FACULTY OF ENGINEERING & THE BUILT ENVIRONMENT
DEPARTMENT OF CIVIL ENGINEERING

A Deep Learning-based Approach Towards Automating Visual Reinforced Concrete Bridge Inspections

Submitted in partial fulfilment of the requirements for the degree of

Master of Science in Civil Engineering

By

Bright N. Dube

Supervised by:
Professor Pilate Moyo

Co-Supervised by:
Doctor Kabani Matongo

15 September 2020
The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.
Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. This thesis/dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

Bright Dube

September 2020
Acknowledgements

I would like to extend my gratitude to the following:

- My supervisor and co-supervisor, Professor Pilate Moyo and Doctor Kabani Matongo, for sparking in me the curiosity to explore the world of artificial intelligence and how it can be exploited for civil engineering applications. I am grateful for their continuous guidance and support.

- The wider Concrete Materials and Structural Integrity Research Unit (CoMSIRU) community for their advice and constructive critique of my work.

- Thomas Tumiel from the Cryptocurrency and Artificial Intelligence Society (CryptoSoc) for introducing me to fastai.

- Zindi data scientist, Jonathan Whitacker, for providing invaluable advice on the practical aspects of deep learning and starter code.

- The Scientific Computing Research Unit (SCRU) and Department of Statistical Science for hosting interactive talks with industry experts on computer vision and machine learning.

- My colleague, Taona Chagwedera for proof reading my work.

- My family for their support. None of this would have been possible without them.
Abstract

Visual inspections are fundamental to the maintenance of RC bridge infrastructure. However, their highly subjective nature often compromises the accuracy of inspection results and ultimately leads to inaccurate prioritisation of repair and rehabilitation activities. Visual inspections are also known to expose inspectors to height and traffic-related hazards, and sometimes require the use of costly access equipment. Therefore, the present study investigated state-of-the-art Unmanned Aerial Vehicles (UAVs) and algorithms capable of automating visual RC bridge inspections in order to reduce inspector subjectivity, minimise inspection costs and enhance inspector safety.

Convolutional neural network (CNN) algorithms are state-of-the-art in relation to the automatic detection of RC bridge defects. However, much of the prior research in this area focused on detecting the presence of defects and gave little to no attention to characterizing them according to defect type and degree (D) or extent (E) ratings. Four proof-of-concept CNN models were therefore developed, namely a defect-type detector, crack-type detector, exposed-rebar detector and a shrinkage crack D-rating model. Each model was built by first compiling defect images, labelling them according to defect/crack type and creating training and test sets at a 90-10% split. The training sets were then used to train the CNN models through transfer learning and fine-tuning using the fastai deep learning python library. The performance of each model was ultimately evaluated based on prediction accuracies on the test sets and their robustness to noise.

Test accuracies $\geq 87\%$ were attained by the trained models. This result shows that CNNs are capable of accurately identifying RC bridge corrosion, spalling, ASR, cracking and efflorescence, and assigning appropriate D ratings to shrinkage cracks. It was concluded that CNN models can be built to identify and allocate D and E ratings to any visible defect type, provided the requisite training data that sufficiently represents noisy real-world inspection conditions can be acquired. This formed the basis upon which a practical framework for UAV-enabled and deep learning-based RC bridge inspections was developed.
# Table of Contents

Chapter 1: Introduction .........................................................................................................................1

1.1. Background to Study ......................................................................................................................1

1.2. Problem Statement ..........................................................................................................................4

1.3. Objectives of Study ..........................................................................................................................4

1.4. Significance of Study .......................................................................................................................5

1.5. Research Scope and Limitations .....................................................................................................6

1.6. Organisation of the Study ..............................................................................................................6

Chapter 2: Overview of Bridge Management Systems .............................................................................8

2.1. Introduction ....................................................................................................................................8

2.2. Typical Bridge Management Systems ............................................................................................8

2.3. Overview of Rating Systems ........................................................................................................12

2.3.1. South African and Namibian Rating Systems ..........................................................................12

2.3.2. British Rating System ..............................................................................................................15

2.3.3. Austrian Rating System ...........................................................................................................17

2.3.4. The German Rating System ....................................................................................................19

2.3.5. NBI Condition Ratings ............................................................................................................21

2.3.6. Summary ..................................................................................................................................23

Chapter 3: Overview of the Inspection Process and Typical RC Bridge Defects .....................................24

3.1. Introduction ....................................................................................................................................24

3.2. The Inspection Process ..................................................................................................................24

3.3. Types of RC Bridge Defects ..........................................................................................................26

3.3.1. Cracking and Spalling ..............................................................................................................26

3.3.2. Leaching .................................................................................................................................29

3.3.3. Disintegration, Abrasion and Scaling of Concrete .................................................................30

3.3.4. Excessive Deflection and Vibration .......................................................................................30

3.3.5. Accidental Damage ...................................................................................................................30

3.3.6. Defects to Deck Surface Material .........................................................................................30

3.3.7. Foundation Displacement .......................................................................................................30

3.3.8. Damage to Bridge Accessories ...............................................................................................31

3.3.9. Other Factors that Contribute to RC Bridge Deterioration ....................................................31

Chapter 4: Literature Review ................................................................................................................33

4.1. Introduction ....................................................................................................................................33

4.2. Overview of Bridge Inspection Automation ..................................................................................33

4.3. UAV-Enabled Bridge Inspections ................................................................................................36

4.3.1. UAV Capabilities Required for Bridge Inspections .................................................................36

4.3.2. Levels of Available UAV Autonomy ......................................................................................38

4.3.3. Deliverables from UAV-enabled Inspections .........................................................................39

4.3.4. Aviation Authority Regulations ...............................................................................................41

4.4. Robotic Bridge Inspections ..........................................................................................................42

4.5. Computer-Vision Based Bridge Inspections ................................................................................44

4.6. Summary ......................................................................................................................................47

Chapter 5: Overview of Deep Learning and Convolutional Neural Networks .........................................49

5.1. Introduction ....................................................................................................................................49
Appendix A: Sample Inspection Sheet (DERU) .................................................................106
Appendix B: Loss vs earning Rate Graphs .................................................................107
Appendix C: Validation Accuracy Graphs .................................................................108
List of Tables

