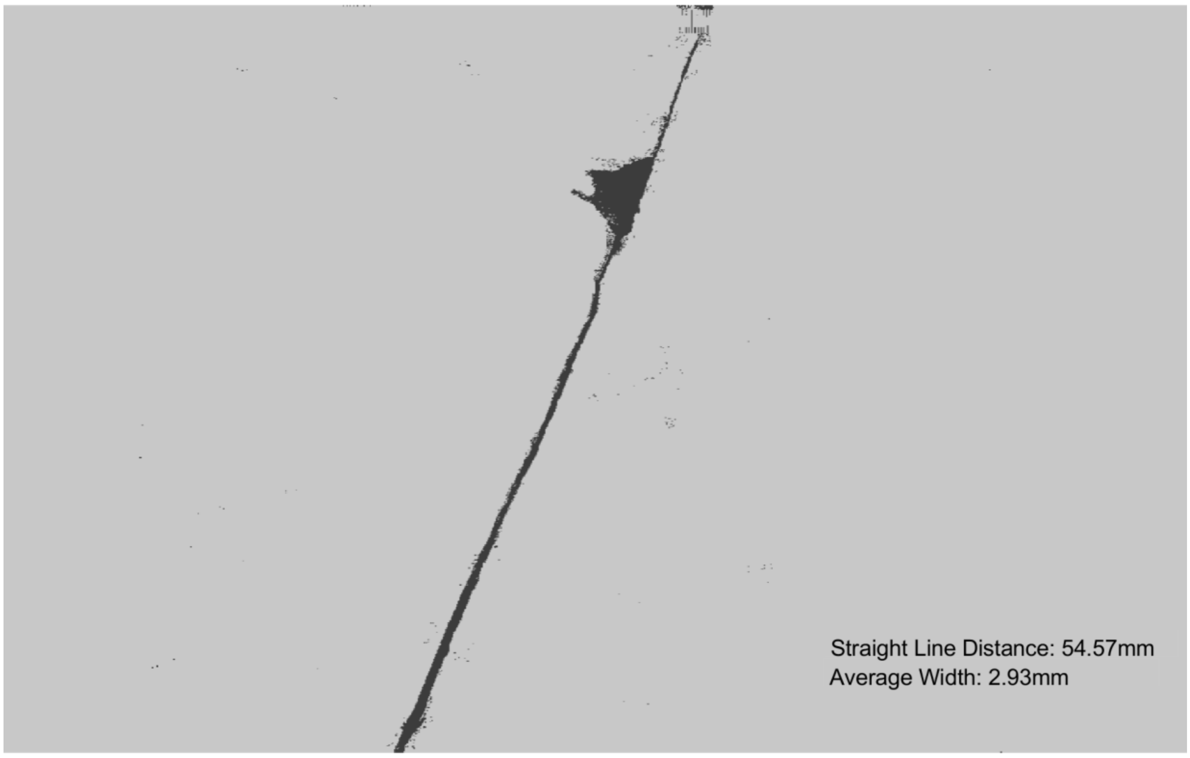


Figure 4.47 Final measurement of crack width and linear distance using image processing.

4.5.2 Image processing results

The final image processing quantitative measurements are shown in Table 4.18 and Table 4.19 respectively.

Table 4.18 presents the results achieved for the average width computation using image processing techniques and the error percentage in comparison with the physical measurements.

Table 4.18 Average width image processing results and error percentages obtained.

Crack	Average Width (mm)	Average Width Image Processing (mm)	Error (%)
a	3	2.93	2.33
b	3.5	3.16	9.71
c	1	1.18	18.00
d	2	1.45	27.50
e	2.5	2.38	4.80
f	2	2.36	18.00
g	2	1.86	7.00
h	2.5	2.10	16.00
i	3	3.06	2.00
j	3	2.94	2.00

Table 4.18 shows the measurement obtained through image processing. Some of the measurements were observed to be smaller than the physical measurements, such as cracks “a”, “b”, “d”, “e”, “g”, “h” and “j”, while some were noted to be greater than the physical measurements, such as cracks “c”, “f” and “I”.

Furthermore, it can be seen that the error percentage ranges from a minimum value of 2% to a maximum value of 27.5%, with an average error of 10.73% obtained overall. It was also observed that cracks “c”,” d”, “f” and “h” produced much higher error rates than the other images. These cracks all measured less than 2mm in width with the exception of crack “h” which had an average width of 2.5mm but ranged from 1mm to 4mm in width. It can thus be seen that the image processing algorithm is highly sensitive to the finer cracks, extracting its features and geometry with a slight error.

Although an average error of 10.73% was obtained, the crack width image processing measurements vary only slightly in comparison to the physical crack measurements.

Table 4.19 presents the results obtained for the length computation using image processing and the error percentage achieved when compared with the physical on site measurements.

Table 4.19 Average width image processing results and error percentages obtained.

Crack	Linear length (mm)	Linear Length Image Processing (mm)	Error (%)
a	25	54.54	118.16
b	29	3.5	87.93
c	20	8.97	55.15
d	26	139.54	436.69
e	43	197.73	359.84
f	26	0.46	98.23
g	21	28.86	37.43
h	30	58.59	95.30
i	30	0.53	98.23
j	27	48.75	80.56

The results in Table 4.19 shows that high error percentages were obtained for the length computation. The error percentages range from 37.43% to a very high percentage of 436.69% and vary across all cracks. These values show the inaccuracy in the computation of the length, with the final results achieved below a value of random guessing. One of the main reasons for this was that the computation found it difficult to establish the correct endpoints of the cracks partly due to their complexity and varying shapes.

Differences between the physical measurements and those obtained through image processing are also related to:

- Image processing limitations: the image processing techniques applied show limitations, being highly sensitive to changes in the crack structure, crack shape and noise in images. The crack dataset used comprised irregular crack shapes which ranged in size and width. Additionally, these images contained noisy backgrounds which could not be fully removed by thresholding. This led to inaccuracies in results as the skeletonisation technique struggled to detect the true endpoint of cracks.
- Thresholding and binarisation: this forms another important aspect of the image processing procedure. This is because thresholding ultimately provides the crack edges or boundaries. If thresholding is not carried out accurately and the edges of the crack are not well defined, this will lead to errors when skeletonisation is carried out.

- Image quality: noise in the form of background elements, shadows or texture changes that result in edges being formed in the image pose difficulties to the thresholding procedure. They affect the detection of crack boundaries and edges and can therefore affect segmentation and subsequent image processing measurements.
- Manual measurements: manual measurements are subjective and there is always a possibility of human error. This can result in differences between physical on site measurements and those obtained through image processing.
- Scale and image resolution: any discrepancies in the image scale and resolution of the image taken and those applied in the image processing algorithm can result in differences. It is important that units and scales be consistent and as accurate as possible.

The results show the complexity and challenges involved when using image processing to assess and extract the characteristics of crack damage. These challenges were highlighted when using image processing to determine the length of cracks, where crack endpoints could not be accurately determined, resulting in an error rate which varied across the different cracks. Despite this, the methodology obtained fairly accurate results when used to determine the average crack width. Due to the varying results obtained, there still is a need to refine the image processing methodology and enhance its accuracy so that more consistent results can be obtained.

4.6 Chapter summary

This chapter presented the results of the performance of the machine learning models and deep learning models in the detection of defects in South African Heritage structures. The following observations were made:

- In the decision tree models, the fine decision tree model performed better with an accuracy of 76.62%, a misclassification rate of 23.38%, a precision of 76.85% and a recall of 76.62%.
- In the SVM models, the cubic SVM model performed better with an accuracy of 88.31%, a misclassification rate of 11.69%, a precision of 88.19% and a recall of 88.31%.
- Among the k-nearest neighbour models, the fine k-nearest neighbour model performed best with an accuracy of 82.25%, a misclassification rate of 17.75%, a precision of 83.39% and a recall of 82.25%.
- In the artificial neural network models, the medium artificial neural network model performed better with an accuracy of 87.45%, a misclassification rate of 12.55%, a precision of 82.48% and a recall of 84.42%.
- The CNN performed best among all models with an accuracy of 97.40%, a misclassification rate of 2.60% and a precision and recall of 97.49%.

In the detection of damage extent for cracks, the image processing techniques could not accurately determine the length of cracks, with error rates varying across the different cracks and producing an average error rate of 146.75%. The image processing techniques could however be used to determine the average crack width obtaining an average error rate of 10.73%, where only slight variations were observed between the physical on site measurement and those obtained through image processing.

The results shown in Chapter 4 highlight the capacity of the machine learning models to detect defects in heritage structures, while also showing the complexity and difficulty involved in determining the damage extent using image processing.

Chapter 5 Conclusions and recommendations

Given the immense role of heritage structures in the development of economic growth through tourism and preserving cultural, traditional, and religious heritage, prioritising their conservation becomes very important. Structural health monitoring is important in ensuring their longevity. The main objective of this research was to compare the performance of machine learning algorithms in detecting wall defects within South African heritage structures using image processing techniques. For this, machine learning and deep learning algorithms were used to classify defects in images gathered from the Castle of Good Hope, Dal Josafat, and Robben Island heritage sites. These machine learning and deep learning models were trained using the captured images and subsequently tested, determining their accuracy in classifying defects within the heritage structures.

Another area focus was determining the extent of crack damage through an image processing approach. This was completed through a combination of image processing techniques such as colour modification, binarisation, thresholding and skeletonisation. The effectiveness of the image processing methodology was assessed by determining the percentage error or difference between physical crack measurements with those obtained through the image processing framework. This chapter summarises the research findings and provides ideas and recommendations for future research in this field.

5.1 Conclusions

According to the literature reviewed, the integration of machine learning and image processing into structural health monitoring tasks presents a promising approach to addressing issues related to the on-going deterioration of heritage structures. Researchers have used machine learning as a resource for a range of structural health monitoring activities which include crack detection in buildings, damage detection in ancient timber structures, damage detection in limestone walls and even the monitoring of structures for early vulnerability warnings.

Although a large amount of research has been completed in the machine learning and structural health monitoring field, the use of machine learning as a tool for damage classification still faces a number of challenges. These issues were highlighted by Mishra (2021:240), who found that models are often not robust enough to generalise to unseen or new data, which results in misclassifications. This is because heritage structures have complex typologies, architectural features and construction materials that require a complex database of information for sufficient training and testing of machine learning models. Choi (2020:110-111) found that algorithms were affected by the quality of images, with noise and changes in lighting playing a role in how effectively the algorithm performed.

In this regard, the main objective of this research study was to compare the performance of different machine learning and deep learning algorithms in detecting defects in heritage structures. Machine learning and deep learning algorithms such as decision trees, SVM, k-nearest neighbour, artificial neural networks and a pre-trained CNN model were used to classify crack and spalling damage as well as intact areas.

From the analysis, it can be seen that models performed well with a high degree of accuracy and a low misclassification rate on the defect dataset. It was found that in the decision trees category, the fine tree performed better. In the SVM category, the cubic SVM performed best. In k-nearest neighbour category, the fine k-nearest neighbour performed best, and in the artificial neural network category, the medium artificial neural network performed best. However, overall the deep learning CNN produced the highest accuracy, precision and recall scores among all models. The CNN obtained scores of 97.40% for accuracy, precision and recall, while a misclassification rate of 2.60% was obtained.

Comparing the results of the deep learning CNN with that of the best performing machine learning algorithm, the cubic SVM, the CNN obtained an accuracy and recall which was higher by 9.09%, a misclassification rate that was lower by 9.09% and a precision which was higher by 9.21%. This research found the deep transfer learning strategy to be more effective in classifying defects in comparison to the traditional machine learning models that were trained from scratch.

For the traditional machine learning models, the Bag of Visual Words algorithm was used to detect patterns and extract damage features. This differed from the deep learning approach where patterns and features were extracted directly by the CNN. The features extracted included different image textures, changes in colour and structural changes within the images which were indicative of damage. The models were trained on this data and this enabled them to differentiate between different damage types. The ability of the models to learn the distribution of extracted damage features in images is directly reflected in the accuracy, misclassification, precision and recall results achieved when new unseen data is input into the model.

Another objective of the research was to determine the characteristics of crack damage using image processing techniques. For this, techniques such as grey scaling, binarisation, thresholding and skeletonisation were used in hand with the Euclidean distance transform to extract crack characteristics. The image processing methodology worked accurately in determining the average crack width. For this, an average error rate of 10.73% was obtained, with measurements obtained through image processing varying only slightly in comparison with the physical measurements. However, where the methodology lacked and showed limitations was in the extraction of linear crack length. Here the methodology obtained highly inaccurate results, with error rates varying greatly across each crack. This is due to the sensitivity of the skeletonisation technique, which was found to be highly sensitive to image quality, noise in images and the structure of the crack, where any interference would result in unreliable and erroneous results.

It must be noted that this research only focused on three types of defects where deteriorating heritage structures would present much more. By testing the models on datasets with varying characteristics, one might better understand their potential limitations and capabilities beyond the scope of this research.

Results showed the capacity of image processing, machine learning and deep learning to be used as a resource for classifying and determining the extent of defects in heritage structures, within a structural health monitoring framework. These techniques are being increasingly used to analyse non-destructive test data, sensor data, and images obtained during inspections, and can provide us with valuable information regarding the structural health of heritage structures. The integration of image processing and machine learning into structural health monitoring also provides a potential tool that would assist engineers and heritage experts in making informed decisions about the conservation and restoration of heritage structures, ultimately ensuring their continued contribution to tourism, cultural identity, and economic growth.

5.2 Recommendations

This research highlighted the considerable potential of incorporating machine learning and image processing into structural health monitoring systems, although within a specific scope limited to defect classification. While this study serves as an initial step, future research must extend beyond defect classification and the determination of the extent of crack damage. Researchers should explore additional avenues using machine learning and image processing techniques, including damage localisation. Moreover, the classification of distinct construction materials found in heritage structures should also be investigated, as it is important for creating intervention methods and selecting the appropriate restoration materials.

There is a clear need to develop a comprehensive image database encompassing a wider range of heritage sites and structural typologies. This database would lead to the creation of a more robust machine-learning model, capable of accommodating the diverse characteristics of heritage structures. This larger image database would ensure the robustness and adaptability of the machine learning model in defect classification across varying localities and different climatic and environmental conditions where structures are exposed to different aspects and types of degradation.

To enhance the holistic assessment of heritage structures, researchers should explore incorporating multi-modal data, including non-destructive testing data and sensor information, alongside image data. In this way, one could create a more comprehensive structural assessment.

Due to the varying results achieved when determining the extent of crack damage, research should be conducted into image processing techniques currently employed to extract crack characteristics. Research should be aimed towards refining and improving image processing algorithms, binarisation and thresholding to better handle irregular crack shapes, noisy backgrounds and variations in crack dimensions allowing for more accurate extraction of crack characteristics.

An emphasis should also be placed on determining the extent of a range of damage types through a holistic image processing approach, allowing for the accurate extraction of damage characteristics.

Lastly, the economic feasibility of implementing image processing and machine learning in heritage preservation should be investigated. Conducting cost-effectiveness analyses can offer insight into the long-term benefits of adopting a machine-learning approach.

References

Aasly, K.A., Meyer, G.B., Keiding, J.K., Langås, R. & Lund, V. 2017. Uniting geology and craftsmanship to find the optimal soapstone for restoration of the Nidaros soapstone Cathedral in Norway. *EGU2017 Conference, EGU General Assembly Conference Abstracts*. 23-28 April 2017, Vienna, Austria, p. 13003.

Adamopolous, E. 2021. Learning-based classification of multispectral images for deterioration mapping of historic structures. *Journal of Building Pathology and Rehabilitation,* 6(1):41. DOI: 10.1007/s41024-021-00136-z

Aicardi, I., Chiabrando, F., Lingua, A. & Noardo, F. 2018. Recent trends in cultural heritage 3D survey: The photogrammetric computer vision approach. *Journal of Cultural Heritage*, 32(2018):257-266. DOI: 10.1016/j.culher.2017.11.006

Al Chanti, D. & Caplier, A. 2018. Improving Bag of Visual Words towards effective facial expressive image classification. *VISIGRAPP, the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. January 2018. Portugal: Madeira DOI:10.5220/0006537601450152

Allie, Z. 2022. Episode 111 Dal Josafat National Heritage Site [Video]. https://www.youtube.com/watch?v=L8dSP-4LpDE [06 December 2022].

Alkadri, M.F., Alam, S., Santosa, H., Yudono, A. & Beselly, S.M. 2022. Investigating surface fractures and materials behaviour of cultural heritage buildings based on the attribute information of point clouds stored in the TLS dataset. *Remote Sensing*, 14(2):1-24. DOI: 10.3390/rs14020410

Almac, U., Duppel, C., Schweizerhof, K. & Wenzel, F. 2014. Recent studies on the structural characteristics of Hagia Sophia. *International Workshop on Seismic Risk of Historic Structures*. 17th November 2014: 1-5. Turkey: Istanbul.

Anton, D., Pineda, P., Medjdoub, B. & Iranzo, A. 2019. As-built 3D heritage city modelling to support numerical structural analysis: Application to the assessment of an archaeological remain. *Remote Sensing,* 11(11):1-37. DOI: 10.3390/rs11111276

Baladram, M. S., Koike, A. & Yamada, K.D. 2020. Introduction to supervised machine learning for data science. *Interdisciplinary Information Sciences,* 26(1):87-121. DOI: 10.4036/iis.2020.A.03

Bassier, M., Vergauwen, M. & Van Genechten, B. 2017. Automated classification of heritage building for as-built BIM using machine learning techniques. *Remote Sensing and Spatial Information Sciences*, 4(2):25-30. DOI: 10.5194/isprs-annals-IV-2-W2-25-2017

Burland, J.B., Jamiolkowski, M. & Viggiani, C. 2003. The stabilisation of the Leaning Tower of Pisa. *Soils and Foundations,* 43:63-80. DOI: 10.1080/13556207.2002.10785324

Architects Associated. 2018. *Report on existing housing condition and assessment of recommended work on City owned house in Bo-Kaap*. https://sahris.sahra.org.za [15 October 2022].

Breiman, L., Friedman, J., Stone, C.J. & Olshen, R.A. 1984. *Classification and regression trees*. New York, NY: Taylor & Francis.

Carmo, R.N.F., Valenca, J. & Da Costa, D. 2015. Assessing steel strains on reinforced concrete members from surface cracking patterns. *Construction and Building Materials*, 98:265-275. DOI: 10.1016/j.conbuildmat.2015.08.079

Carrasco, M., Araya-Letelier, G., Velazquez, R. & Visconti, P. 2021. Image-based automated width measurement of surface cracking. *Sensors,* 21(22): 7534. DOI: 10.3390/s21227534

Castle of Good Hope. 2018. *Cape's bastion of history*. https://castleofgoodhope.co.za [10 October 2022].

Chaiyasarn, K., Sharma, M., Ali, L., Khan, W. & Poovarodom, N. 2018. Crack detection in historical structures based on convolutional neural network. *GEOMATE Journal*, 15(51):240-251. DOI: 10.21660/2018.51.35376

Charles Consult Consortium. 2018. *Conditional assessment of buildings on Robben Island: Outline description of material testing required*. https://sahris.sahra.org.za [10 October 2022].

Choi, W. 2020. Deep learning implemented structural defect detection on digital images. Unpublished PhD thesis, The University of Manitoba, Winnipeg, Canada.

Cortes, C. & Vapnik, V. 1995. Support-vector networks. *Machine Learning*, 20(3):273-297. DOI: 10.1007/BF00994018

Crespo, C., Armesto, J., Gonzalez-Aguilera, D. & Arias, P. 2010. Damage detection on historical buildings using unsupervised classification techniques. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(5):184-188. https://www.researchgate.net/publication/264842818_Damage_detection_on_historical_buildings_using_unsupervised_classification_techniques [10 August 2022].

Csurka, G., Dance, C.R., Fan, L., Williamowski, J., Bray, C. Visual categorization with bag of keypoints. *Workshop on Statistical Learning in Computer Vision, European Conference of Computer Vision*. 15 May 2004: 1-16. Czech Republic: Prague.

Ćurčić, A., Momčilović Petronijević, A., Topličić Ćurčić, G. & Keković, A. 2020. An approach to building heritage and its preservation in Serbia and surrounding areas. *Architecture and Civil Engineering,* 18(1):15-31. DOI:10.2298/FUACE200511002C

Daneshtalab, M. & Modarressi, M. 2020. *Hardware architectures for deep learning*. London: Institution of Engineering and Technology.

Davids, G. & Blacky, N. 2019. Governance and management challenges in establishing Robben Island as a national museum and a world heritage institution in post-apartheid South Africa. *New Contree*, 82:1-15. Boloka Institutional Repository, Northwest University. https://repository.nwu.ac.za/handle/10394/33934 [10 October 2022].

Da Silva Ramos, J.L.F. 2007. Damage identification on masonry structures based on vibration signatures. Unpublished PhD thesis, University of Minho, Braga, Portugal.

De Castro Mota, M.L.F. 2021. Structural health monitoring: a machine learning approach. Unpublished Master's thesis, University of Porto, Porto, Portugal.

De Paiva, P. V., Cogima, C.K., Dezen-Kempter, E., de Carvalho, M.A. & Cerqueira, L.R. 2018. Intelligent digital built heritage models: An approach from image processing and building information modelling technology. *In Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 4:558-565. DOI: 10.5220/0006615005580565

Ďurčíková, M. & Vidholdová, Z. 2021. Notre-Dame Cathedral fire. *Delta,* 15(1): 15-26. DOI: 10.17423/delta.2021.15.1.93

Feng, D., Feng, M.Q., Ozer, E. & Fukuda, Y. 2015. A vision-based sensor for noncontact structural displacement measurement. *Sensors*, 15(7):16557-16575. DOI: 10.3390/s150716557

Feilden, B. 2003. *Conservation of historic buildings*. Architectural Press, Elsevier.

Gales, J.A., Bisby, L.A. & Stratford, T. 2012. New parameters to describe high temperature deformation of prestressing steel determined using digital image correlation. *Structural Engineering International*, 22(4):476-486. DOI: 10.2749/101686612X13363929517730

Gertenbach, M. 2019. *Non-Pareille: Historical Overview*. https://sahris.sahra.org.za [04 December 2022].

Ghalyan, N.F., Ghalyan, I.F., Ray, A. 2019. Modelling of microscope images for early detection of fatigue cracks in structural materials. *The International Journal of Advanced Manufacturing Technology*, 104:3899-3913. DOI: 10.1007/s00170-019-04108-z

Gilbert, C. 1994. The Castle of Good Hope: An examination of controversies and conflicting perceptions- A case study in public history. Honours Degree thesis, University of Cape Town, South Africa.

Goodfellow, I., Bengio, Y. & Courville, A. 2016. *Deep learning*. Cambridge, Massachusetts: MIT Press. http://www.deeplearningbook.org.

Hatir, M.E., Barstugan, M. & Ismail Ince, I. 2020. Deep learning-based weathering type recognition in historical stone monuments. *Journal of Cultural Heritage*, 45:193-203. DOI: 10.1016/j.culher.2020.04.008

Heidari, M., Torabi-Kaveh, M., Chastre, C., Ludovico-Marques, M., Mohseni, H. & Akefi, H. 2017. Determination of weathering degree of the Persepolis stone under laboratory and natural conditions using fuzzy inference system. *Construction and Building Materials*, 145:28-41. DOI: 10.1016/j.conbuildmat.2017.03.230

Henke, K., Pawlowski, R., Schregle, P. & Winter, S. 2015. Use of digital image processing in monitoring of deformations in building structures. *Journal of Civil Structural Health Monitoring*, 5(2):141-152. DOI: 10.1007/s13349-014-0091-6

Hoang, N.D. 2018. Image processing-based recognition of wall defects using machine learning approaches and steerable filters. *Computational Intelligence and Neuroscience*, 2018: 7913952. DOI: 10.1155/2018/7913952

Housner, G.W., Bergman, A., Caughey, T.K., Chassiakos, A.G., Claus, R.O., Masri, S.F., Skelton, R.E., Soong, T.T., Spencer, B.F. & Yao, J.T.P. 1997. Structural control: past, present, and future. *Journal of Engineering Mechanics*, 123:897-971. http://ascelibrary.org/journal/jenmdt [6 June 2022].

Idjaton, K., Desquesnes, X., Treuillet, S. & Brunetaud, X. 2022. Transformers with YOLO network for damage detection in limestone wall images. *Lecture Notes in Computer Science*, 13374:302-313. DOI: 10.1007/978-3-031-13324-4_26

Ierimonti, L., Cavalagli, N., Venanzi, I., García-Macías, E. & Ubertini, F. 2021. A transfer Bayesian learning methodology for structural health monitoring of monumental structures. *Engineering Structures*, 247:1-17. DOI: 10.1016/j.engstruct.2021.113089

Jain, N. 2015. Matlab as a tool in image processing. *International Journal of Scientific Research,* 4(11):484-486. DOI: 10.36106/IJSR

Jauregui, D.V., White, K.R., Woodward, C.B. & Leitch, K.R. 2003. Noncontact photogrammetric measurement of vertical bridge deflection. *Journal of Bridge Engineering,* 8(4):212-222. DOI: 10.1061/(ASCE)1084-0702(2003)8:4(212)

Joubert, J.D. 2020. Design and implementation of a convolutional neural network to classify pecan nut cultivars in a post-harvest application. Unpublished Master's thesis, Cape Peninsula University of Technology, Cape Town, South Africa. https://etd.cput.ac.za/handle/20.500.11838/3261 [14 June 2022].

Kim, B. & Cho, S. 2018. Automated vision-based detection of cracks on concrete surfaces using deep learning technique. *Sensors*, 18(10): 1-18. DOI: 10.3390/s18103452

Kim, H., Ahn, E., Shin, M., Sim, S. 2018. Crack and non-crack classification from concrete surface images using machine learning. Structural Health Monitoring, 18(3): 1-14. DOI: 10.1177/1475921718768747

Kulkarni, A., Batarseh, F.A. & Chong, D. 2020. *Data democracy*. London: Academic Press.

Loke, M.E. 2020. Standard protocols for restoring heritage cementing materials. Unpublished Master's thesis, Cape Peninsula University of Technology, Cape Town, South Africa. https://etd.cput.ac.za/handle/20.500.11838/3050 [9 May 2022].

Liu, Y.F., Cho, S.J., Spencer, F. & Fan, J.S. 2016. Concrete crack assessment using digital image processing and 3D scene reconstruction. *Journal of Computing in Civil Engineering*, 30(1):1-19. DOI: 10.1061/(ASCE)CP.1943-5487.0000446

Ma, J., Yan, W., Liu, G., Xing, S., Niu, S. & Wei, T. 2022. Complex texture contour feature extraction of cracks in timber structures of ancient architecture based on YOLO algorithm. *Advances in Civil Engineering*, 2022:7879302. DOI: 10.1155/2022/7879302

Mandrekar, J. 2010. Receiver operating characteristic curve in diagnostic test assessment. *Journal of Thoracic Oncology*, 5(9): 1315-1316. DOI: 10.1097/JTO.0b013e3181ec173d

Marini, A., Cominelli, S., Zanotti, C. & Giuriani, E. 2018. Improved natural hydraulic lime mortar slab for compatible retrofit of wooden floors in historical buildings. *Construction and Building Materials*, 158:801-813. DOI: doi.org/10.1016/j.conbuildmat.2017.10.010

Merono, J.E., Perea, A.J., Aguilera, M.J. & Laguna, A.M. 2021. Recognition of materials and damage on historical buildings using digital image classification. *South African Journal of Science*, 111(1-2):1-9. DOI: 10.17159/sajs.2015/20140001

Mishra, M. 2021. Machine learning techniques for structural health monitoring of heritage buildings: a state-of-the-art review and case studies. *Journal of Cultural Heritage*, 47:227-245.
DOI: 10.1016/j.culher.2020.09.005

Murphy, A.L. 2020. The application of computer vision, machine and deep learning algorithms utilizing MATLAB. Unpublished Master's thesis, University of New Hampshire, Durham, United States of America. https://scholars.unh.edu/thesis/1346 [20 June 2022].

Nguyen, D.H., Bui, T.T., De Roeck, G. & Abdel Wahab, M. 2019. Damage detection in Ca-Non Bridge using transmissibility and artificial neural networks. *Structural Engineering and Mechanics*, 71(2):175-183.
DOI: 10.12989/sem.2019.71.2.175

Noor, S.M., Shing Mei, C., Ibrahim, I.S., Sarbini, N.N., Osman, M.H. & Khiyon, N.A. 2019. Heritage building condition assessment: A case study from Johor Bahru, Malaysia. *Earth and Environmental Science*, 220(2019):1-11. DOI: 10.1088/1755-1315/220/1/012024

Oses, N., Dornaika, F. & Moujahid, A. 2014. Image based delineation and classification of built heritage masonry. *Remote Sensing*, 6(3):1863-1889. DOI: 10.3390/rs6031863

Palma, V. 2019. Towards deep learning for architecture: A monument recognition mobile app. *Remote Sensing and Spatial Information Sciences*, 42(2):551-556. DOI: 10.5194/isprs-archives-XLII-2-W9-551-2019

Park, S.W., Park, S., Kim, J.H. & Adeli, H. 2015. 3D displacement measurement model for the health monitoring of structures using a motion capture system. *Measurement*, 59:352-362. DOI: 10.1016/j.measurement.2014.09.063

Pettit, R.W., Fullem, R., Cheng, C. & Amos, C.I. 2021. Artificial intelligence, machine learning, and deep learning for clinical outcome prediction. *Emerging Topics in Life Sciences*, 5(6):729-745. DOI: 10.1042/ETLS20210246

Ploskas, N. & Samaras, N. 2016. *GPU Programming in MATLAB*. Burlington, Massachusetts: Morgan Kaufmann United States of America.
Professional Engineers Ontario. 2016. *Structural condition assessment of existing buildings and designated structures guidelines*. https://peo.on.ca/index.php/ [19 October 2022].

Quinlan, J.R. 1985. Induction of decision trees. *Machine Learning*, 1:81-106. DOI: 10.1007/BF00116251

Raszczuk, K. & Karolak, A. 2021. Correlation between the cracking pattern pf historical structure and soil properties: the case of the church in Kożuchów. *Heritage Science*, 9(1):43. DOI: 10.1186/s40494-021-00516-1

Robben Island. 2013. *Integrated conservation management plan*. https://www.robben-island.org.za/news#publications [10 October 2022].

Rymarczyk, T., Kłosowski, G., Hoła, A., Hoła, J., Sikora, J., Tchórzewski, P. & Skowron, L. 2021. Historical buildings dampness analysis using electrical tomography and machine learning algorithms. *Energies*, 14(5):1-24. DOI: 10.3390/en14051307

Santos, S.P. 2010. *Guide for the structural rehabilitation of heritage buildings*. Lisbon, Portugal: CIB Commission.

Sarker, I. 2021. Deep Learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2:420. DOI:10.1007/s42979-021-00815-1

Shalunts, G., Haxhimusa, Y. & Sablatnig, R. 2011. Architectural style classification of building facade windows. Advances in visual computing. 7th *International Symposium on Visual Computing.* 26-28 September 2011: 280-289. United States of America: Las Vegas.

Sharma, T., Agrawal, P., Verma, N.K. (2019). Detection of dust deposition using convolutional neural network for heritage images. In: Verma, N., Ghosh, A. (eds) *Computational intelligence: theories, applications and future directions - Volume II. Advances in intelligent systems and computing, vol 799.* Springer, Singapore. https://doi.org/10.1007/978-981-13-1135-2_27

South African Heritage Resource Agency. 2018. *Request for expression of interest for the design and execution of restoration work to heritage buildings and surroundings on Non Pareille, Dal Josafat Farm*. https://sahris.sahra.org.za [06 December 2022].

South African Heritage Resource Agency. 2020. *Sites under protection: National heritage sites*. https://www.sahra.org.za/national-sites/ [25 June 2023].

Tibaduiza, D., Torres Arredondo, M.A., Vitola, J., Anaya, M. & Pozo, F. 2018. A damage classification approach for structural health monitoring using machine learning. *Complexity*, 2018: 5081283. DOI: 10.1155/2018/5081283

Tibaduiza, D.A., Mujica, L, E., Rodellar, J. 2011. Comparison of several methods for damage localization using indices and contributions based on PCA. *Journal of Physics: Conference Series*, 305:1-10. DOI: 10.1088/1742-6596/305/1/012013

Topolnicki, M. 2001. Strengthening and underpinning of stone foundations of St. John's Church in Gdansk. *15th International conference on Soil Mechanics and Geotechnical Engineering*. 27-31 August 2001: 1863-1866. Turkey: Istanbul.

Turek, M.E. 2007. A method for implementation of damage detection algorithms for civil structural health monitoring systems. Unpublished PhD thesis, The University of British Columbia, Kelowna, Canada.

Umzarulazijo Umar, M., Hanizun Hanafi, M. & Abdul Latip, N. 2015. Analysis of non-destructive testing of historic building structures. *Australian Journal of Basic and Applied Sciences*, 9(7):326-330.

United Nations Educational, Scientific and Cultural Organization. 2021. *World heritage convention: Properties inscribed on the world heritage list*. https://whc.unesco.org/en/statesparties/za/ [25 June 2023].

Valero, E., Bosche', F. & Forster, A. 2018. Automatic segmentation of 3D point clouds of rubble masonry walls, and its application to building surveying, repair and maintenance. *Automation in Construction*, 96:29-39. DOI: 10.1016/j.autcon.2018.08.018

Valero, E., Forster, A., Bosche', F., Hyslop, E., Wilson, L. & Turmel, A. 2019. Automated defect detection and classification in ashlar masonry walls using machine learning. *Automation in Construction*, 106:1-14. DOI: 10.1016/j.autcon.2019.102846

Wang, N., Zhao, Q., Li, S. & Zhao, X. 2018. Damage classification for masonry historic structures using convolutional neural networks based on still images. *Computer-Aided Civil and Infrastructure Engineering*, 33(12):1-17. DOI: 10.1111/mice.12411

World Travel and Tourism Council. 2022. *Economic impact report*. https://wttc.org/research/economic-impact [11 October 2022].

Yapuncich, G.S. 2018. Alternative methods for calculating percentage prediction error and their implications for predicting body mass in fossil taxa. *Journal of Human Evolution*, 2018(115):140-145. DOI: 10.1016/j.jhevol.2017.03.001

Ye, X.W., Dong, C.Z. & Liu, T. 2016. A review of machine vision based structural health monitoring: methodologies and applications. *Journal of Sensors*, 2016:1-11. DOI: 10.1155/2016/7103039

Yeum, C.M. & Dyke, S.J. 2015. Vision based automated crack detection for bridge inspection. *Computer Aided Civil and Infrastructure Engineering*, 30(10):759-770. DOI: 10.1111/mice.12141

Yoneyama, S., Kitagawa, A., Tani, K. & Kikuta, H. 2007. Bridge deflection measurement using digital image correlation. *Experimental Techniques*, 31(1):34-40. DOI: 10.2320/matertrans.I-M2011843

Zhu, L., Li, Z., Li, C., Wu, J. & Yue, J. 2018. High performance vegetable classification from images based on AlexNet deep learning model. *International Journal of Agricultural and Biological Engineering*, 11(4): 217-223. DOI: 10.25165/j.ijabe.20181104.2690

Zou, Z., Zhao, X., Zhao, P., Qi, F. & Wang, N. 2019. CNN-based statistics and location estimation of missing components in routine inspection of historic buildings. *Journal of Cultural Heritage* 38:221-230. DOI: 10.1016/j.culher.2019.02.002

Appendix A. Machine learning model training results

The following figures and tables present the results of the conventional machine learning models' performance during the training phase of this research. The results present a breakdown of the performance of the models in the defect classification of the crack, spall and intact areas.

Fine decision tree model

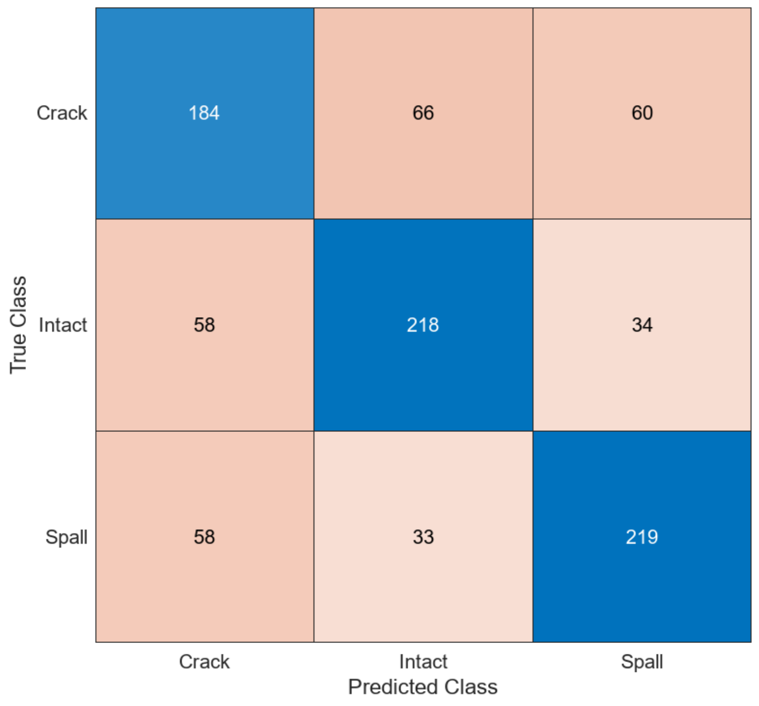


Figure A.1 Confusion matrix showing the classification results of the fine decision tree model.

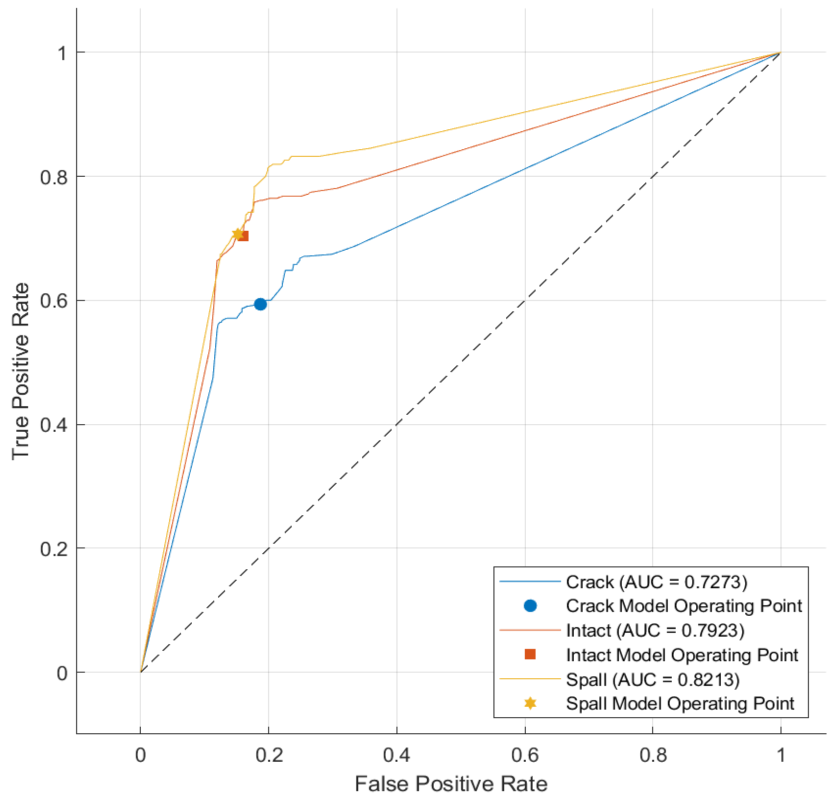


Figure A.2 ROC curve showing the fine decision tree model's performance.

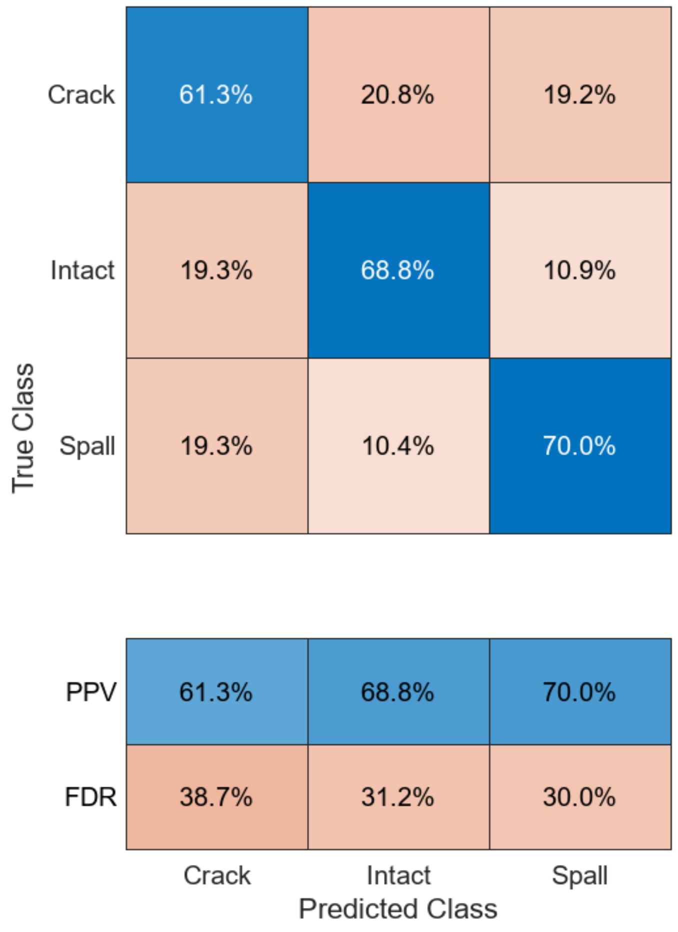


Figure A.3 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the fine decision tree model.

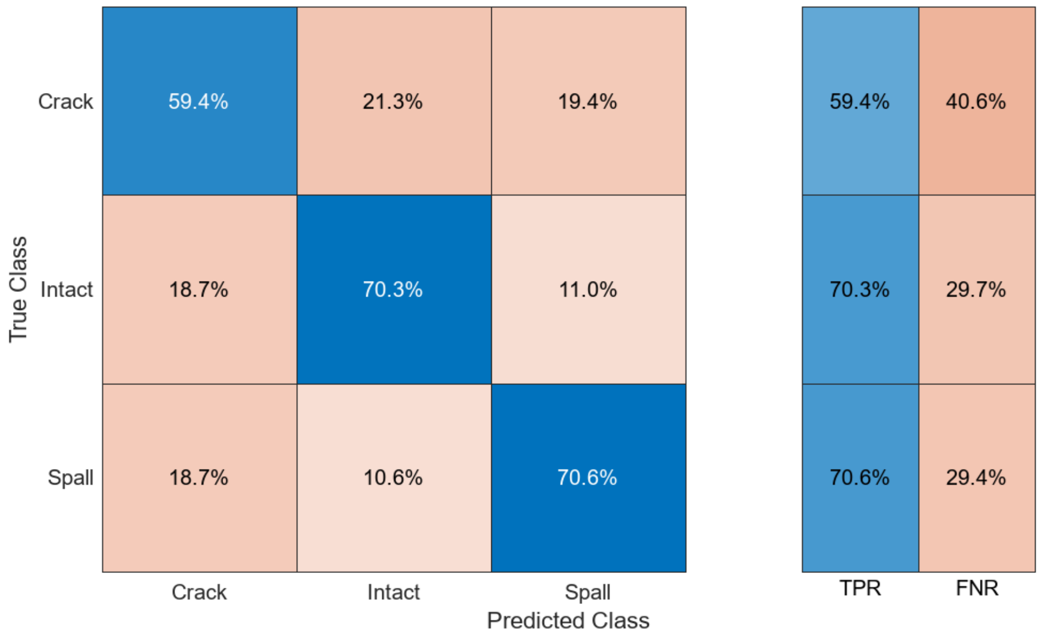


Figure A.4 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the fine decision tree model.

Medium decision tree model

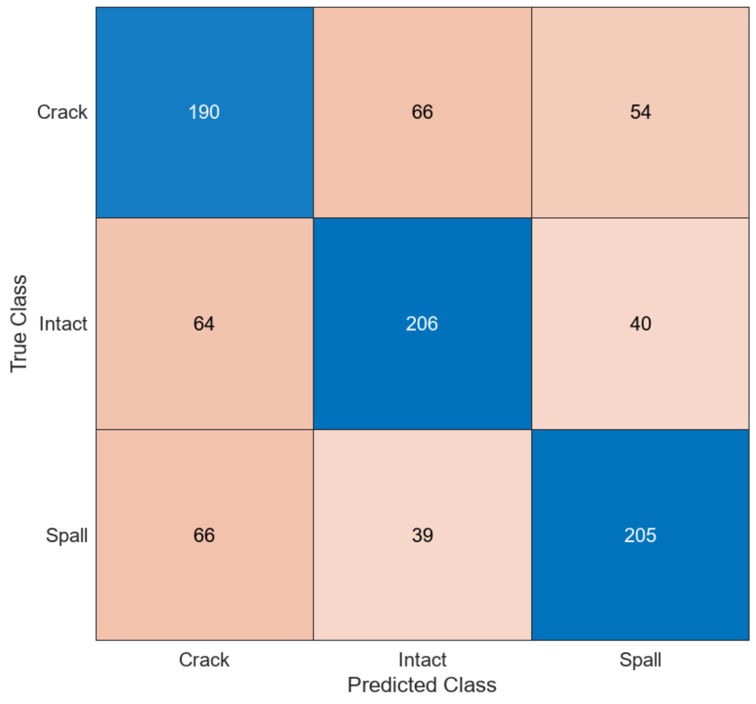


Figure A.5 Confusion matrix showing the classification results of the medium decision tree model.

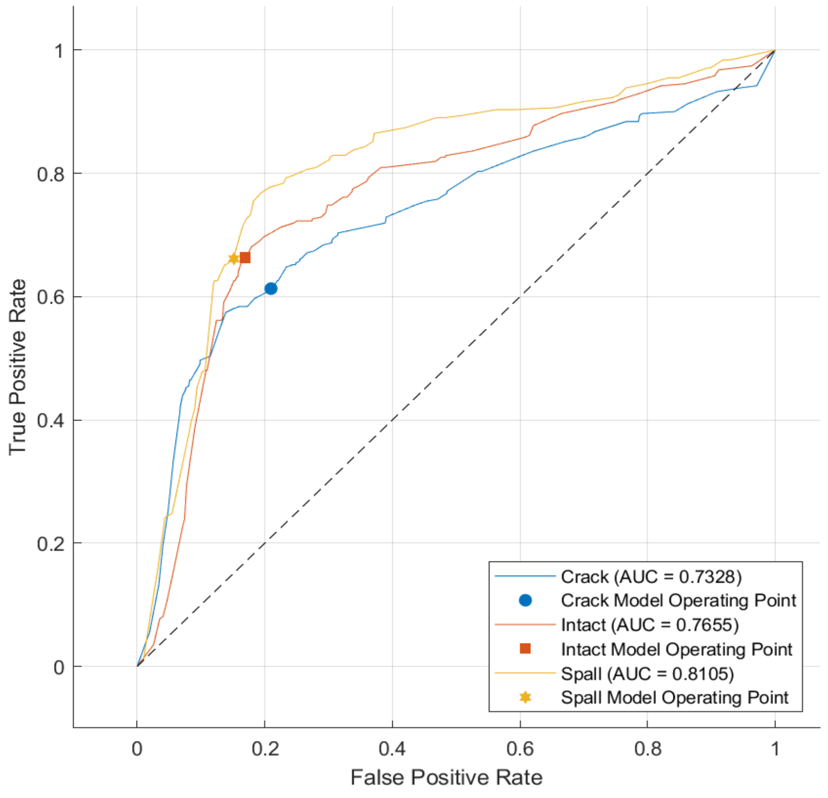


Figure A.6 ROC curve showing the medium decision tree model's performance.