Table 2.1. The four-point DERU rating system .......................................................... 13
Table 2.2. The CSS Severity Rating Scale .................................................................. 15
Table 2.3. The CSS Extent Rating Scale ...................................................................... 15
Table 2.4. Determination of ECS values ...................................................................... 16
Table 2.5. Allocation of EIF values .............................................................................. 16
Table 2.6. Austrian Quantitative Condition Rating System ......................................... 18
Table 2.7. Austrian Qualitative Condition Rating System ........................................... 19
Table 2.8. German Ratings for Structural Stability ...................................................... 20
Table 2.9. German Ratings for Traffic Safety ............................................................... 20
Table 2.10. German Ratings for Durability .................................................................... 20
Table 2.11. German BCI .............................................................................................. 21
Table 2.12. Description of NBI Element Ratings ....................................................... 22
Table 2.13. NBI Overall Condition Rating Scale ......................................................... 22
Table 3.1. Typical Damage to Bridge Accessories ....................................................... 31
Table 4.1. Sheridan’s scale of human-machine interaction ........................................... 35
Table 5.1. Effect of hyperparameters on model capacity ............................................. 65
Table 6.1. Attributes of compiled datasets .................................................................... 74
Table 7.1. Summary of Model Performance Results ................................................... 82
List of Figures

Figure 1.1. Typical parabolic condition deterioration curve for concrete bridges ........ 2
Figure 1.2. Maintenance strategies ................................................................. 3
Figure 2.1. Basic components of a BMS ............................................................. 8
Figure 2.2. Example of coded segmentation of a bridge pier ............................... 9
Figure 2.3. Diagrammatic illustration of the Extent rating ................................... 14
Figure 3.1. Possible micro-climate conditions ................................................... 25
Figure 3.2. Identifying areas exposed to salt spray .......................................... 25
Figure 3.3. An under-bridge inspection vehicle in operation ............................... 26
Figure 3.4. Typical RC cracks ............................................................................ 27
Figure 3.5. A close-up view of different crack patterns ....................................... 27
Figure 3.6. Detailed classification of cracks in hardened reinforced concrete ......... 28
Figure 3.7. Spalling due to reinforcement corrosion .......................................... 29
Figure 3.8. Efflorescence due to lime leaching ................................................. 29
Figure 3.9. Types of automation ....................................................................... 34
Figure 3.10. Examples of different UAV types .................................................. 37
Figure 3.11. A collision-tolerant rotorcraft ....................................................... 37
Figure 3.12. Example of a traditional inspection photolog ................................... 40
Figure 3.13. Example of a Pix4D 3D photolog .................................................. 40
Figure 3.14. Example of an orthomosaic image of a bridge site ......................... 41
Figure 3.15. Example of an orthoplane of a bridge façade .................................. 41
Figure 3.16. The RABIT bridge deck inspection vehicle ................................... 43
Figure 3.17. An example of a climbing inspection robot ..................................... 43
Figure 3.18. Deep learning for image classification ......................................... 45
Figure 3.19. Image classification as applied to crack detection ......................... 46
Figure 3.20. Application of image segmentation algorithms ............................. 46
Figure 3.21. Application of object detection algorithms ..................................... 47
Figure 3.22. Traditional feature learning versus deep learning ......................... 50
Figure 3.23. LSVRC results between 2010 and 2017 ....................................... 51
Figure 3.24. Effect of amount of training data on performance ....................... 52
Figure 3.25. A high-level overview of a CNN ................................................. 53
Figure 3.26. An overview of the model training process ................................... 54
Figure 3.27. Minimising the loss function through SGD .................................... 56
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.7</td>
<td>Conceptual illustration of the convolution process</td>
</tr>
<tr>
<td>5.8</td>
<td>An example of the convolution operation</td>
</tr>
<tr>
<td>5.9</td>
<td>A high-level illustration of hierarchical learning</td>
</tr>
<tr>
<td>5.10</td>
<td>An example of zero-padding</td>
</tr>
<tr>
<td>5.11</td>
<td>An example of the downsampling operation</td>
</tr>
<tr>
<td>5.12</td>
<td>Examples of max pooling operations</td>
</tr>
<tr>
<td>5.13</td>
<td>Transfer learning</td>
</tr>
<tr>
<td>5.14</td>
<td>Fine-tuning a CNN</td>
</tr>
<tr>
<td>5.15</td>
<td>Conceptual illustration of underfitting, overfitting and ideal fit</td>
</tr>
<tr>
<td>5.16</td>
<td>The typical relationship between model capacity and error</td>
</tr>
<tr>
<td>5.17</td>
<td>Model divergence due to a very high learning rate</td>
</tr>
<tr>
<td>5.18</td>
<td>The ‘bouncing’ effect due to a very high learning rate</td>
</tr>
<tr>
<td>5.19</td>
<td>The effect of a very low learning rate</td>
</tr>
<tr>
<td>5.20</td>
<td>The non-convex nature of error surfaces</td>
</tr>
<tr>
<td>5.21</td>
<td>The effect of cyclic learning rates on model convergence</td>
</tr>
<tr>
<td>5.22</td>
<td>Snapshot Ensembling</td>
</tr>
<tr>
<td>5.23</td>
<td>Application of trained models</td>
</tr>
<tr>
<td>6.1</td>
<td>Desired interaction of different CNN models</td>
</tr>
<tr>
<td>6.2</td>
<td>Methodology flowchart</td>
</tr>
<tr>
<td>6.3</td>
<td>Sample labelling of SDNET2018 images</td>
</tr>
<tr>
<td>6.4</td>
<td>Example of dataset pre-processing using the fastai python library</td>
</tr>
<tr>
<td>6.5</td>
<td>The ResNet architecture</td>
</tr>
<tr>
<td>6.6</td>
<td>Illustrative example: Loss vs Learning Rate graph</td>
</tr>
<tr>
<td>6.7</td>
<td>Training of the Defect-type Classifier</td>
</tr>
<tr>
<td>6.8</td>
<td>Typical downward trend of train_loss and valid_loss during training</td>
</tr>
<tr>
<td>6.9</td>
<td>Example showing improvement of model accuracy during training</td>
</tr>
<tr>
<td>7.1</td>
<td>Images most poorly predicted by the defect-type classifier</td>
</tr>
<tr>
<td>7.2</td>
<td>Images most poorly predicted by the rebar detection classifier</td>
</tr>
<tr>
<td>7.3</td>
<td>Image poorly predicted by the crack-type classifier</td>
</tr>
<tr>
<td>7.4</td>
<td>Images most poorly predicted by the D-rating classifier</td>
</tr>
<tr>
<td>7.5</td>
<td>Confusion matrices of final models</td>
</tr>
<tr>
<td>8.1</td>
<td>Overview of proposed RC visual bridge inspection methodology</td>
</tr>
<tr>
<td>8.2</td>
<td>Proposed hardware setup</td>
</tr>
<tr>
<td>8.3</td>
<td>Continuous improvement of image classification models</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