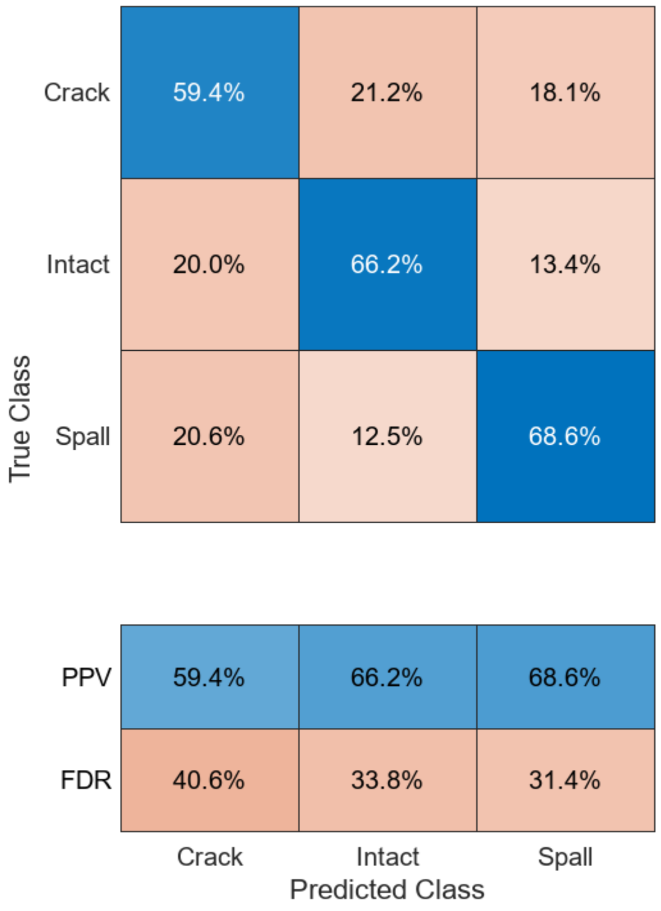


Figure A.7 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium decision tree model.

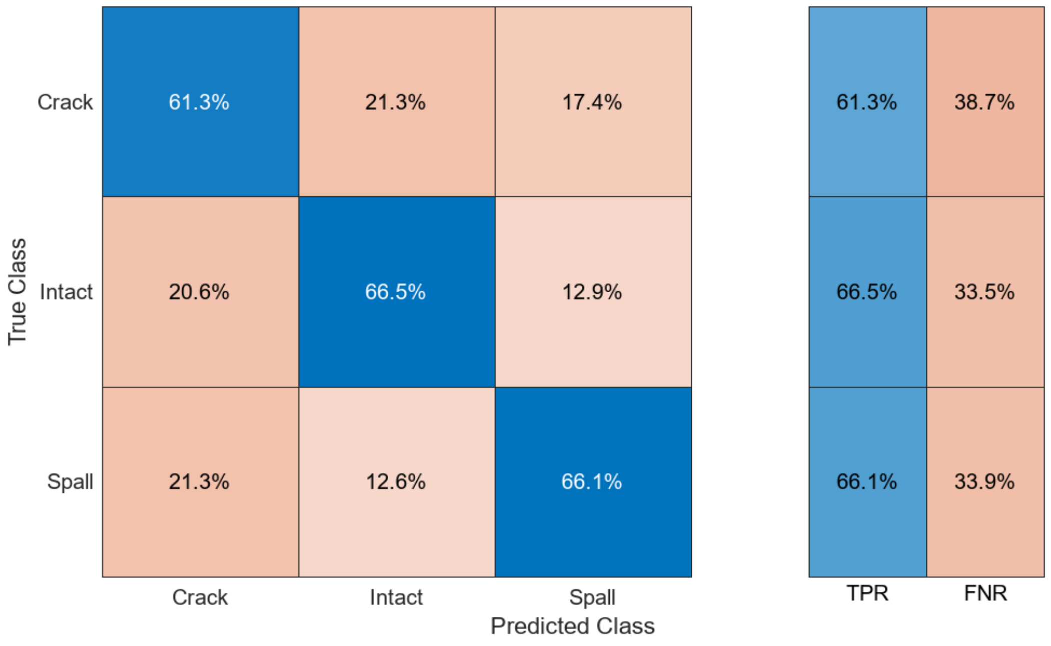


Figure A.8 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium decision tree model.

Coarse decision tree model

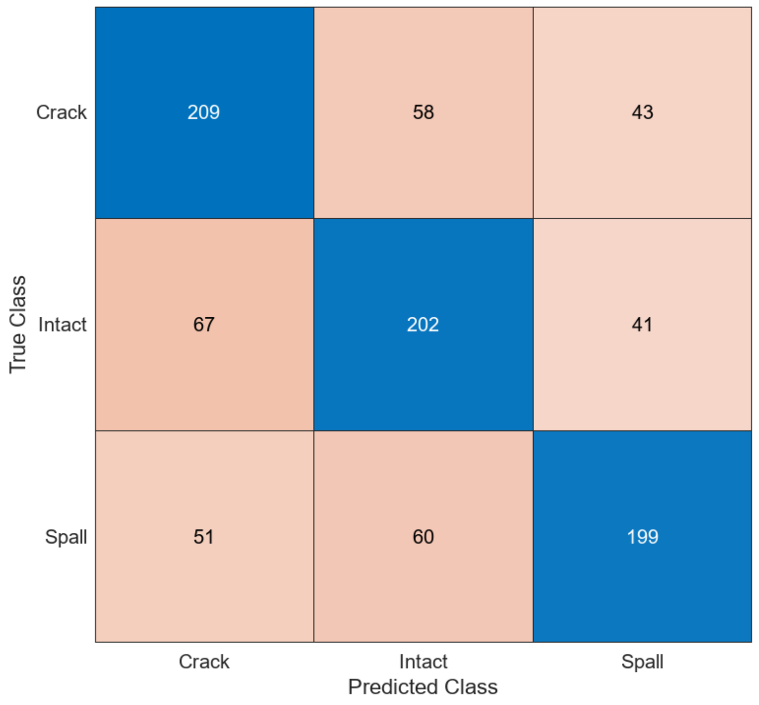


Figure A.9 Confusion matrix showing the classification results of the coarse decision tree model.

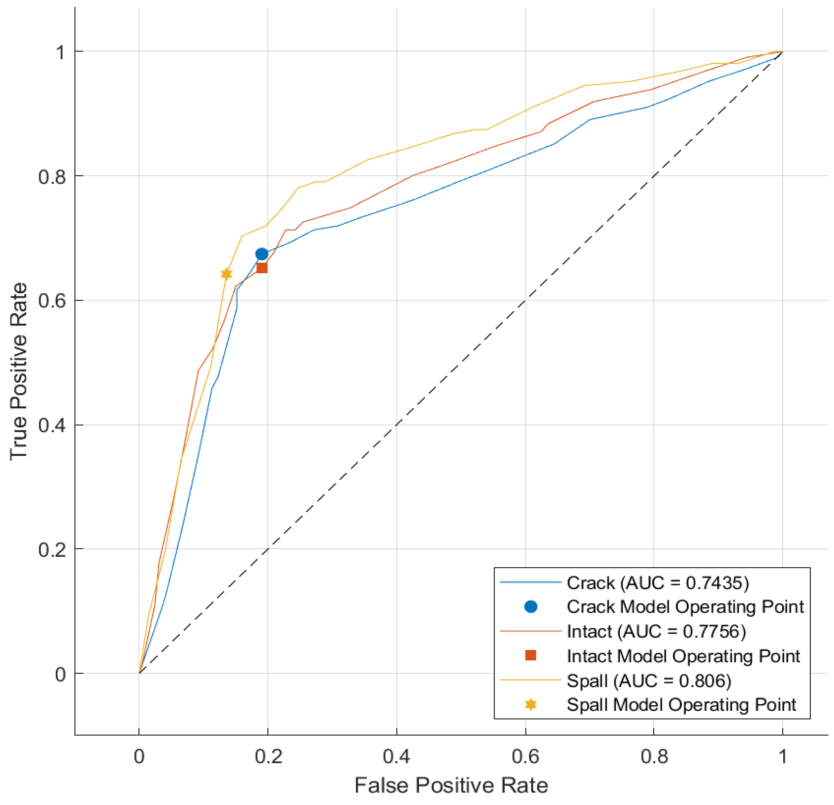


Figure A.10 ROC curve showing the coarse decision tree model's performance.

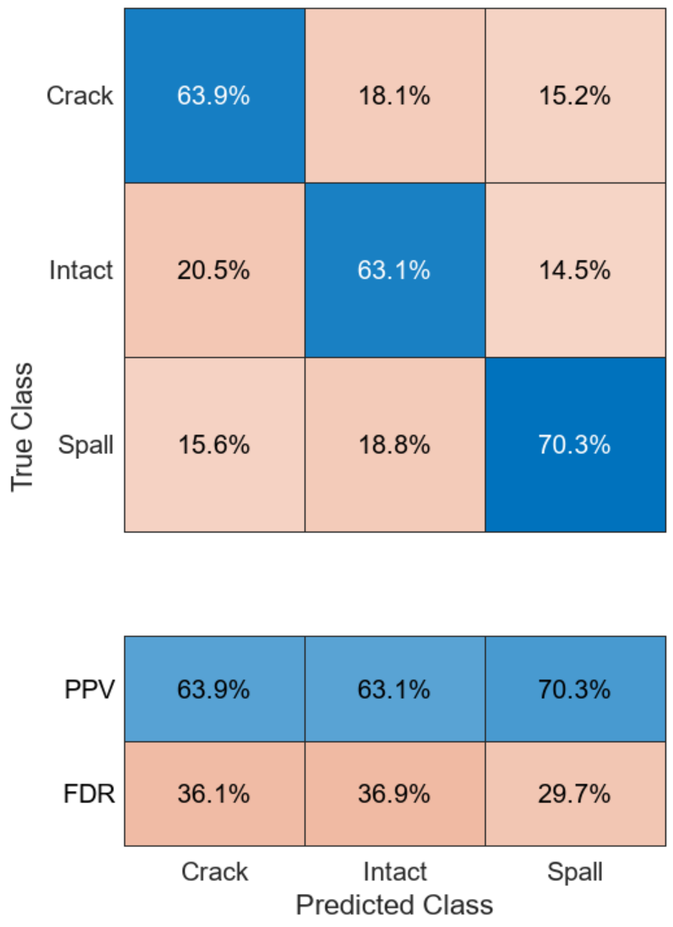


Figure A.11 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse decision tree model.

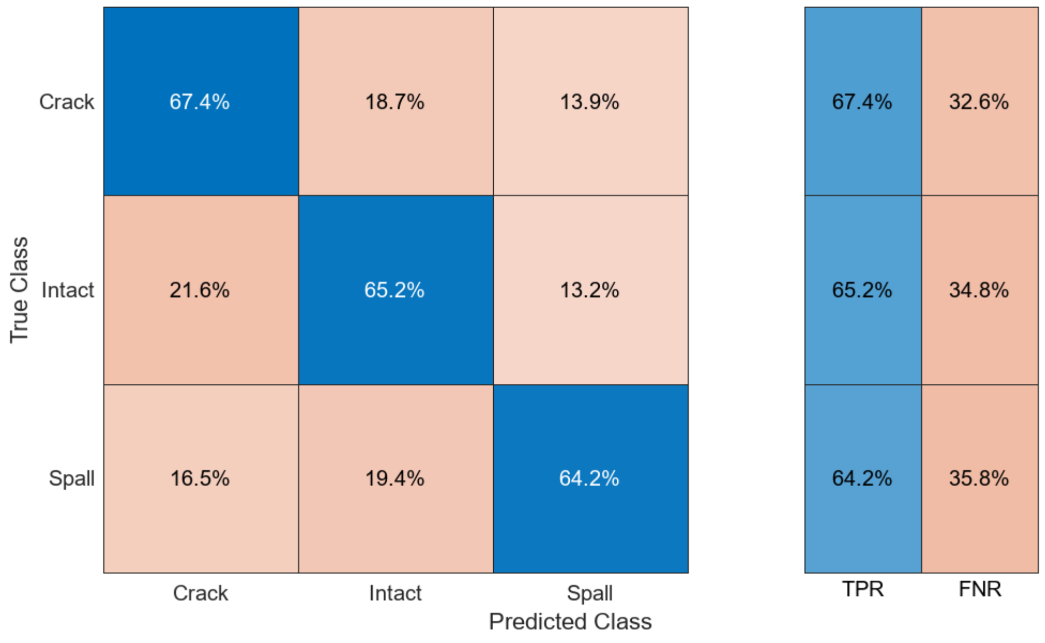


Figure A.12 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse decision tree model.

Linear SVM model

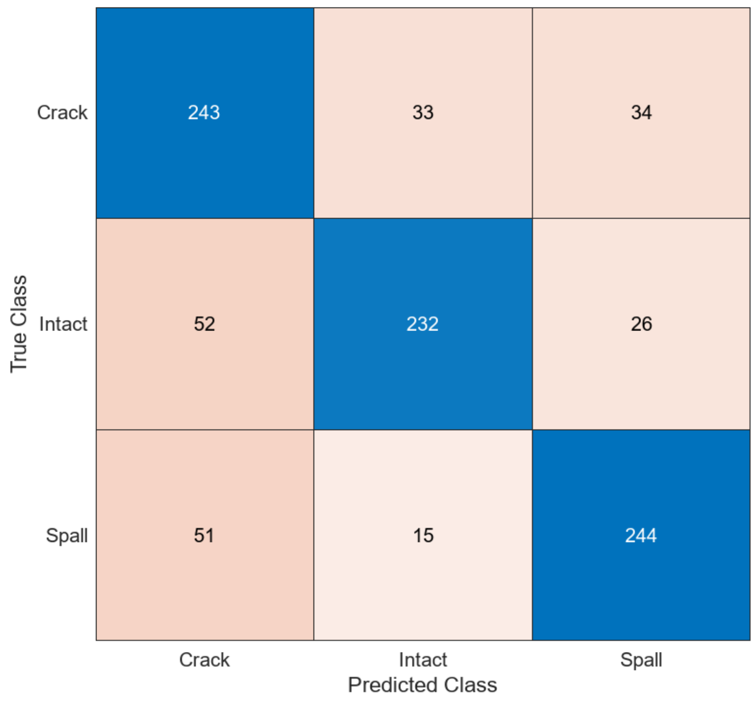


Figure A.13 Confusion matrix showing the classification results of the linear SVM model.

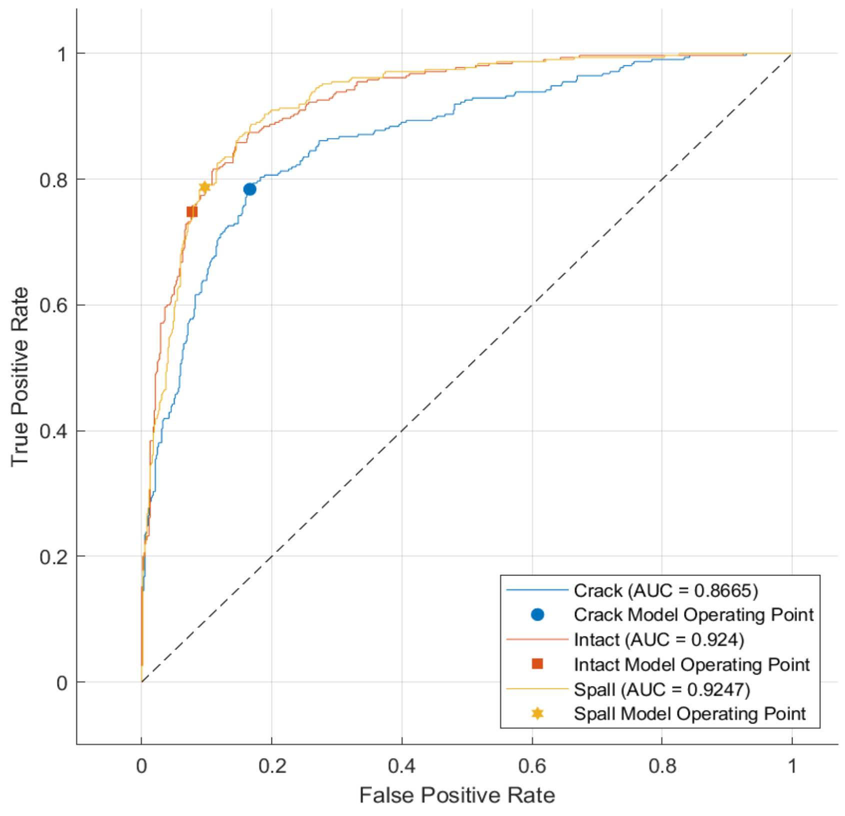


Figure A.14 ROC curve showing the linear SVM model's performance.

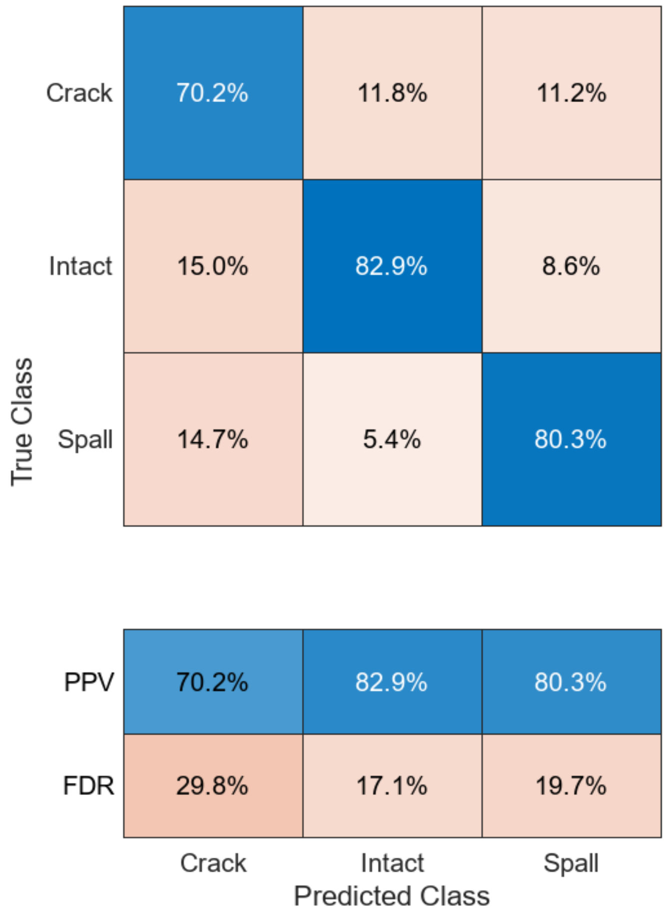


Figure A.15 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the linear SVM model.

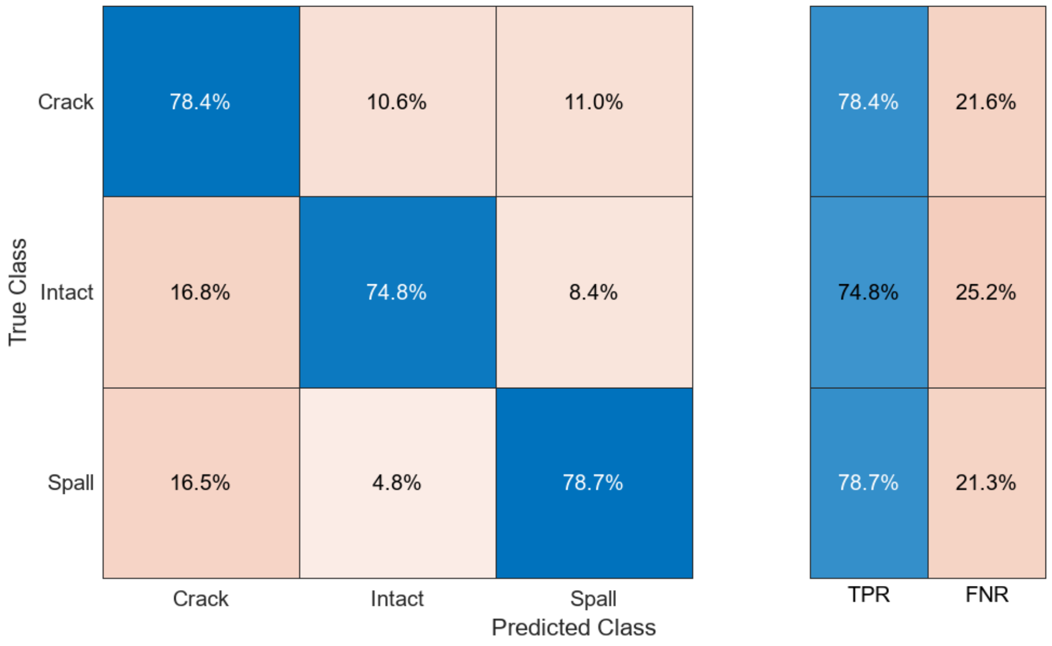


Figure A.16 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the linear SVM model.

Quadratic SVM model

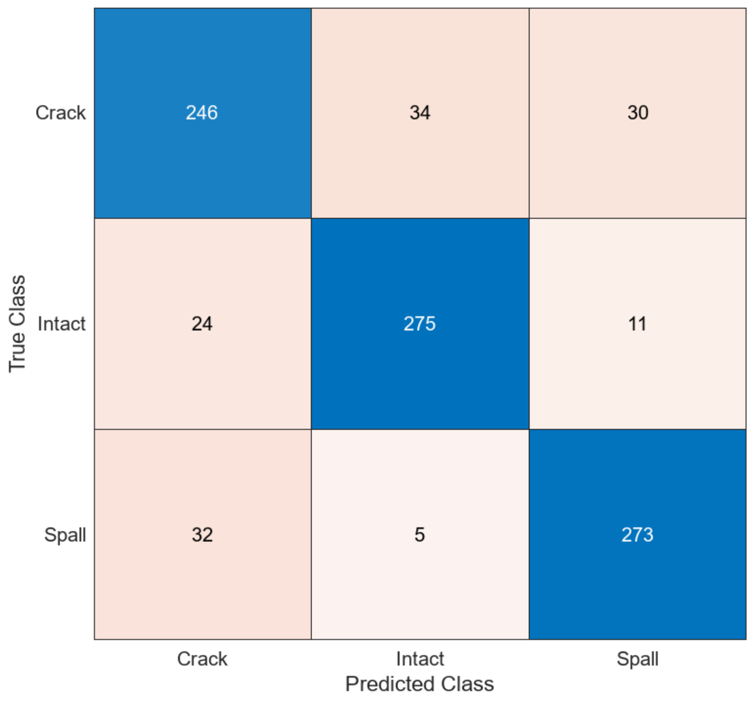


Figure A.17 Confusion matrix showing the classification results of the quadratic SVM model.

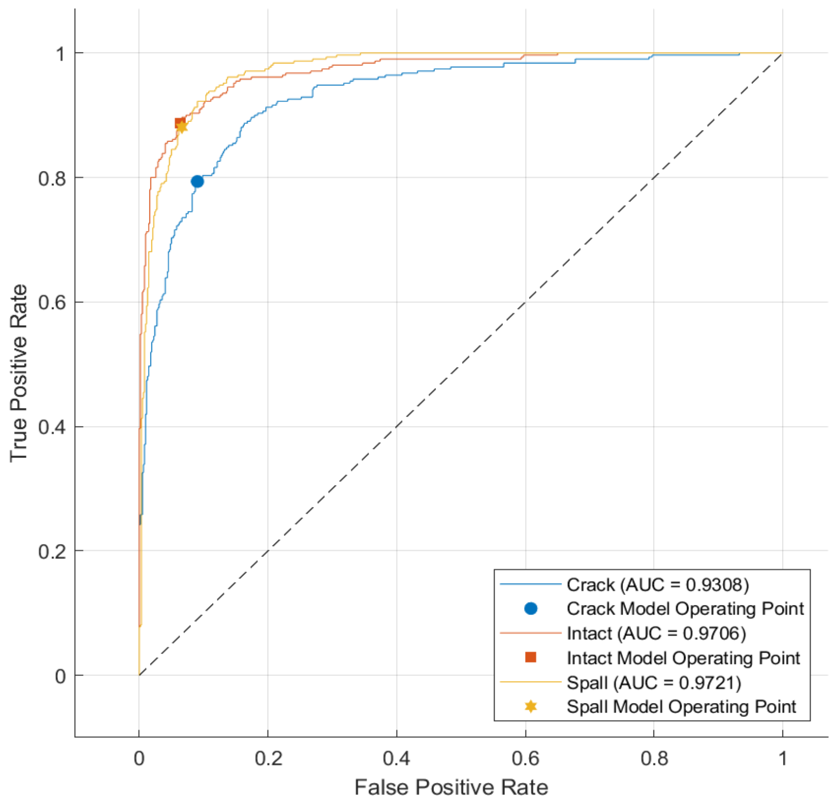


Figure A.18 ROC curve showing the quadratic SVM model's performance.

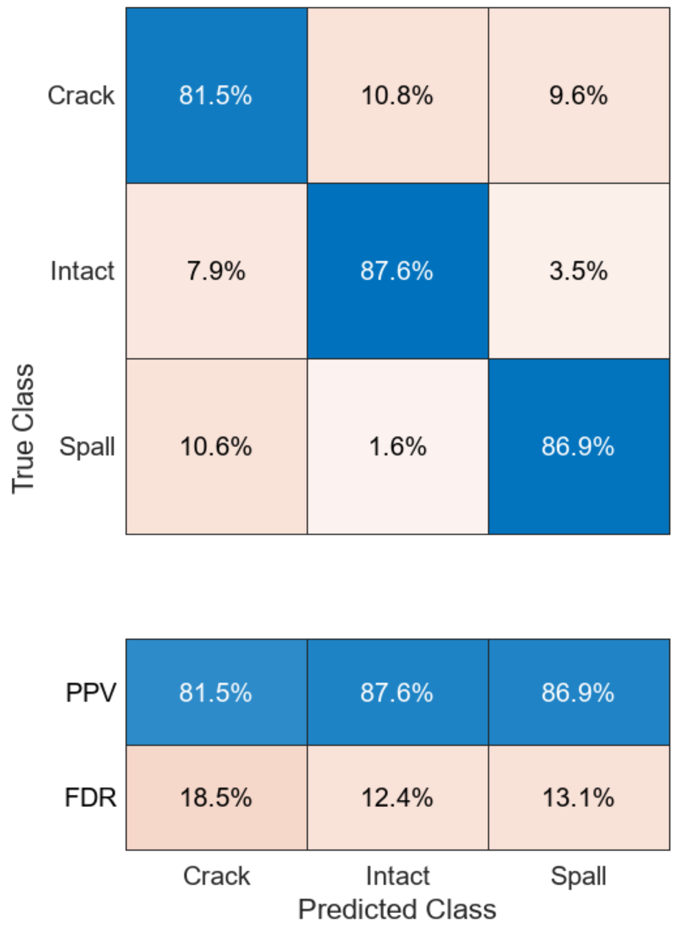


Figure A.19 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the quadratic SVM model.

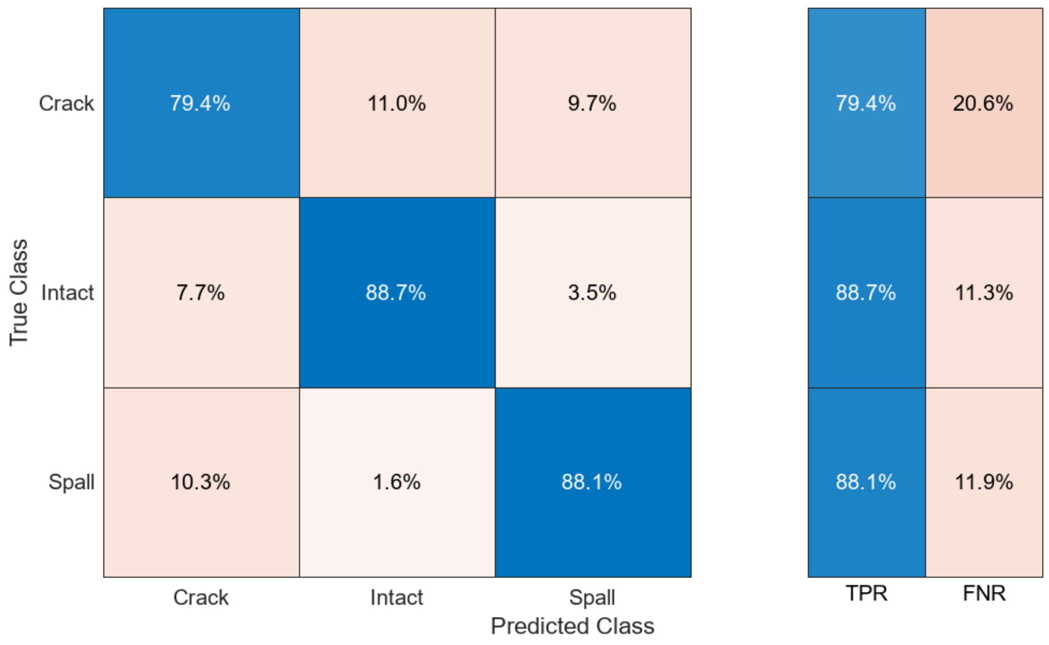


Figure A.20 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the quadratic SVM model.

Cubic SVM model

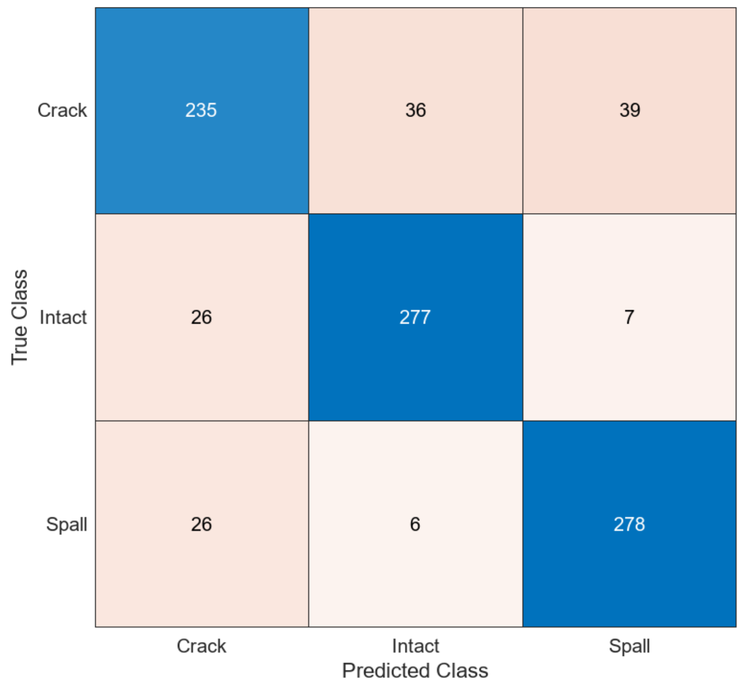


Figure A.21 Confusion matrix showing the classification results of the cubic SVM model.

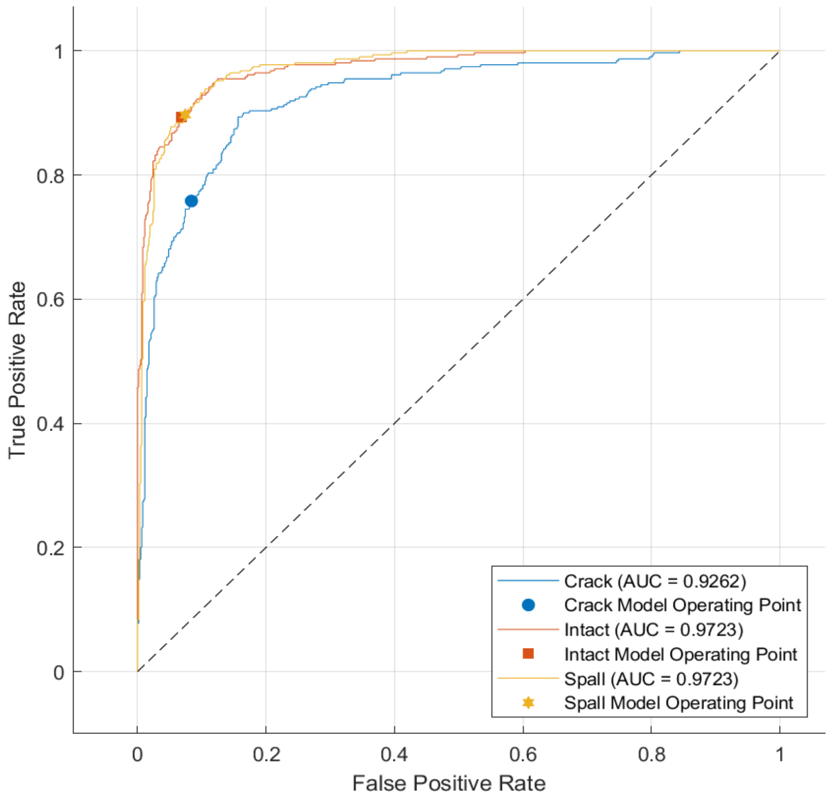


Figure A.22 ROC curve showing the cubic SVM model's performance.

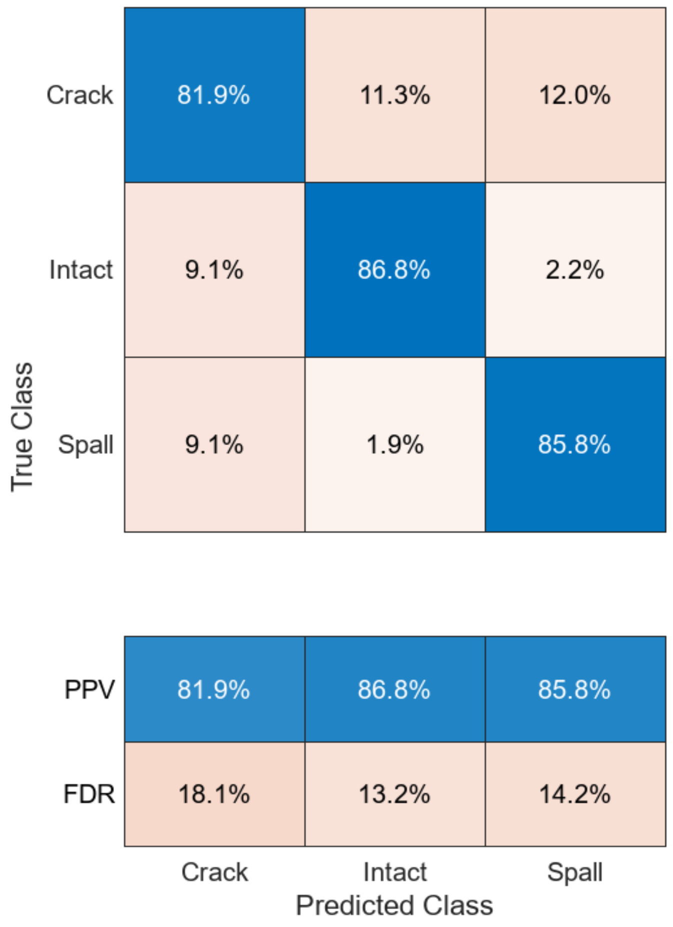


Figure A.23 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the cubic SVM model.

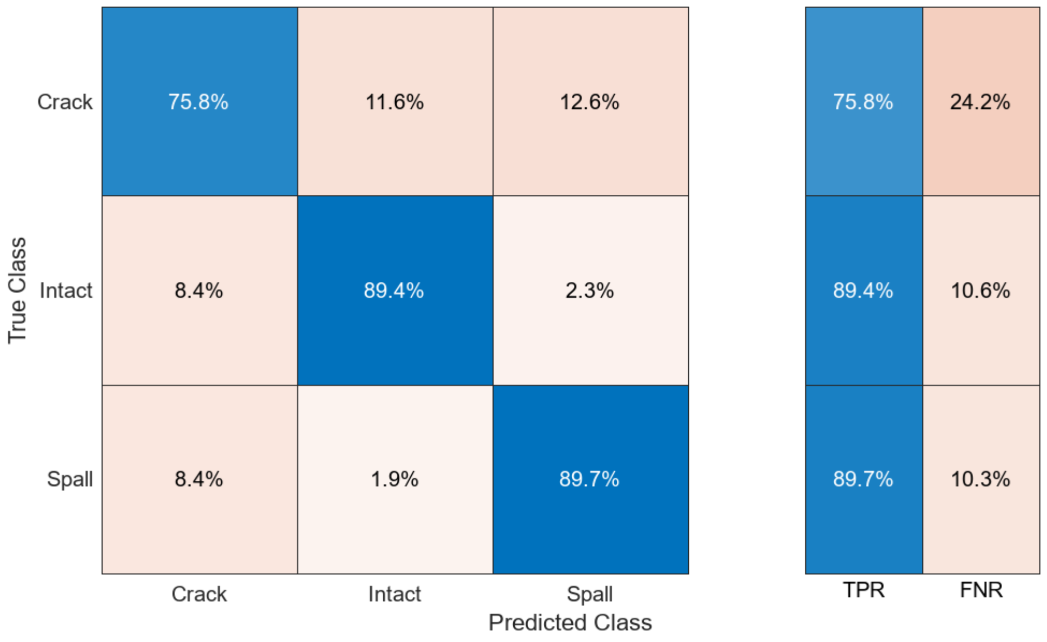


Figure A.24 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the cubic SVM model.

Fine Gaussian SVM model

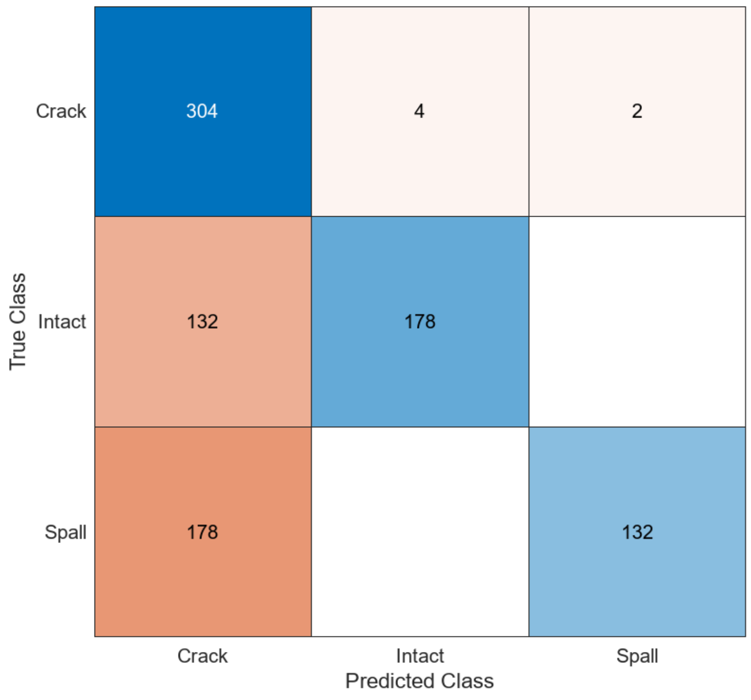


Figure A.25 Confusion matrix showing the classification results of the fine Gaussian SVM model.

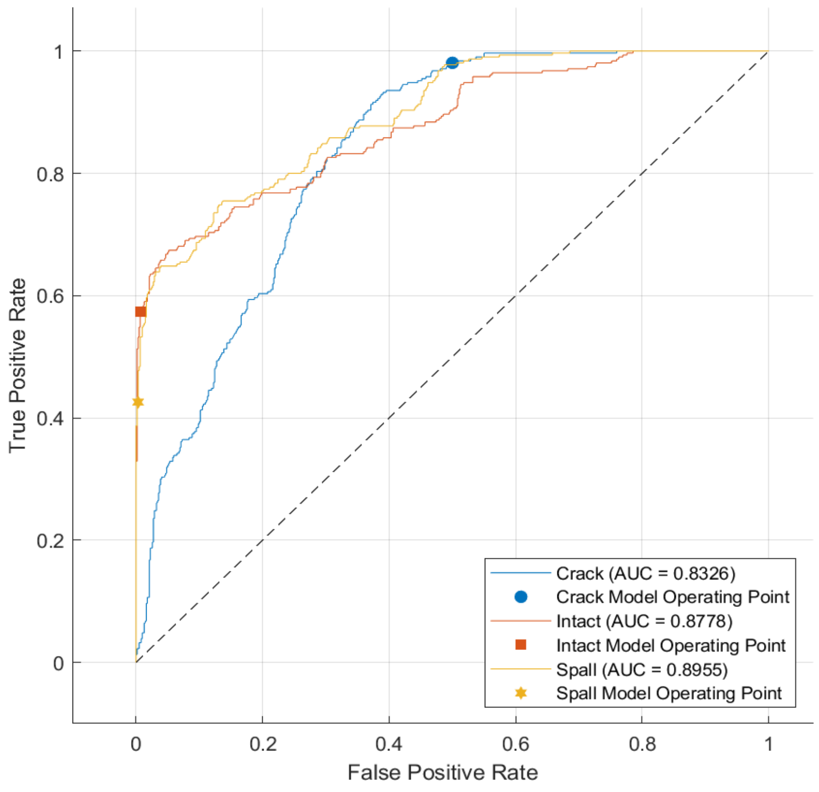


Figure A.26 ROC curve showing the fine Gaussian SVM model’s performance.

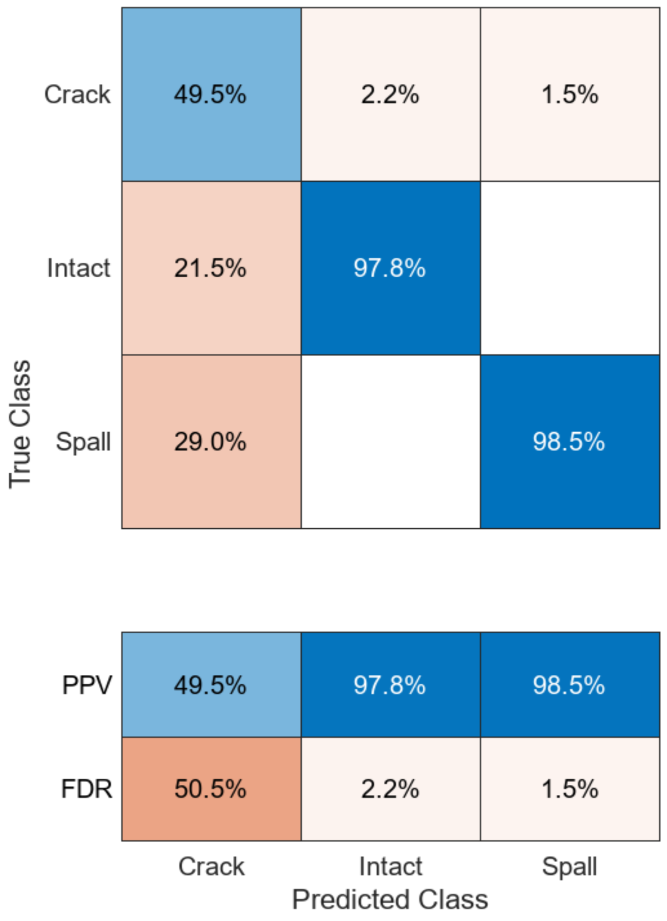


Figure A.27 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the fine Gaussian SVM model.

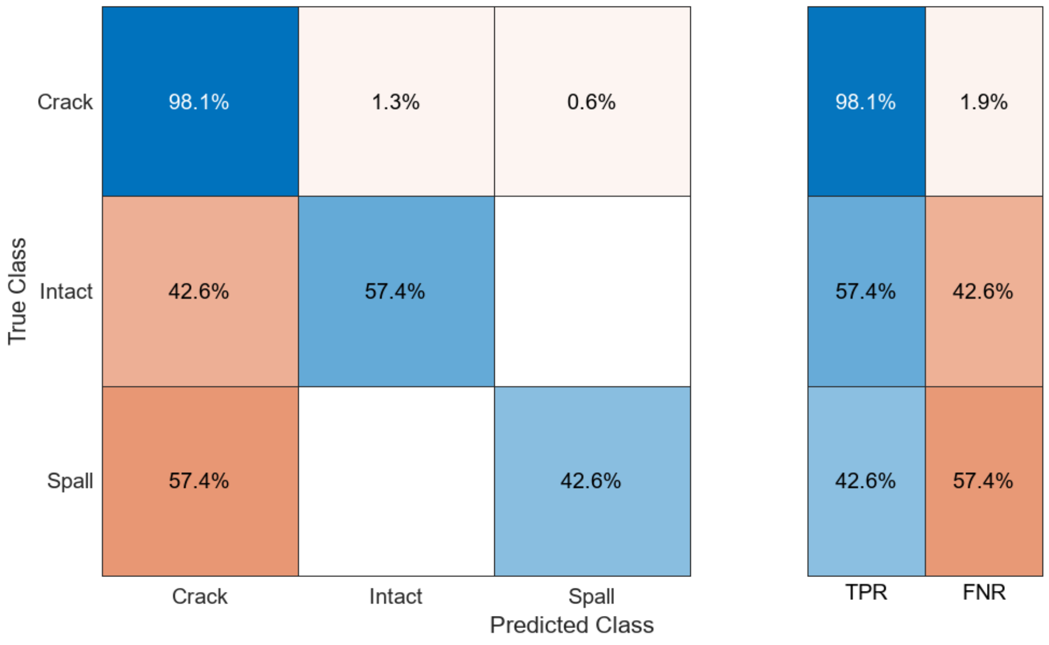


Figure A.28 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the fine Gaussian SVM model.

Medium Gaussian SVM model

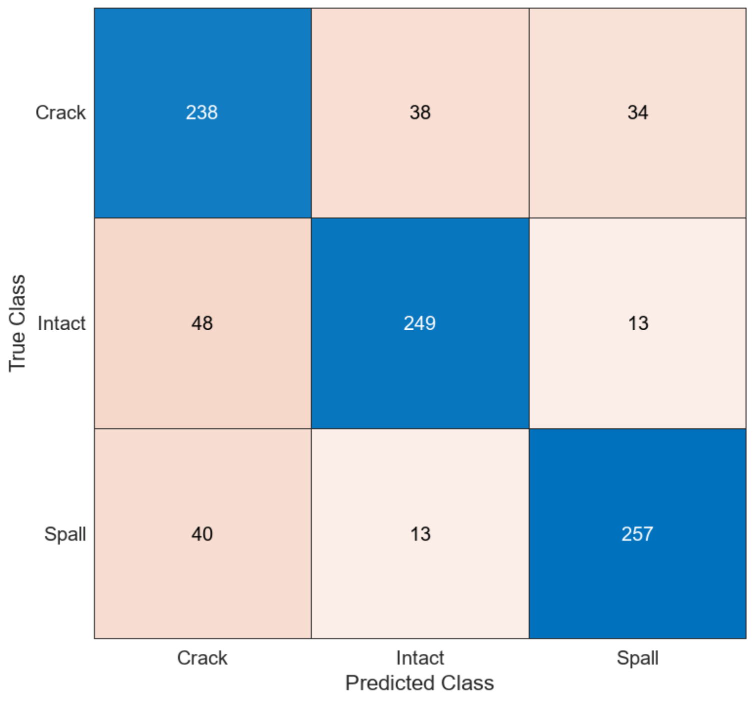


Figure A.29 Confusion matrix showing the classification results of the medium Gaussian SVM model.

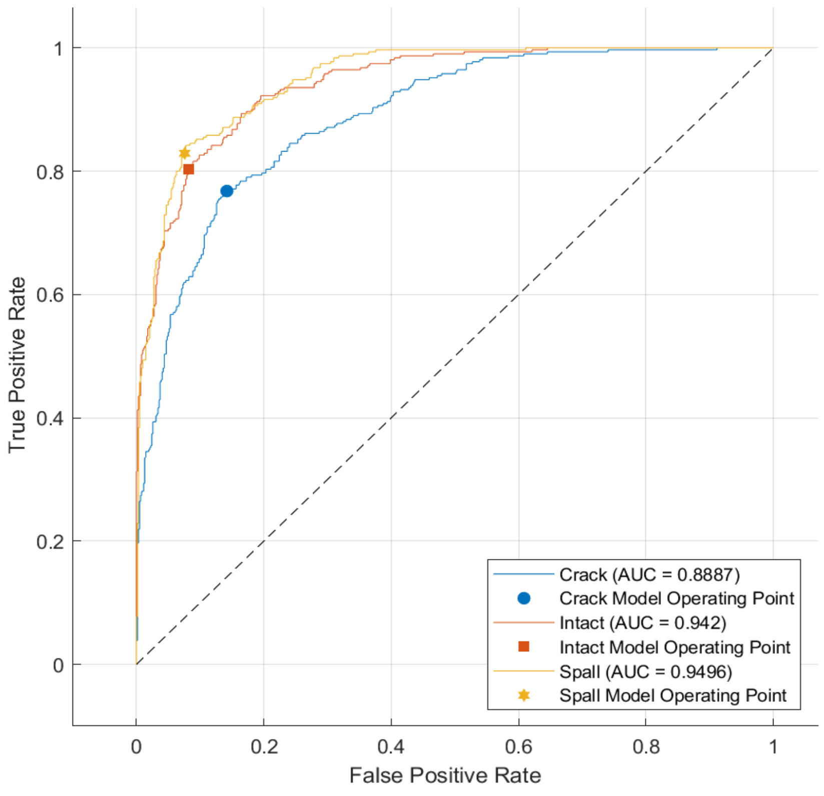


Figure A.30 ROC curve showing the medium Gaussian SVM model’s performance.