1.1. Background to Study
Bridges are essential components of transportation networks worldwide, connecting communities and fostering trade. In the United States of America (USA) alone, over 500,000 bridges are currently in service (Sprung et al., 2018) and this number continues to grow as new highway projects are commissioned every year. In 2016, however, approximately 40% of American bridges were 50 years or older; with 1 in 11 bridges deemed structurally deficient (ASCE, 2017). A large number of trips occur across such structurally deficient bridges each day in the USA, averaging 188 million daily trips in 2016 (ASCE, 2017). Urgent rehabilitation measures are needed to mitigate the significant safety risks this poses to bridge users. However, recent estimates indicate that the USA has a bridge maintenance backlog amounting to US$123 billion (ASCE, 2017). Notably, such backlogs are not limited to the USA alone. It is indeed a global problem characterised by inadequate worldwide investment in critical civil infrastructure (Boshoff et al., 2018). In the South African context, for example, the City of Johannesburg had a total bridge maintenance backlog of approximately R6.5 billion in 2017, with a budget of only R140 million made available that year (JRA, 2017). The Johannesburg Road Agency reported approximately 78% of the city’s bridges to be in ‘poor’ or ‘very poor’ condition (JRA, 2017). Unless there is significant investment in rehabilitating structurally deficient bridges, this percentage will continue to rise.

The deterioration of concrete bridges typically follows the parabolic pattern shown in Figure 1.1 (Boshoff et al., 2018). In the absence of adverse natural or manmade events that directly impair bridge condition significantly (e.g. severe flooding), it can be seen that the rate of deterioration is typically slow but tends to accelerate towards the end of their service life. This degradation often occurs due to a combination of several deterioration mechanisms, namely Alkali-Silica Reaction (ASR), soft water attack, acid attack, sulphate attack, corrosion of steel reinforcement and physical processes such as abrasion, erosion and cavitation (Alexander et al., 2017). These deterioration mechanisms are accentuated by increasing traffic loads over time and poor construction practices such as poor concrete mix design and material selection, inadequate reinforcement cover and insufficient curing of concrete. The rate of
deterioration ultimately depends on the severity of the local environment and the structure’s capacity to withstand it (Alexander et al., 2017).

Figure 1.1. Typical parabolic condition deterioration curve for concrete bridges (Boshoff et al., 2018).

The purpose of maintenance is to ensure that bridge structures reach their intended design lives while meeting defined performance standards. This makes it necessary to periodically reinvest in bridge infrastructure in order to restore a bridge’s service potential or expected useful life to that which it had originally. Maintenance interventions are typically implemented following a reactive maintenance strategy, preventative maintenance strategy or a combination of both (Figure 1.2) (Raupach & Büttner, 2014). The reactive maintenance strategy entails performing repairs only after the structure reaches the limit state which is usually characterised by visible signs of deterioration. A preventative maintenance plan, in contrast, is implemented when it is advantageous to periodically repair the structure before defects become visible e.g. renewing a surface protecting system every 5 years regardless of the original surface protecting system’s condition.

Regular bridge inspections are typically conducted to monitor bridge condition and maintain acceptable levels of service. These involve visual assessments meant to identify defects that could have adverse effects on the safety, serviceability, durability and aesthetics of bridge structures. The outputs from inspections subsequently become a basis from which repair interventions are prescribed and implemented (Frangopol & Tsompanakis, 2014).
Bridge inspections are conducted at 2 levels; namely at network level and at project level. At network level, visual inspections are conducted to identify and prioritise bridges in a transportation network that require the most attention (Tonias & Zhao, 2007). The individual bridges identified from network-level inspections are subsequently subjected to project-level inspections that are more detailed and may involve non-destructive and laboratory testing (Parke & Hewson, 2008).
1.2. Problem Statement

Traditional network-level bridge inspections are highly subjective due to their visual nature. This compromises the accuracy of inspection results and has been shown to lead to inaccurate prioritisation of bridge repair and rehabilitation activities (Nordengen & Nell, 2005). In a study involving the deployment of 49 accredited inspectors for the inspection of 6 bridges in the USA, Phares et al. (2001) found that 95% of primary bridge element condition ratings varied within an average of 2 rating points, while 68% varied within 1 rating point as a direct consequence of inspector subjectivity. Nordengen (2012) also found that the subjective nature of inspections tends to result in the allocation of conservative defect ratings that may not accurately portray the true condition of the bridge. This has been shown to be the case despite extensive inspector training supplemented by the use of standard visual assessment reference guides such as Part B of the TMH19 manual.

Conventional visual inspections typically involve the use of access equipment ranging from simple ladders to heavy mobile platforms (Parke & Hewson, 2008). Inspectors are often required to work at heights, in confined spaces, around water and in the vicinity of road traffic, exposing them to significant safety hazards. The use of ladders has particularly been cited as the main contributor to bridge inspection-related accidents (Wells & Lovelace, 2018). Furthermore, the mobilisation and operation of heavy inspection vehicles often necessitates disruptive lane closures and extensive traffic control (Parke & Hewson, 2008). Equipment mobilisation and traffic control typically consumes 40-50% of the inspection budget and often accounts for 40-50% of the overall inspection time (Choset, 2000). These costs, coupled with resource and personnel scarcity, has been shown to lead to maintenance-activity backlogs (Chan et al., 2015).

1.3. Objectives of Study

This dissertation aims to investigate the potential exploitation of state-of-the-art Unmanned Aerial Vehicle (UAV) technology and deep learning algorithms for network-level RC visual bridge inspections. In order to achieve this aim, the following hypotheses are tested:

i. The use of state-of-the-art inspection-grade UAVs can significantly reduce inspection costs and enhance inspector safety.
ii. State-of-the-art deep learning models are capable of automating the defect detection and rating allocation process during visual inspections, thereby reducing the effect of inspector subjectivity on inspection results.

Therefore, the specific objectives of this study are to:

i. Identify UAV features and capabilities required to enable UAV-aided visual RC bridge inspections.

ii. Assess the cost-saving potential of UAV use during RC visual bridge inspections.

iii. Review aviation authority legislation governing the use of UAVs for bridge inspections.

iv. Identify and review deep learning algorithms capable of automatically detecting and allocating ratings to visible RC bridge defects solely from image inputs.

v. Produce and evaluate proof-of-concept deep learning models capable of RC defect detection and rating allocation.

vi. Propose a practical framework for the application of UAVs and deep learning models, in visual RC bridge inspections.

1.4. Significance of Study

The fact that visual bridge inspections are highly subjective has been well-known as far back as 2001 (Phares et al., 2001), with inspectors tending to allocate conservative defect ratings that do not accurately portray true bridge condition and lead to inaccurate prioritisation of repair and rehabilitation activities (Nordengen & Nell, 2005). In spite of this, most transport authorities still consider such inspections to be the most reliable network-level inspection method (Dorafshan et al., 2018). The manufacturing and aviation industries, however, have shown that inspector subjectivity and human error can be reduced significantly through the introduction of automation to manual processes (Dorafshan et al., 2018). This study proposes a practical computer vision and UAV-based approach towards visual RC bridge inspections. Current research in this area has predominantly focused on crack-presence detection. The detection of non-crack defects has generally received little attention in literature and, to the author’s knowledge, no studies exploring the automatic allocation of defect ratings (in terms of severity and/or extent) have been published at the time of writing. In addition to filling
these gaps in literature, this study ultimately proposes a deep learning-based practical framework that can help transport agencies achieve more consistent inspection results while negating the need for specialised access equipment and enhancing inspector safety.

1.5. Research Scope and Limitations
This is a proof-of-concept study concerned with assessing the feasibility of deploying state-of-the-art computer vision algorithms to automate the defect detection and rating process during visual RC bridge inspections. Based on the findings, a practical framework for the joint application of deep learning models and UAVs during network-level RC bridge inspections is proposed. A field test of the proposed framework, however, could not be conducted within the research timeframe.

1.6. Organisation of the Study
This study is organized into 9 chapters described as follows:

**Chapter 1**
The first chapter provides a background to the research, problem statement, research objectives, significance of study and the study’s scope and limitations.

**Chapter 2**
This chapter reviews the broad bridge management system (BMS) and rating system landscape.

**Chapter 3**
The third chapter provides a general overview of the traditional visual inspection process. It also discusses the types of defects that are typically encountered during inspections.

**Chapter 4**
This chapter presents a literature review that interrogates previous studies in relation to the automation of RC bridge inspections. It discusses in detail the types of automation that have successfully been introduced to the inspection process using unmanned aerial vehicles (UAVs), robots and deep learning algorithms. The chapter
identifies convolutional neural network (CNN) algorithms to be state-of-the-art regarding the automatic detection of defects during visual RC bridge inspections.

**Chapter 5**
Chapter 5 provides a detailed overview of CNN algorithms in relation to image classification and object detection.

**Chapter 6**
This chapter describes the methodology that was used to develop proof-of-concept CNN models capable of accepting inspection images as input and automatically assigning accurate defect-type and defect-severity labels to image contents.

**Chapter 7**
Chapter 7 investigates the performance of the CNN models produced in Chapter 6 based on their classification accuracies and robustness.

**Chapter 8**
This chapter proposes a practical framework for the joint application of CNN models and UAVs to automate the defect identification and rating process in real-time during visual RC bridge inspections. The framework’s limitations are also discussed.

**Chapter 9**
Chapter 9 presents the conclusions of the study, recommends ways in which the proposed framework could be adapted for other related applications and suggests topics for further research.
Chapter 2: Overview of Bridge Management Systems

2.1. Introduction
Transport agencies typically face the task of tracking the condition of a large group of bridges in a transport network in order to make sound repair and rehabilitation decisions. As a result, Bridge Management Systems (BMSs) have widely been employed to assist with bridge management at network and project levels (Tonias & Zhao, 2007). This chapter aims to provide a general overview of how BMSs operate. It starts by exploring the basic features of a typical BMS. Thereafter, examples of some of the most commonly used BMSs and rating systems are discussed.

2.2. Typical Bridge Management Systems
BMSs typically consist of 5 core modules, namely the inventory, inspection, condition, maintenance and cost modules (Nordengen & Nell, 2005). Together, these modules are a tool that allow bridge managers to decide how best limited funds can be used to the benefit of the entire bridge network (Parke & Hewson, 2008). The general structure of a typical BMS is provided in Figure 2.1.

![Figure 2.1. Basic components of a BMS (Ryall, 2010).](image)

Central to a BMS is the database. Its role is to store all past and present technical,
administrative, financial and anecdotal information about bridges in a network (Nordengen & Nell, 2005).

The inventory module stores information that specifically identifies and describes each bridge in a network, such as names, locations and construction (Parke & Hewson, 2008). The information stored in this module is typically obtained from drawings, maintenance records and site visits.

The inspection module stores information gathered during bridge inspections (Ryall, 2010). The exact nature of this information largely depends on the bridge rating approach that the BMS adopts. A number of BMSs, including PONTIS and BRIDGIT, use a condition-based rating system. The application of this rating system during inspections typically involves dividing bridge elements into coded segments and assigning a numerical rating that corresponds to the inspector’s perception of the condition of each. The inspection module of condition-based BMSs, therefore, stores information that include general bridge condition, prescribed remedial measures and, sometimes, inspection drawings indicating components that need further attention. An example of bridge-element segmentation is provided in Figure 2.2.

![Figure 2.2. Example of coded segmentation of a bridge pier (Ryall, 2010).](image)

Other BMSs make use of a defect-based rating system. In this case, numerical ratings are only assigned to observable defects on each bridge element (Nordengen & Nell, 2005). Unlike condition-based BMSs, the inspection module of defect-based BMSs stores information such as defect ratings and photographs of all observed defects. Since ratings are only conducted on elements with visible defects and no element
segmentation is required, the defect-based rating system is known for reduced inspection times (Nordengen & Nell, 2005). Examples of defect-based BMSs include the STRUMAN BMS which is prominently deployed in South Africa and Namibia, and BMSs used by Austrian and Finnish transport authorities (Adu-Gyamfi et al., 2016).

The maintenance module stores maintenance records for every bridge in a given network (Parke & Hewson, 2008). This allows bridge owners to keep track of the nature of maintenance carried out on each bridge over time and the associated costs. The cost (or budget) module processes such cost information in order to produce reliable financial reports (Ryall, 2010).