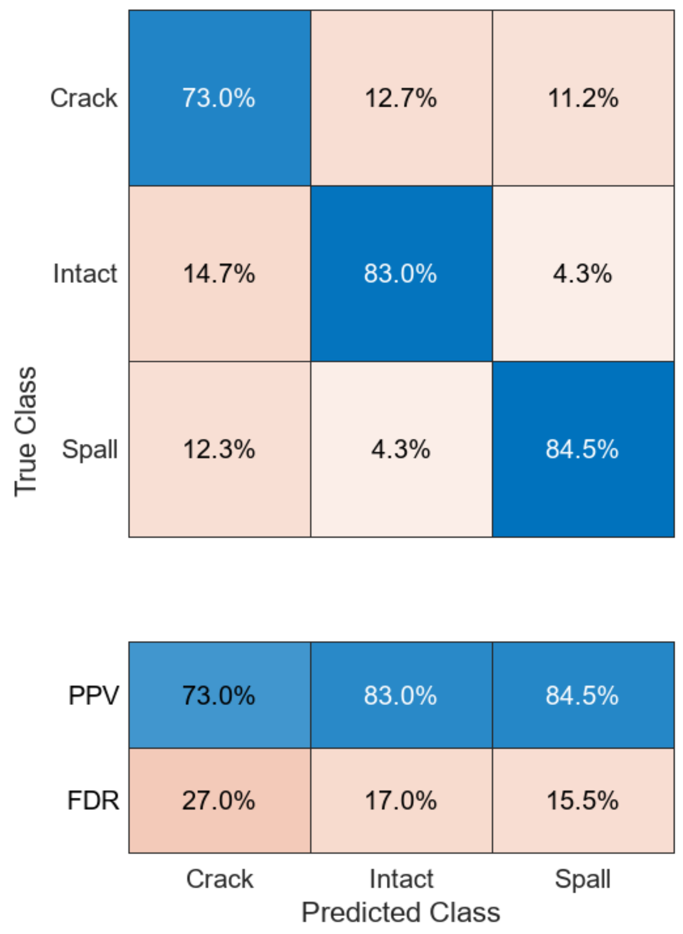


Figure A.31 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium Gaussian SVM model.

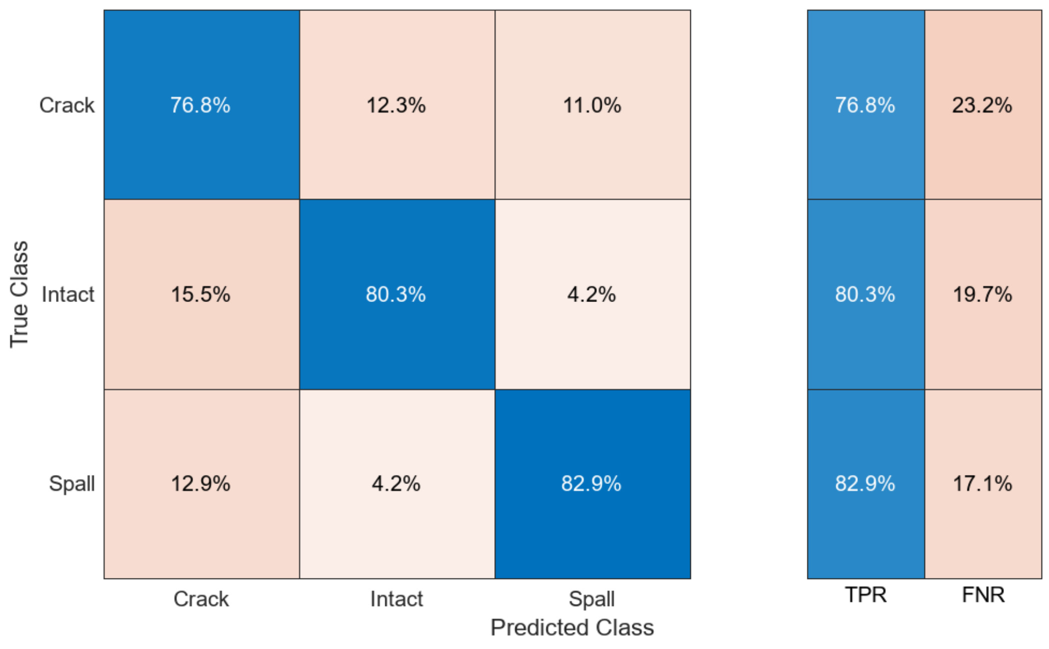


Figure A.32 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium Gaussian SVM model.

Coarse Gaussian SVM model

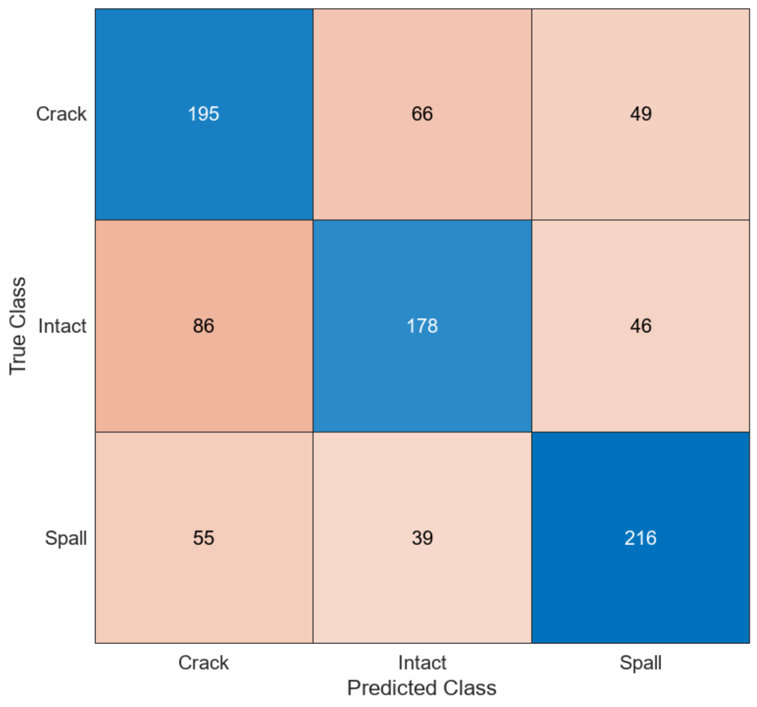


Figure A.33 Confusion matrix showing the classification results of the coarse Gaussian SVM model.

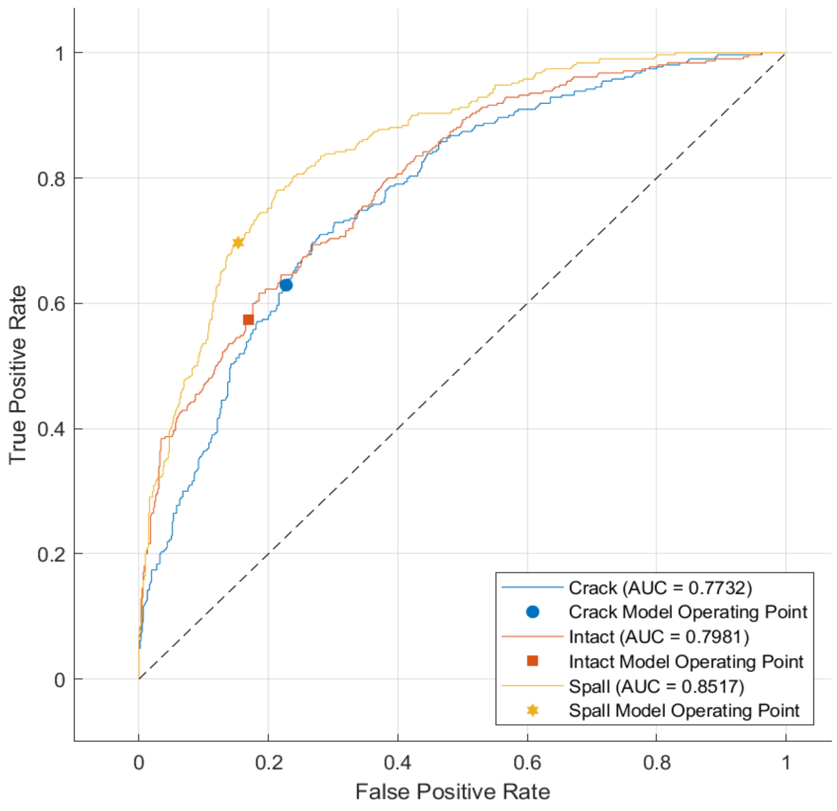


Figure A.34 ROC curve showing the coarse Gaussian SVM model's performance.

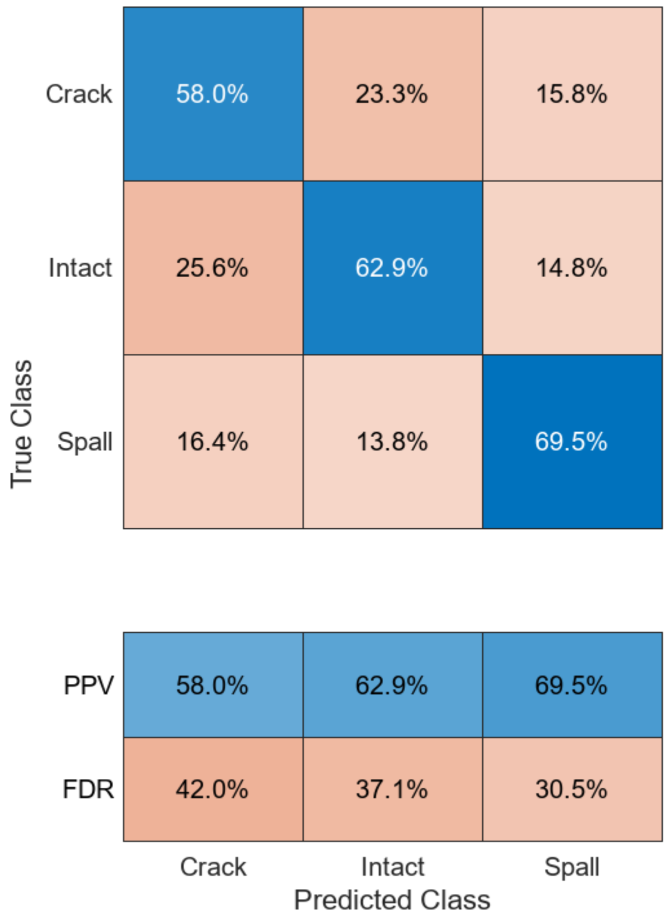


Figure A.35 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse Gaussian SVM model.

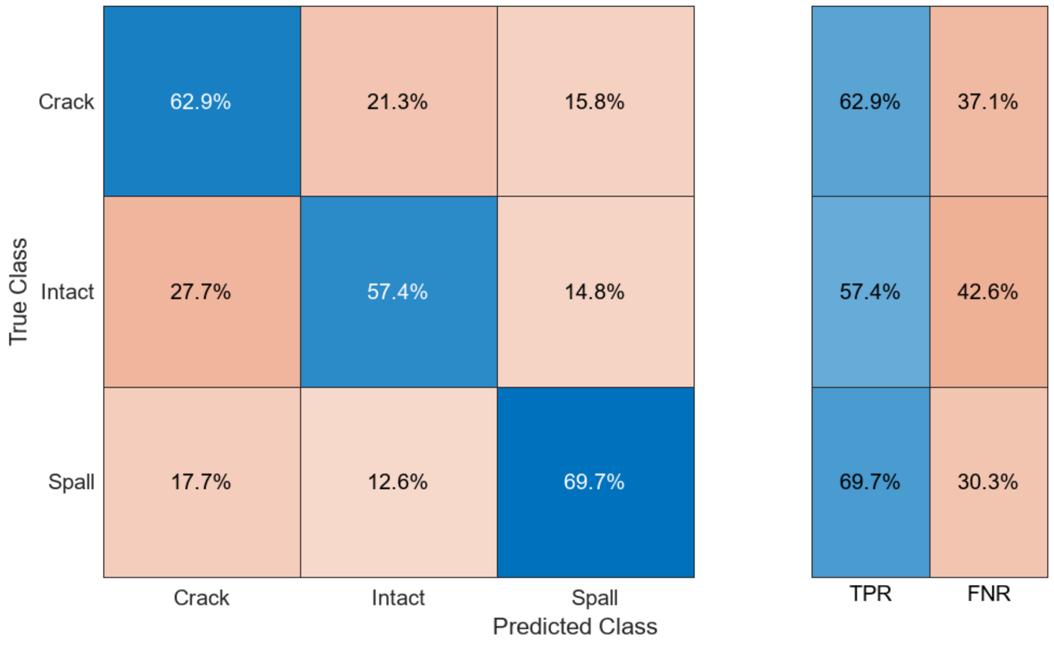


Figure A.36 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse Gaussian SVM model.

Fine KNN model

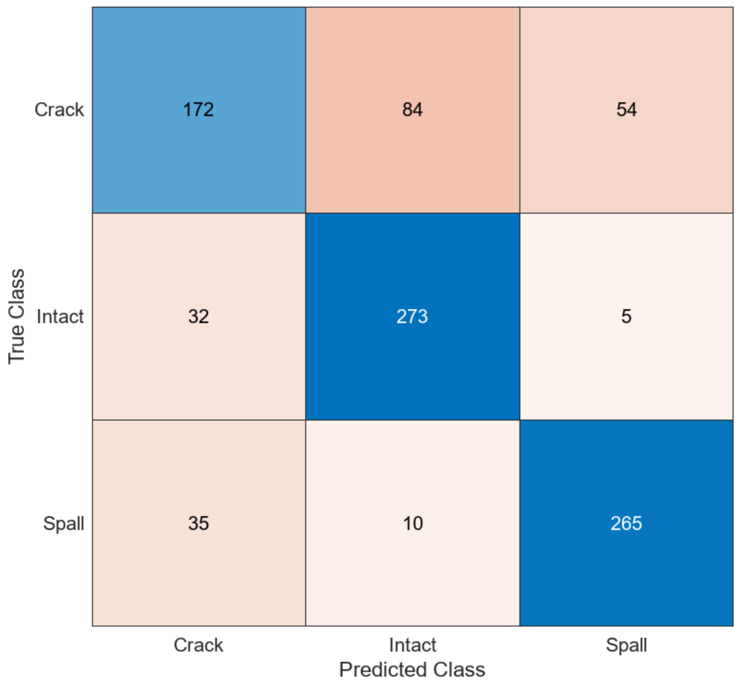


Figure A.37 Confusion matrix showing the classification results of the fine KNN model.

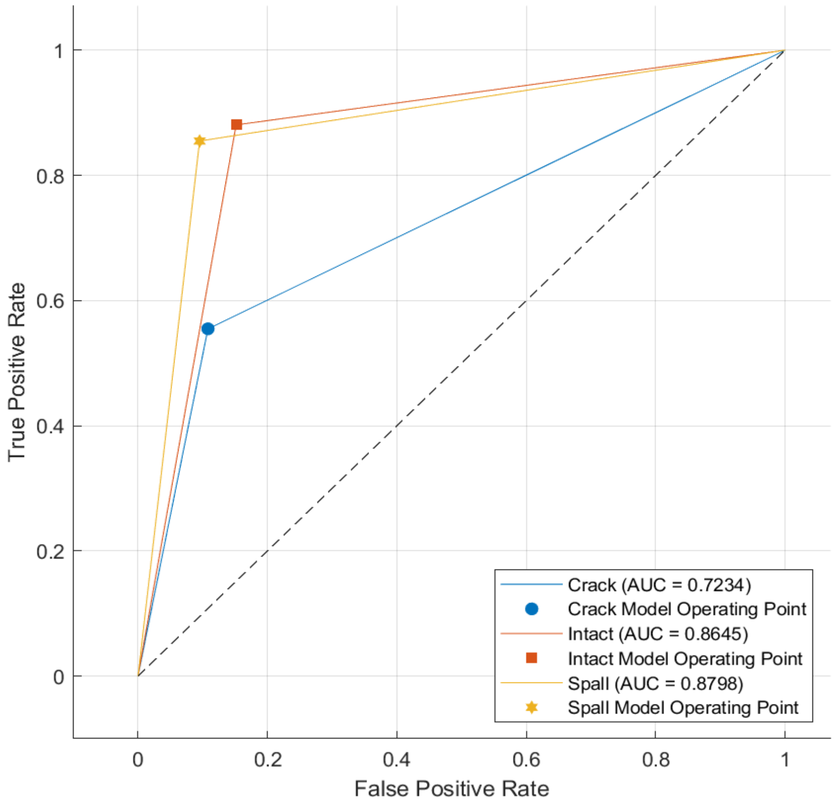


Figure A.38 ROC curve showing the fine KNN model's performance.

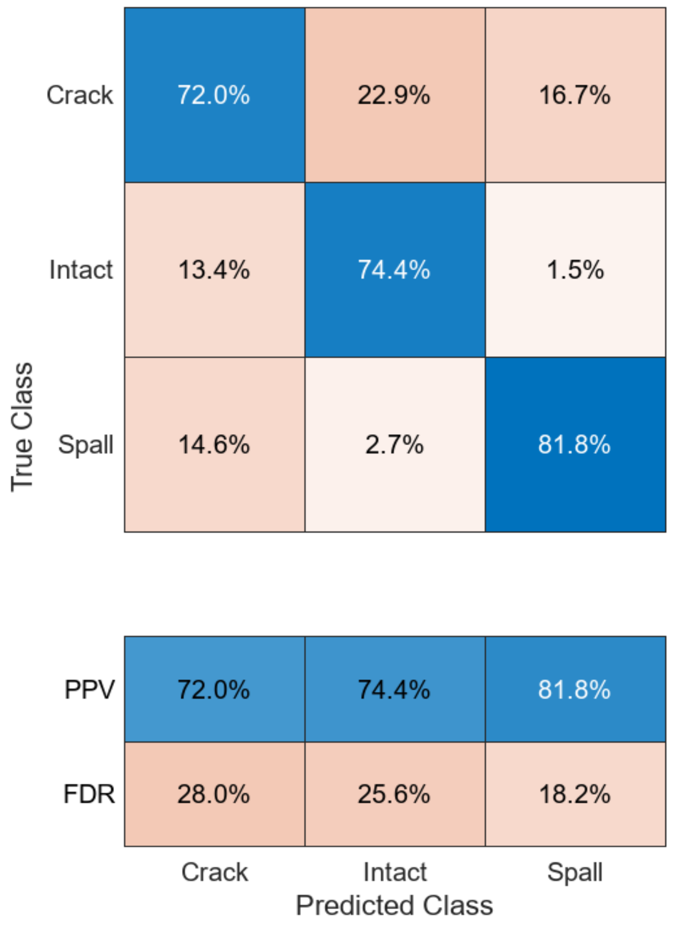


Figure A.39 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the fine KNN model.

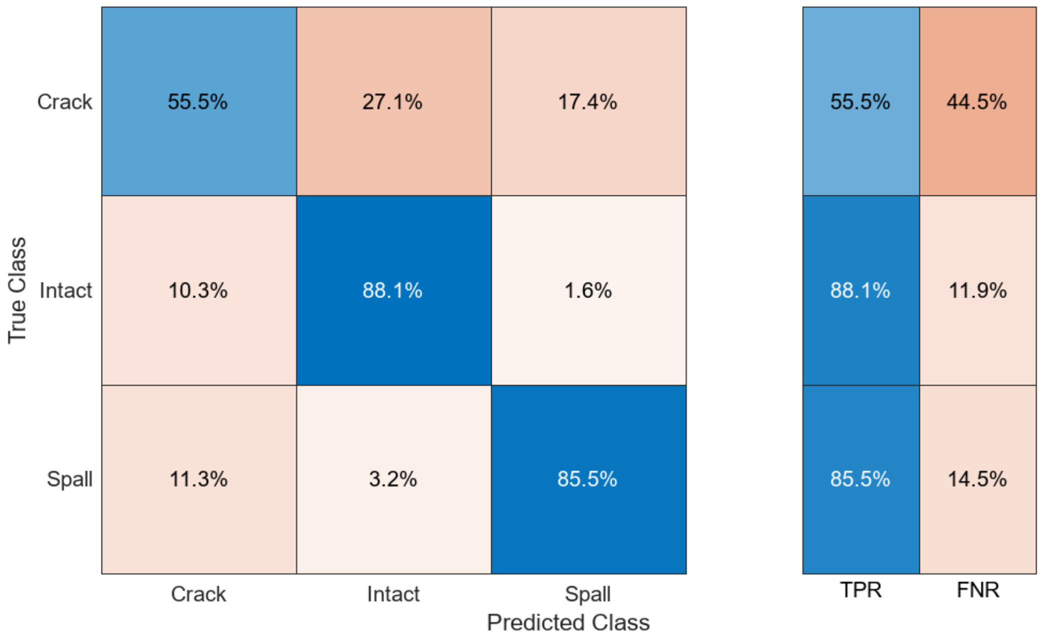


Figure A.40 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the fine KNN model.

Medium KNN model

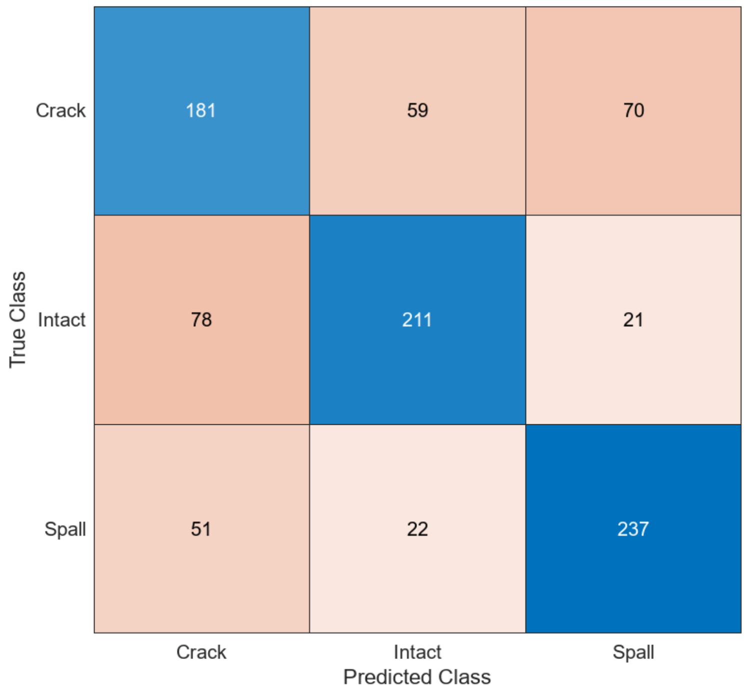


Figure A.41 Confusion matrix showing the classification results of the medium KNN model.

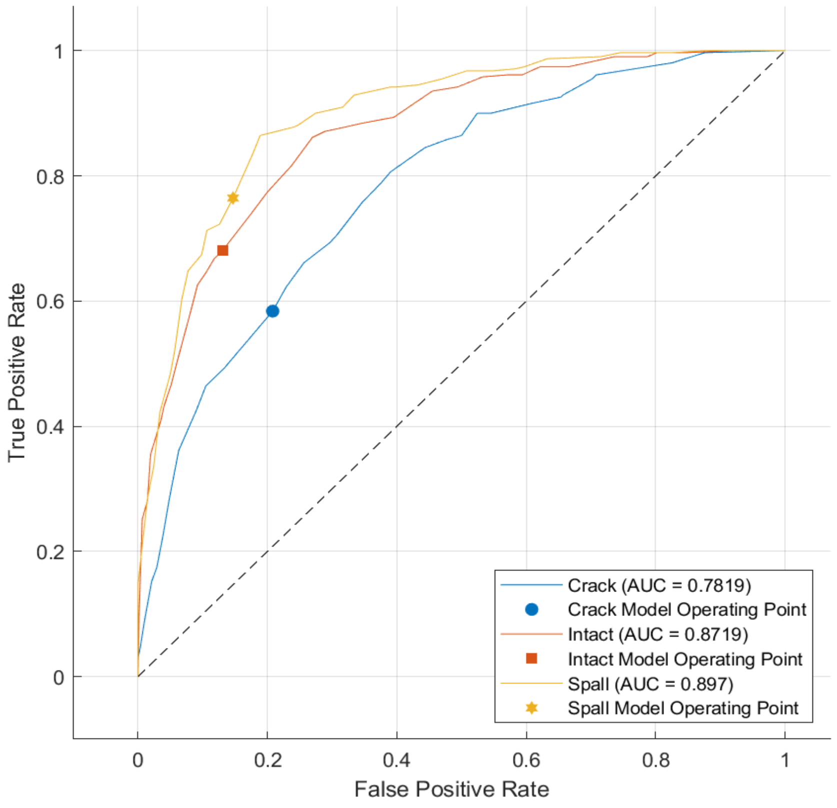


Figure A.42 ROC curve showing the medium KNN model’s performance.

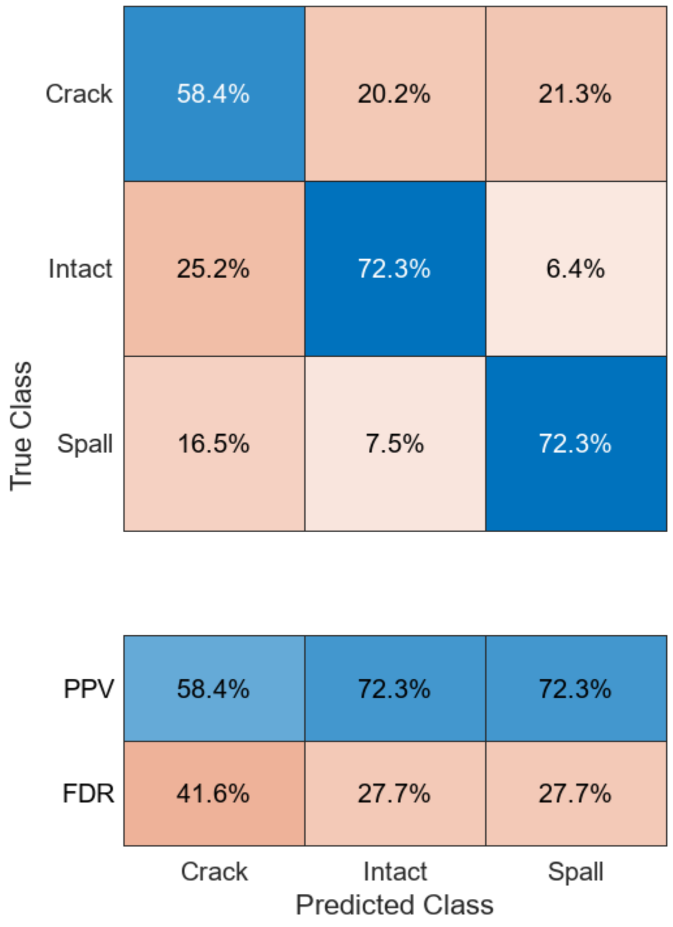


Figure A.43 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium KNN model.

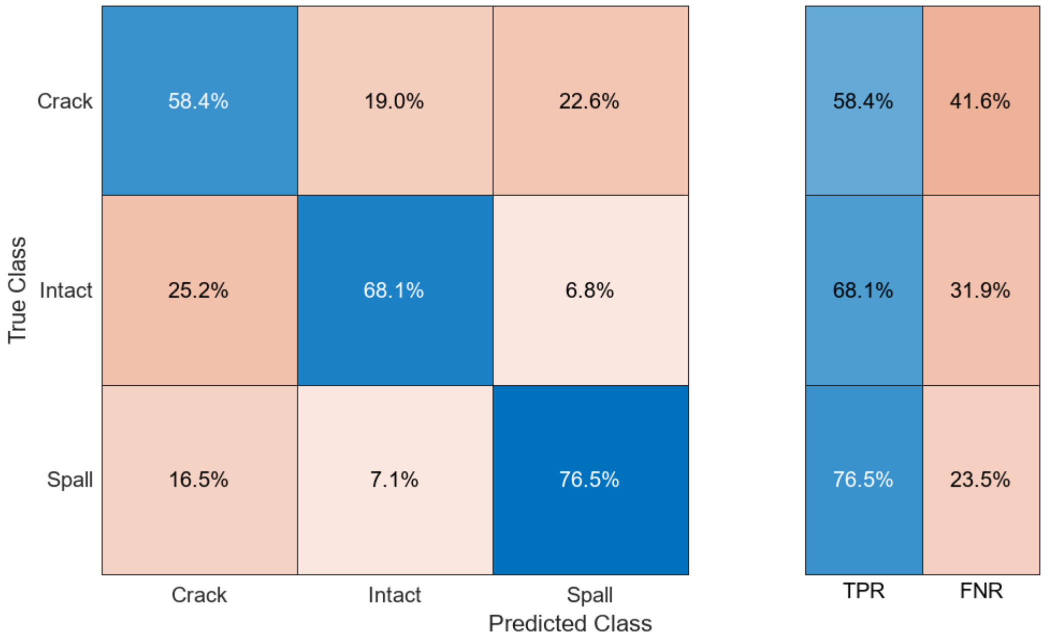


Figure A.44 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium KNN model.

Coarse KNN model

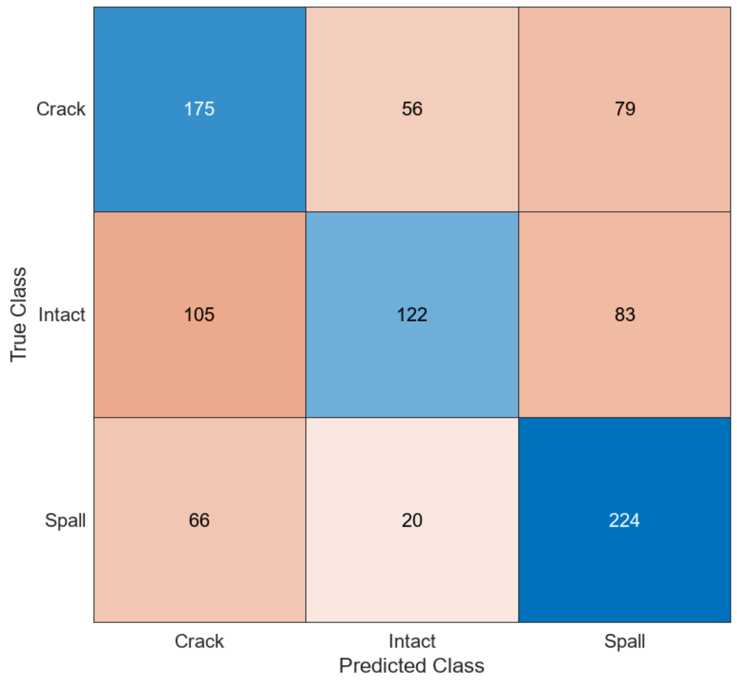


Figure A.45 Confusion matrix showing the classification results of the coarse KNN model.

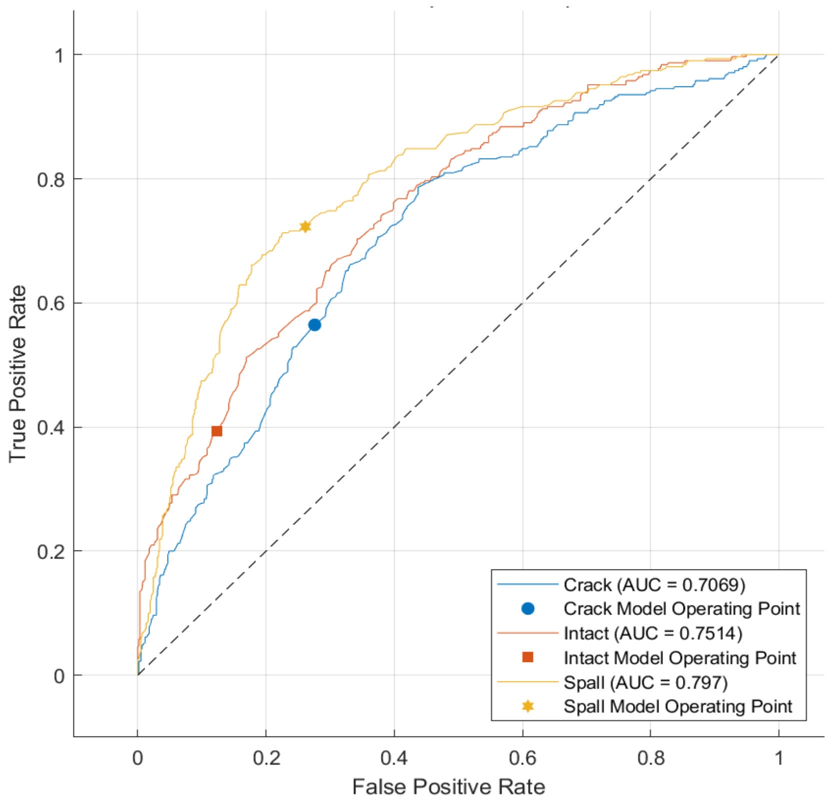


Figure A.46 ROC curve showing the coarse KNN model's performance.

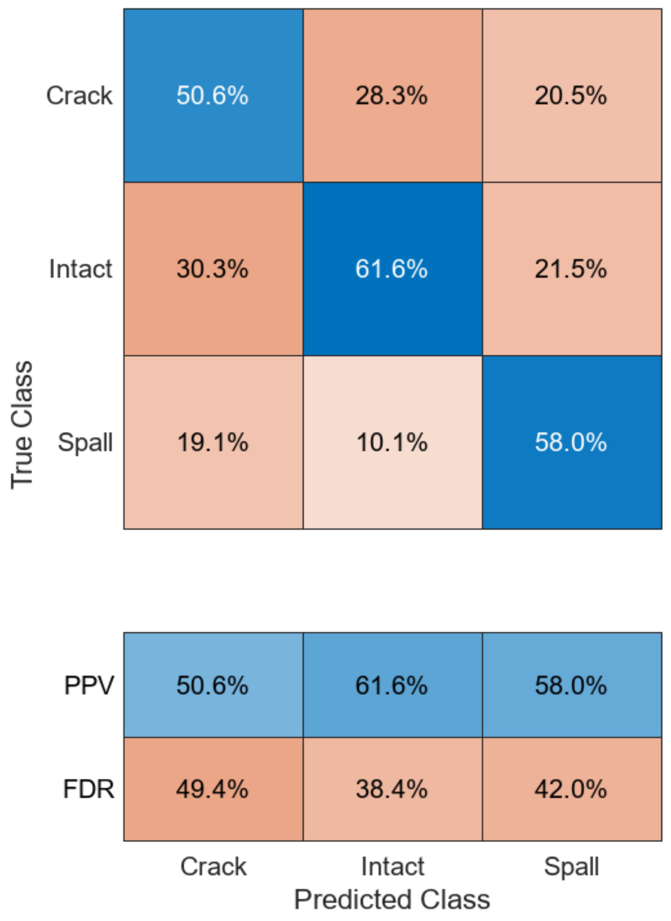


Figure A.47 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse KNN model.

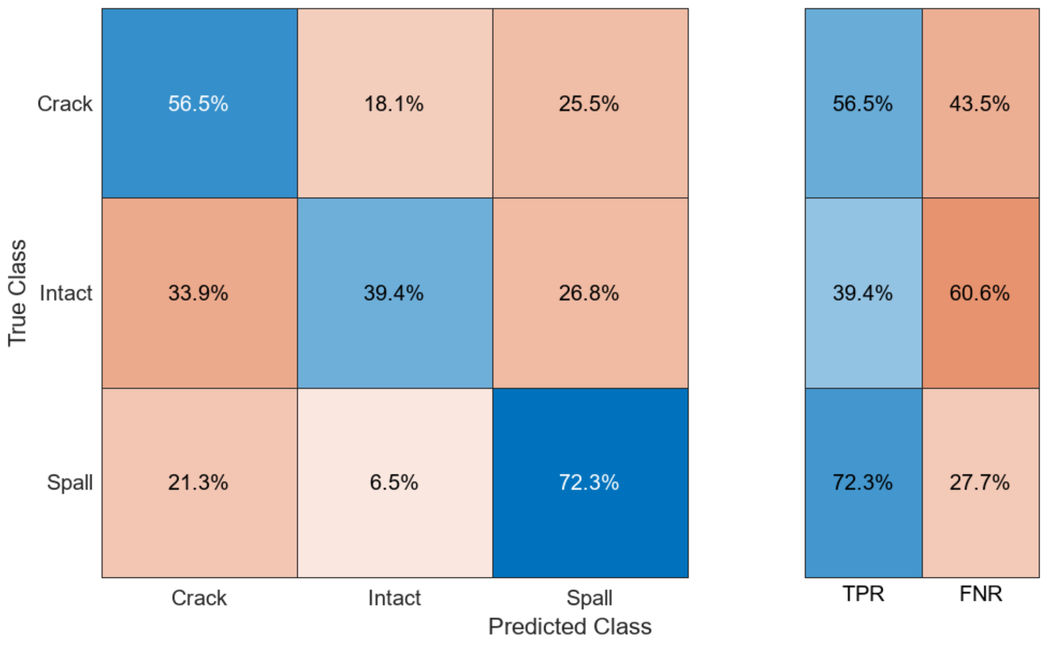


Figure A.48 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse KNN model.

Cosine KNN model

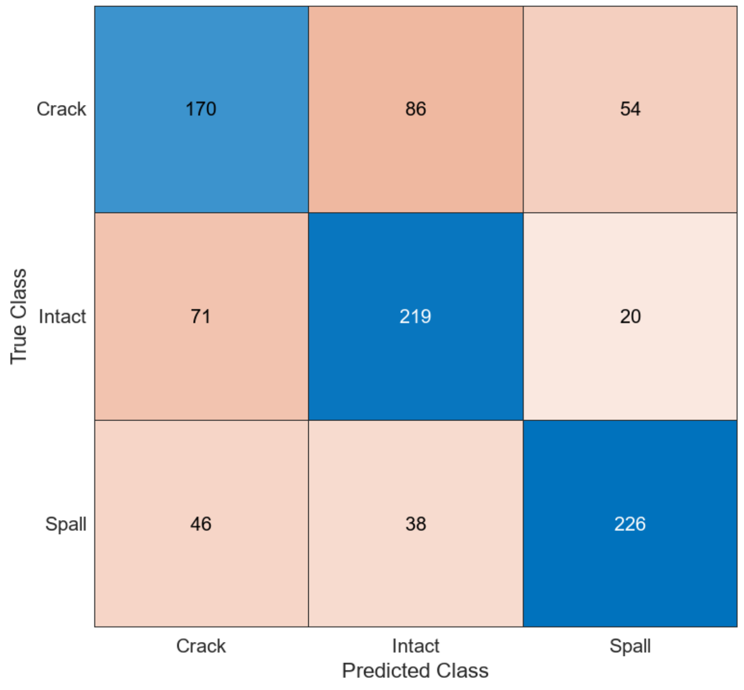


Figure A.49 Confusion matrix showing the classification results of the cosine KNN model.

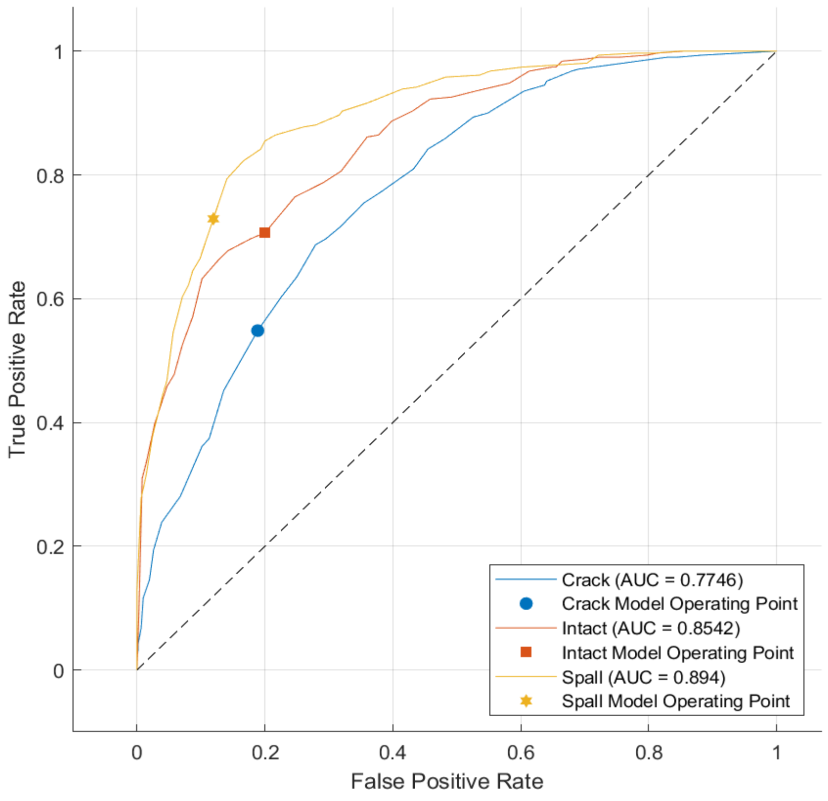


Figure A.50 ROC curve showing the cosine KNN model’s performance.

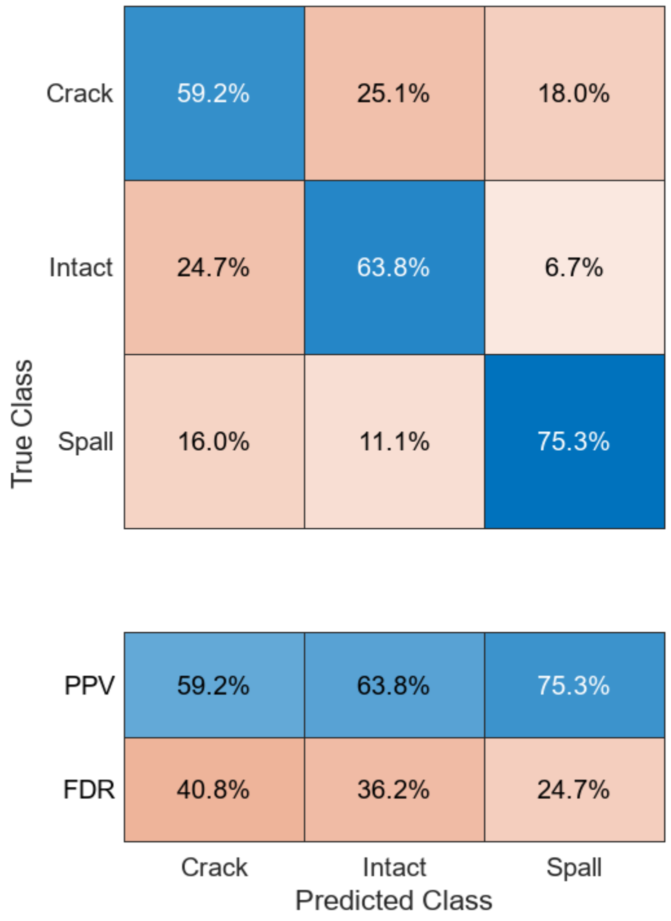


Figure A.51 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the cosine KNN model.

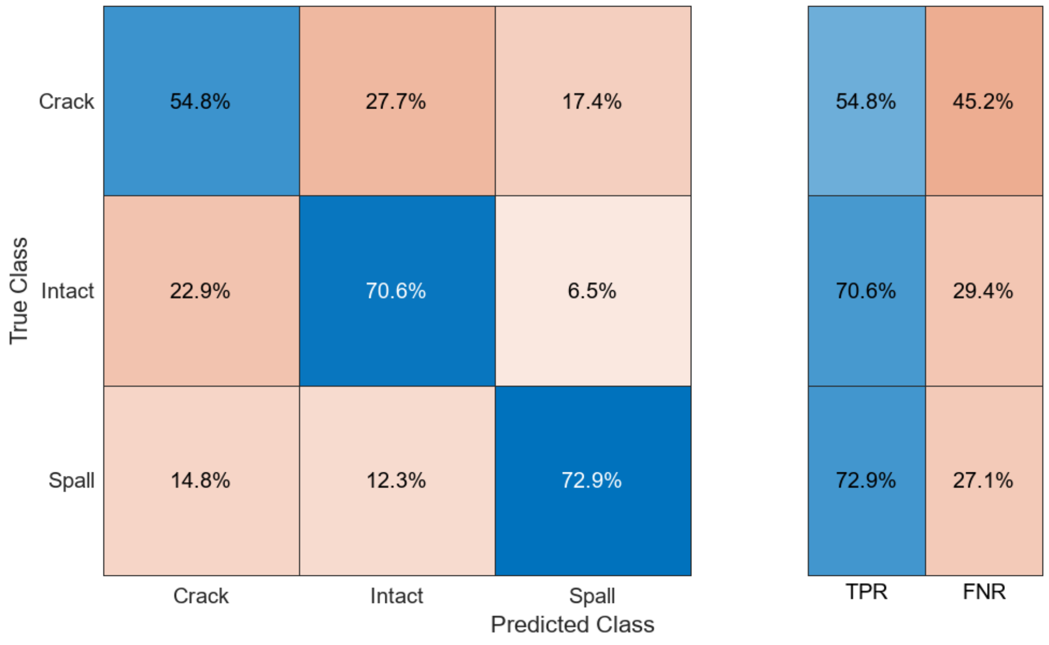


Figure A.52 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the cosine KNN model.

Cubic KNN model

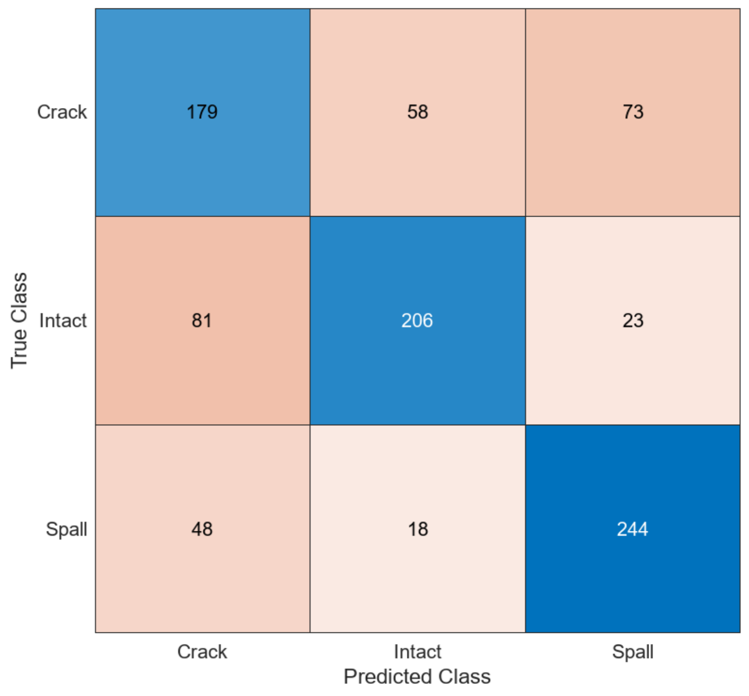


Figure A.53 Confusion matrix showing the classification results of the cubic KNN model.