The means by which the budget module optimises costs of future maintenance depends on whether the BMS is condition-based or defect-based. Condition-based BMSs make use of Markov chains to predict the future condition of bridge elements and calculate the costs of all possible maintenance options (Parke & Hewson, 2008). By identifying the path on the Markov chain with the least total cost, optimisation of costs is achieved and an Optimised Maintenance Program (OMP) is produced. Defect-based BMSs, however, optimise costs based on the severity of each observed defect and the urgency of necessary repairs. For example, cost optimisation by the STRUMAN BMS is based on the Relevancy-to-cost ratio per defect (Nordengen & Nell, 2005). Due to budgetary constraints, a proportion of the required maintenance works may be delayed in favor of maintenance activity that addresses defects with higher Urgency (U) defect ratings (Nordengen & Nell, 2005).

The function of the condition module is to derive a Condition Index (CI) for individual bridges in a network based on historical and latest inspection information (Parke & Hewson, 2008). This index reflects the overall condition of each bridge, allowing bridge managers to identify the most deteriorated structures in order to prioritise repair and rehabilitation projects (Ryall, 2010). The CI is often calculated using one or a combination of four approaches (Adu-Gyamfi et al., 2016):

i. A ratio-based approach where the CI is derived based on the current bridge condition to new bridge condition ratio. This method is underpinned by the need to improve the value of the structure by improving its condition. Notable examples of BMSs that make use of this approach include the Californian
BMS and PONTIS (now AASHTOWare BrM).

ii. A weighted average approach in which the overall condition of a bridge is estimated based on inspection ratings that are weighted according to each element’s significance to the structure’s integrity. Examples of systems that use this approach include the South African, British, Australian and Austrian BMSs.

iii. A worst-conditioned component approach in which the CI is based on the rating assigned to the bridge component deemed to be in the worst condition. This method is often used when inspections are only carried out on critical bridge elements. BMSs adopted in Germany, Japan and by the Michigan Department of Transportation follow this approach.

iv. A qualitative approach whereby overall bridge condition is described using the terms “Good”, “Fair” or “Poor”, instead of a numerical scale. It is often used to capture the overall condition of bridges and quickly identify those that are in need of attention. Since individual bridges flagged to be in “Poor” condition are not ranked against each other in terms of condition and urgency of repairs, the qualitative approach cannot be used to accurately prioritise and plan maintenance and rehabilitation projects. Examples of BMSs that make use of this approach are those used in the states of Washington and Florida in the United States.

Ultimately, transport agencies decide on the approach(es) they wish to take in order to arrive at a means of assessing the overall condition of their bridge networks (Ryall, 2010). With the exception of BMSs that use the qualitative approach, the condition module of most BMSs generate the CI in the form of a number (on a fixed scale) that is associated with a particular set of actions to be taken e.g. in the case of the STRUMAN BMS, $\text{CI} \geq 85 \Rightarrow \text{“Do nothing” and CI} < 30 \Rightarrow \text{“Substantial renewal/upgrades required”} \ (\text{COTO, 2016}).$

While the CI reflects the overall condition of individual bridges in terms of structural adequacy, it does not take the strategic importance of each bridge into account (Adu-Gyamfi et al., 2016). In order to address this, BMSs typically combine structural condition information from inspections with functional information (e.g. traffic volumes and structure type) to prioritise maintenance and rehabilitation activities. The
STRUMAN BMS (through the Functional Index) and the BMS adopted by the Finnish Road Administration (through the Rehabilitation Index) are examples of systems that use this approach (Nordengen & Nell, 2005; Adu-Gyamfi et al., 2016).

2.3. Overview of Rating Systems

It is evident that transport authorities are afforded some degree of flexibility when it comes to deciding the specific means by which they prioritise their bridge structures for repair and rehabilitation within their jurisdiction. This flexibility includes deciding on the rating techniques to use during bridge inspections, whether defect-based or condition-based. This section aims to provide an overview of some of the rating systems adopted in South Africa, the UK, Austria, Germany and in some American states. While a limited number of systems are reviewed, they are representative of the rating system landscape and adequately demonstrate the underlying philosophies of defect or condition ratings.

2.3.1. South African and Namibian Rating Systems

South Africa and Namibia adopted the STRUMAN BMS which relies on a defect-based ‘DERU’ rating system. Under this system, inspectors look to rate the Degree (D), Extent (E) and Relevancy (R) of observed defects, as well as the Urgency (U) of necessary remedial action for each defect. The D, E and R components are described as follows (COTO, 2016):

- **Degree** – represents how bad or severe a defect looks. It does not consider the defect’s consequences on the inspection element and overall structure.
- **Extent** – indicative of how widespread a defect is on the inspection item.
- **Relevancy** – represents how severe the consequences of a defect are with regards to the bridge’s functionality, structural integrity and user safety.

A four-point scale (1-4) is used for each component. Degree ratings of X, U or 0 are used to indicate ‘item not applicable’, ‘unable to inspect’ or ‘no visible defects’ respectively (COTO, 2016). The four-point scale is described in Table 2.1. Figure 2.3 provides a schematic illustration of the E-rating scale.
Table 2.1. The four-point DERU rating system (COTO, 2016).

<table>
<thead>
<tr>
<th>Category</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree (D)</td>
<td>No visible defects</td>
<td>Minor</td>
<td>Moderate</td>
<td>Warning</td>
<td>Severe</td>
</tr>
<tr>
<td>Extent (E)</td>
<td>N/A</td>
<td>Local</td>
<td>More than local</td>
<td>Less than general</td>
<td>General</td>
</tr>
<tr>
<td>Relevancy (R)</td>
<td>N/A</td>
<td>Minimum</td>
<td>Moderate</td>
<td>Major</td>
<td>Critical</td>
</tr>
<tr>
<td>Urgency (U)</td>
<td>Monitor only</td>
<td>Routine</td>
<td>Within 10 years</td>
<td>Within 5 years</td>
<td>As soon as possible</td>
</tr>
</tbody>
</table>

(Adapted from COTO, 2016 and Branco & Brito, 2004)

D, E and R ratings are used to compute a CI for each defect \( I_c \) defined in Equation 2.1 (Hearn et al., 2005):

\[
I_c = 100 \times \left[ 1 - \frac{(D+E)+R}{32} \right]
\]  

(2.1)

A defect is considered critical when \( I_c < 40 \). The overall CI of the bridge \( BCI_n \) is then defined as the sum of \( I_c \) values weighted by the average daily traffic as shown in Equation 2.2 (Hearn et al., 2005):

\[
BCI = \frac{\sum I_c \cdot ADT_n}{\sum ADT_i},
\]