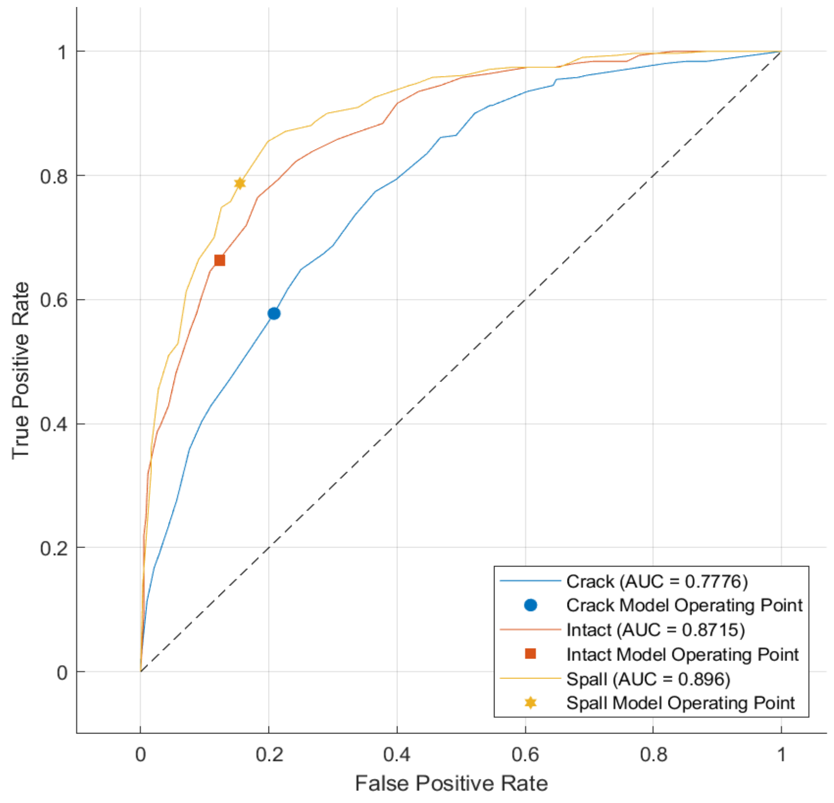


Figure A.54 ROC curve showing the cubic KNN model’s performance.

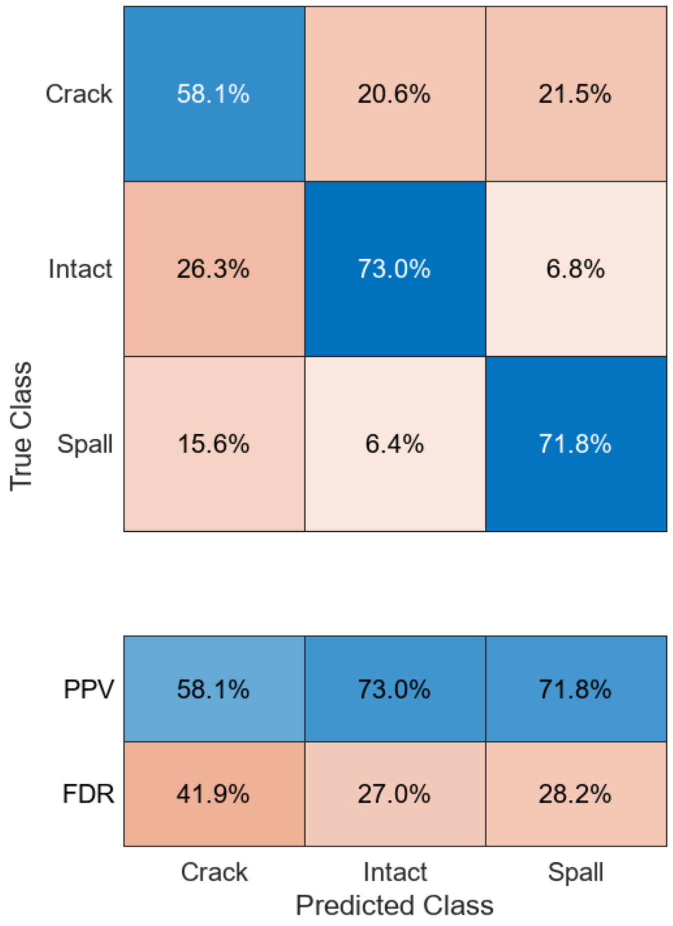


Figure A.55 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the cubic KNN model.

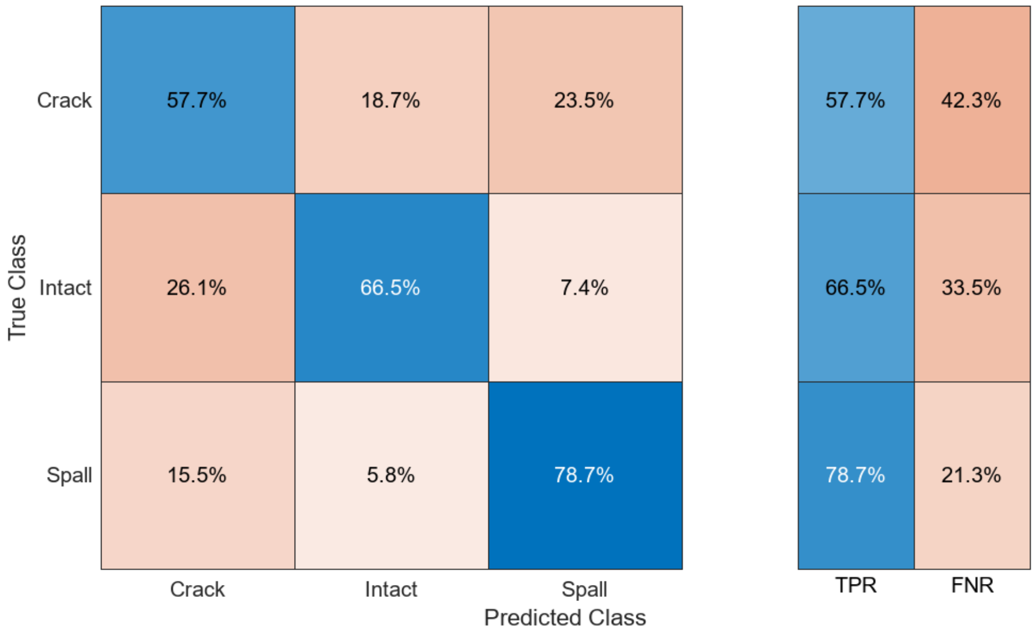


Figure A.56 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the cubic KNN model.

Weighted KNN model

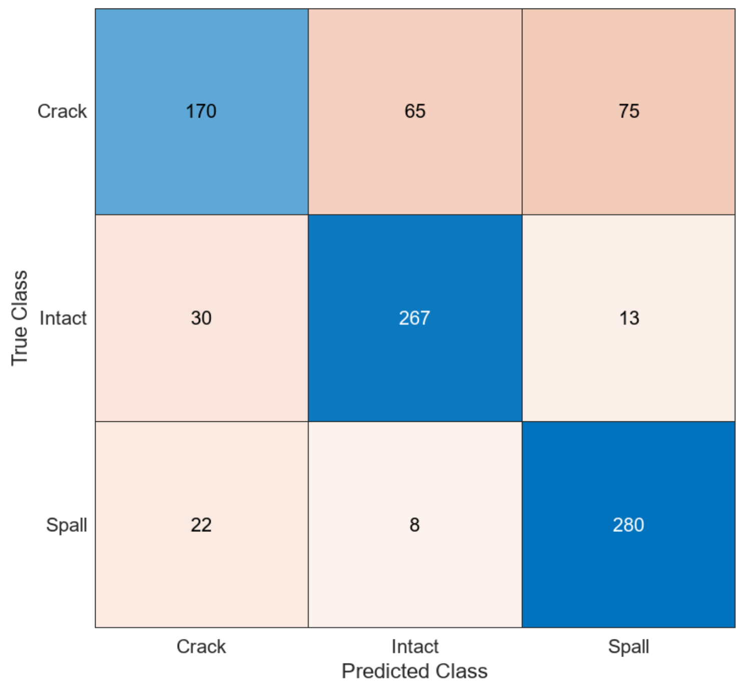


Figure A.57 Confusion matrix showing the classification results of the weighted KNN model.

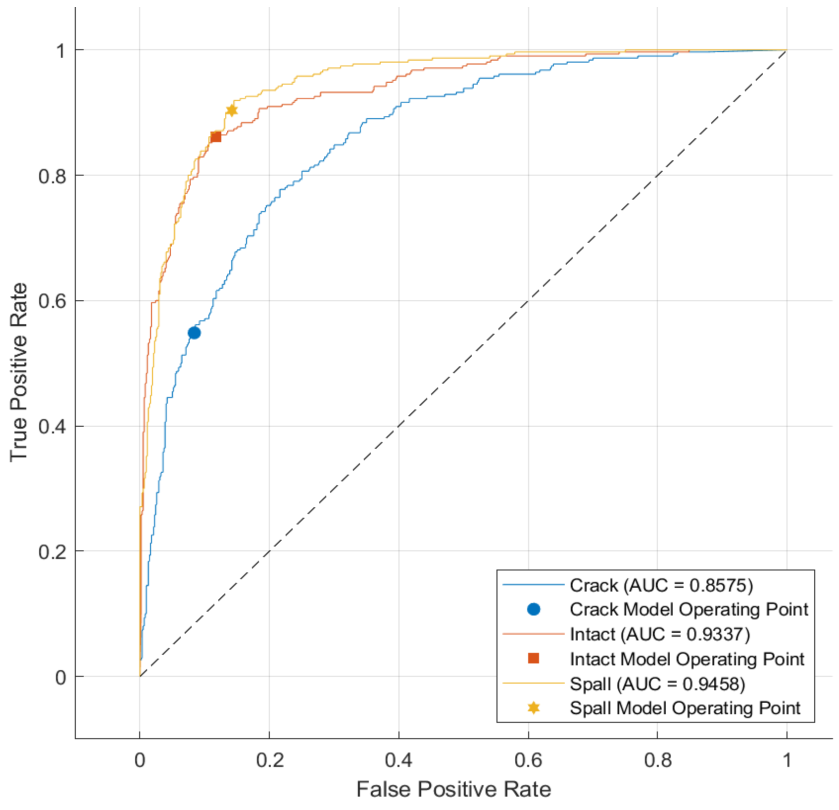


Figure A.58 ROC curve showing the weighted KNN model’s performance.

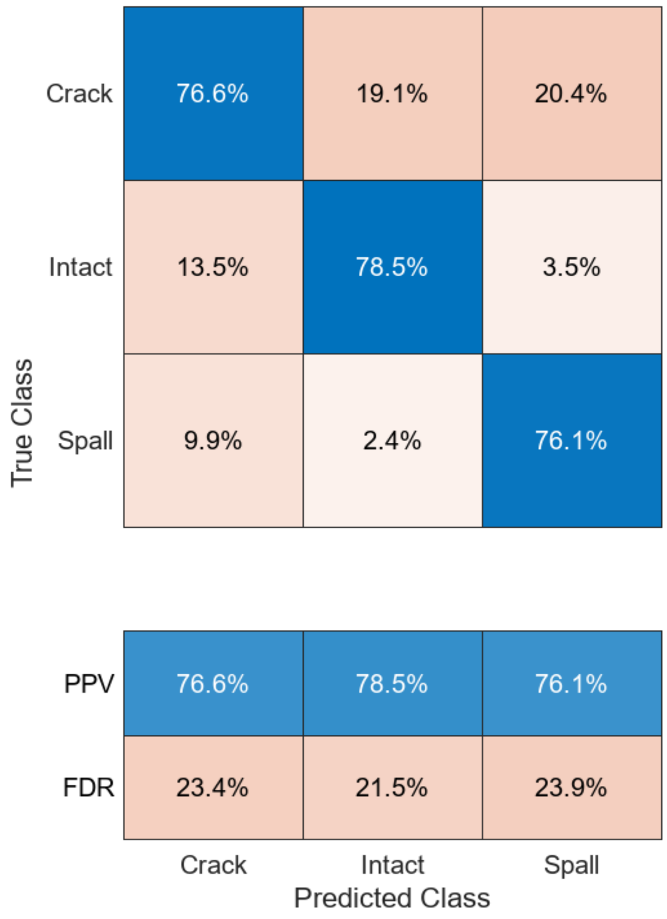


Figure A.59 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the weighted KNN model.

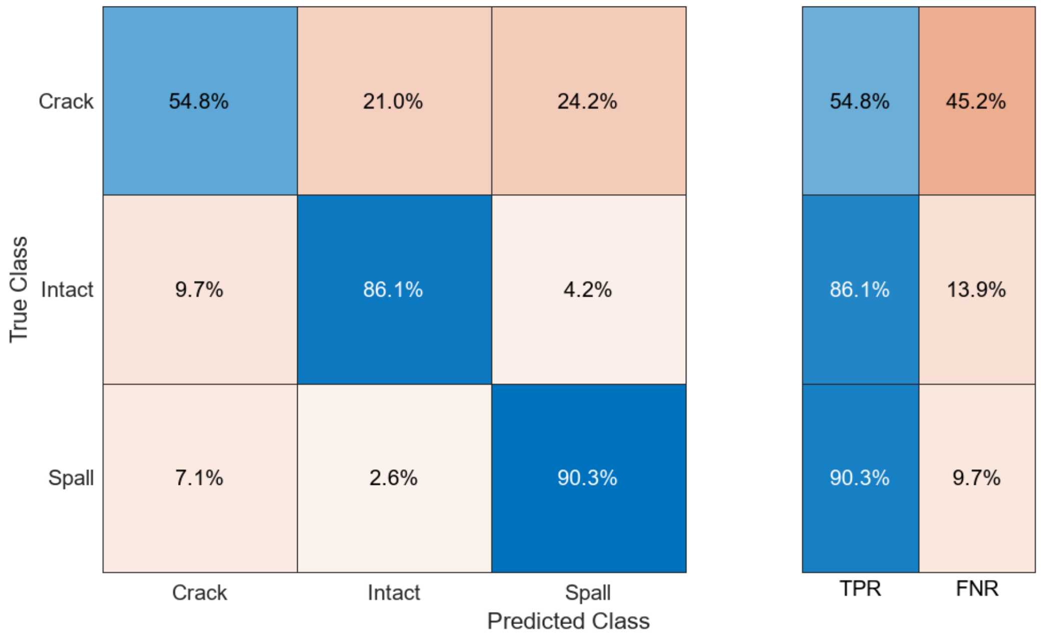


Figure A.60 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the weighted KNN model.

Narrow artificial neural network model

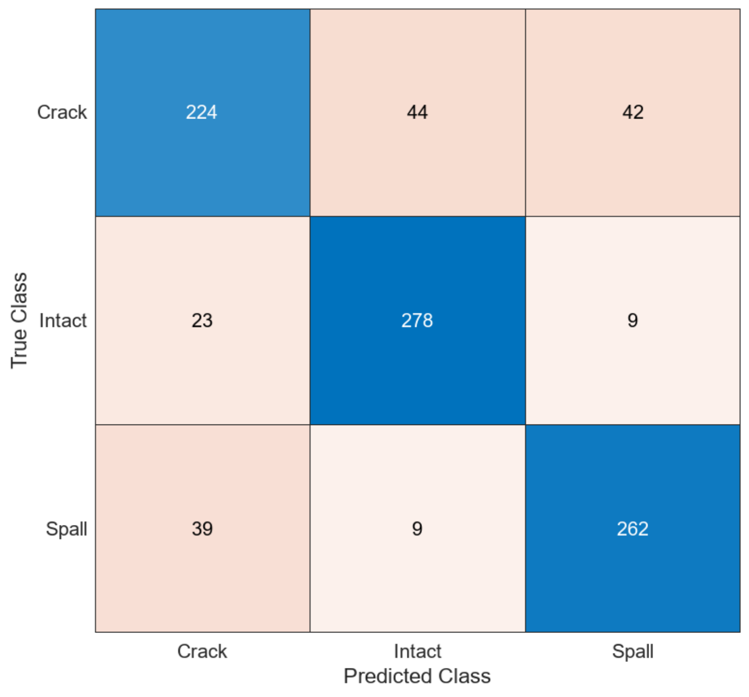


Figure A.61 Confusion matrix showing the classification results of the narrow artificial neural network.

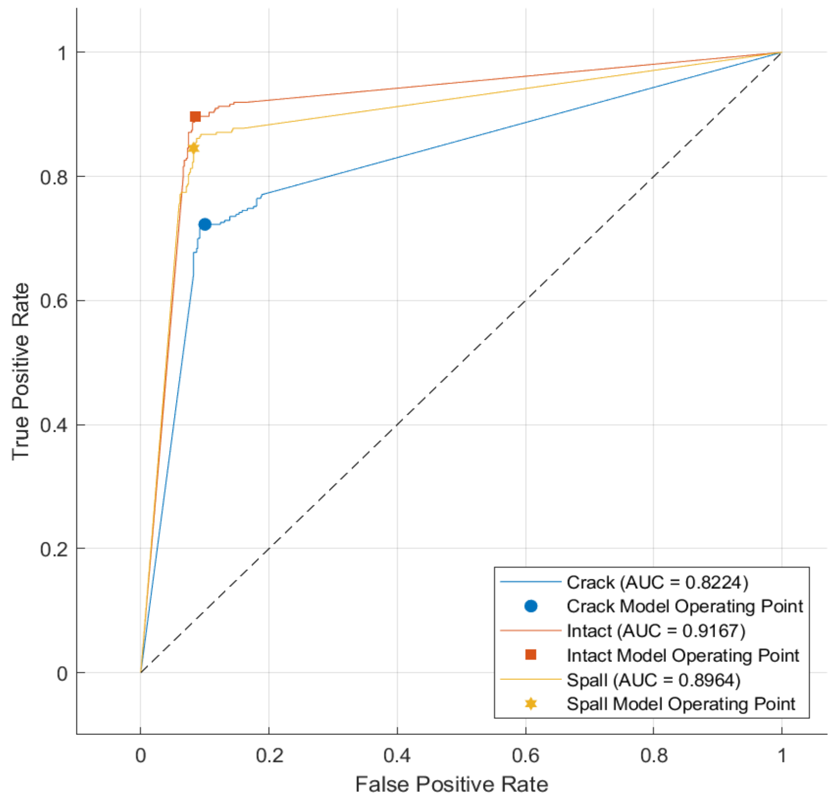


Figure A.62 ROC curve showing the narrow artificial neural network performance.

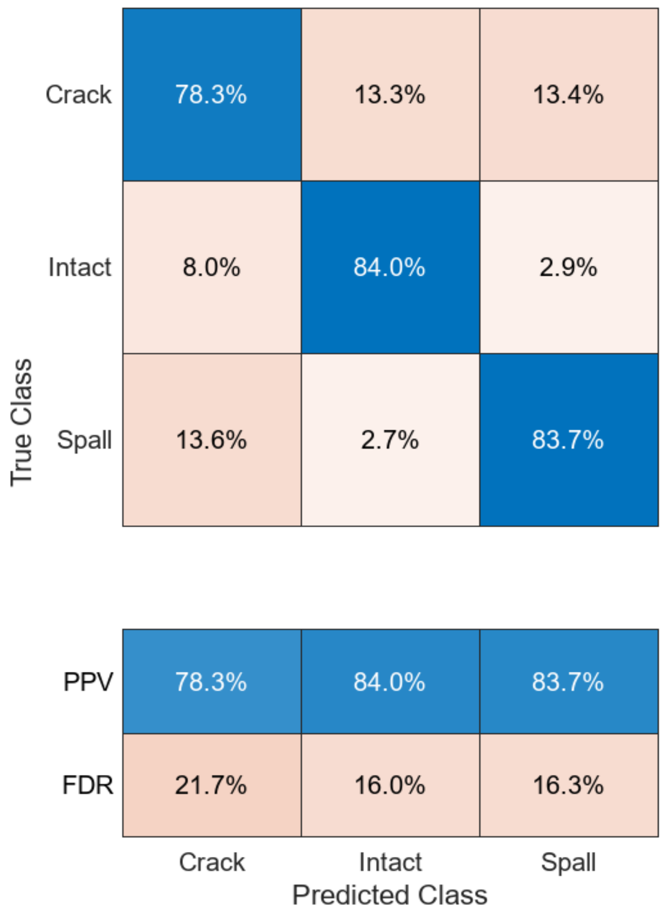


Figure A.63 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the narrow artificial neural network.

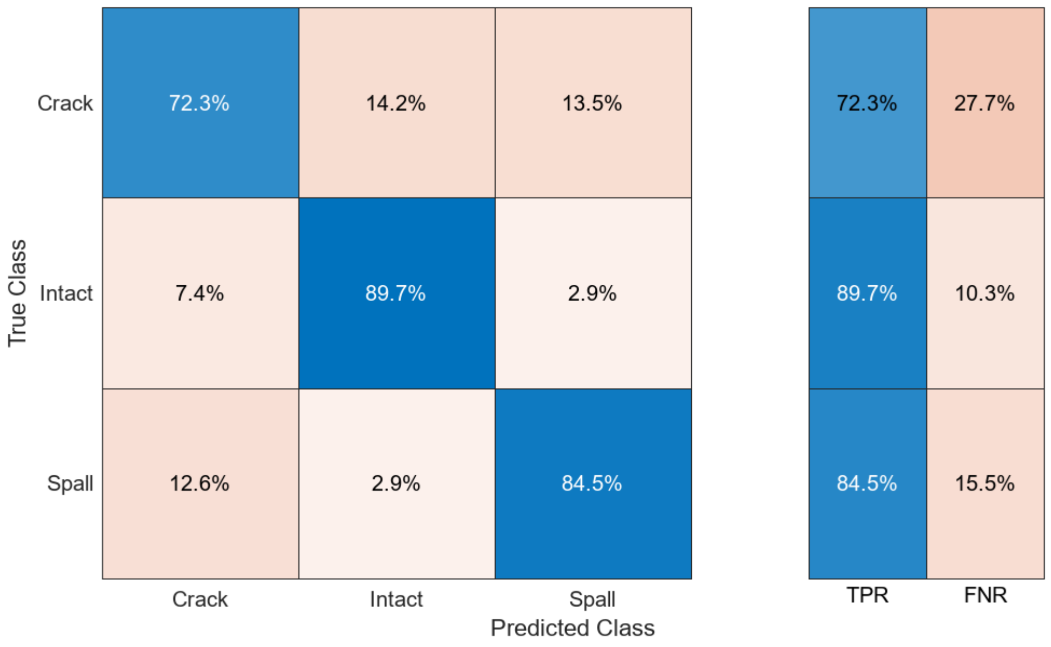


Figure A.64 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the narrow artificial neural network.

Medium artificial neural network model

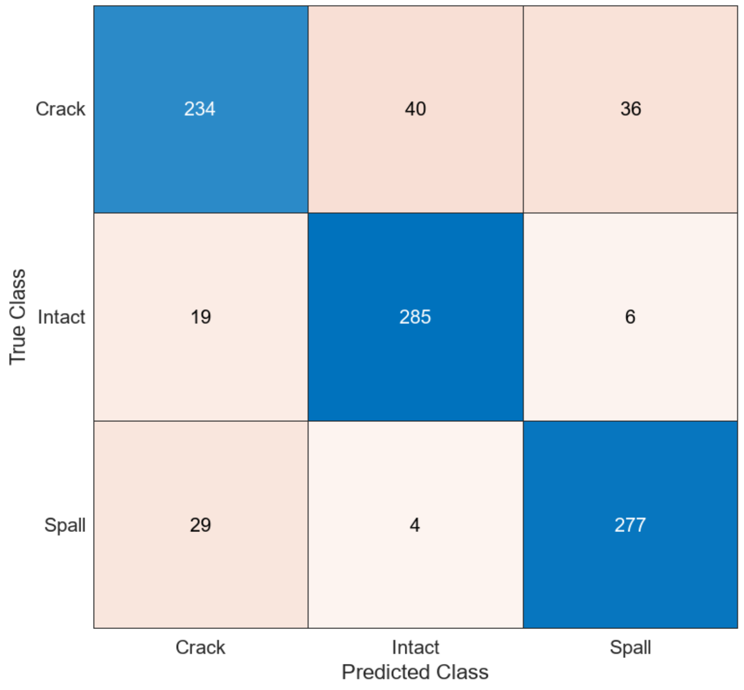


Figure A.65 Confusion matrix showing the classification results of the medium artificial neural network.

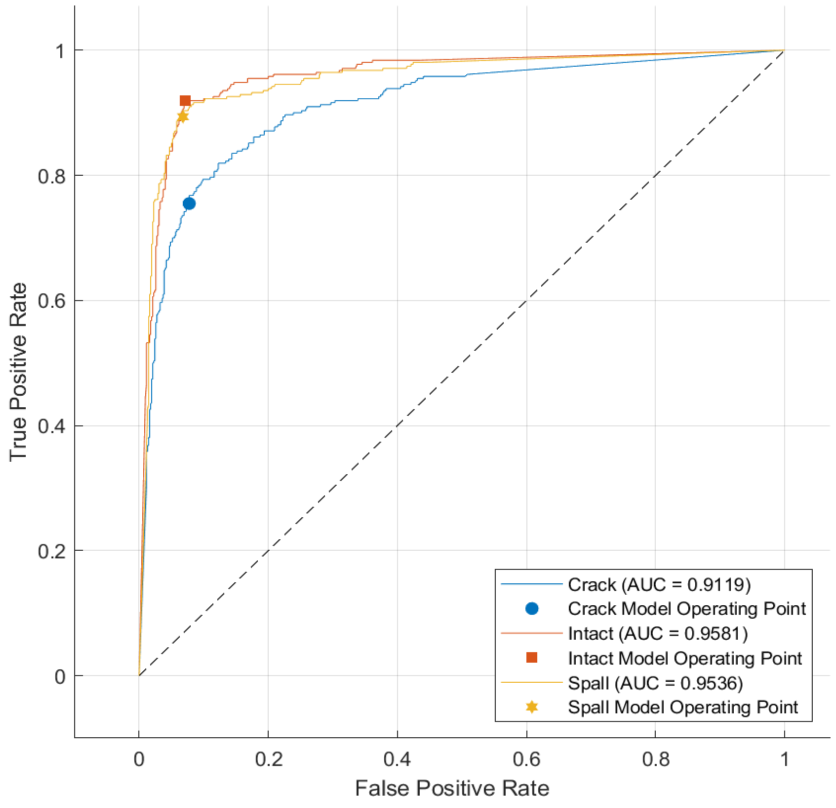


Figure A.66 ROC curve showing the medium artificial neural network performance.

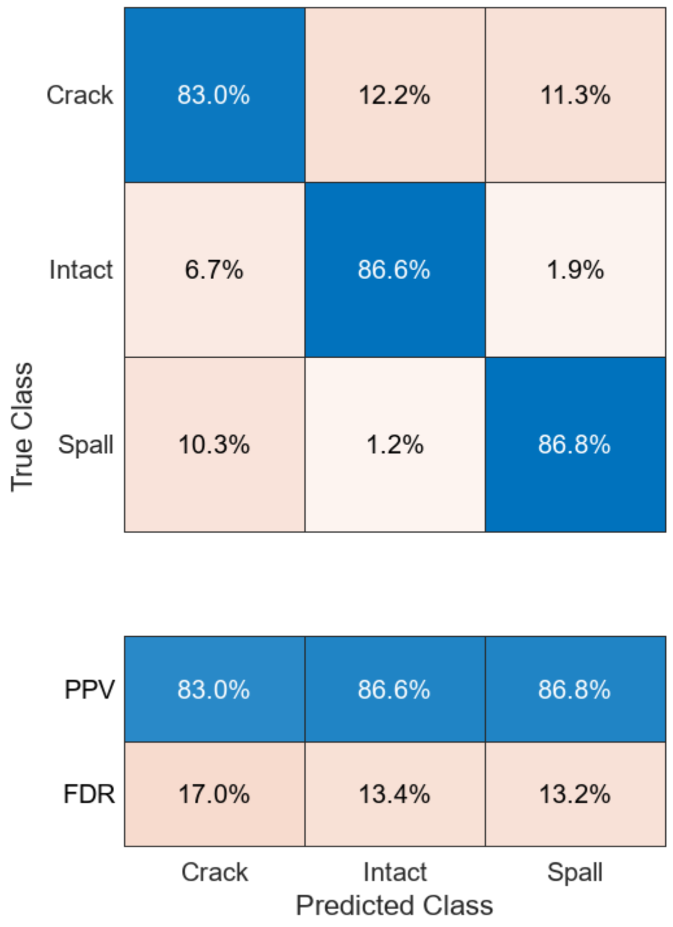


Figure A.67 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium artificial neural network.

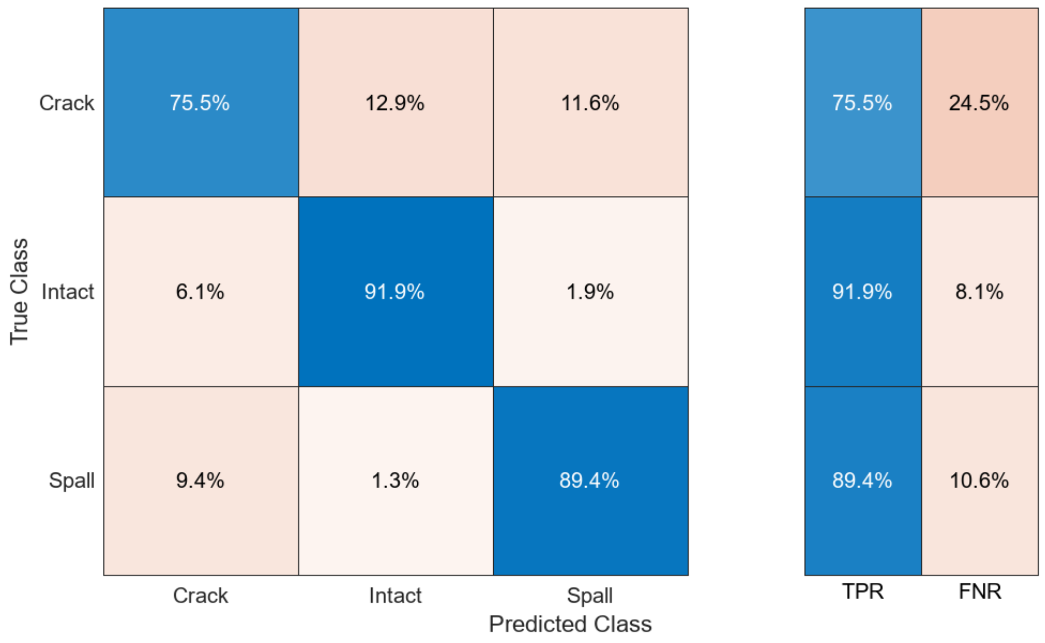


Figure A.68 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium artificial neural network.

Bi-layered artificial neural network model

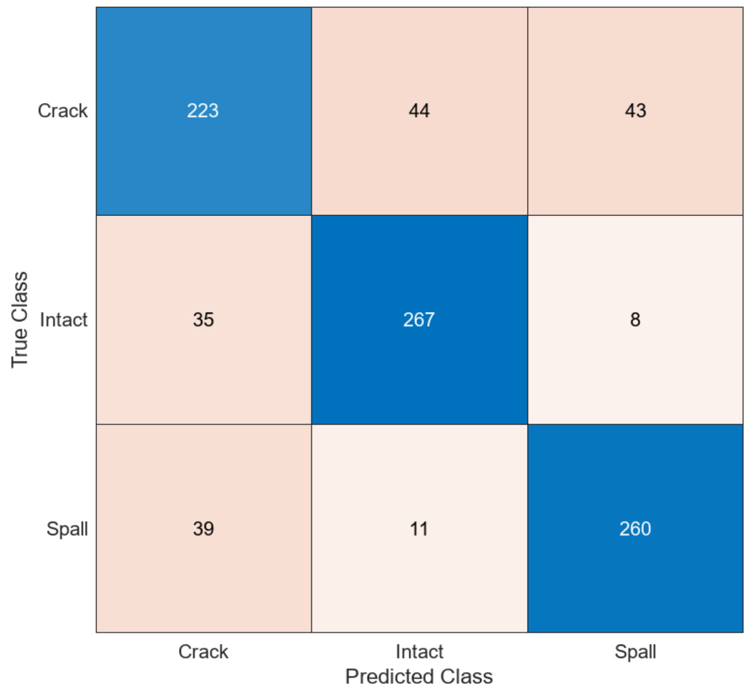


Figure A.69 Confusion matrix showing the classification results of the bi-layered artificial neural network.

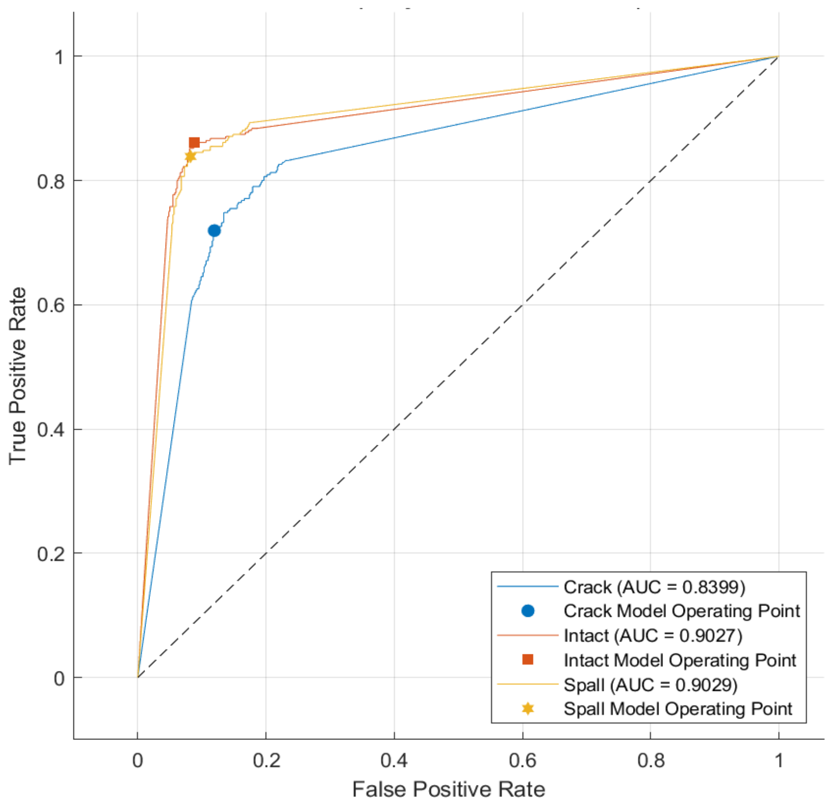


Figure A.70 ROC curve showing the bi-layered artificial neural network's performance.

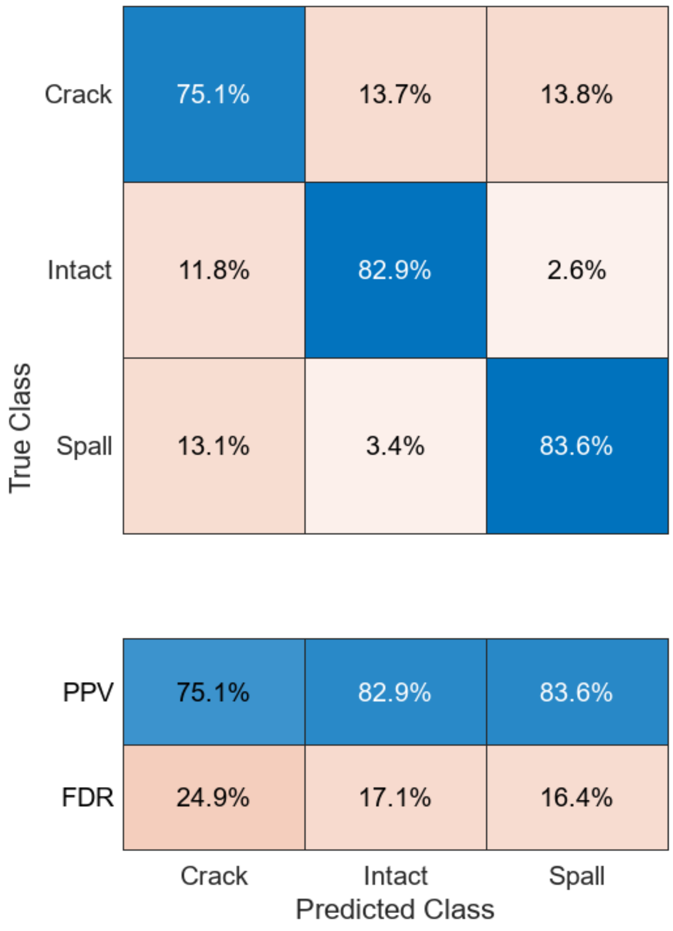


Figure A.71 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the bi-layered artificial neural network.

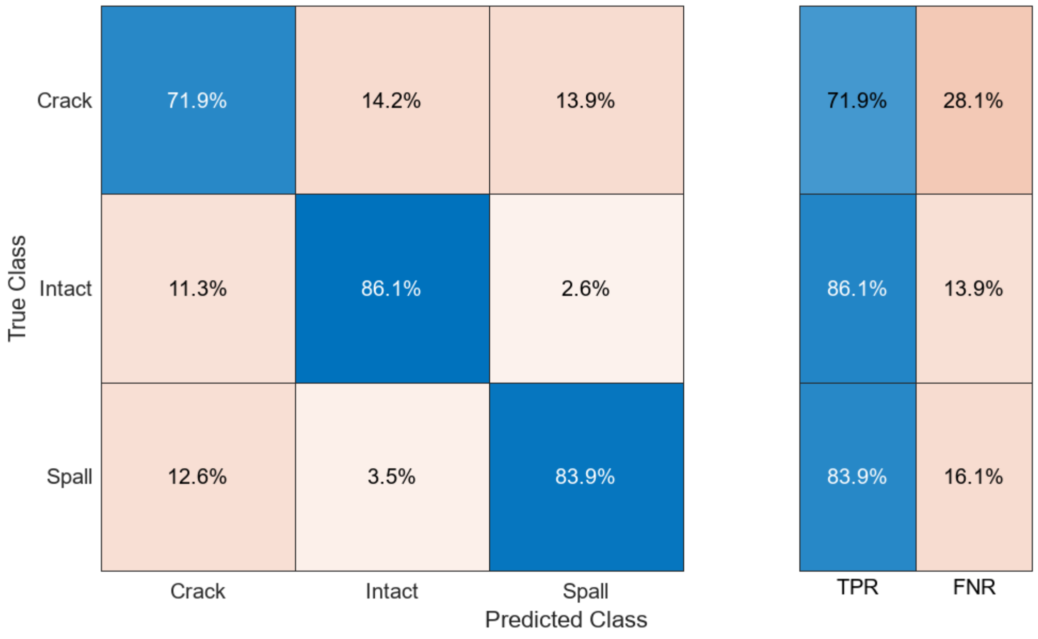


Figure A.72 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the bi-layered artificial neural network.

Tri-layered artificial neural network model

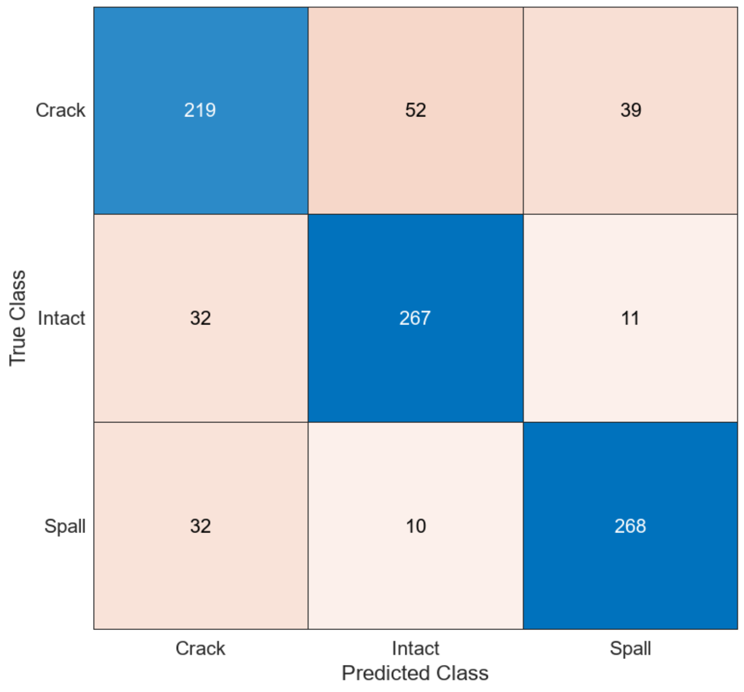


Figure A.73 Confusion matrix showing the classification results of the tri-layered artificial neural network.

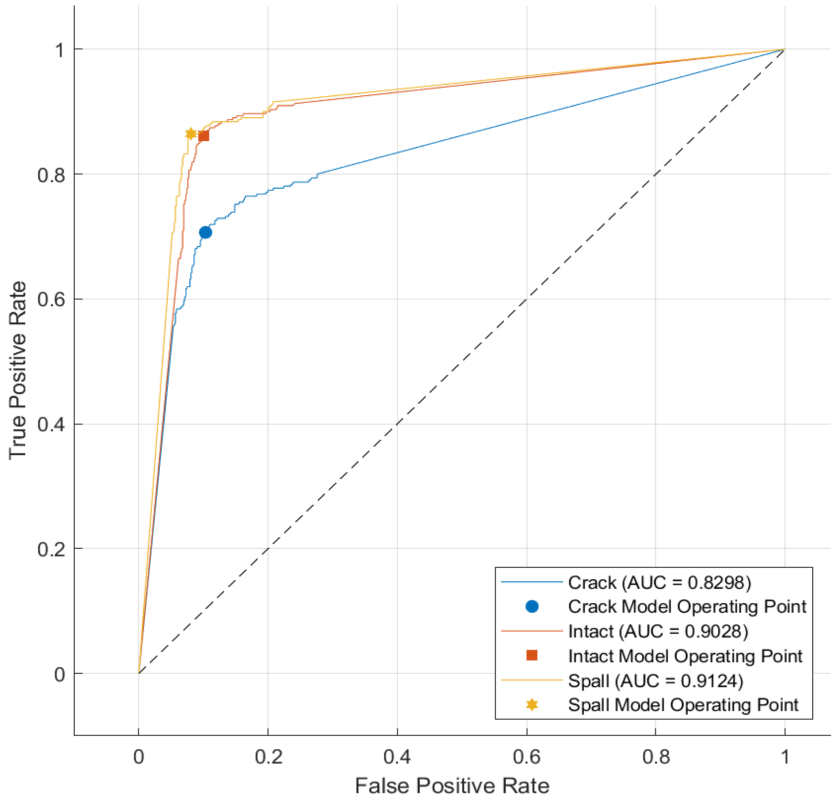


Figure A.74 ROC curve showing the tri-layered artificial neural network’s performance.

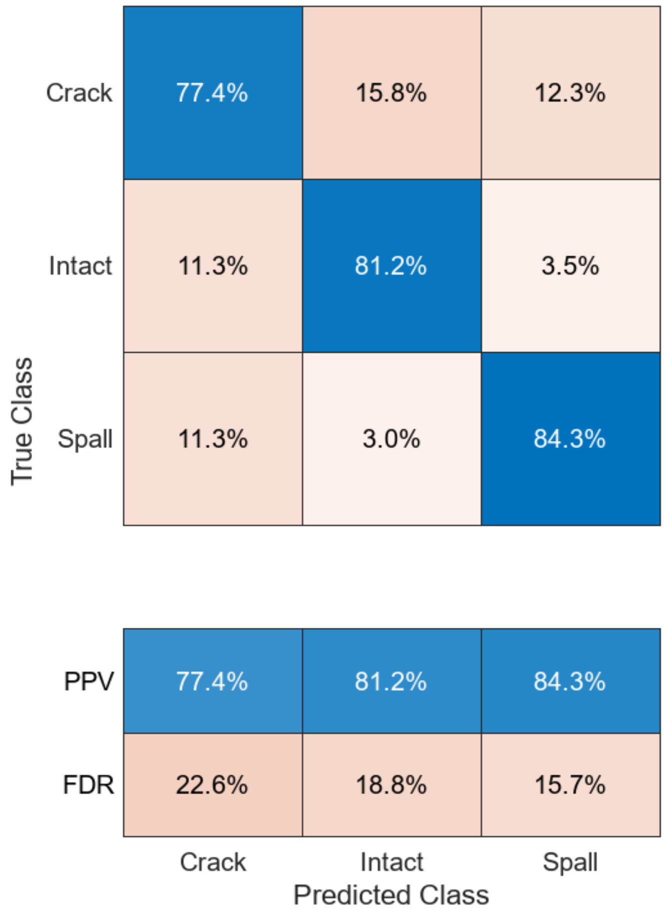


Figure A.75 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the tri-layered artificial neural network.

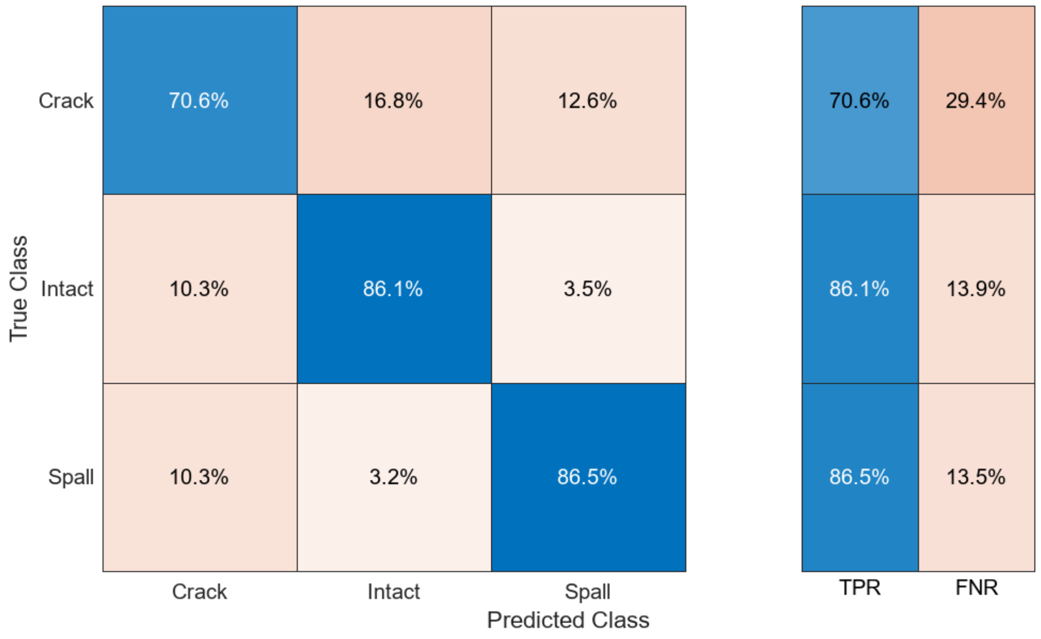


Figure A.76 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the tri-layered artificial neural network.

Optimisable artificial neural network model

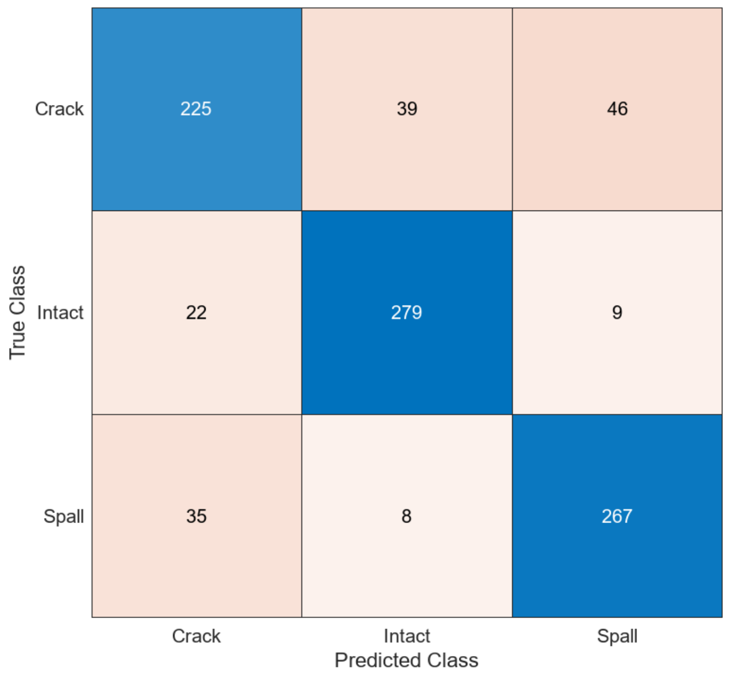


Figure A.77 Confusion matrix showing the classification results of the optimisable artificial neural network.