(2.2)

where \( ADT_n \) represents the average daily traffic for structure \( n \).
<table>
<thead>
<tr>
<th>Description</th>
<th>Local</th>
<th>More than local</th>
<th>Less than general</th>
<th>General</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>⋆</td>
<td>⋆</td>
<td>⋆</td>
<td>⋆</td>
</tr>
<tr>
<td>E = 2</td>
<td>⋆</td>
<td>⋆</td>
<td>⋆</td>
<td>⋆</td>
</tr>
<tr>
<td>E = 3</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
</tr>
<tr>
<td>E = 4</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
<td>⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆ ⋆</td>
</tr>
</tbody>
</table>

*Figure 2.3.* Diagrammatic illustration of the Extent rating (COTO, 2016).
2.3.2. British Rating System

A number of rating systems have been adopted by various transport agencies in the UK. Each rating system generally incorporated one or more of the following approaches (ATKINS, 2002):

i. Use of qualitative attributes ("Good", "Fair", "Poor").

ii. Rating the Extent (A – D) and Severity of defects (1 – 4).

iii. Application of a Condition Factor that relates to element importance.

In order to ensure consistency in inspection reporting in the UK, the County Surveyors' Society (CSS) developed a five-point defect-based rating system for gradual adoption by UK local authorities (ATKINS, 2002). As with the DERU rating method, the CSS inspection reporting system attempts to rate the severity and extent of defects on bridge elements (ATKINS, 2002). The CSS rating system for severity and extent is described in Table 2.2 and Table 2.3 respectively.

### Table 2.2. The CSS Severity Rating Scale (ATKINS, 2002).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>As new condition or defect has no significant effect on element (visually or functionally).</td>
</tr>
<tr>
<td>2</td>
<td>Early signs of deterioration, minor defect/damage, no reduction in functionality of element.</td>
</tr>
<tr>
<td>3</td>
<td>Moderate defect/damage, some loss of functionality could be expected.</td>
</tr>
<tr>
<td>4</td>
<td>Severe defect/damage, significant loss of functionality and/or is close to failure/collapse.</td>
</tr>
<tr>
<td>5</td>
<td>The element is non-functional/failed.</td>
</tr>
</tbody>
</table>

### Table 2.3. The CSS Extent Rating Scale (ATKINS, 2002).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>No significant defect.</td>
</tr>
<tr>
<td>B</td>
<td>Slight, not more than 5% of surface area/length/number.</td>
</tr>
<tr>
<td>C</td>
<td>Moderate, 5% - 20% of surface area/length/number.</td>
</tr>
<tr>
<td>D</td>
<td>Wide: 20% - 50% of surface area/length/number.</td>
</tr>
<tr>
<td>E</td>
<td>Extensive, more than 50% of surface area/length/number.</td>
</tr>
</tbody>
</table>

By taking the consequences of defects on the functionality of the element or overall bridge structure, the CSS Severity rating can be perceived as a combination of DERU-
based D and R ratings. The CSS and DERU Extent ratings are also comparable, although the scale of the former is defined in more quantitative terms.

To compute the BCI, CSS Extent and Severity ratings are first combined to produce an element condition score (ECS) according to Table 2.4 (ATKINS, 2002).

<table>
<thead>
<tr>
<th>Extent</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>B</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1.1</td>
<td>2.1</td>
<td>3.2</td>
<td>4.1</td>
<td>5.0</td>
</tr>
<tr>
<td>D</td>
<td>1.3</td>
<td>2.3</td>
<td>3.3</td>
<td>4.3</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>1.7</td>
<td>2.7</td>
<td>3.7</td>
<td>4.7</td>
<td></td>
</tr>
</tbody>
</table>

Factors that account for the importance of each element (i.e. Element Importance Factor or EIF) are then assigned to each element according to Table 2.5.

<table>
<thead>
<tr>
<th>Element Importance</th>
<th>EIF Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>2.0</td>
</tr>
<tr>
<td>High</td>
<td>1.5</td>
</tr>
<tr>
<td>Medium</td>
<td>1.2</td>
</tr>
<tr>
<td>Low</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Factors that account for each element’s independent contribution to overall bridge condition (i.e. Element Condition Factor or ECF) are subsequently obtained using Equation 2.3:

\[ ECF = X - \left( \text{ECS} - 1 \right) \cdot \frac{X}{4} \]  

(2.3)

where X is 0.3, 0.6 or 1.2 for elements of high, medium or low importance respectively.

The Element Condition Index (ECI) is then calculated as the difference between the ECS and ECF per element. Next, the overall Bridge Condition Score (BCS) is obtained as the aggregate sum of ECI values for each element, weighted by the EIF as Equation
2.4 shows.

\[
BCS = \frac{\sum_{i=1}^{N} EGI_i EIF_i}{\sum_{i=1}^{N} EIF_i}
\]  

(2.4)

Finally, the BCI is calculated on a scale of 0 to 100 (0 being the worst and 100 being the best) according to Equation 2.5:

\[
BCI = 100 - (2 \times [(BCS)^2 + (6.5 \times BCS) - 7.5])
\]

(2.5)

### 2.3.3. Austrian Rating System

Austria’s rating system resembles the South African and British systems by attempting to score the severity, extent, relevancy and urgency of bridge defects. The rating system assigns ratings based on the following defect groups (Casas & Bien, 2007):

i. Defects on the concrete surface. These are grouped into 7 categories.
ii. Crack defects, whose intensities are divided into 3 categories.
iii. Open joints.
iv. Reinforcement damage. These are divided into 6 categories.
v. Damage to post-tensioned tendons.
vi. Damage to waterproofing, drainage or sealing.
vii. Damage to bearings.
viii. Damage to expansion joints.
ix. Damage to the carriageway. These are divided into 8 categories.
x. Damage to the drainage system.
xi. Damage to bridge equipment.
xii. Deficiencies in the bridge landscape e.g. riverbank slopes.

These groups and related categories give a total of 32 defect types. Rating scores for each defect type are combined to obtain the overall bridge condition rating \( S \) according to Equation 2.6 (Casas & Bien, 2007).

\[
S = \sum_{1}^{32} G_i \times k_{1i} \times k_{2i} \times k_{3i} \times k_{4i}
\]

(2.6)
where,

- $G_i$ is a rating score on a 5-point scale that relates to the severity of the consequences of a defect on the bridge’s structural and functional integrity. This rating is comparable to the Relevancy rating under the DERU rating system.
- $K_{1i}$ is a rating score that signifies the extent of the defect either on one or more components, or on the entire bridge structure. This rating, ranging between 0 and 1, is obtained qualitatively based on the inspector’s judgement of the quantity, frequency or size of the defect. Similar to the DERU system and unlike the CSS method, the Austrian extent rating is not quantified by measured area, length or number of defects.
- $K_{2i}$ is a rating score that signifies the intensity of defect on a scale ranging between 0 and 1. The defect intensity value refers to the inspector’s perception of how detrimental a defect looks e.g. based on the width of cracks. It is therefore comparable to the Degree rating under the DERU rating system.
- $K_{3i}$ is a factor that signifies the importance of the structural element, ranging between 0 and 1.
- $K_{4i}$ is a rating score that signifies the urgency of necessary interventions, ranging between 0 and 10.

Based on the overall condition rating $S$, bridges are ultimately classified according to Table 2.6:

<table>
<thead>
<tr>
<th>Damage class</th>
<th>Definition</th>
<th>Condition rating value $S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No or very little deterioration</td>
<td>0-3</td>
</tr>
<tr>
<td>2</td>
<td>Little deterioration</td>
<td>2-8</td>
</tr>
<tr>
<td>3</td>
<td>Medium to severe deterioration</td>
<td>6-13</td>
</tr>
<tr>
<td>4</td>
<td>Severe deterioration</td>
<td>10-25</td>
</tr>
<tr>
<td>5</td>
<td>Very severe deterioration</td>
<td>20-70 (k_4 = 10)</td>
</tr>
<tr>
<td>6</td>
<td>Very severe or total deterioration</td>
<td>&gt;50 (k_4 = 10)</td>
</tr>
</tbody>
</table>

As an alternative to the above classification, a number of Austrian infrastructure administrations make use of a simple qualitative rating system (Adu-Gyamfi et al., 2016). This system is qualitative in the sense that no calculation is done to obtain elemental and overall bridge condition ratings. During inspections, inspectors assign
elemental condition ratings on a 5-point scale, taking into account the severity and extent of damage; load-bearing capacity, operability and durability of the structure; as well as the urgency of necessary interventions. Based on the elemental condition ratings, the overall condition of the bridge is ultimately rated on the same 5-point scale. A description of the Austrian qualitative rating system is provided in Table 2.7.

<table>
<thead>
<tr>
<th>Rating</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No problems, minor problems; load-bearing capacity, operability, and durability not limited; no maintenance required.</td>
</tr>
<tr>
<td>2</td>
<td>Minor problems; load-bearing capacity and operability not limited; operability and durability will be limited if defects are not removed in the long-term; no restriction of use.</td>
</tr>
<tr>
<td>3</td>
<td>Moderate problems; indication of limited operability and durability; maintenance required in the medium term (within 6 years).</td>
</tr>
<tr>
<td>4</td>
<td>Severe problems; load-bearing capacity not yet limited but operability and durability already limited; maintenance within 3 years (short term) to reestablish regular use.</td>
</tr>
<tr>
<td>5</td>
<td>Critical condition; load-bearing capacity and operability limited; immediate initiation of repair, restrict use.</td>
</tr>
</tbody>
</table>

### 2.3.4. The German Rating System

The rating system adopted in Germany represents a significant departure from those discussed so far. The German rating system places more emphasis on rating the effects of each instance of damage with specific regard to structural stability, traffic safety and durability (Adu-Gyamfi et al., 2016). The 5-point scales used for these ratings are described in Table 2.8, Table 2.9 and Table 2.10. The Extent of damage (U) is described using the qualitative terms “Small”, “Medium” or “Large” based on the inspector’s judgement (Casas & Bien, 2007). The number of damage occurrences (n) is also taken note of during inspections.

The BCI ($Z_{ges}$) is ultimately computed as a function of the structural stability rating ($S^s$), traffic safety rating ($S^v$), durability rating ($S^d$), extent (U) and number of occurrences (n) as Equation 2.7 describes (Casas & Bien, 2007).

$$Z_{ges} = f (S^v, S^s, S^d, U, n)$$ (2.7)
Table 2.8. German Ratings for Structural Stability (Adu-Gyamfi et al., 2016).

<table>
<thead>
<tr>
<th>Assessment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Defects have no effect on structural stability of elements or overall structure.</td>
</tr>
<tr>
<td>1</td>
<td>Defects affect stability of structure elements but not the overall structure.</td>
</tr>
<tr>
<td>2</td>
<td>Defects affect stability of structure elements and have little effect on stability of overall structure.</td>
</tr>
<tr>
<td>3</td>
<td>The effect of defects on stability of structural elements and the overall structure is beyond permissible tolerance.</td>
</tr>
<tr>
<td>4</td>
<td>The structural stability of structural elements and the structure itself no longer exists.</td>
</tr>
</tbody>
</table>

Table 2.9. German Ratings for Traffic Safety (Adu-Gyamfi et al., 2016).

<table>
<thead>
<tr>
<th>Assessment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Defects have no effect on traffic safety.</td>
</tr>
<tr>
<td>1</td>
<td>Defects affect traffic safety only slightly.</td>
</tr>
<tr>
<td>2</td>
<td>Defects may impair traffic safety.</td>
</tr>
<tr>
<td>3</td>
<td>Defects affect traffic safety.</td>
</tr>
<tr>
<td>4</td>
<td>Traffic safety is no longer given due to defects.</td>
</tr>
</tbody>
</table>

Table 2.10. German Ratings for Durability (Adu-Gyamfi et al., 2016).

<table>
<thead>
<tr>
<th>Assessment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Defects have no effect on durability.</td>
</tr>
<tr>
<td>1</td>
<td>Defects affect durability of structure elements but not the durability of the overall structure.</td>
</tr>
<tr>
<td>2</td>
<td>Defects affect durability of structure elements and, in the long term, can affect the overall structure.</td>
</tr>
<tr>
<td>3</td>
<td>Defects affect durability of structure elements and, in the medium term, can affect the overall structure.</td>
</tr>
<tr>
<td>4</td>
<td>The durability of both the structure element and the overall structure is no longer given due to the defects.</td>
</tr>
</tbody>
</table>

Since the German BMS uses the worst-conditioned component approach, the BCI is based on the rating of the worst bridge component (Adu-Gyamfi et al., 2016).