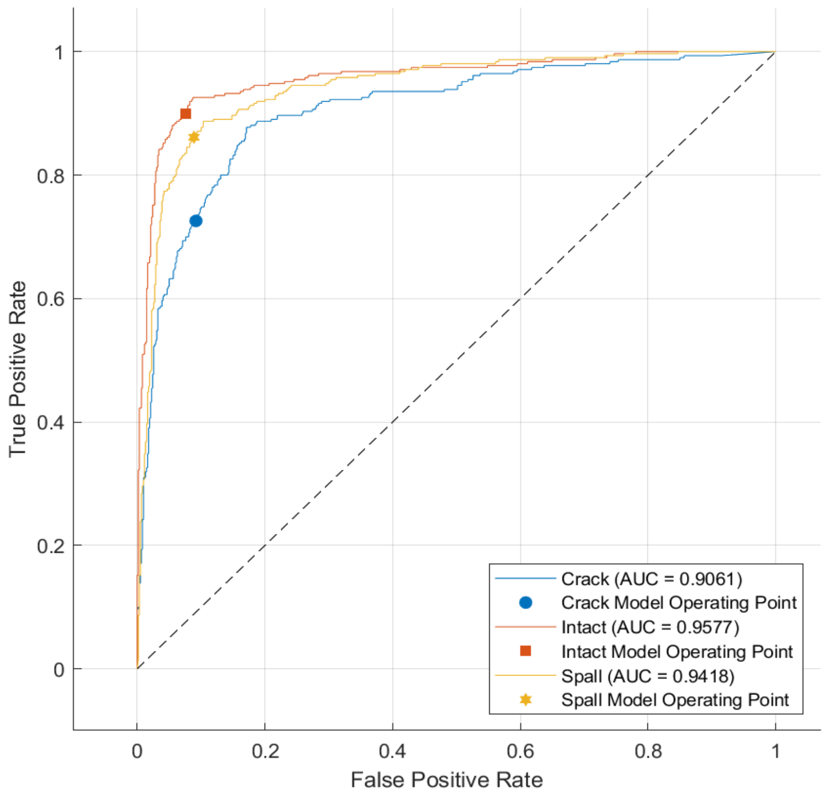


Figure A.78 ROC curve showing the optimisable artificial neural network's performance.

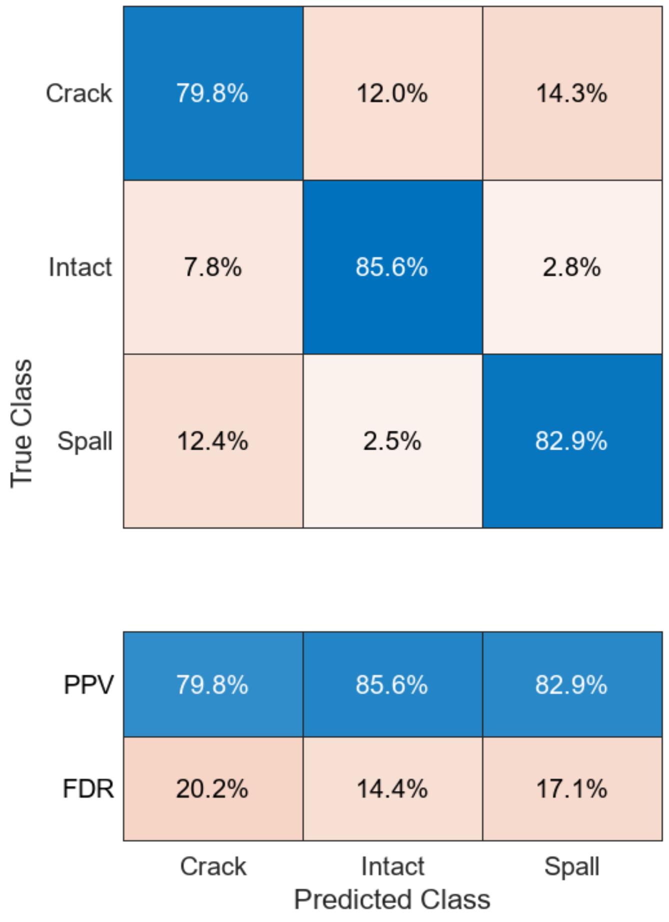


Figure A.79 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the optimisable artificial neural network.

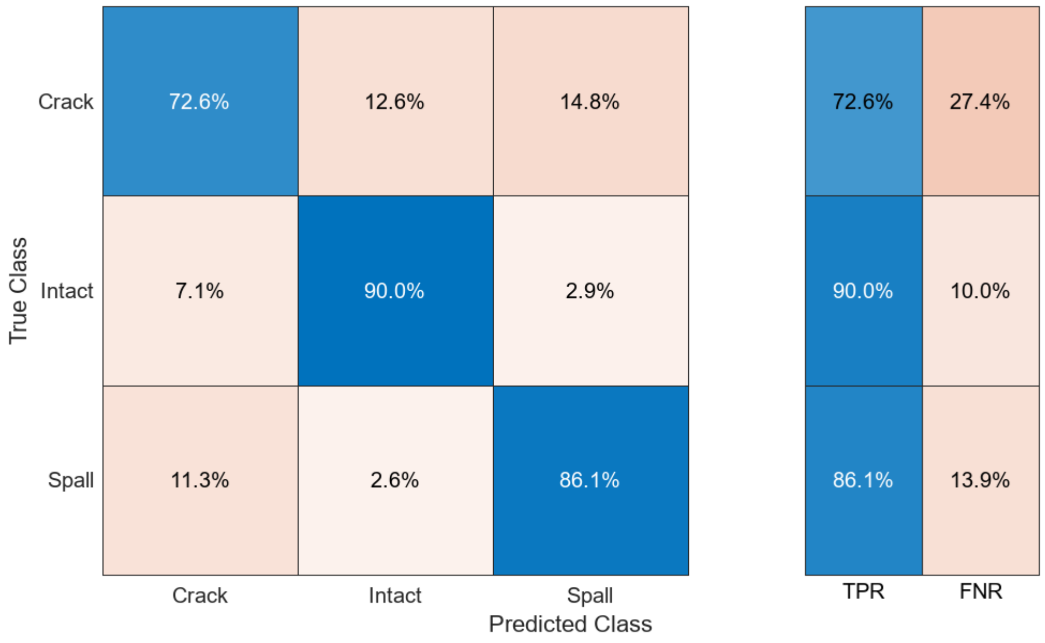


Figure A.80 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the optimisable artificial neural network.

Appendix B. Machine learning model testing results

The following figures and tables present the results of the conventional machine learning models' performance during the testing phase of this research. The results present a breakdown of the performance of the models in the defect classification of the crack and spall damage, and intact areas.

Medium decision tree model

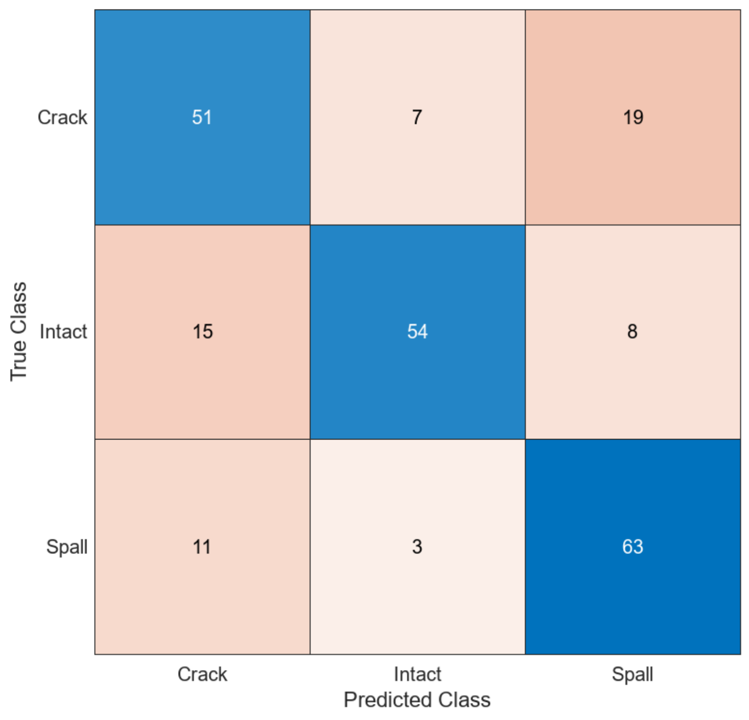


Figure B.1 Confusion matrix showing the classification results of the medium decision tree model.

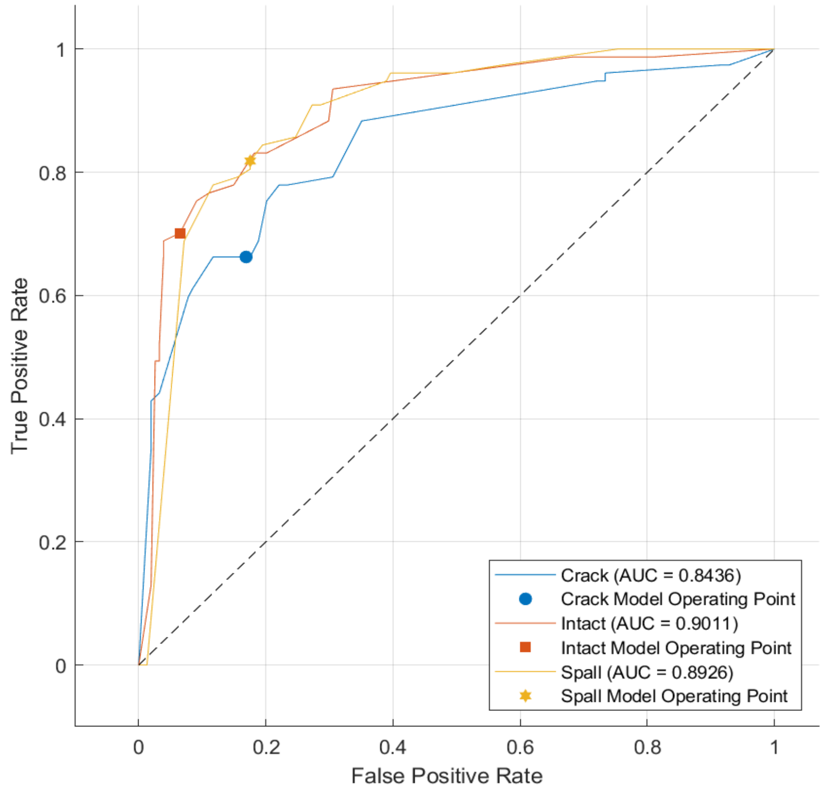


Figure B.2 ROC curve showing the medium decision tree model's performance.

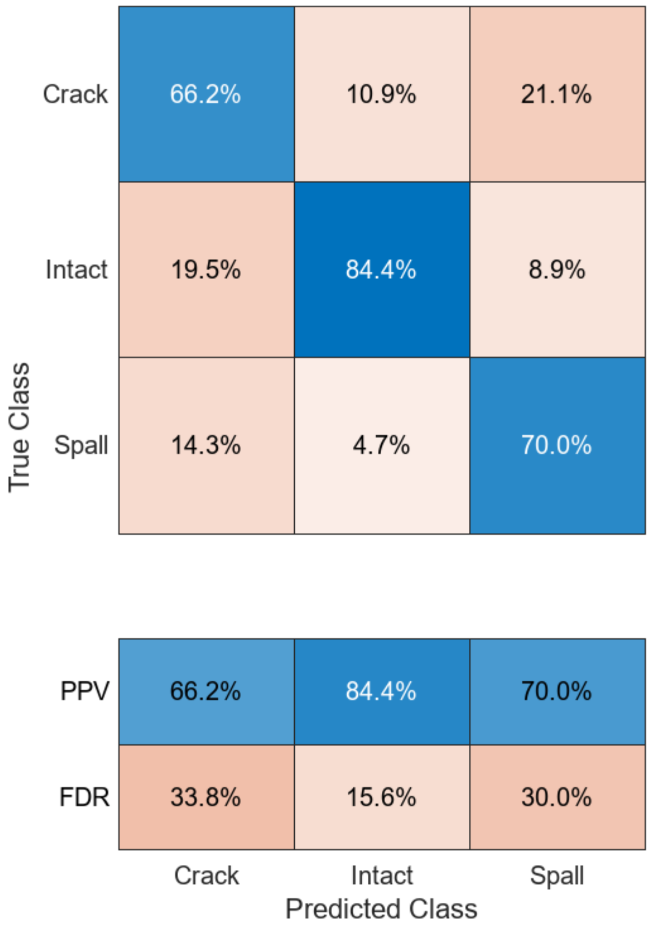


Figure B.3 The precision or positive predictive value (PPV) and false discovery rate (FDR) achieved for each damage class, using the medium decision tree model.

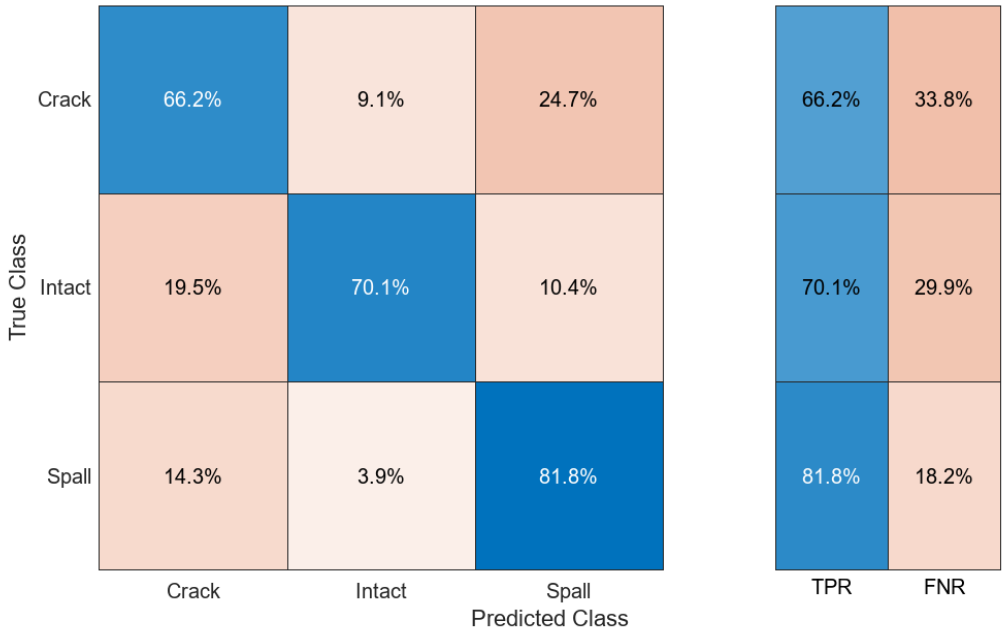


Figure B.4 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium decision tree model.

Coarse decision tree model

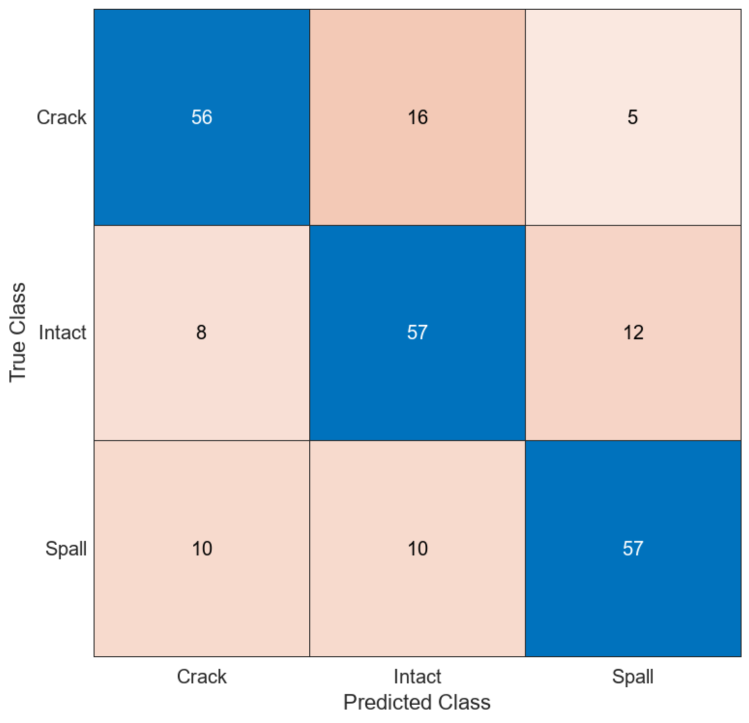


Figure B.5 Confusion matrix showing the classification results of the coarse decision tree model.

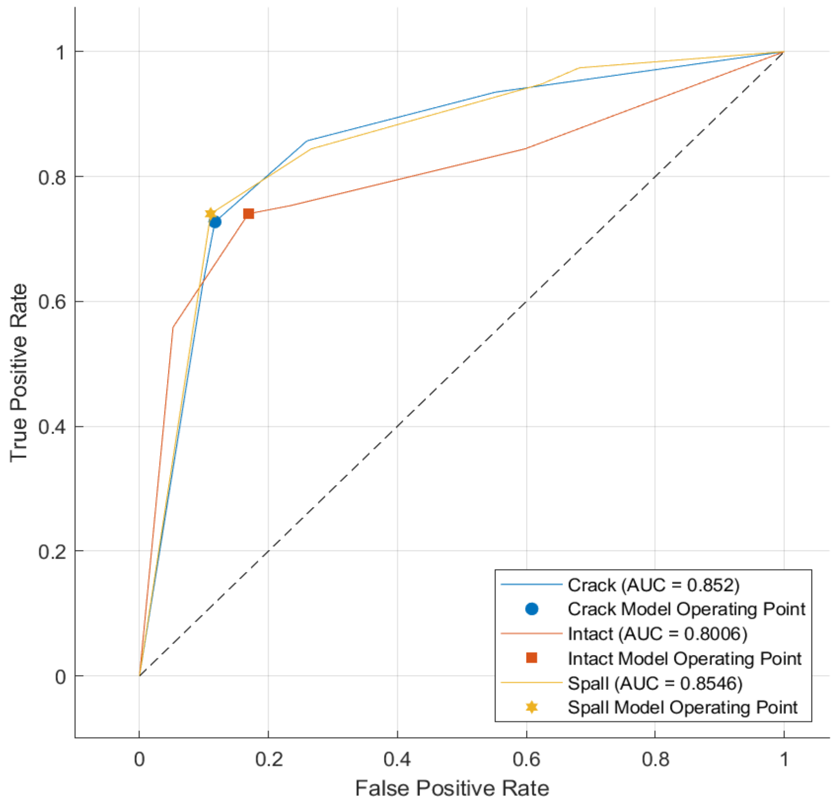


Figure B.6 ROC curve showing the coarse decision tree model’s performance.

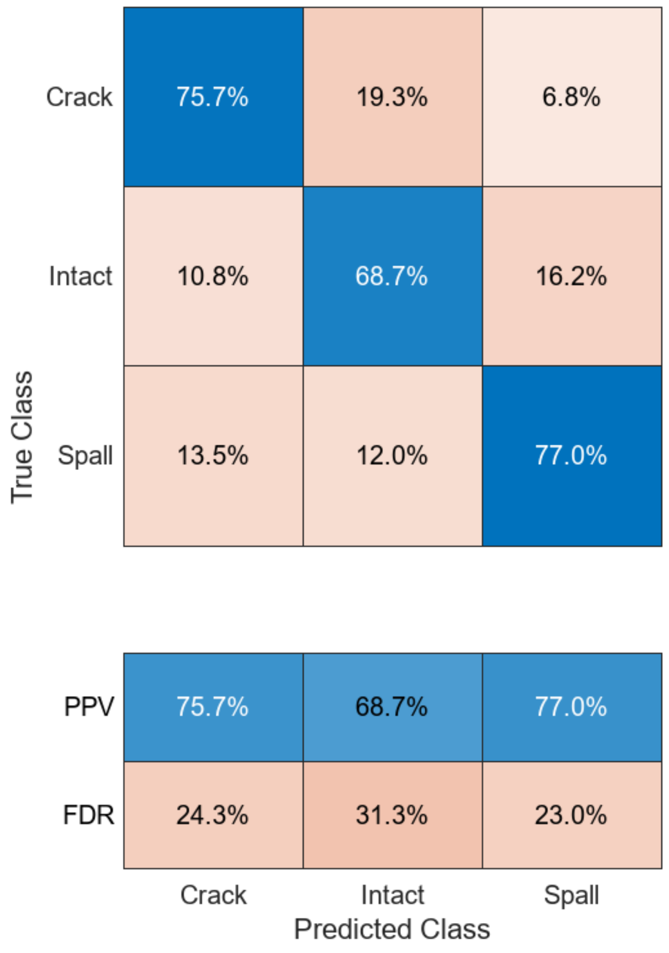


Figure B.7 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse decision tree model.

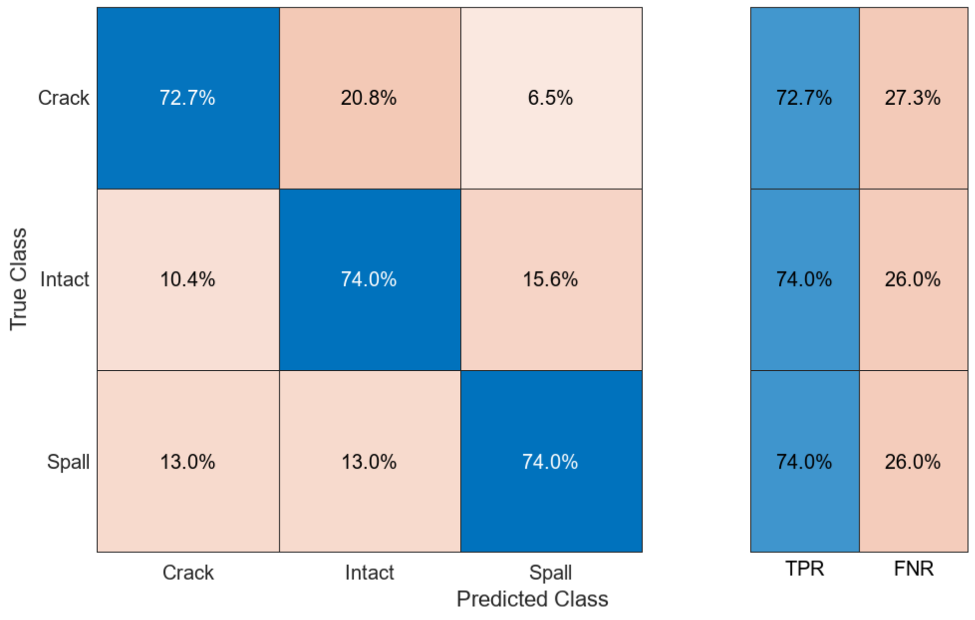


Figure B.8 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse decision tree model.

Optimisable decision tree model

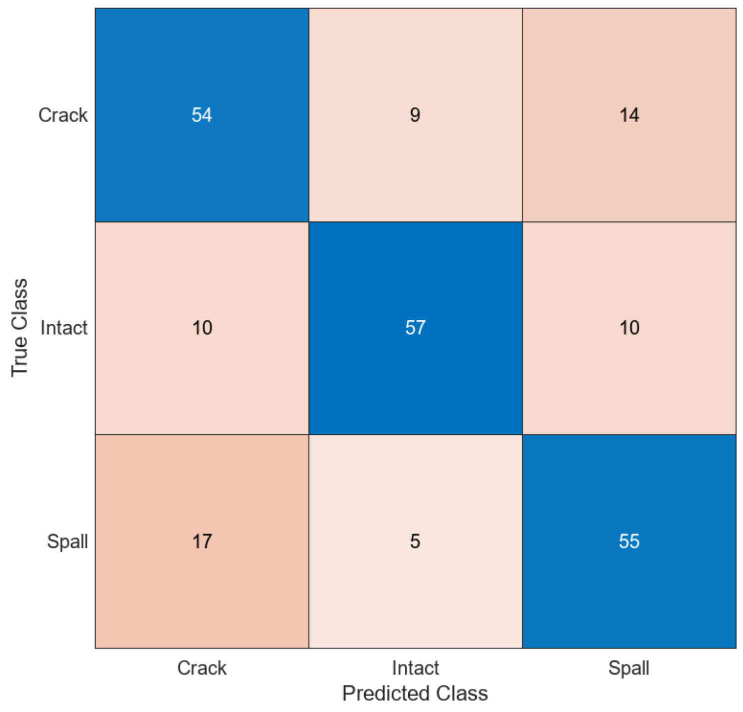


Figure B.9 Confusion matrix showing the classification results of the optimisable decision tree model.

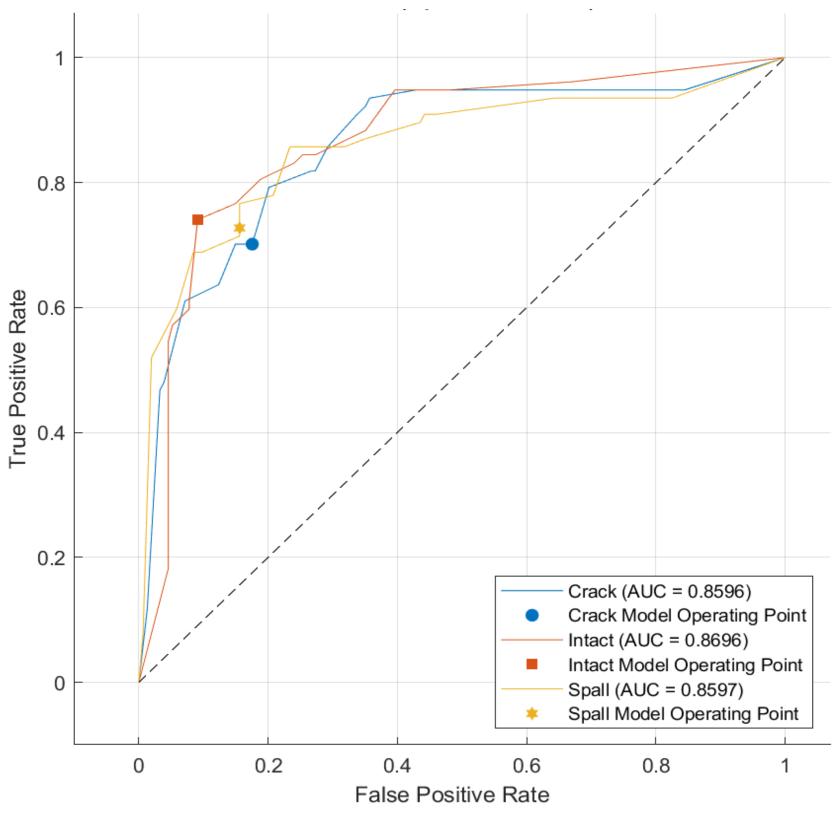


Figure B.10 ROC curve showing the optimisable decision tree model’s performance.

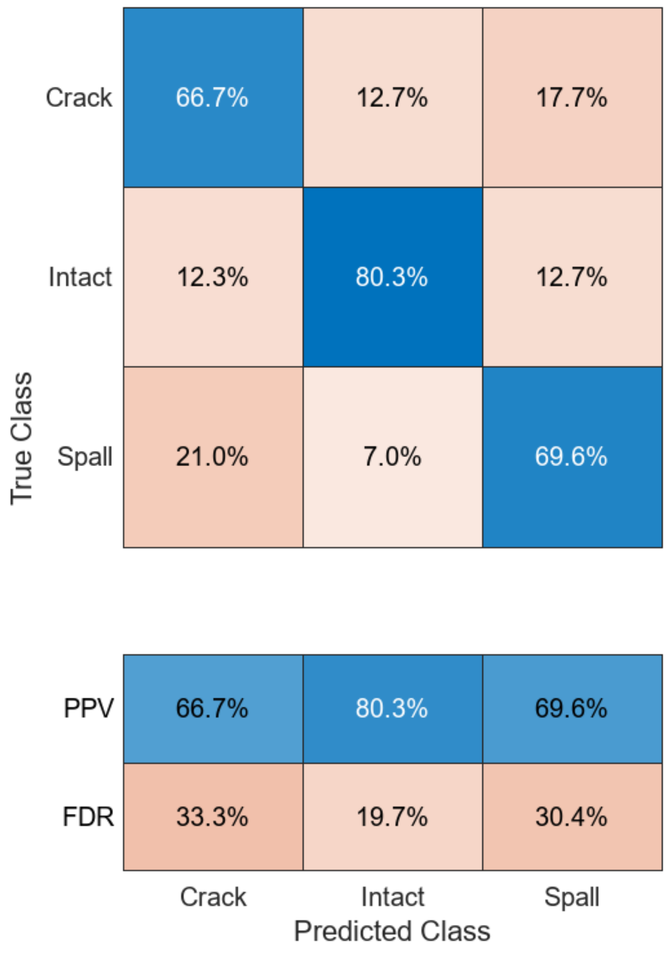


Figure B.11 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the optimisable decision tree model.

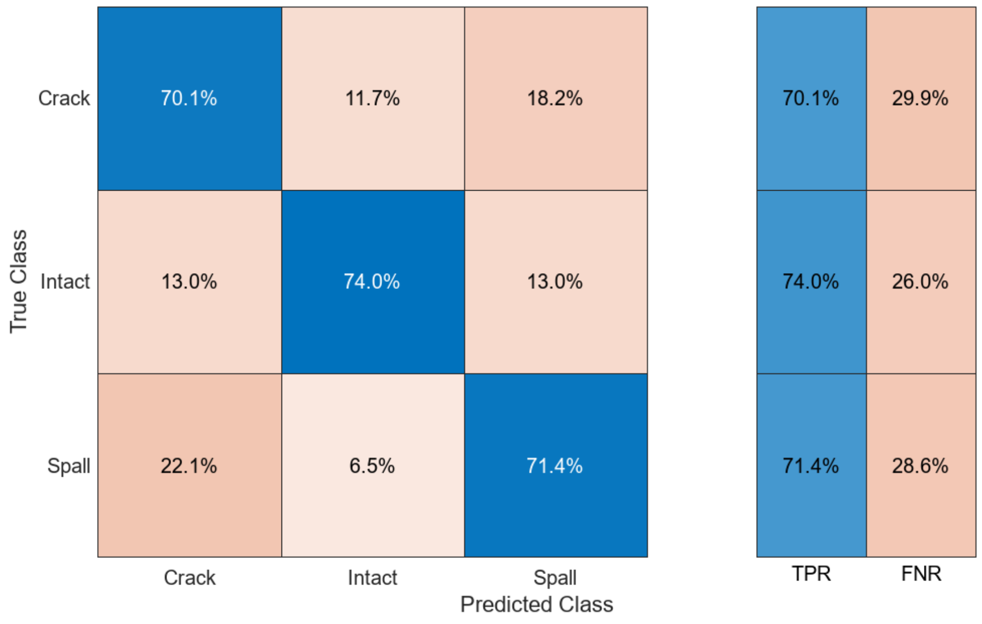


Figure B.12 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the optimisable decision tree model.

Linear SVM model

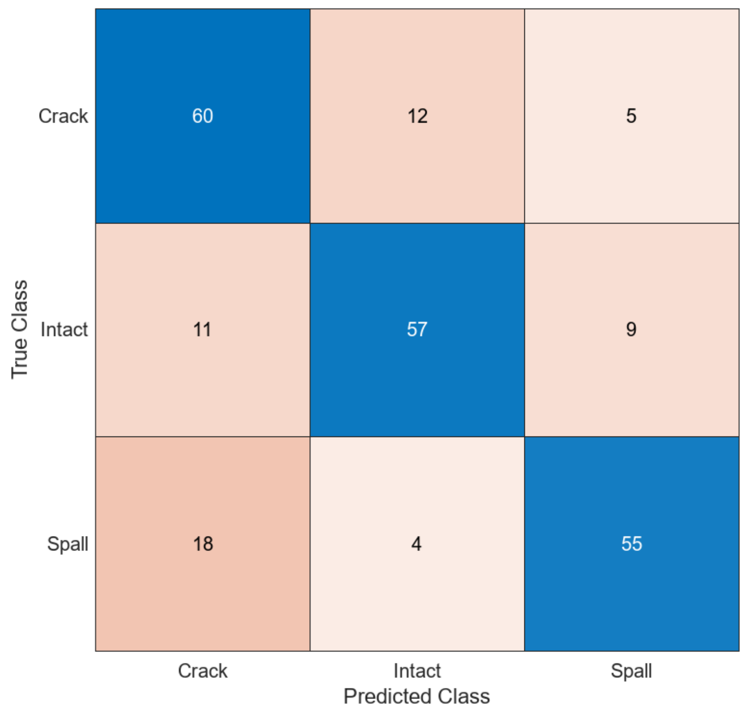


Figure B.13 Confusion matrix showing the classification results of the linear SVM model.

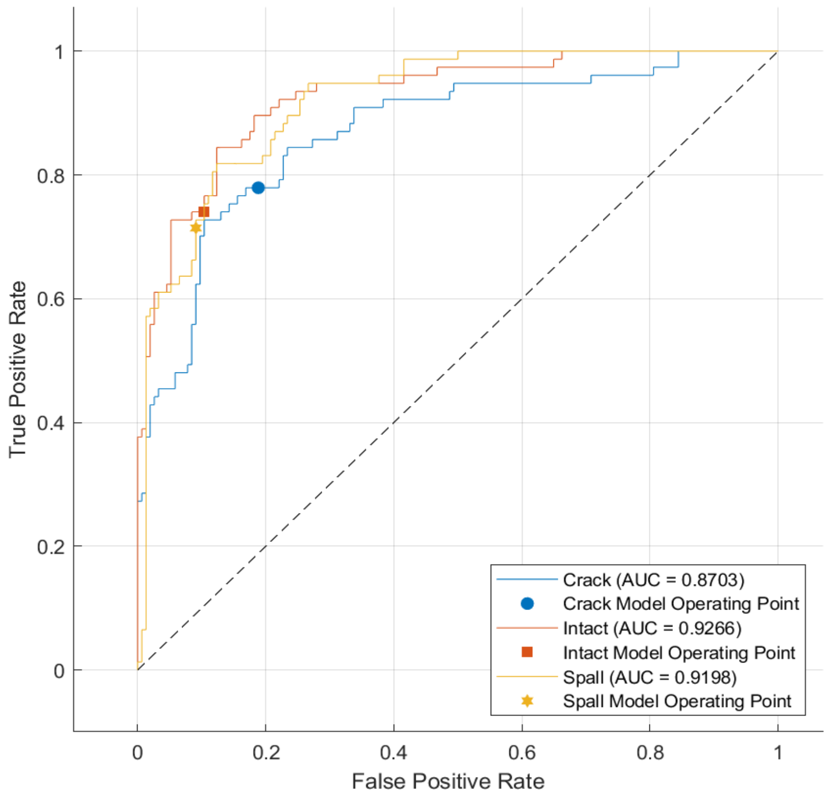


Figure B.14 ROC curve showing the linear SVM model's performance.

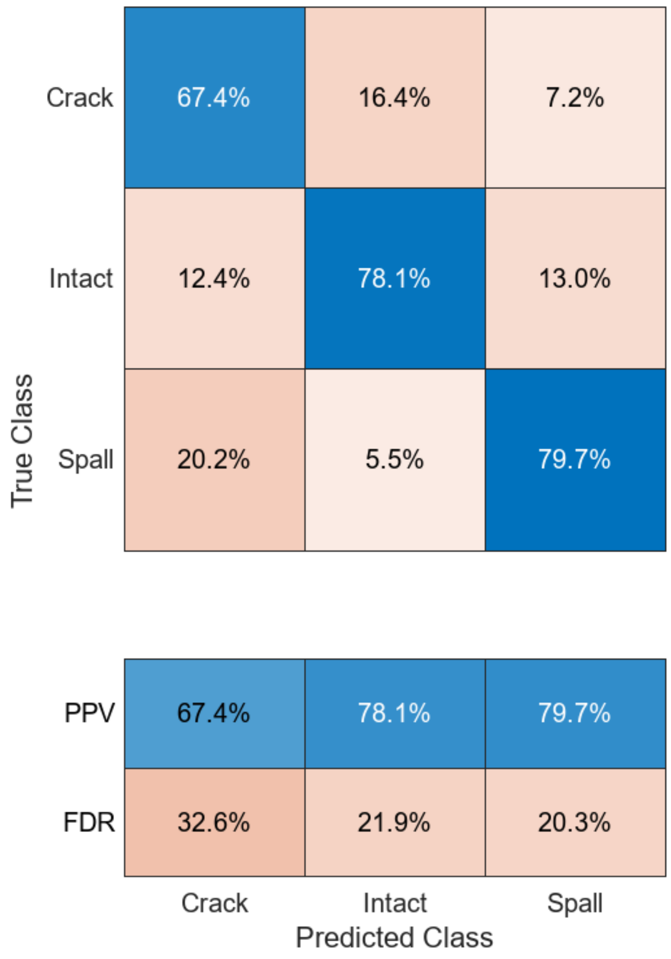


Figure B.15 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the linear SVM model.

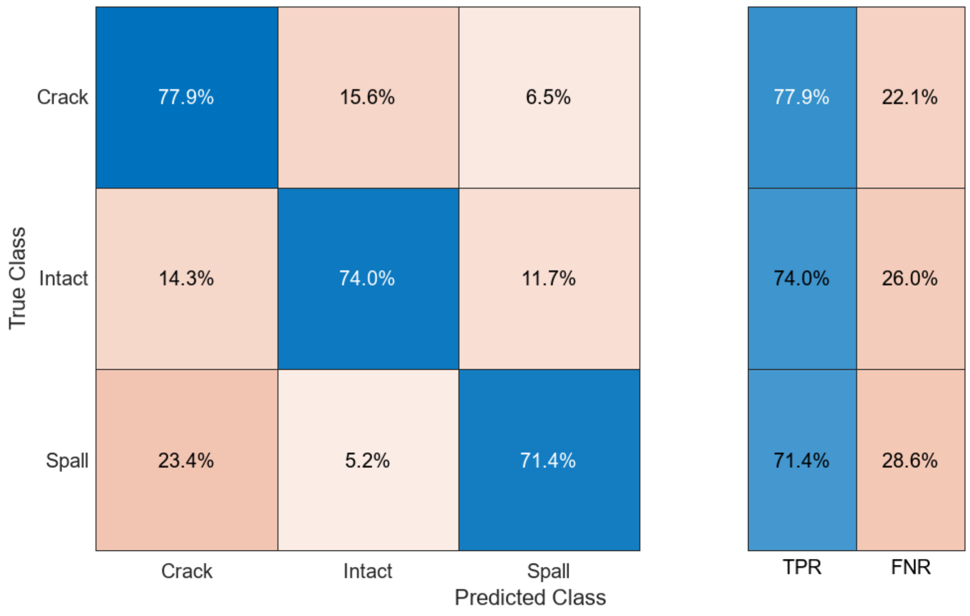


Figure B.16 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the linear SVM model.

Quadratic SVM model

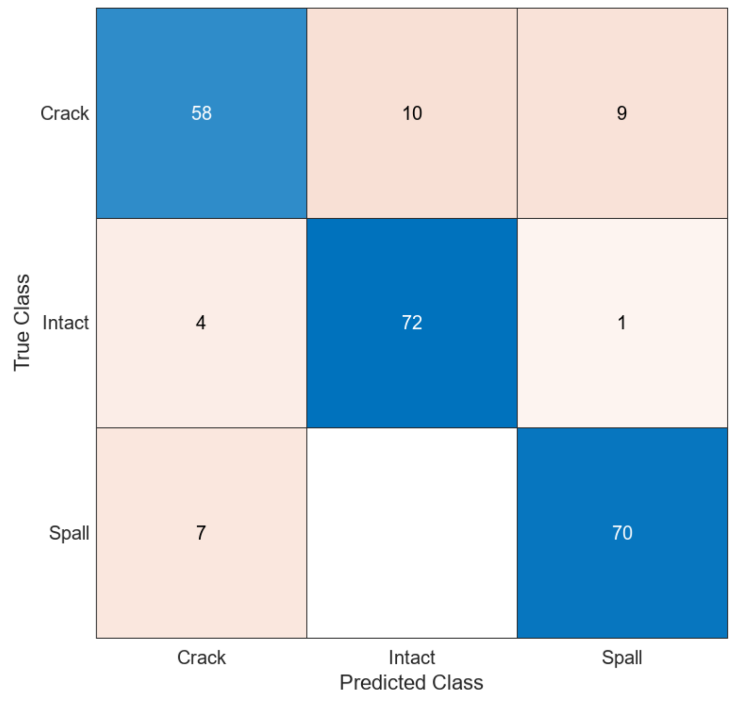


Figure B.17 Confusion matrix showing the classification results of the quadratic SVM model.

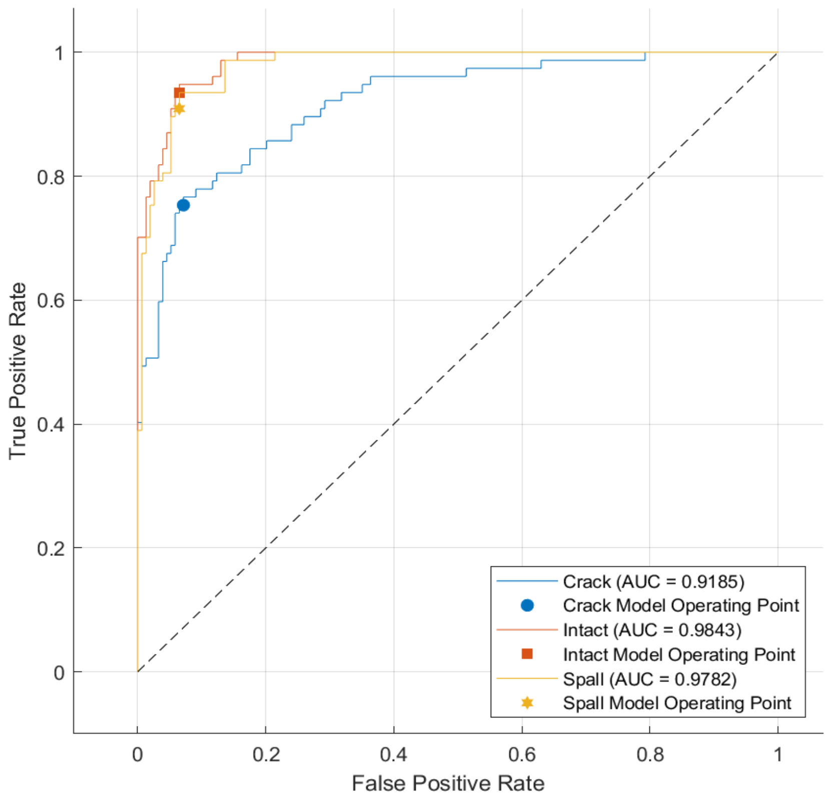


Figure B.18 ROC curve showing the quadratic SVM model's performance.

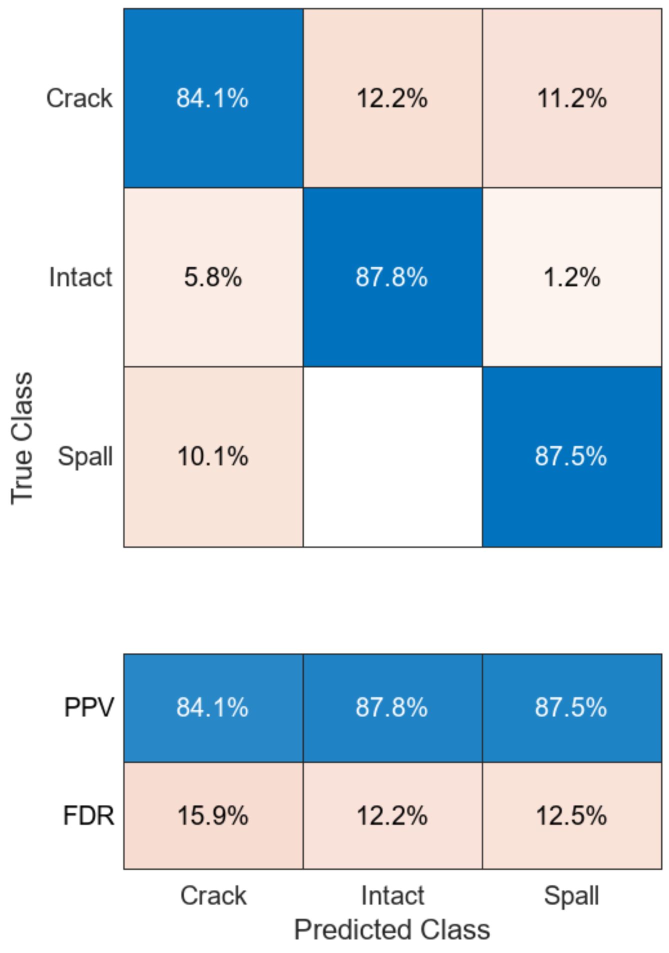


Figure B.19 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the quadratic SVM model.

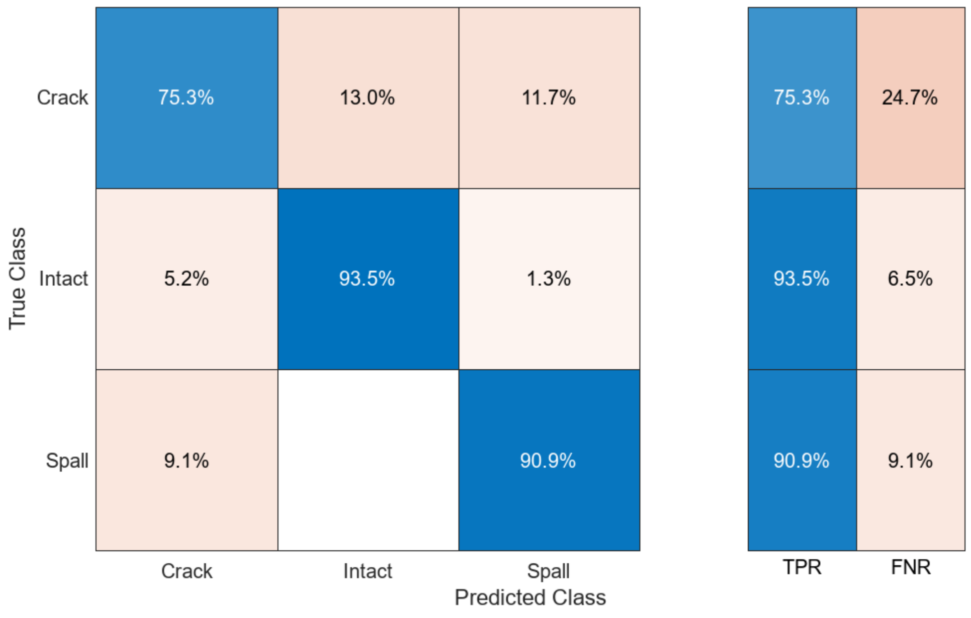


Figure B.20 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the quadratic SVM model.

Fine Gaussian SVM model

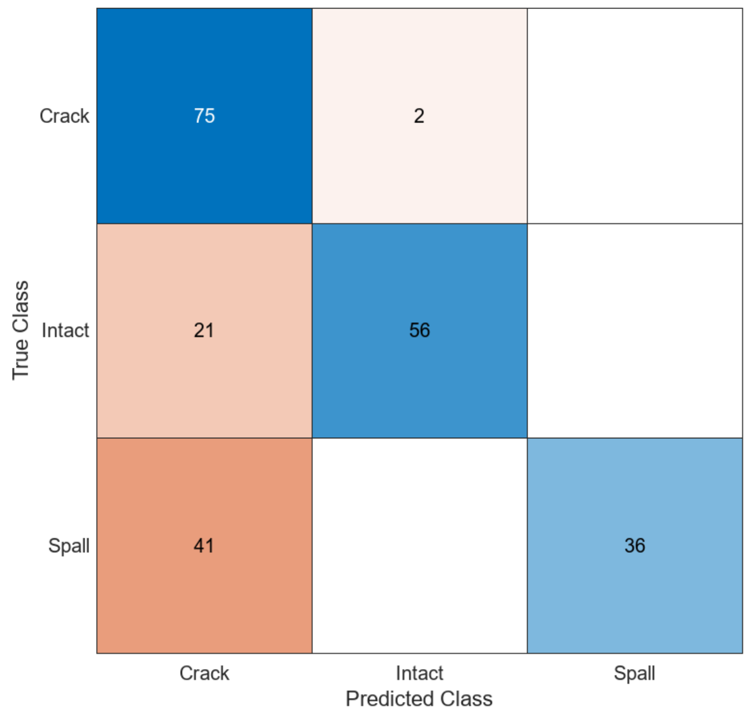


Figure B.21 Confusion matrix showing the classification results of the fine Gaussian SVM model.

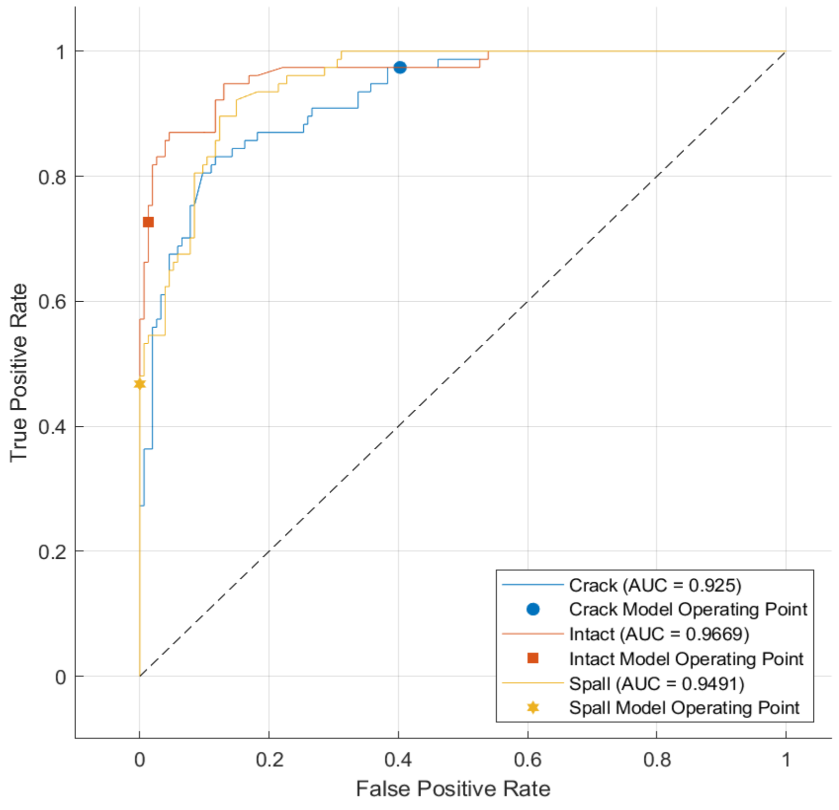


Figure B.22 ROC curve showing the fine Gaussian SVM model's performance.

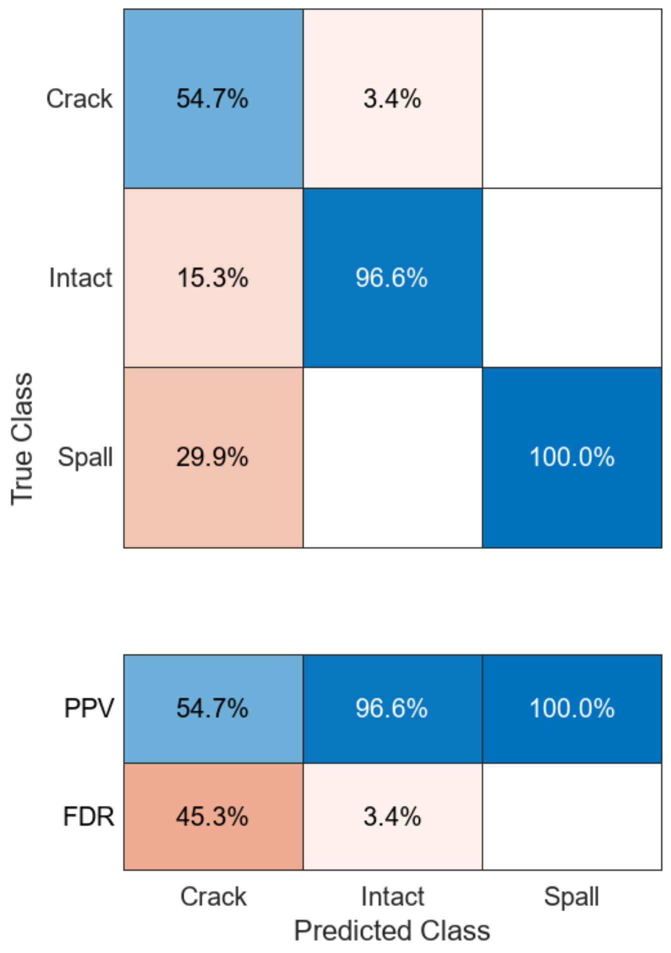


Figure B.23 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the fine Gaussian SVM model.

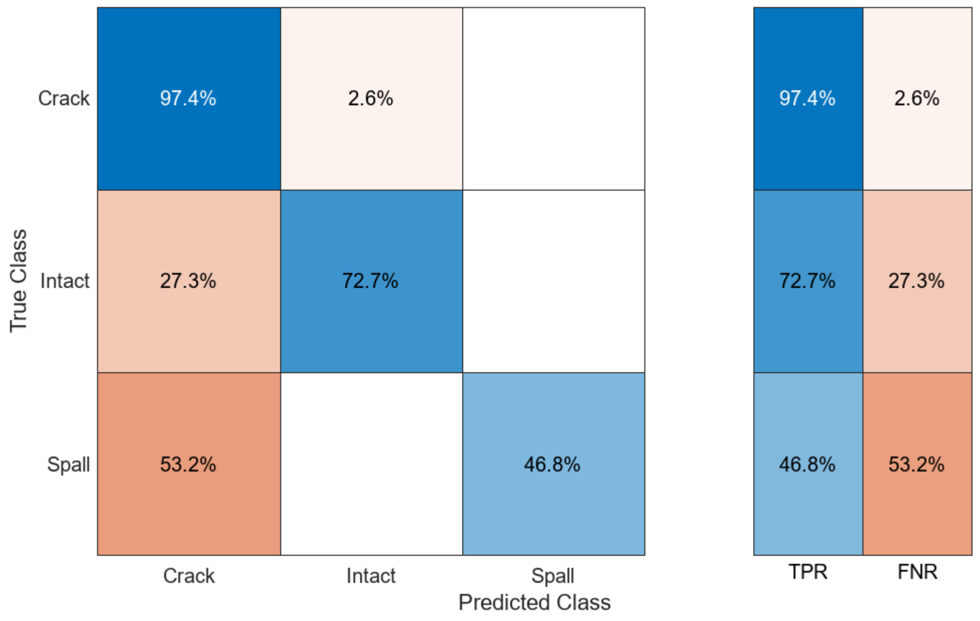


Figure B.24 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the fine Gaussian SVM model.

Medium Gaussian SVM model

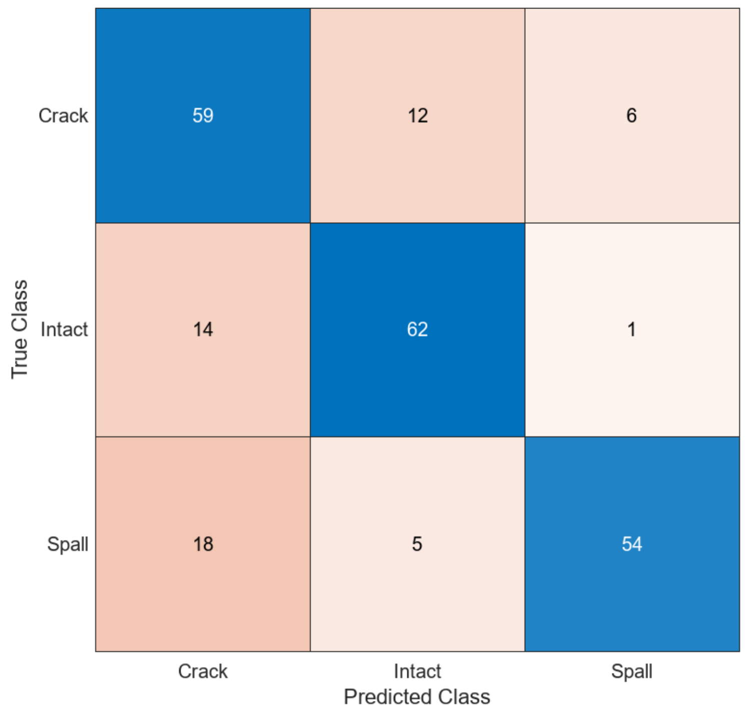


Figure B.25 Confusion matrix showing the classification results of the medium Gaussian SVM model.

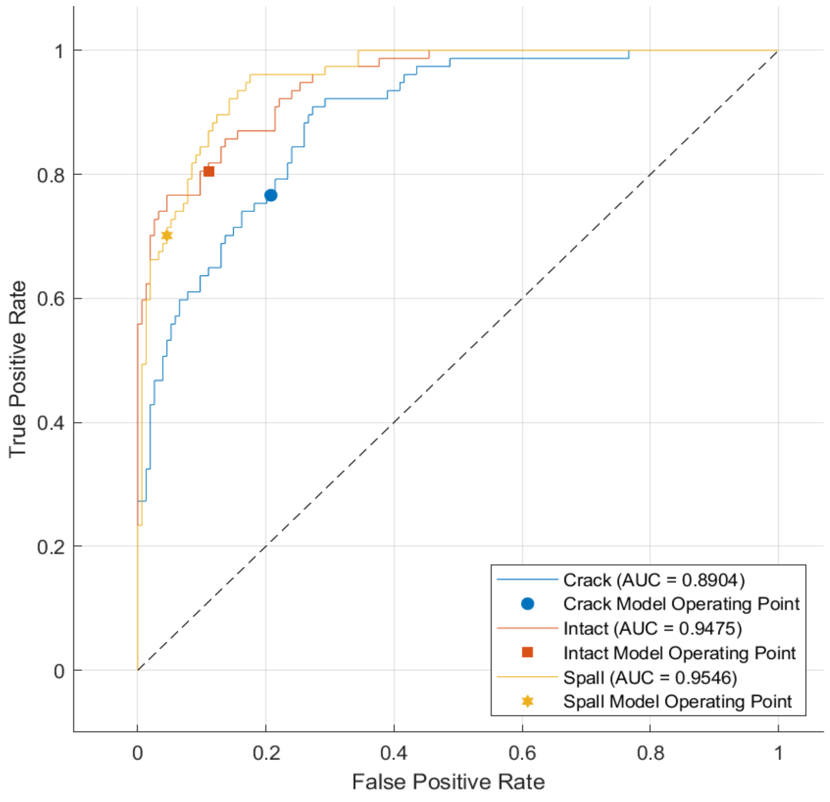


Figure B.26 ROC curve showing the medium Gaussian SVM model’s performance.

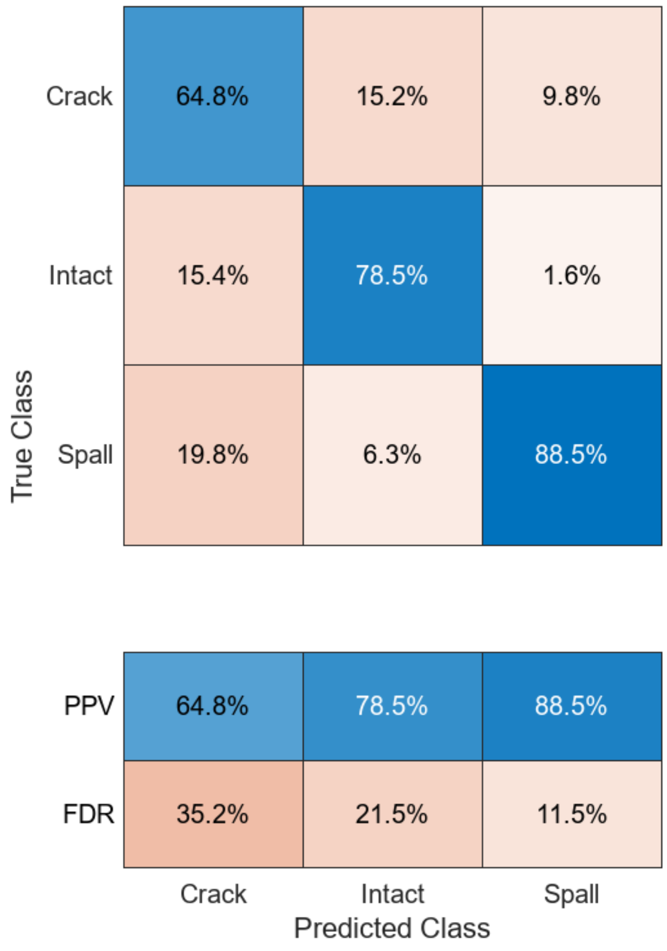


Figure B.27 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium Gaussian SVM model.

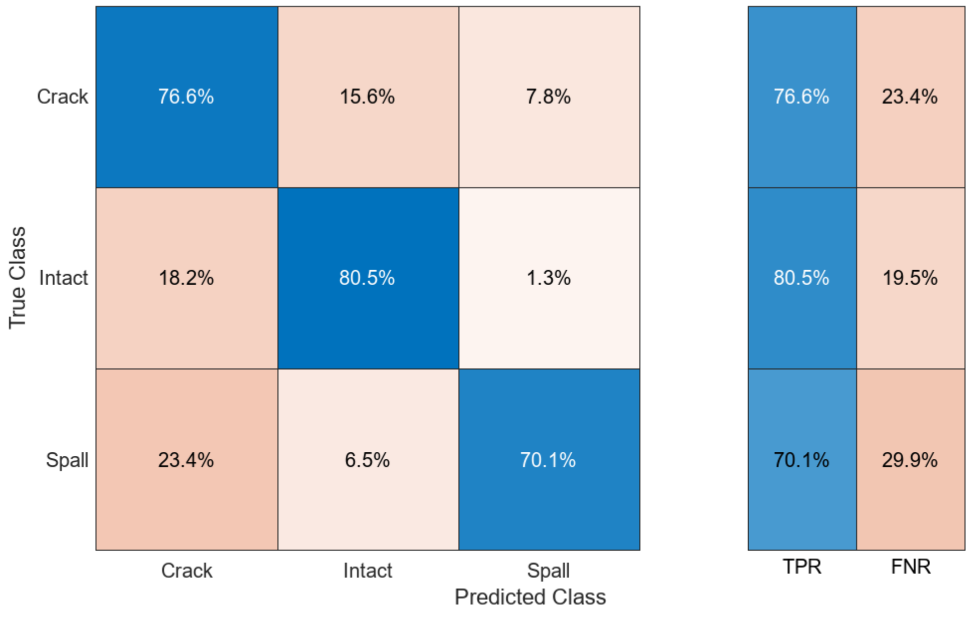


Figure B.28 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium Gaussian SVM model.

Coarse Gaussian SVM model

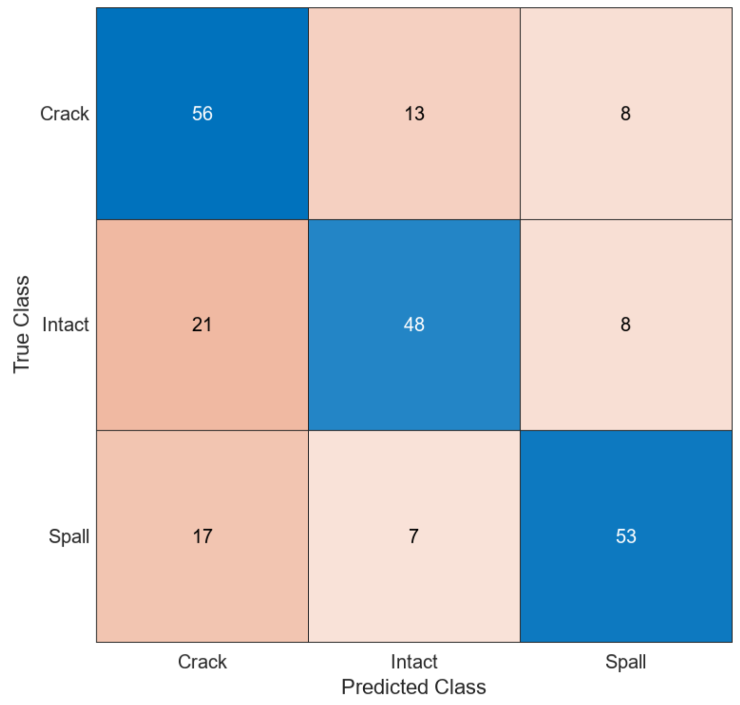


Figure B.29 Confusion matrix showing the classification results of the coarse Gaussian SVM model.

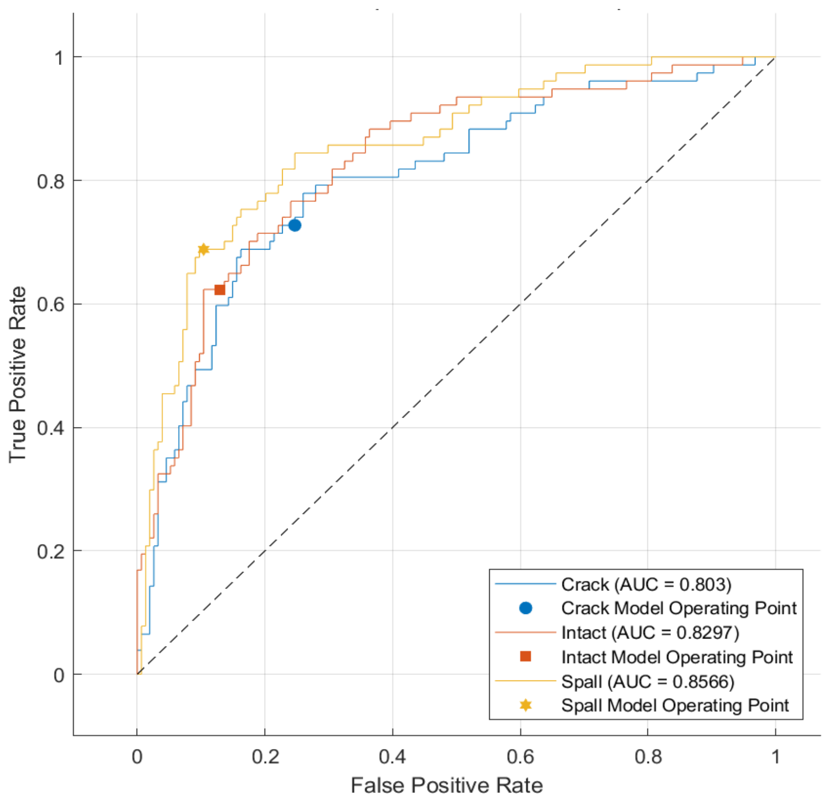


Figure B.30 ROC curve showing the coarse Gaussian SVM model's performance.

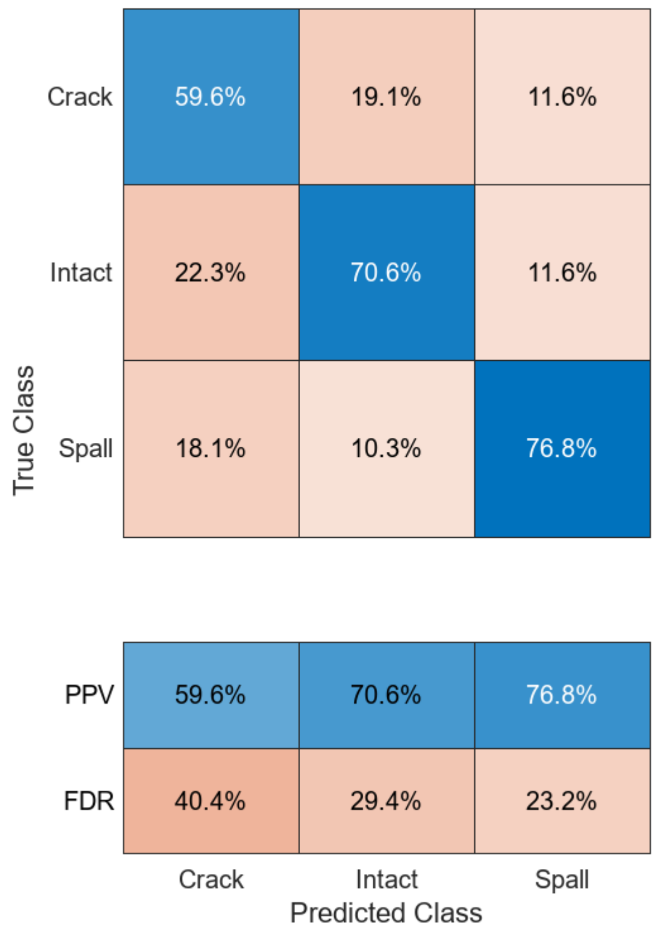


Figure B.31 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse Gaussian SVM model.

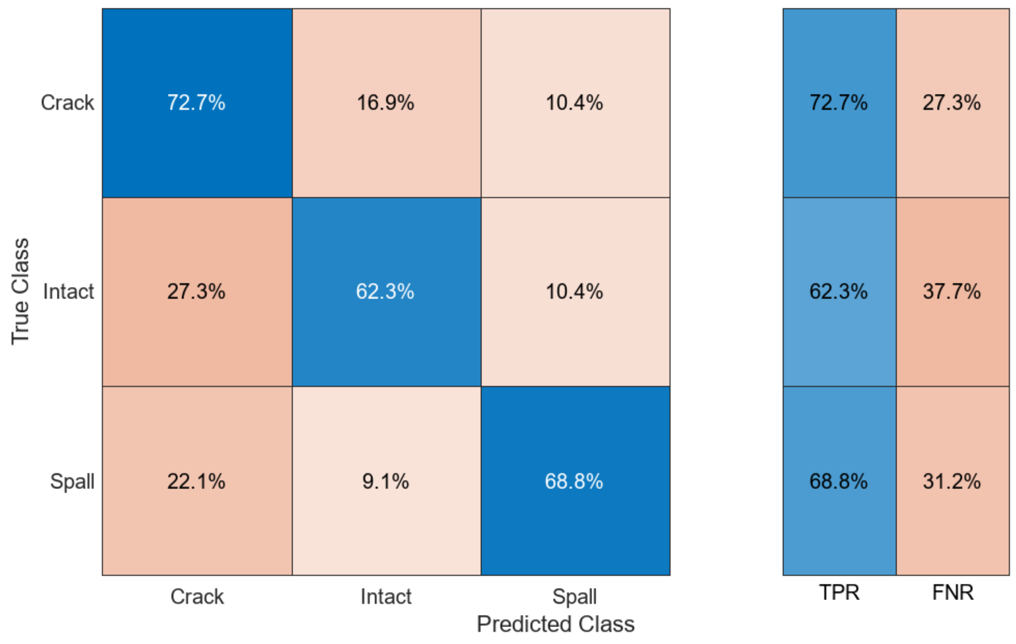


Figure B.32 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse Gaussian SVM model.

Optimisable SVM model

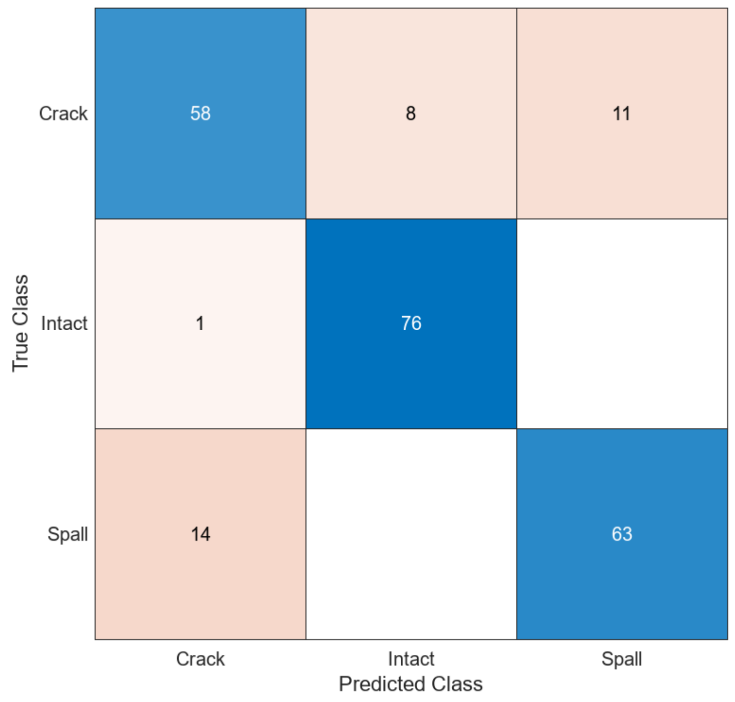


Figure B.33 Confusion matrix showing the classification results of the optimisable SVM model.

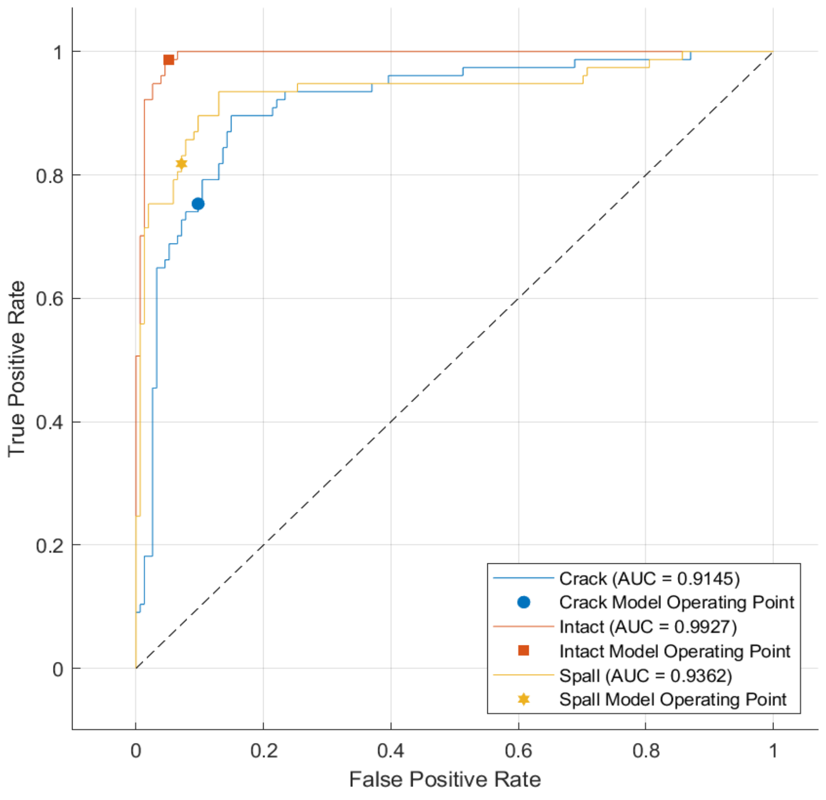


Figure B.34 ROC curve showing the optimisable SVM model's performance.

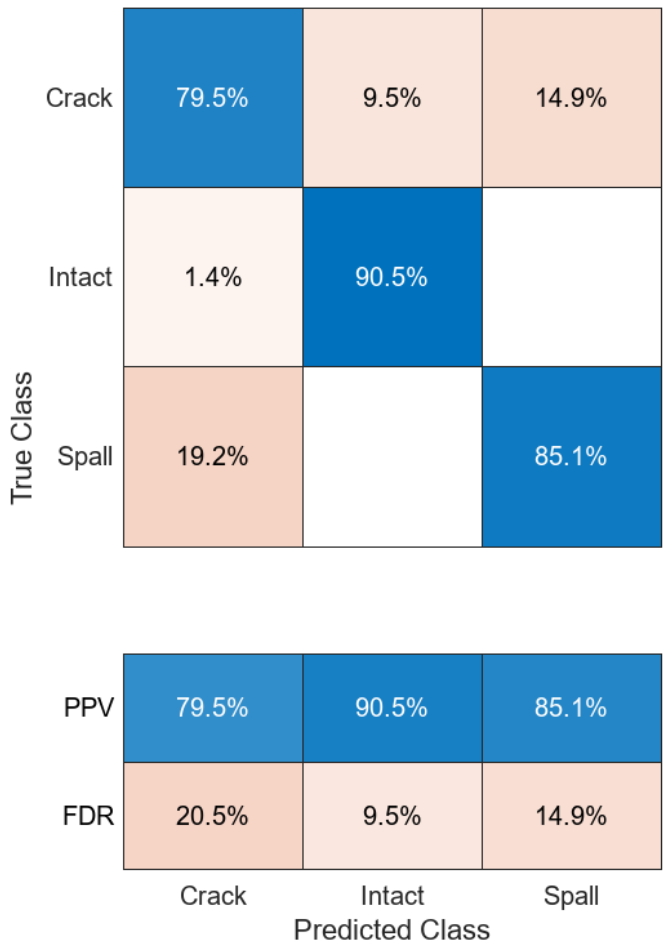


Figure B.35 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the optimisable SVM model.

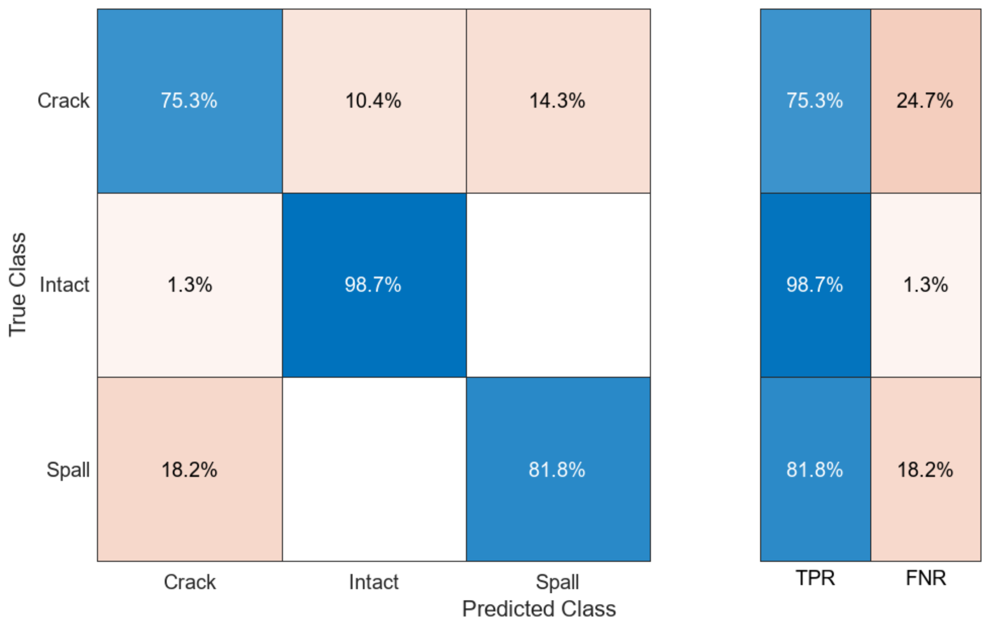


Figure B.36 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the optimisable SVM model.

Medium KNN model

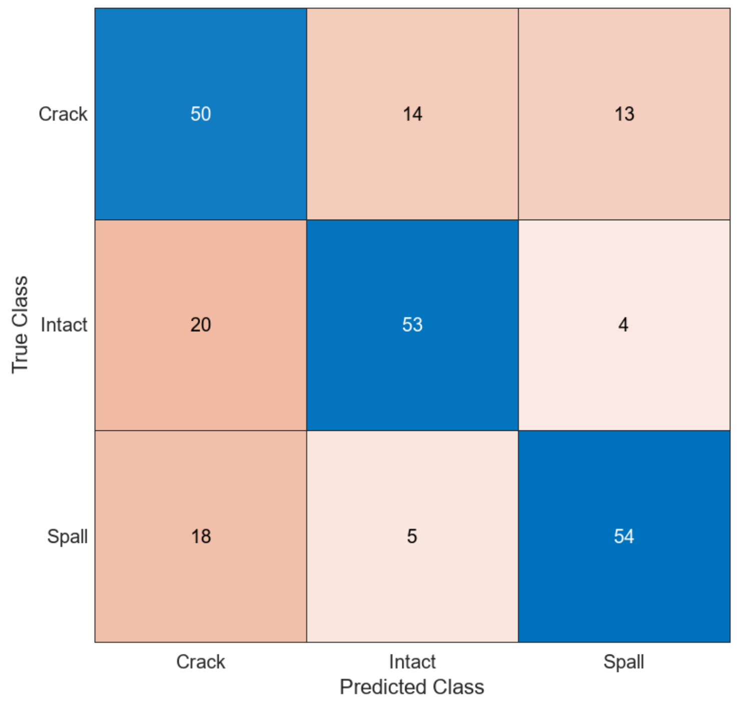


Figure B.37 Confusion matrix showing the classification results of the medium KNN model.

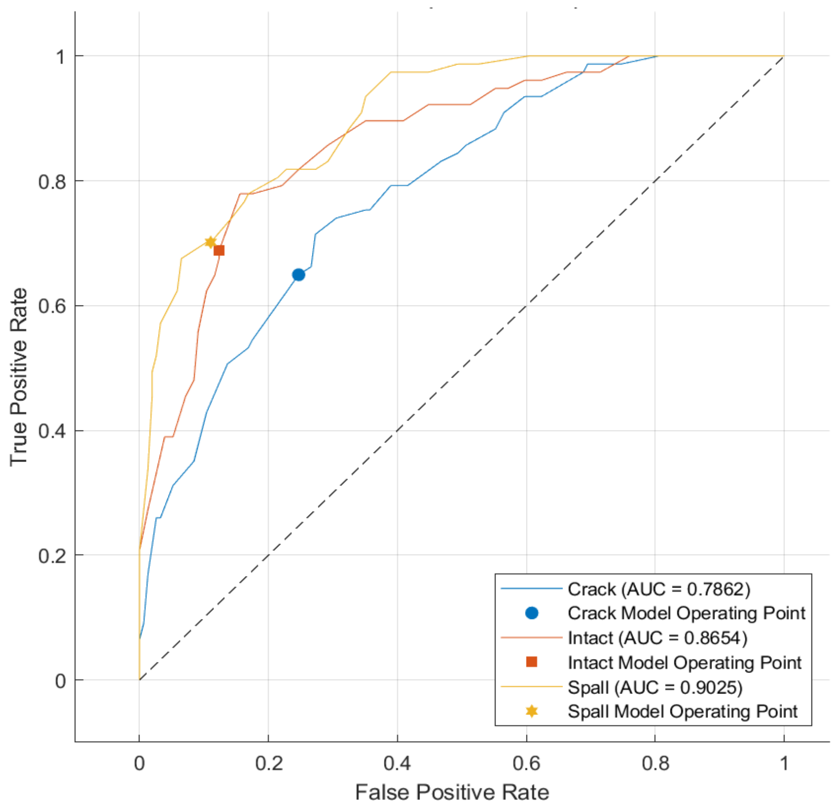


Figure B.38 ROC curve showing the medium KNN model's performance.

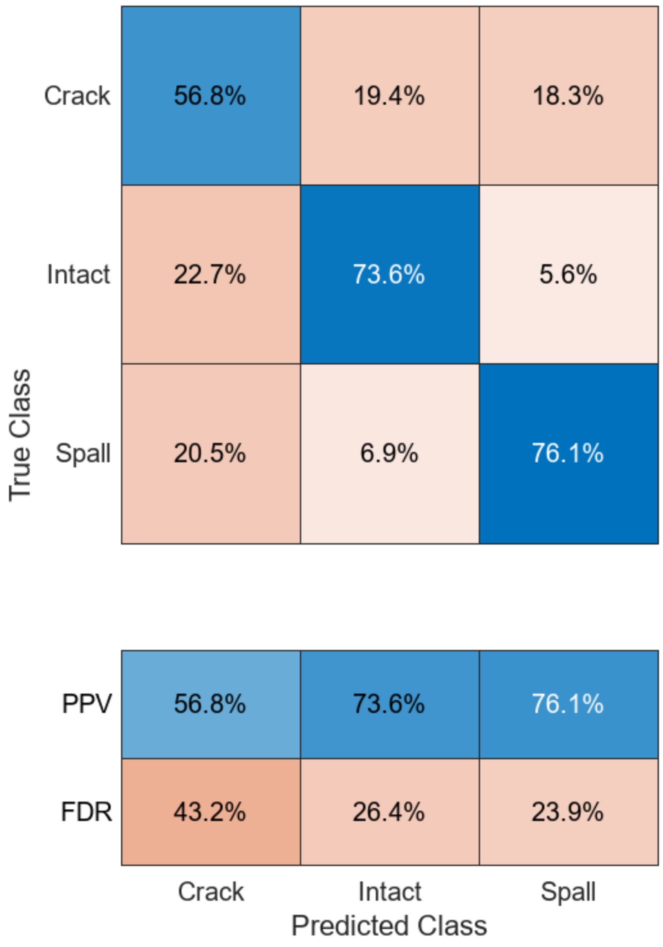


Figure B.39 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the medium KNN model.

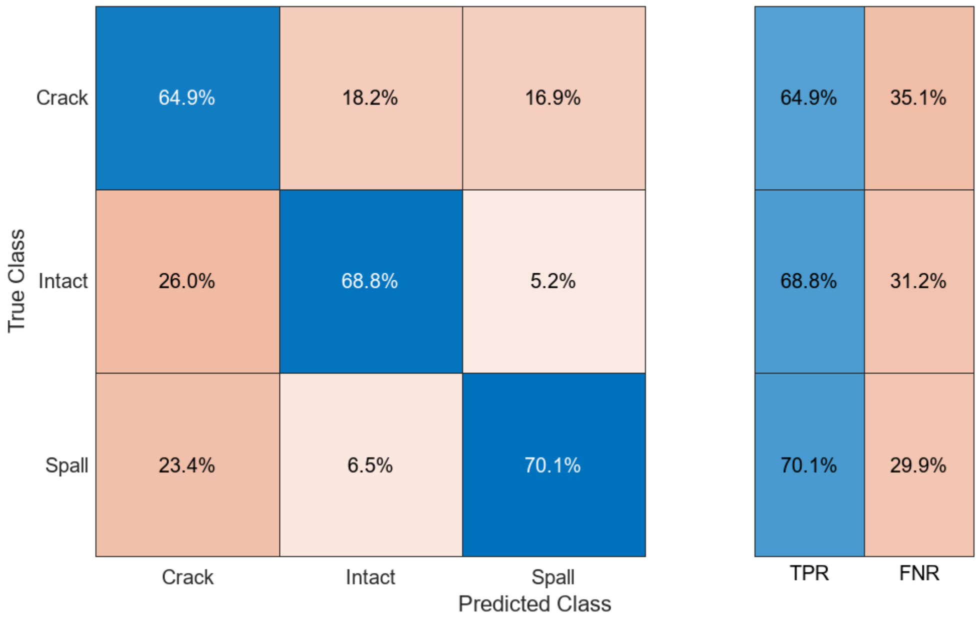


Figure B.40 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the medium KNN model.

Coarse KNN model

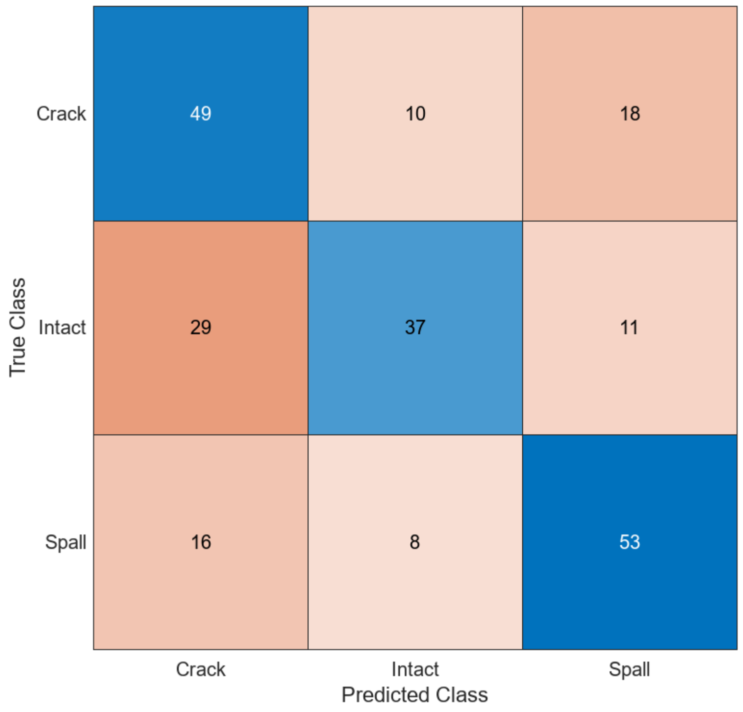


Figure B.41 Confusion matrix showing the classification results of the coarse KNN model.

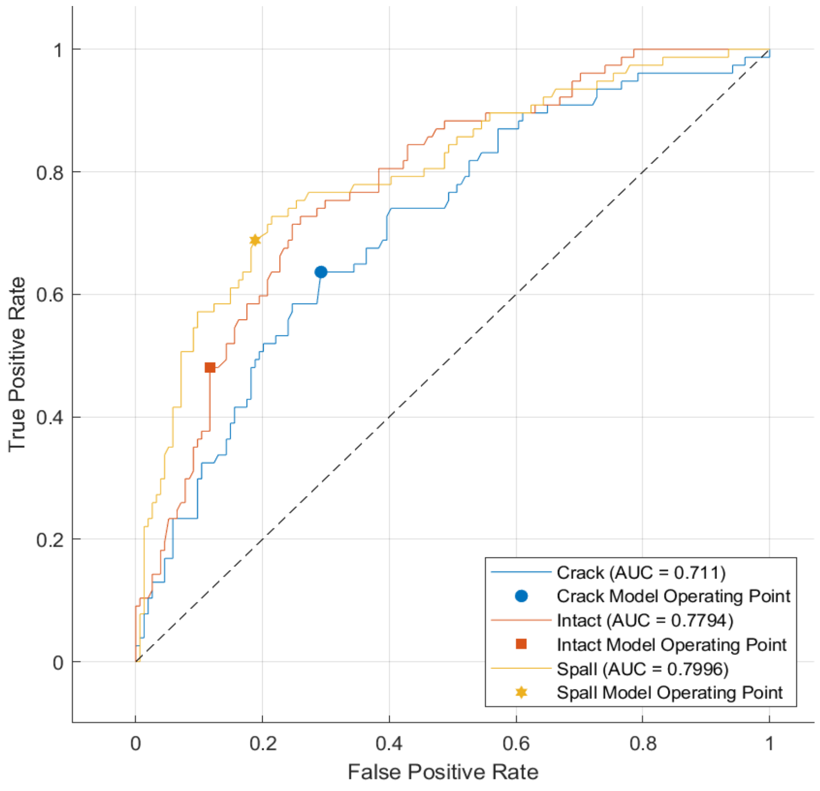


Figure B.42 ROC curve showing the coarse KNN model's performance.

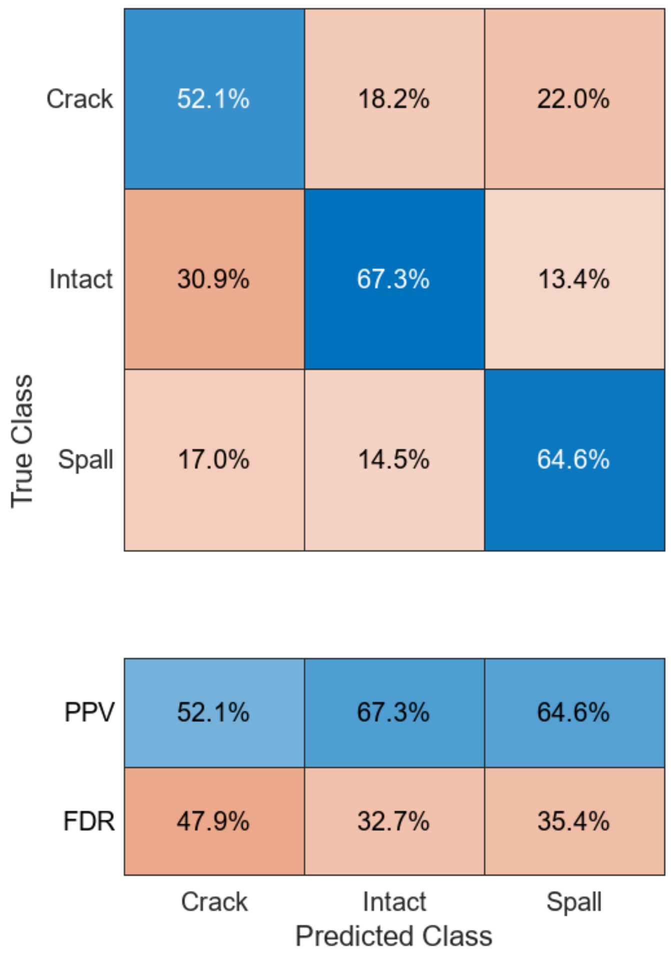


Figure B.43 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the coarse KNN model.

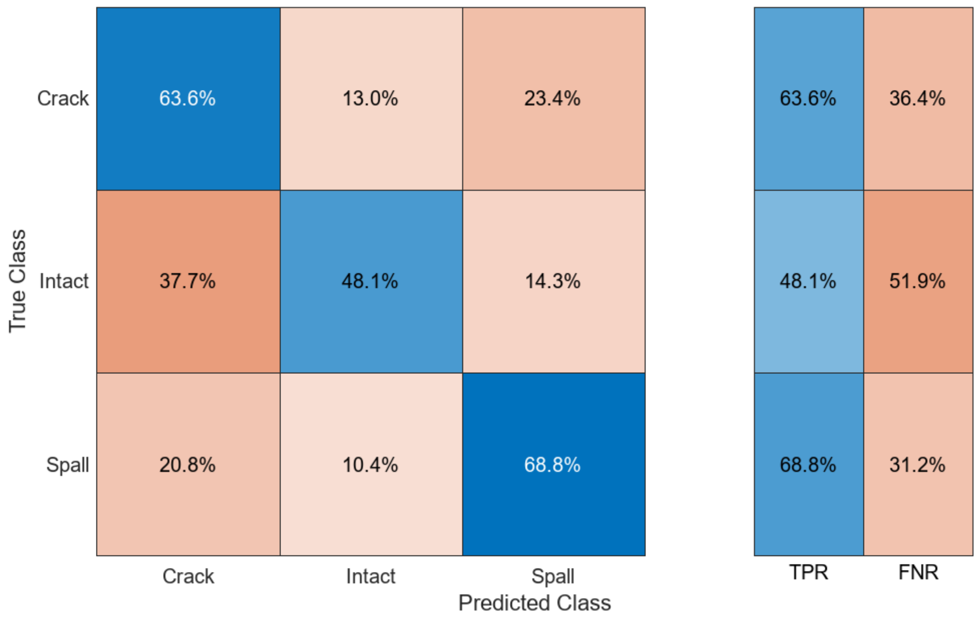


Figure B.44 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the coarse KNN model.

Cosine KNN model

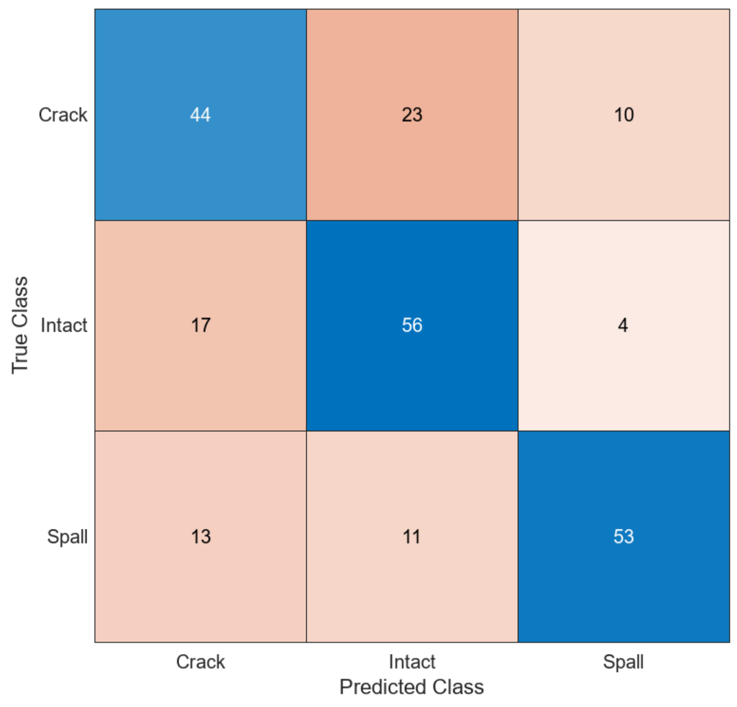


Figure B.45 Confusion matrix showing the classification results of the cosine KNN model.

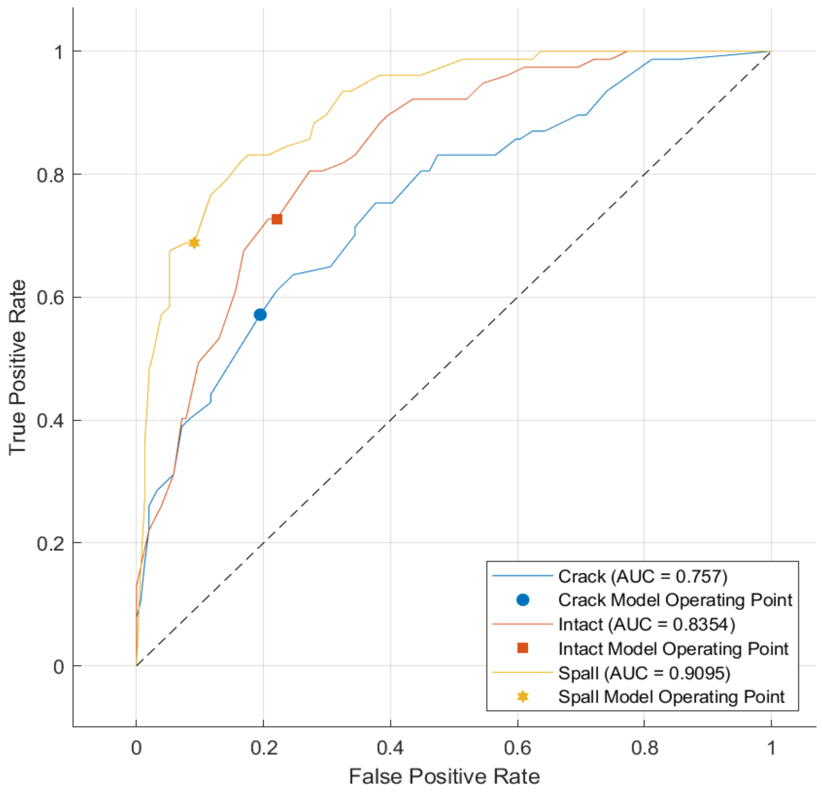


Figure B.46 ROC curve showing the cosine KNN model's performance.

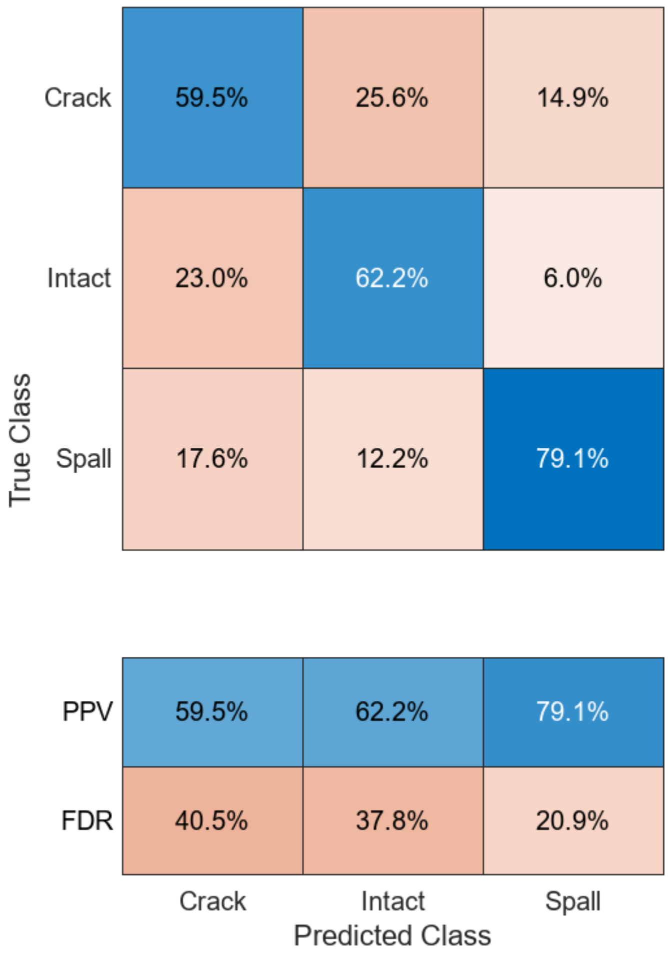


Figure B.47 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the cosine KNN model.

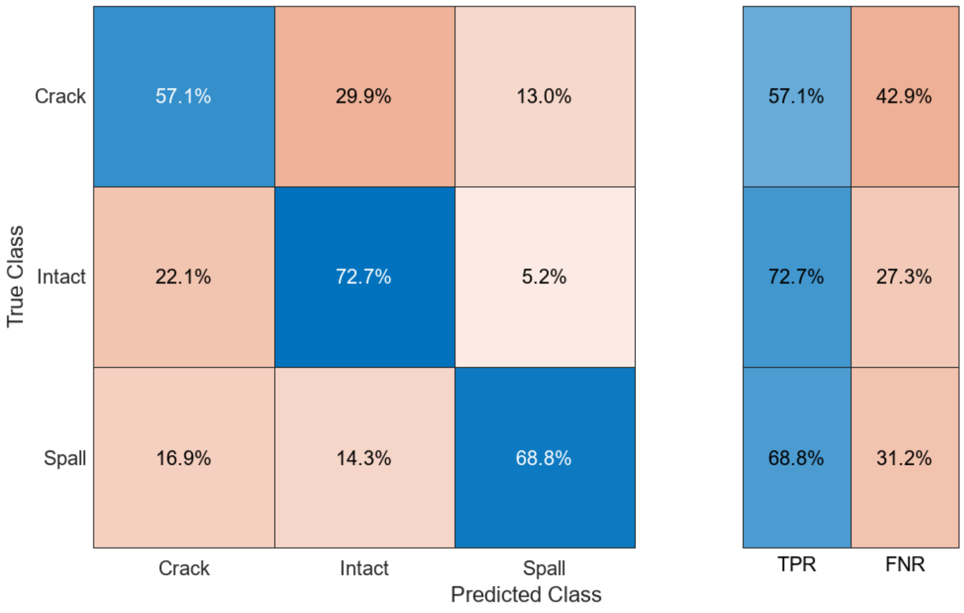


Figure B.48 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the cosine KNN model.

Cubic KNN model

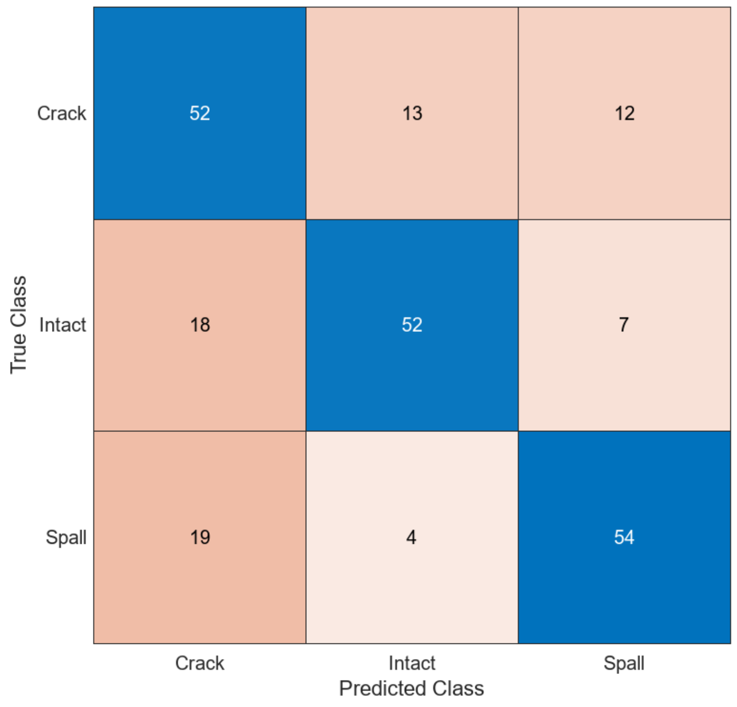


Figure B.49 Confusion matrix showing the classification results of the cubic KNN model.

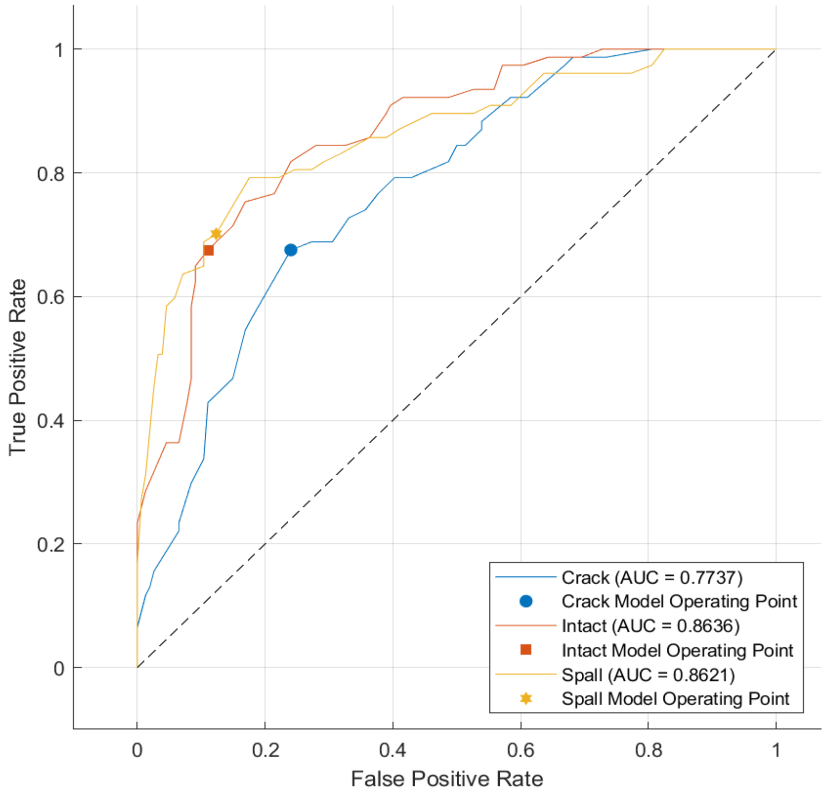


Figure B.50 ROC curve showing the cubic KNN model's performance.

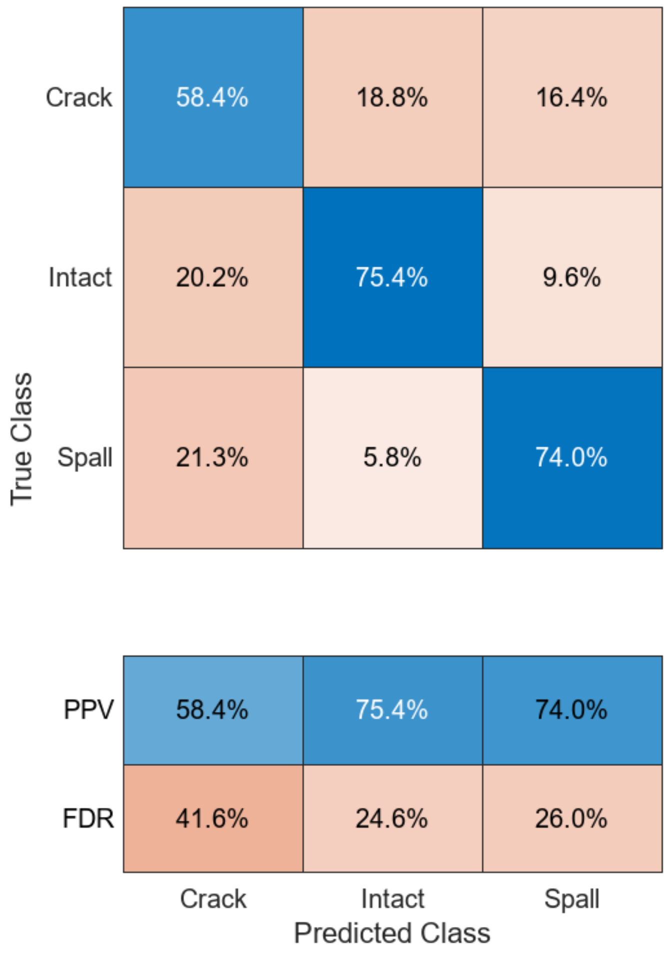


Figure B.51 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the cubic KNN model.

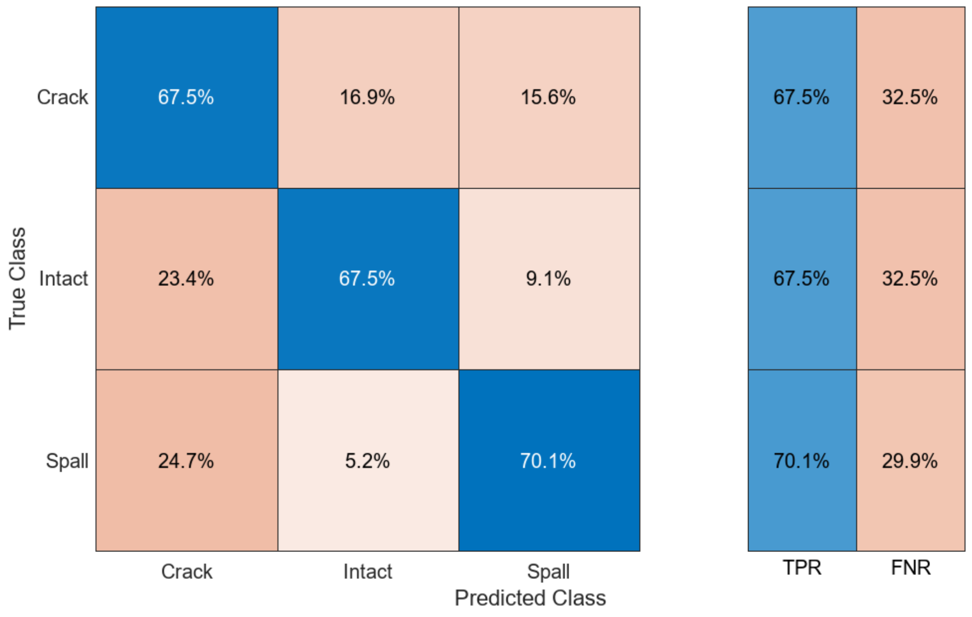


Figure B.52 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the cubic KNN model.

Weighted KNN model

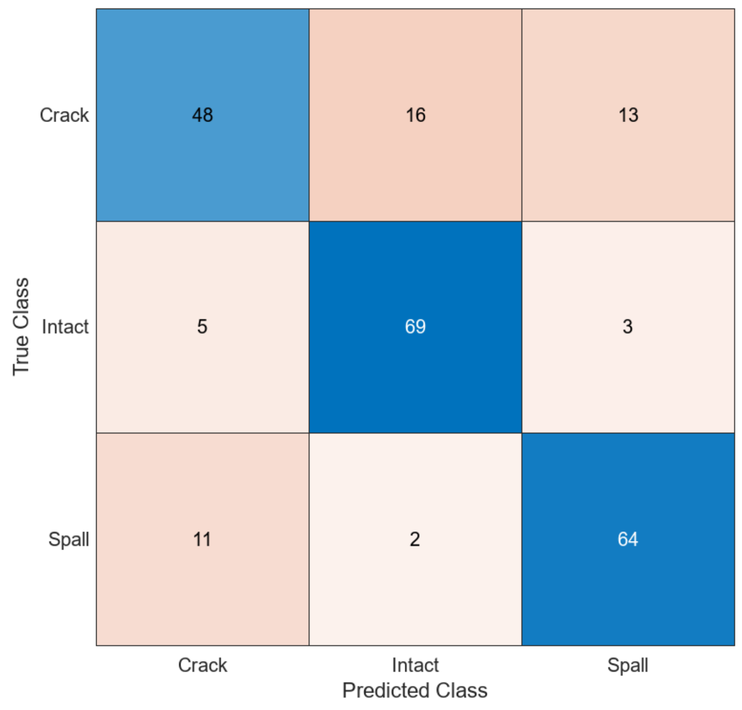


Figure B.53 Confusion matrix showing the classification results of the weighted KNN model.

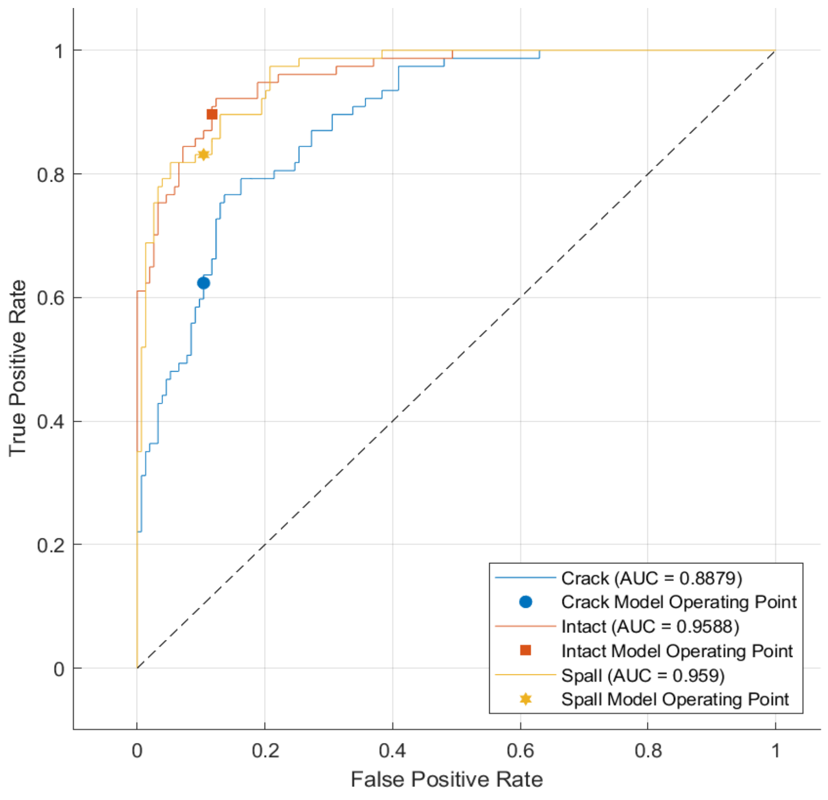
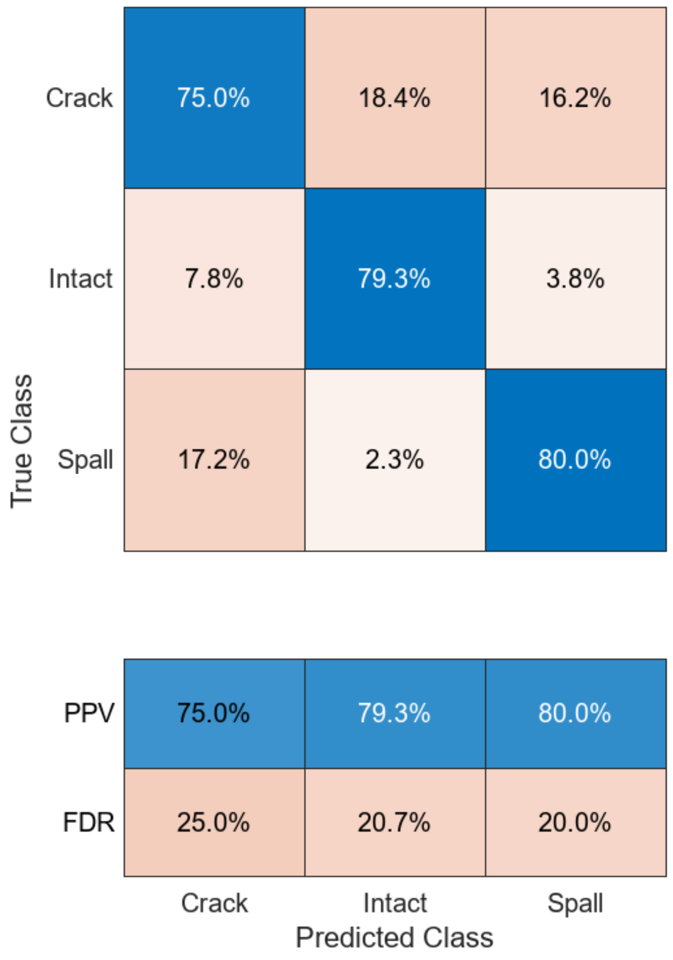


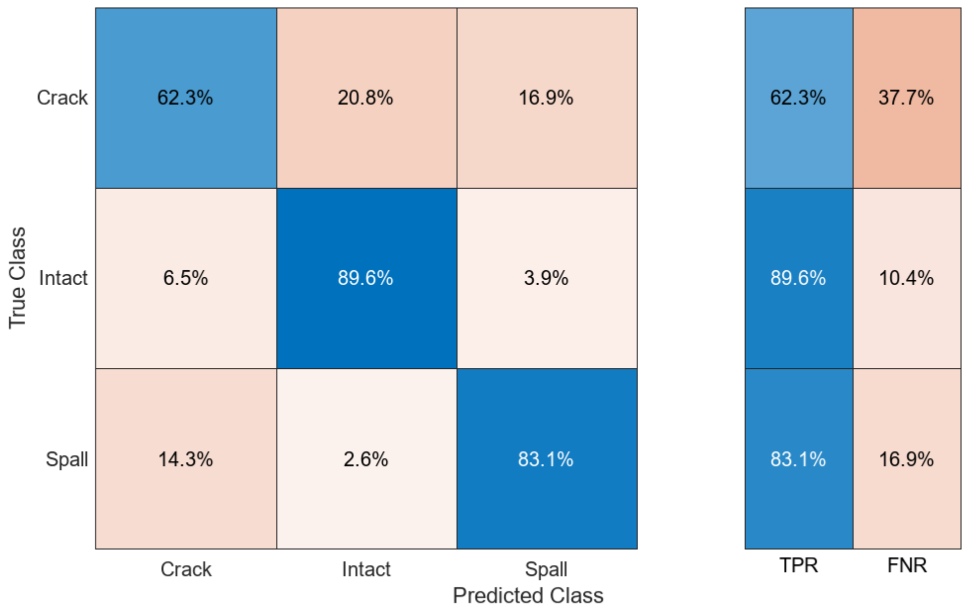
Figure B.54 ROC curve showing the weighted KNN model's performance.



True Class	Crack	Intact	Spall
Crack	75.0%	18.4%	16.2%
Intact	7.8%	79.3%	3.8%
Spall	17.2%	2.3%	80.0%

	Crack	Intact	Spall
PPV	75.0%	79.3%	80.0%
FDR	25.0%	20.7%	20.0%

Figure B.55 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the weighted KNN model.



True Class	Crack	Intact	Spall	TPR	FNR
Crack	62.3%	20.8%	16.9%	62.3%	37.7%
Intact	6.5%	89.6%	3.9%	89.6%	10.4%
Spall	14.3%	2.6%	83.1%	83.1%	16.9%

Figure B.56 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the weighted KNN model.

Optimisable KNN model

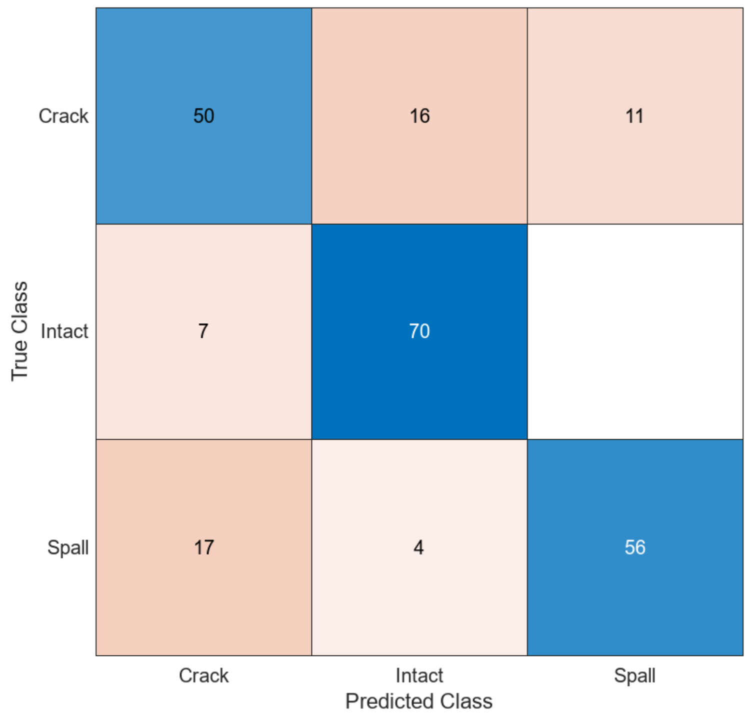


Figure B.57 Confusion matrix showing the classification results of the optimisable KNN model.

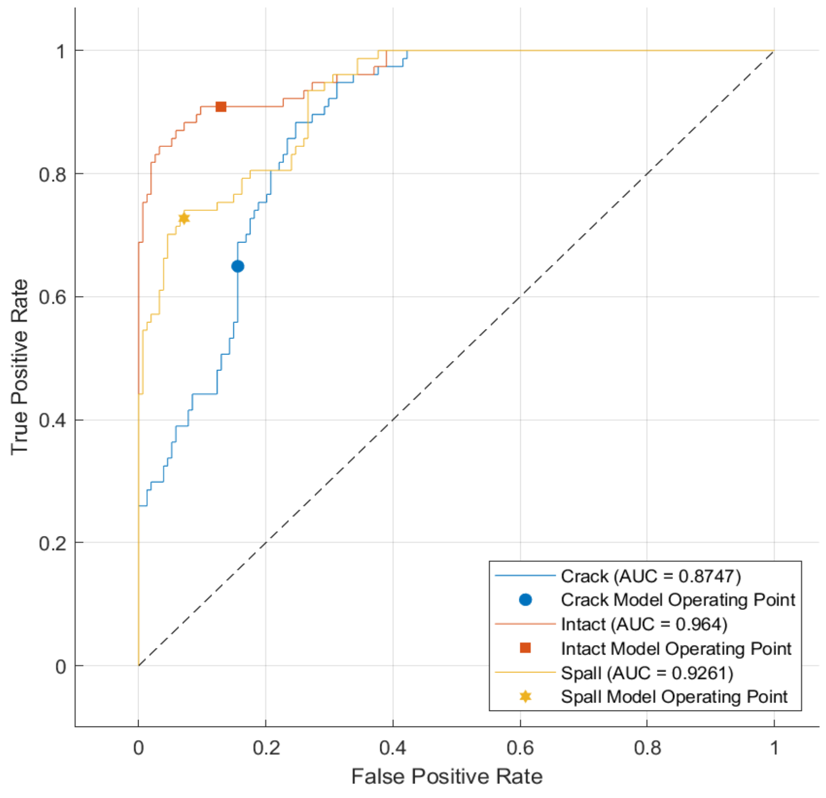


Figure B.58 ROC curve showing the optimisable KNN model’s performance.

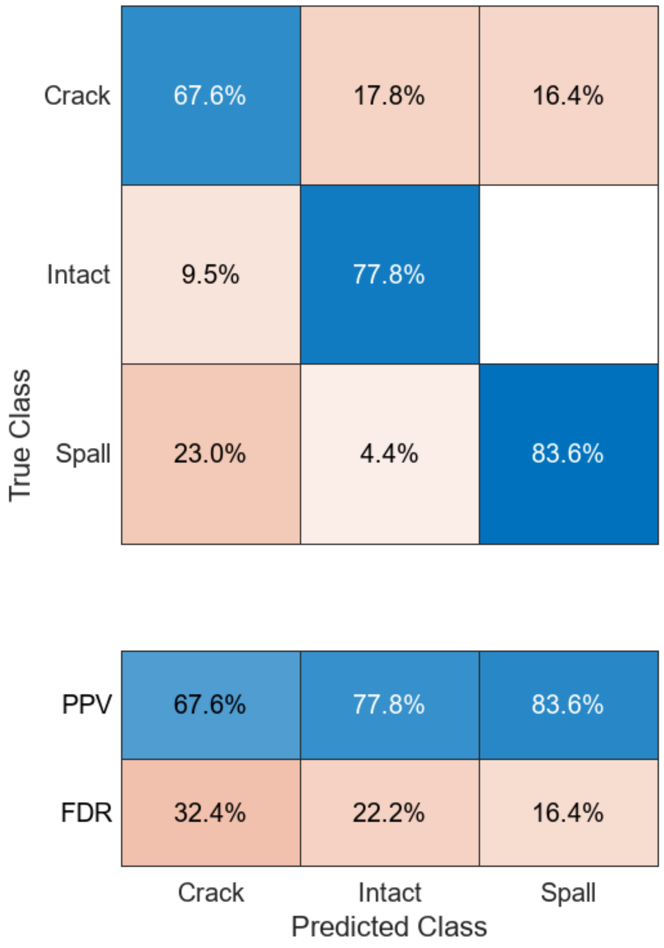


Figure B.59 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the optimisable KNN model.

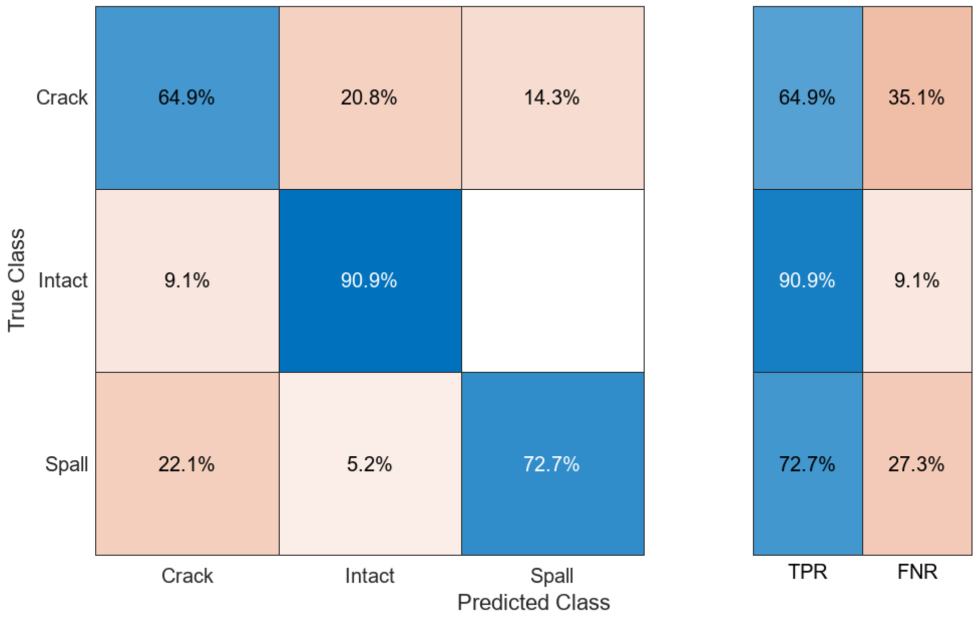


Figure B.60 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the optimisable KNN model.

Narrow artificial neural network model

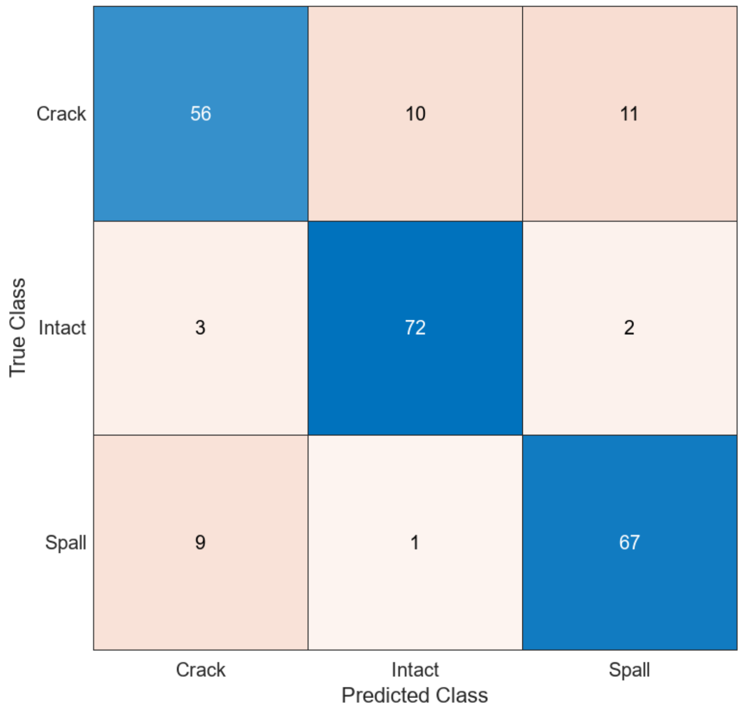


Figure B.61 Confusion matrix showing the classification results of the narrow artificial neural network.

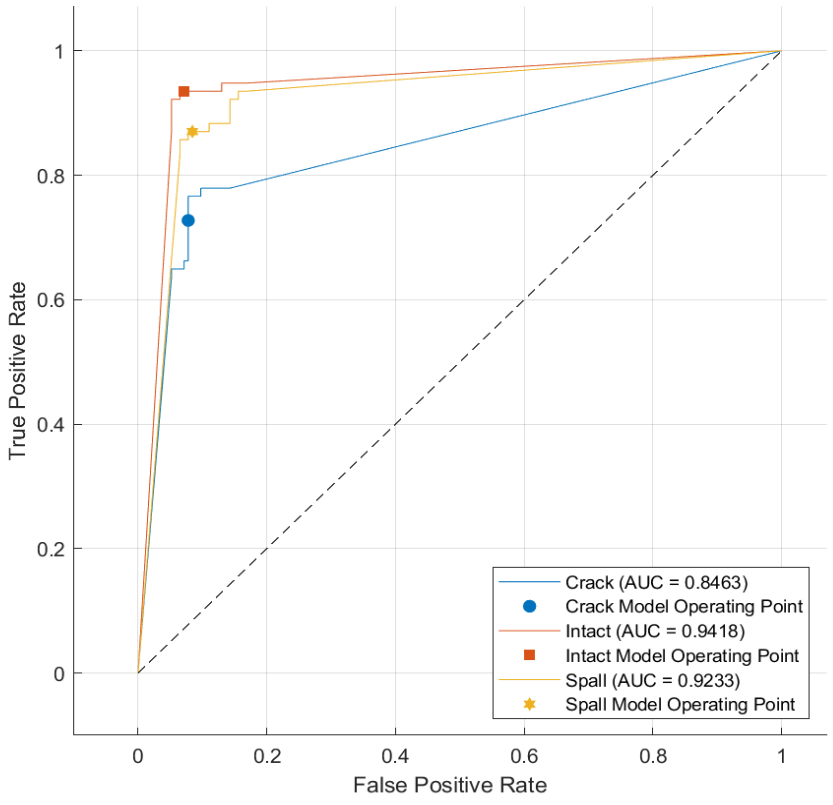


Figure B.62 ROC curve showing the narrow artificial neural network performance.

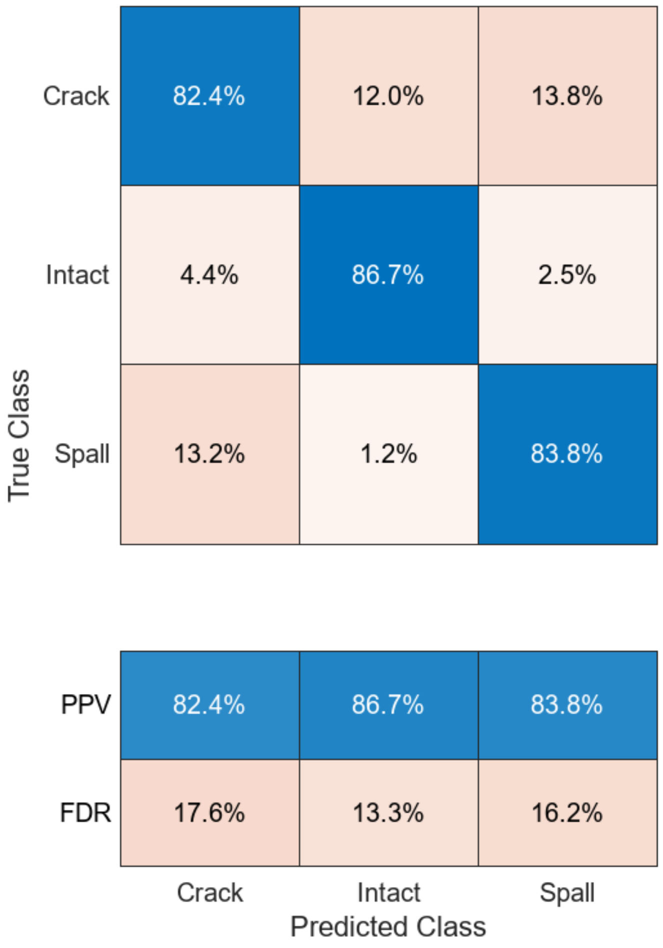


Figure B.63 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the narrow artificial neural network.

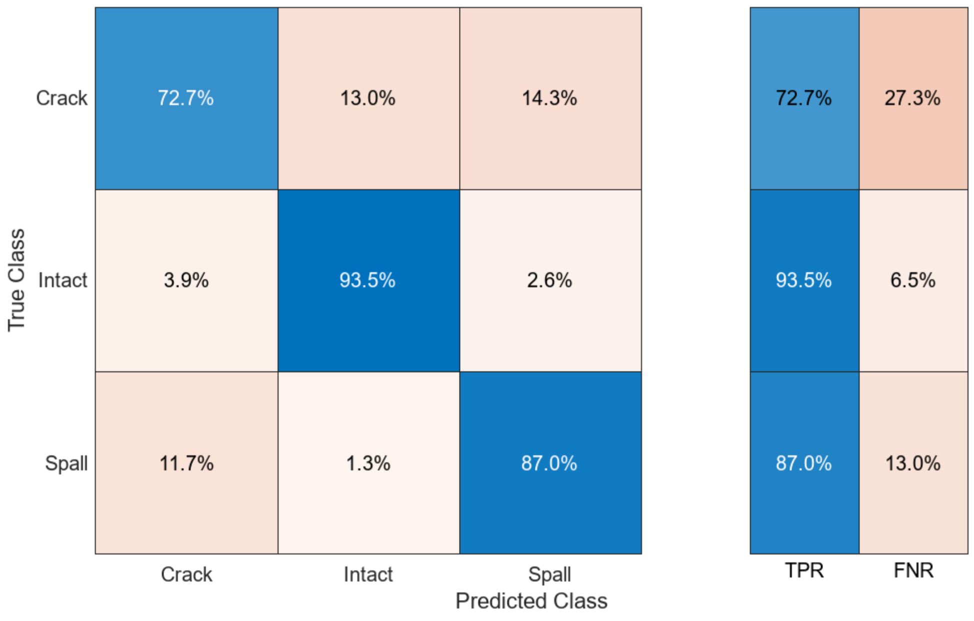


Figure B.64 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the narrow artificial neural network.

Wide artificial neural network model

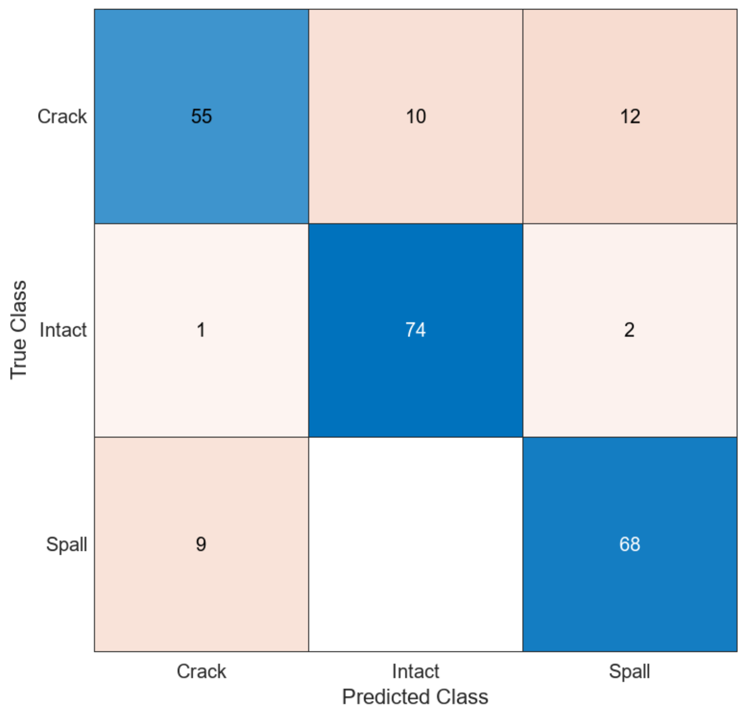


Figure B.65 Confusion matrix showing the classification results of the wide artificial neural network.

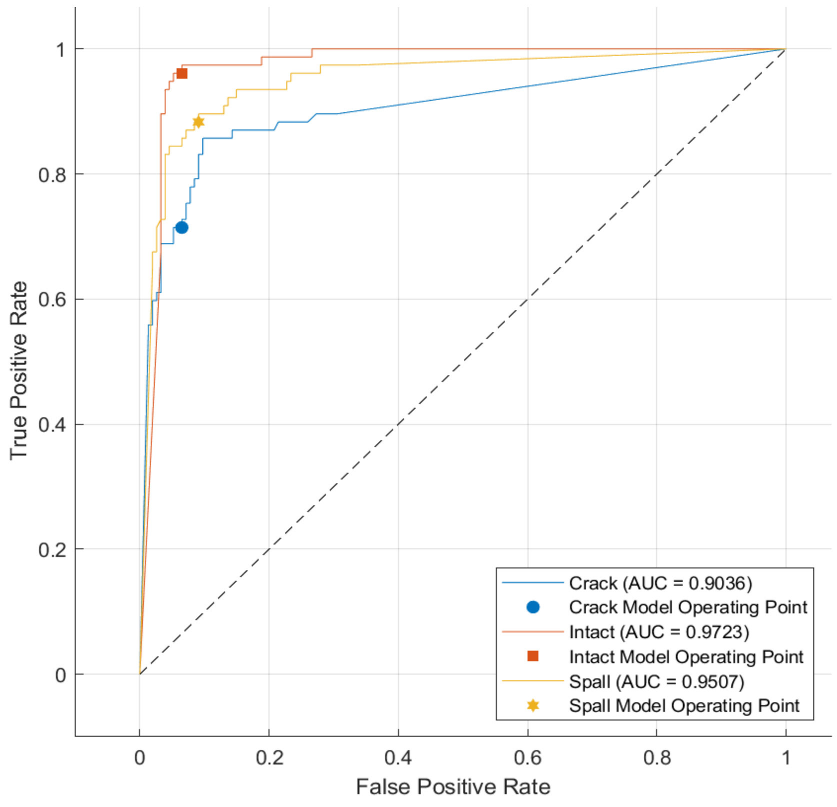


Figure B.66 ROC curve showing the wide artificial neural network performance.

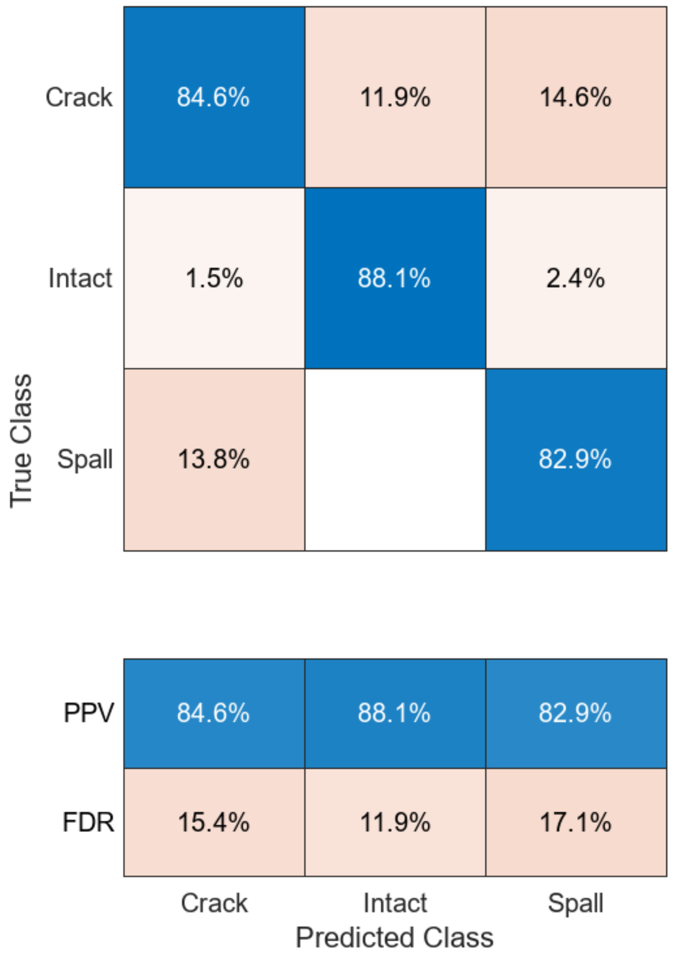


Figure B.67 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class using the wide artificial neural network.

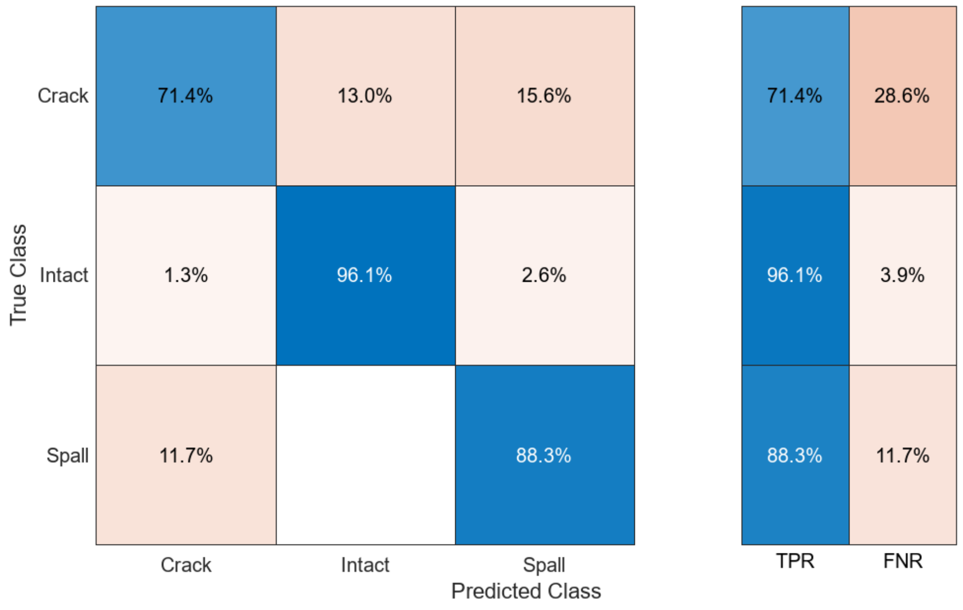


Figure B.68 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the wide artificial neural network.

Bi-layered artificial neural network model

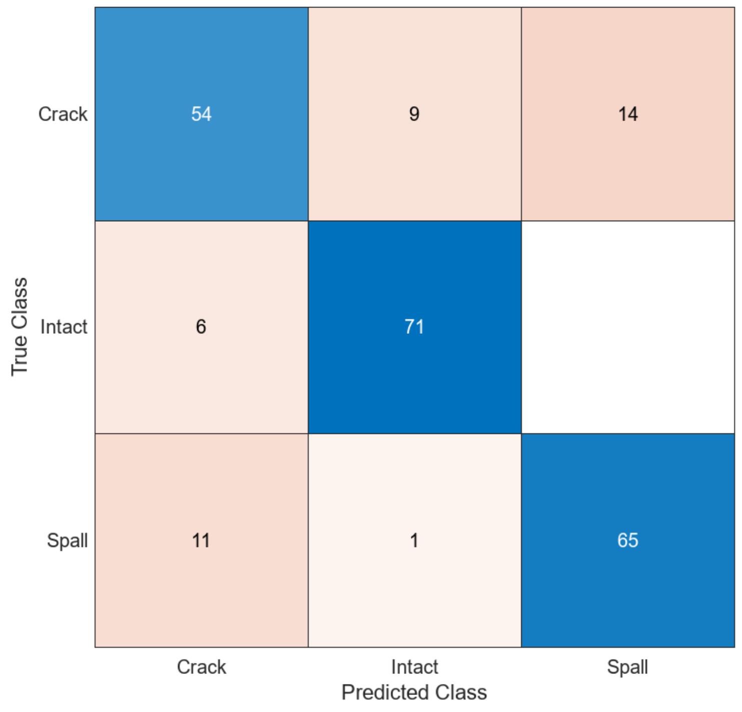


Figure B.69 Confusion matrix showing the classification results of the bi-layered artificial neural network.

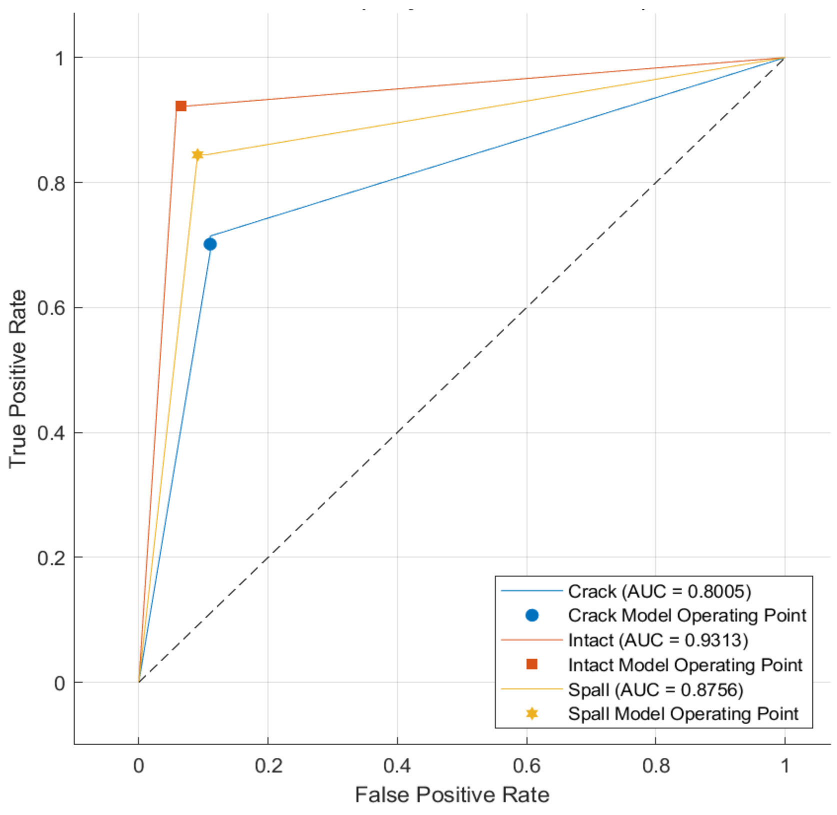


Figure B.70 ROC curve showing the bi-layered artificial neural network performance.

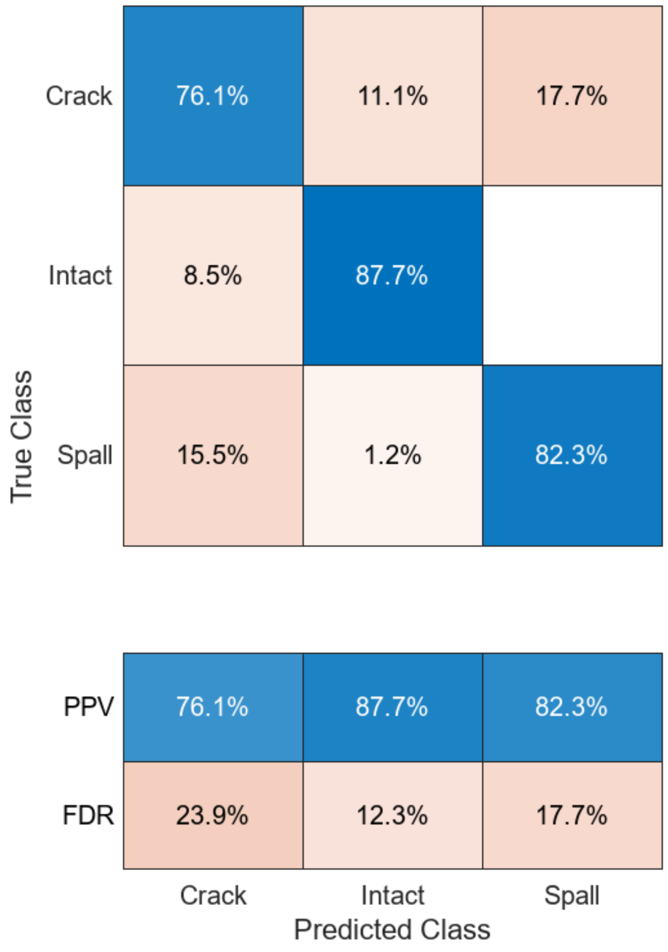


Figure B.71 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the bi-layered artificial neural network.

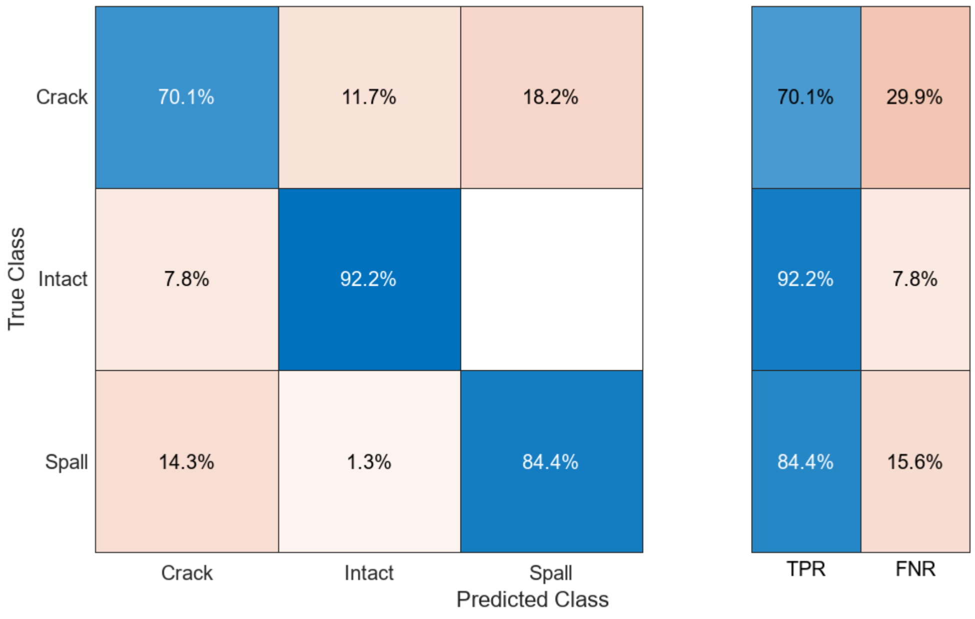


Figure B.72 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the bi-layered artificial neural network.

Tri-layered artificial neural network model

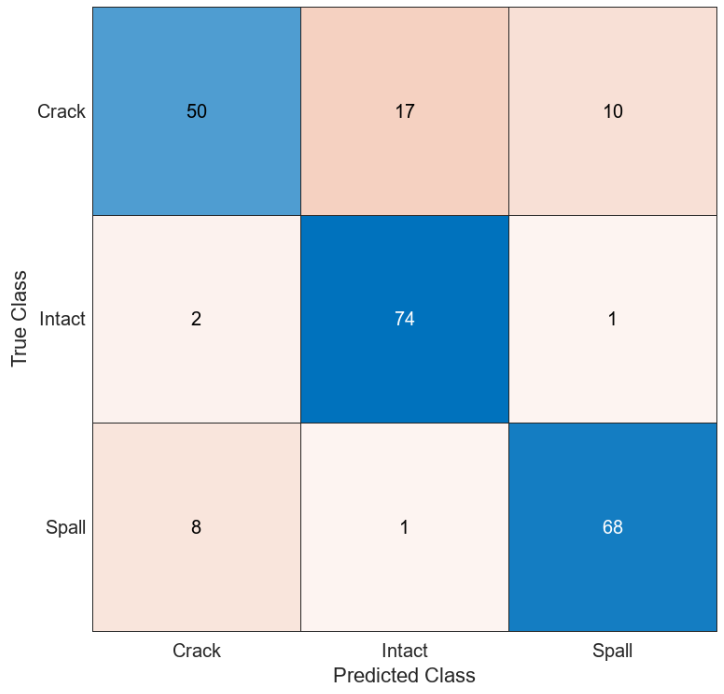


Figure B.73 Confusion matrix showing the classification results of the tri-layered artificial neural network.

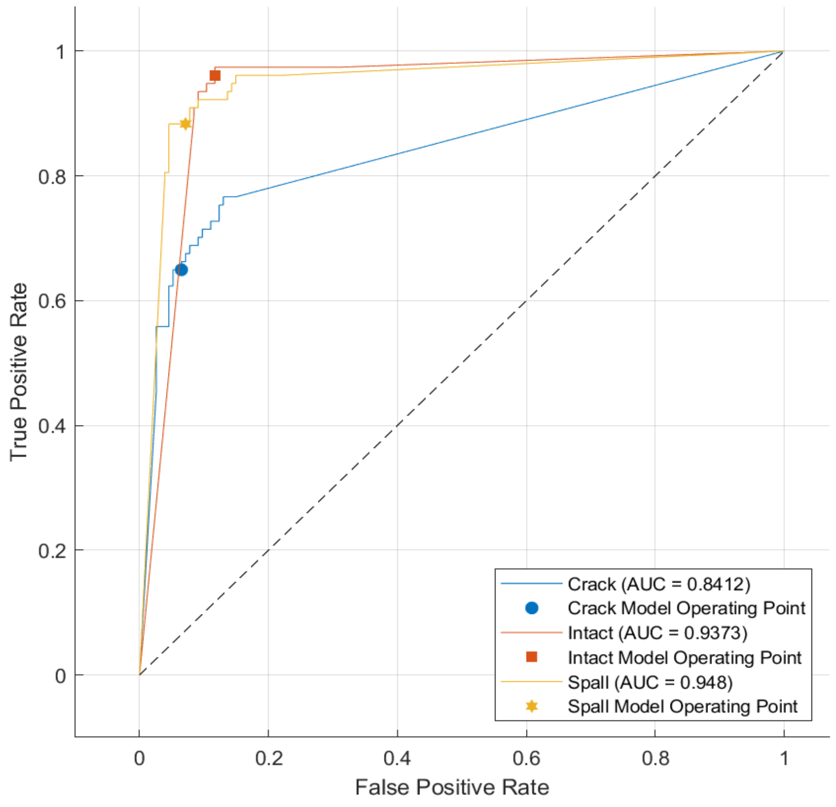


Figure B.74 ROC curve showing the tri-layered artificial neural network performance.

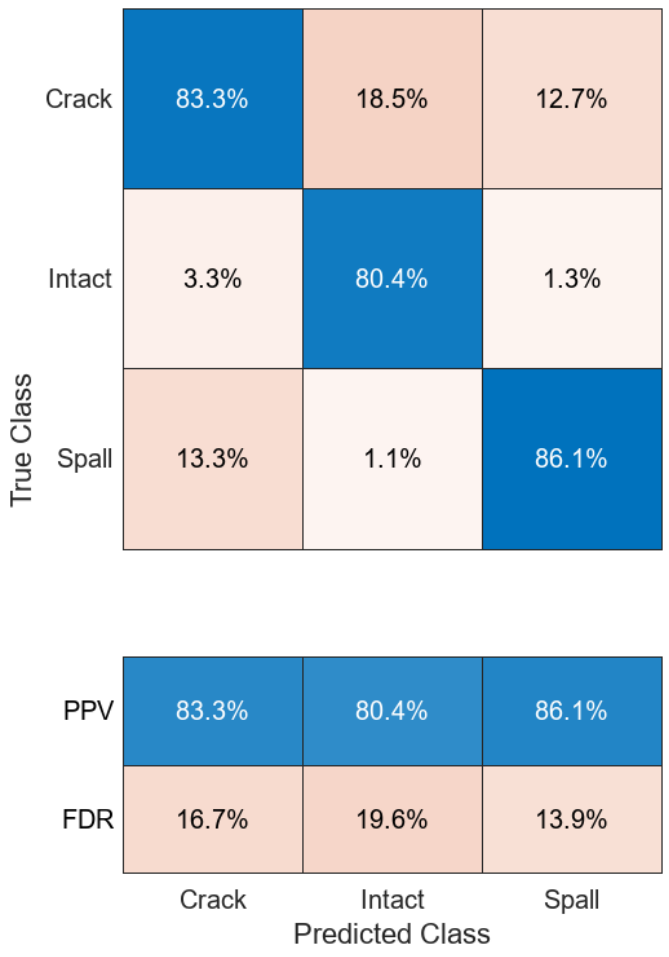


Figure B.75 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the tri-layered artificial neural network.

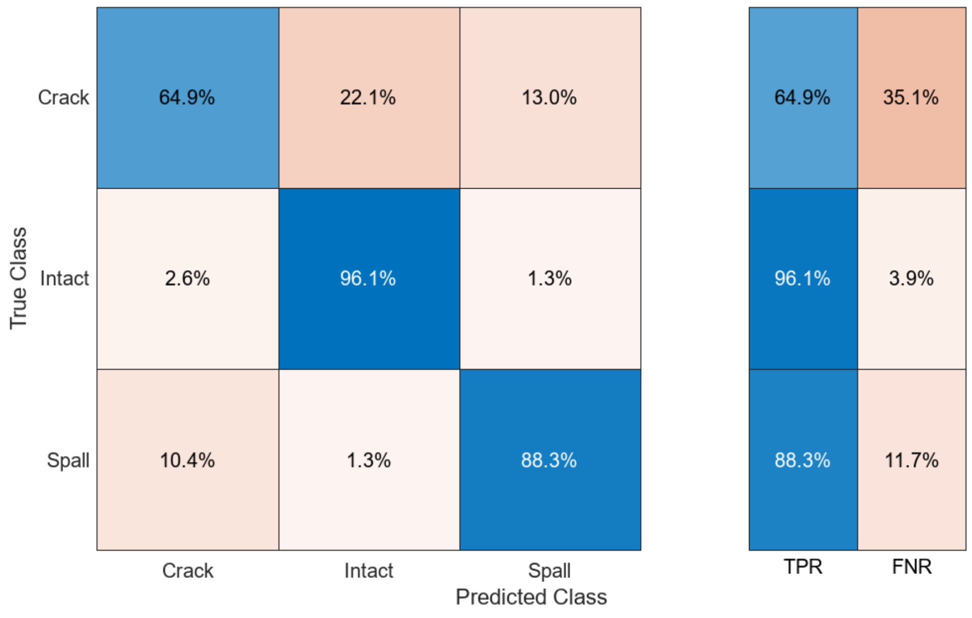


Figure B.76 The recall or true positive rate (TPR) and false negative rate (FNR) were achieved for each damage class, using the tri-layered artificial neural network.

Optimisable artificial neural network model

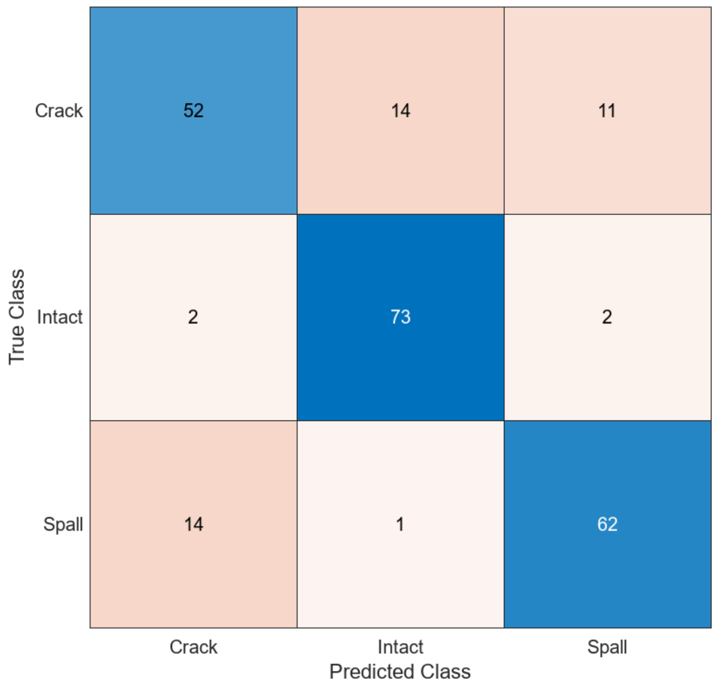


Figure B.77 Confusion matrix showing the classification results of the optimisable artificial neural network.

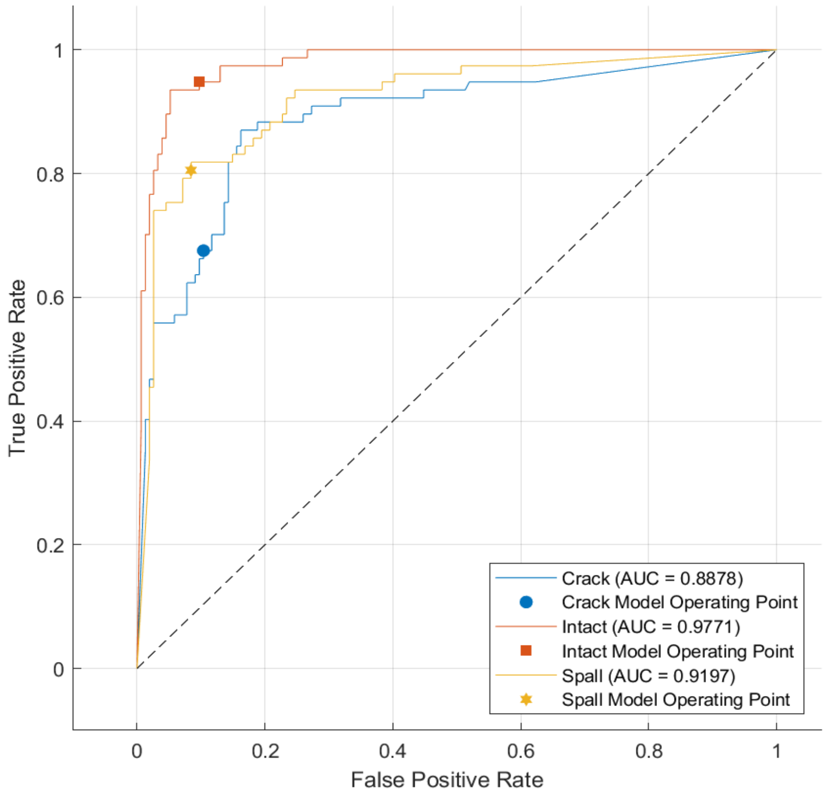


Figure B.78 ROC curve showing the optimisable artificial neural network performance.

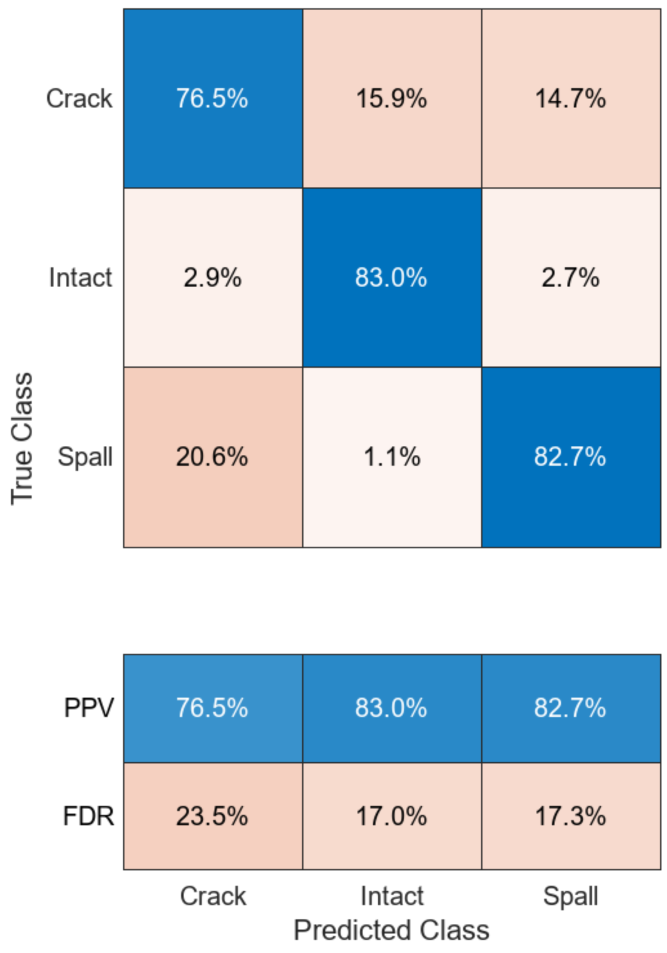


Figure B.79 The precision or positive predictive value (PPV) and false discovery rate (FDR) were achieved for each damage class, using the optimisable artificial neural network.

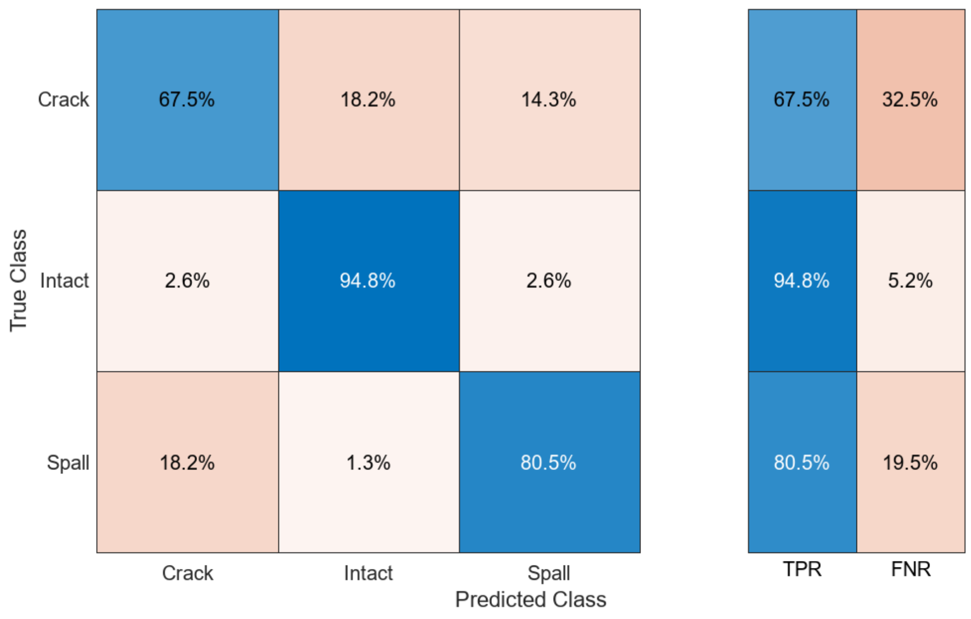


Figure B.80 The recall or true positive rate (TPR) and false negative rate (FNR) achieved for each damage class, using the optimisable artificial neural network.

Appendix C. Image processing and crack extent results

The following figures show the results obtained in determining the extent of crack damage using an image processing methodology.

Crack "a"

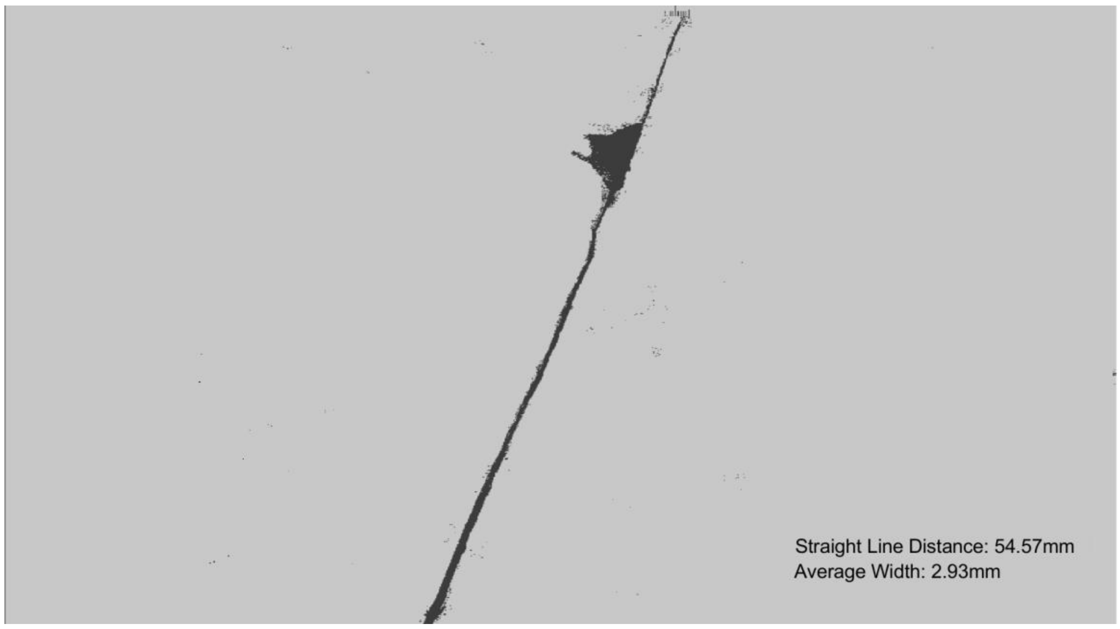


Figure C.1 Linear crack length and average crack width obtained for crack "a".

Crack "b"

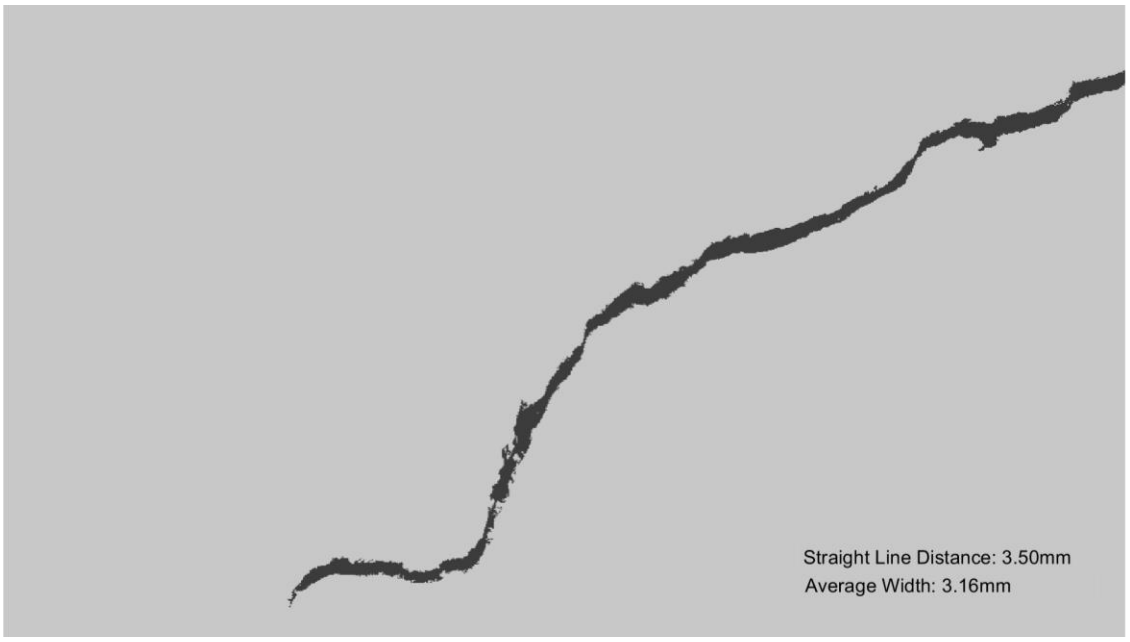


Figure C.2 Linear crack length and average crack width obtained for crack "b".

Crack "c"

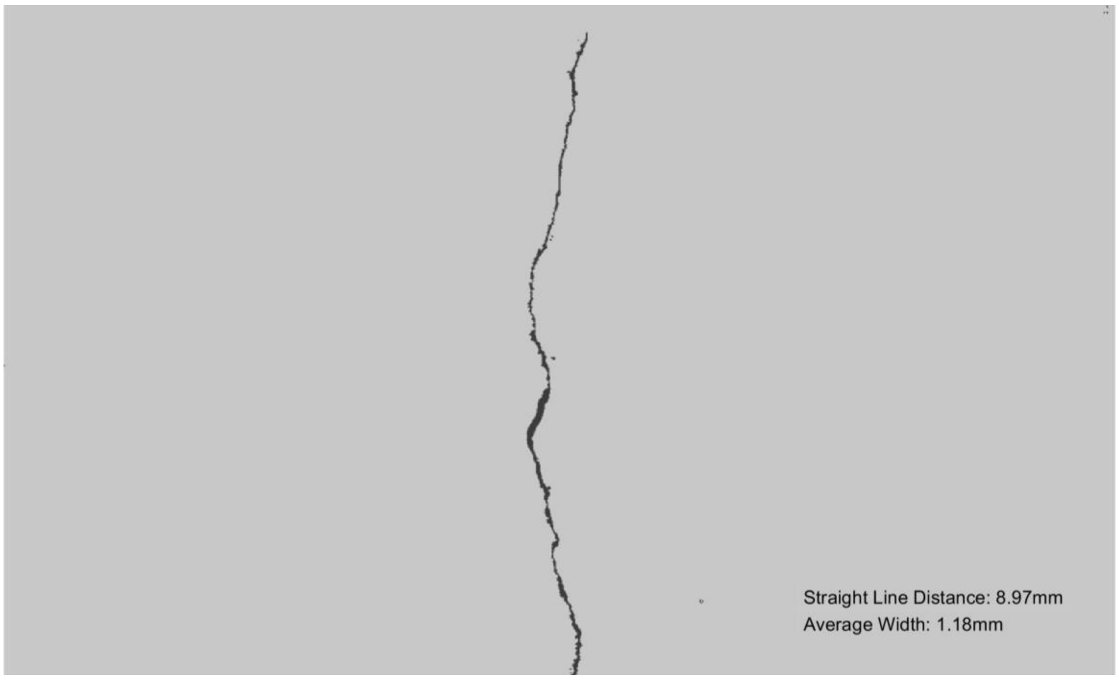


Figure C.3 Linear crack length and average crack width obtained for crack "c".

Crack "d"

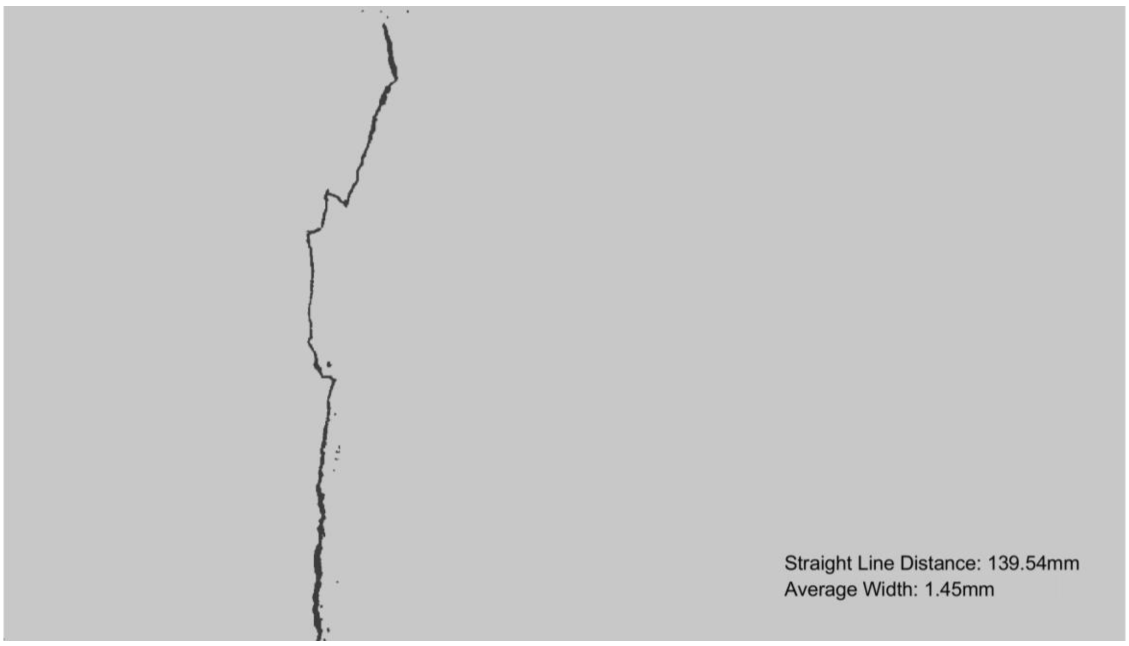


Figure C.4 Linear crack length and average crack width obtained for crack "d".

Crack "e"

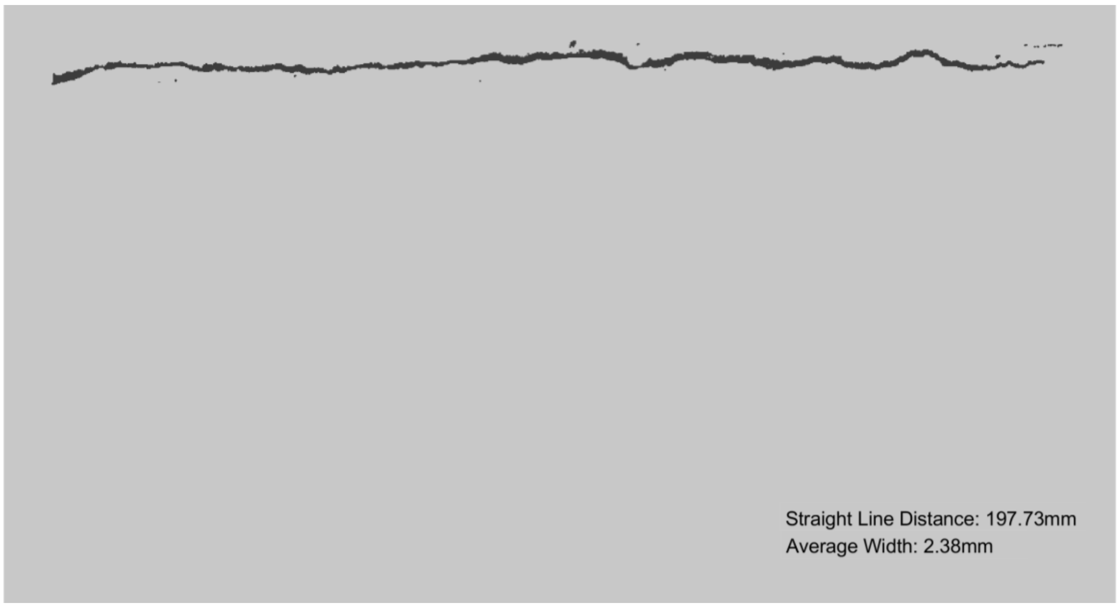


Figure C.5 Linear crack length and average crack width obtained for crack "e".

Crack "f"

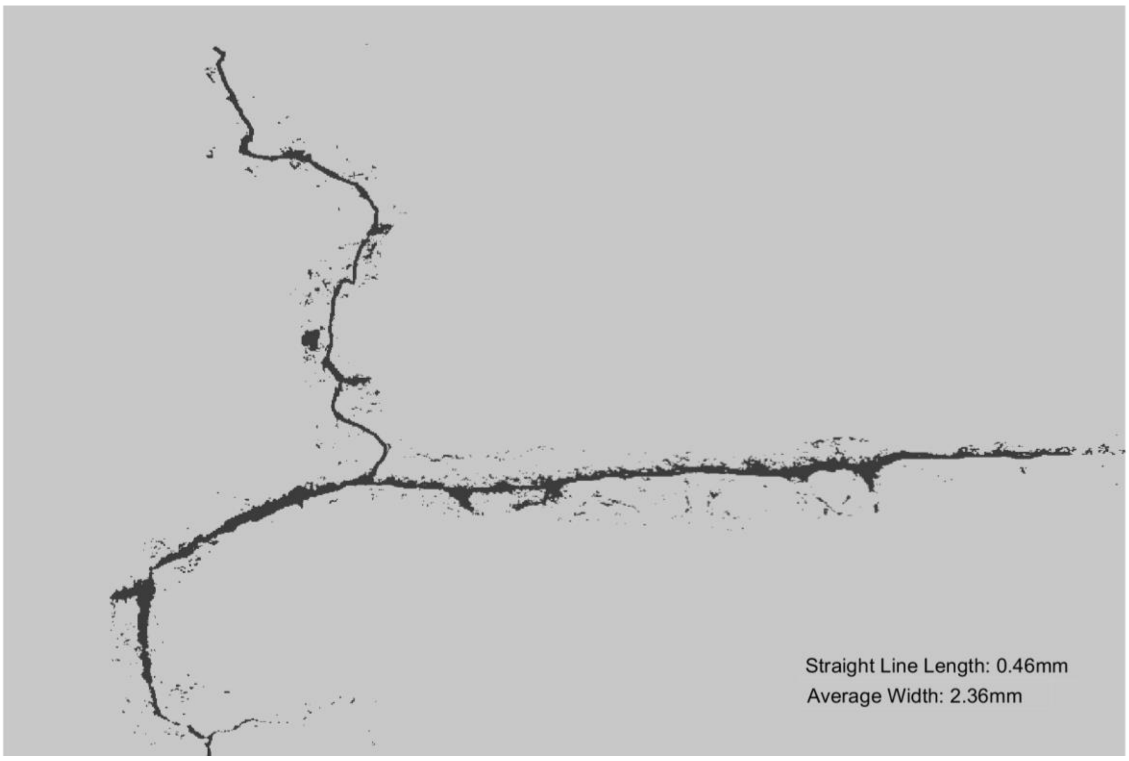


Figure C.6 Linear crack length and average crack width obtained for crack "f".

Crack "g"

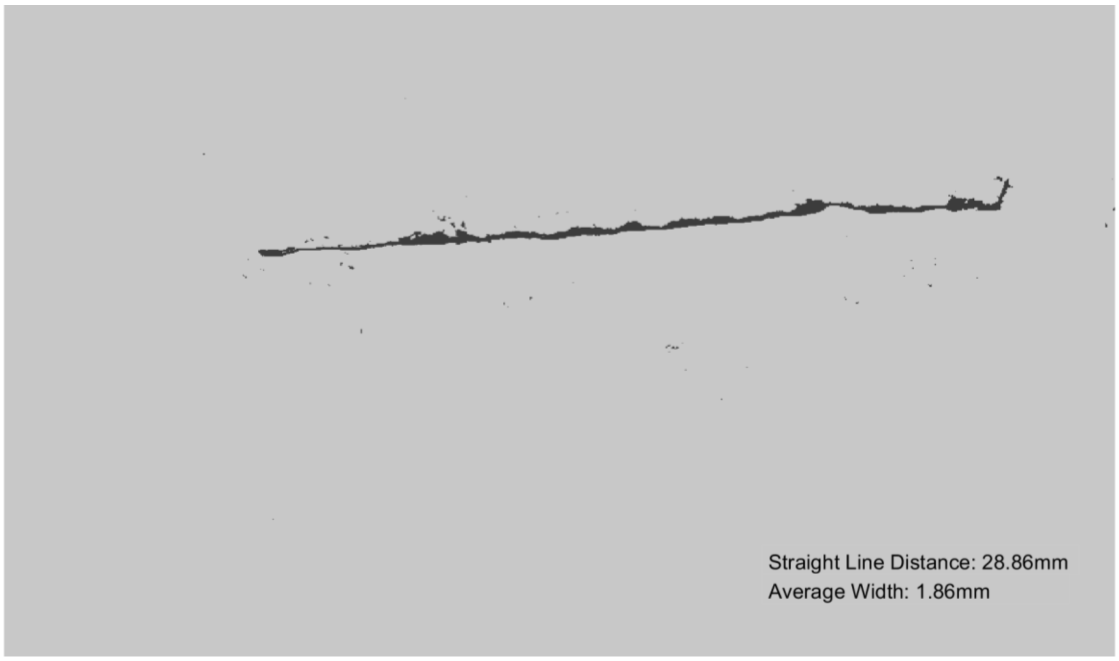


Figure C.7 Linear crack length and average crack width obtained for crack "g".

Crack "h"

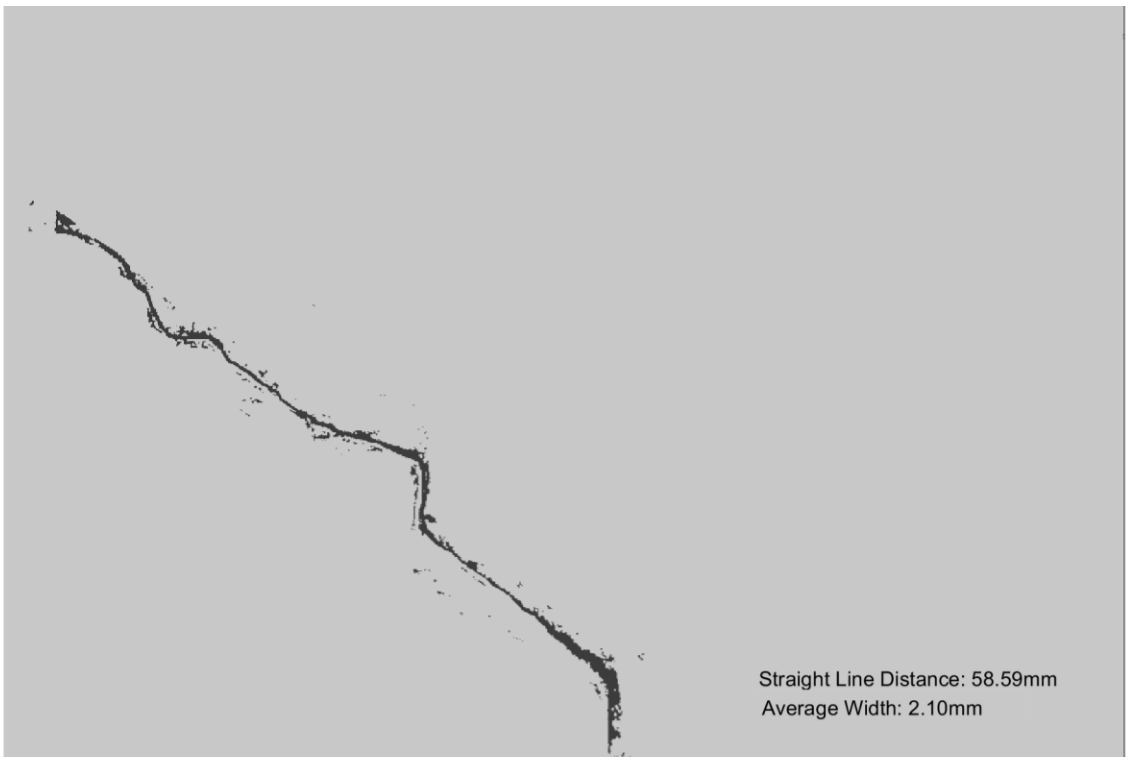


Figure C.8 Linear crack length and average crack width obtained for crack "h".

Crack "i"

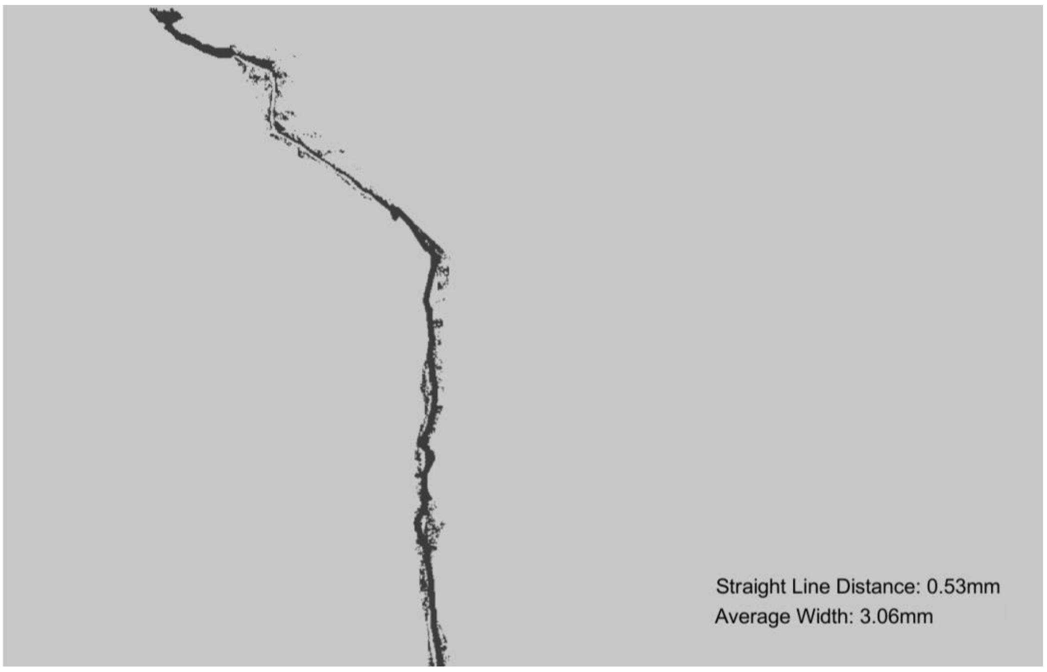


Figure C.9 Linear crack length and average crack width obtained for crack "i".

Crack "j"

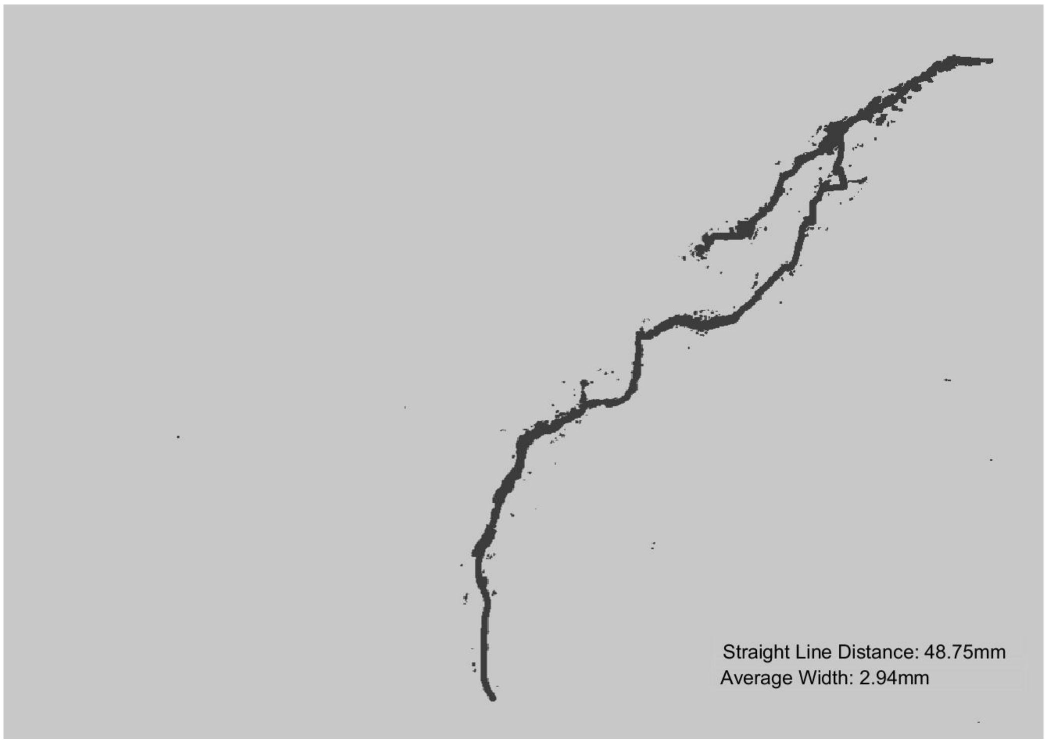


Figure C.10 Linear crack length and average crack width obtained for crack "j".