A description of the German BCI is provided in Table 2.11.
<table>
<thead>
<tr>
<th>Assessment</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1.0-1.4    | • Very good structural condition.  
|            | • The stability, traffic safety and durability of the structure are assured. |
| 1.5-1.9    | • Good structural condition.  
|            | • The stability and traffic safety of the structure are assured.  
|            | • The durability of the structure might be impaired slightly in the long term. |
| 2.0-2.4    | • Satisfactory structural condition.  
|            | • The stability and traffic safety of the structure are assured.  
|            | • The durability of the structure might be impaired considerably in the long term. |
| 2.5-2.9    | • Temporarily satisfactory structural condition.  
|            | • The stability of the structure is assured.  
|            | • The traffic safety can be impaired.  
|            | • The durability of the structure might be impaired considerably in the long term. |
| 3.0-3.4    | • Critical structural condition.  
|            | • Traffic safety is affected.  
|            | • The durability of the structure might no longer be assured.  
|            | • Repair work is required immediately. |
| 3.5-4.0    | • Inadequate structural condition.  
|            | • Traffic safety is not adequate.  
|            | • The durability of the structure might no longer be assured.  
|            | • Repair work, rehabilitation or replacement is required immediately. |

### 2.3.5. NBI Condition Ratings

Some American States, such as Washington and Florida, have adopted the National Bridge Inventory (NBI) condition rating system (Adu-Gyamfi et al., 2016). Under this system, bridges are divided into 3 main components, namely the deck, superstructure and substructure. These components are further divided into their respective elements e.g. drainage systems, girders, bearings, abutments, piers, etc. During inspections, each element is rated qualitatively using the terms “Good”, “Fair” or “Poor” depending on any observed deficiencies (Casas & Bien, 2007). These ratings are described in Table 2.12.
After inspecting all the components, the inspector assigns an assessment value between 0 and 9 to indicate the overall condition of the bridge. This assessment is done solely based on the inspector’s judgement of overall bridge condition, without taking any pre-defined rules into account. A description of the NBI rating scale is provided in Table 2.13 (Casas & Bien, 2007).

<table>
<thead>
<tr>
<th>Condition Rating</th>
<th>Description</th>
<th>Condition Rating</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good</td>
<td>Element is limited to only minor problems.</td>
<td>1</td>
<td>Imminent failure condition – major deterioration or section loss present in critical structural components, or obvious vertical or horizontal movement affecting structure stability. Bridge is closed to traffic but corrective action may put bridge back in light service.</td>
</tr>
<tr>
<td>Fair</td>
<td>Structural capacity of element is not affected by minor deterioration, section loss, spalling, cracking or other deficiency.</td>
<td>2</td>
<td>Critical condition - advanced deterioration of primary structural elements. Fatigue cracks in steel or shear cracks in concrete may be present or scour may have removed substructure support. Unless closely monitored it may be necessary to close the bridge until corrective action is taken.</td>
</tr>
<tr>
<td>Poor</td>
<td>Structural capacity of element is affected or jeopardized by advanced deterioration, section loss, spalling, cracking or other deficiency.</td>
<td>3</td>
<td>Serious condition – loss of section, deterioration, spalling or scour have seriously affected primary structural elements. Local failures are possible. Fatigue cracks in steel or shear cracks in concrete may be present.</td>
</tr>
<tr>
<td>N</td>
<td>Not applicable.</td>
<td>4</td>
<td>Poor condition - advanced section loss, deterioration, spalling, or scour.</td>
</tr>
<tr>
<td>9</td>
<td>Excellent condition.</td>
<td>5</td>
<td>Fair condition – all primary structural elements are sound but may have minor section loss, cracking, spalling, or scour.</td>
</tr>
<tr>
<td>8</td>
<td>Very good condition – no problems noted.</td>
<td>6</td>
<td>Satisfactory condition - structural elements show some minor deterioration.</td>
</tr>
<tr>
<td>7</td>
<td>Good condition – some minor problems.</td>
<td>7</td>
<td>Good condition – some minor problems.</td>
</tr>
<tr>
<td>6</td>
<td>Satisfactory condition - structural elements show some minor deterioration.</td>
<td>8</td>
<td>Excellent condition.</td>
</tr>
<tr>
<td>5</td>
<td>Fair condition – all primary structural elements are sound but may have minor section loss, cracking, spalling, or scour.</td>
<td>9</td>
<td>Excellent condition.</td>
</tr>
</tbody>
</table>

**Table 2.12.** Description of NBI Element Ratings (Casas & Bien, 2007).

**Table 2.13.** NBI Overall Condition Rating Scale (Casas & Bien, 2007).
While the NBI condition rating system captures the severity of deficiencies, it does not provide information on the extent of defects (Casas & Bien, 2007). For this reason, Casas & Bien (2007) opine that it has a limited value when it comes to prioritising bridges according to maintenance and rehabilitation requirements. Unlike the other rating systems described in this study, it may also be noted that the NBI system is especially subjective since the BCI is obtained qualitatively i.e. without computation.

2.3.6. Summary
A number of rating systems have been adopted by different transport authorities around the world. The majority of these systems look to capture both the severity and extent of deficiencies, while taking their effects on the bridge’s functionality and user safety into account.

A recurring feature of rating systems is that they are highly subjective in nature as they are dependent on the inspector’s judgement. Since these ratings form the foundation upon which the BCI is computed for prioritisation, this is a significant drawback. In order to reduce subjectivity, transport authorities ensure that bridge inspectors are well trained and certified. The transport authority in South Africa, for example, requires inspectors to be professionally registered engineers with at least 5 years of bridge design experience and has developed a standard visual assessment reference guide (Part B of the TMH19 manual) (COTO, 2016). Nevertheless, inspection ratings in South Africa remain considerably inconsistent as Nordengen & Nell (2005) observed.
Chapter 3: Overview of the Inspection Process and Typical RC Bridge Defects

3.1. Introduction
During bridge inspections, inspectors identify defects and allocate ratings based on their severity and extent. This chapter aims to provide a general overview of the inspection process and the types of defects typically encountered during visual RC bridge inspections.

3.2. The Inspection Process
Bridge structures have 3 main constituent parts, namely the superstructure, substructure and foundations. These can further be subdivided into separate elements that can be assessed individually. During inspection, inspectors systematically inspect each element, take photographs and complete inspection forms. This information is subsequently fed into the inspection module of a BMS. The deck is often given the highest priority during inspections as it is the component of the bridge that directly carries traffic (Ryall, 2010). Somerville (2008), however, asserts that different bridge structures must be treated as unique in the context of their local macro and micro-climate conditions. According to Somerville (2008), it is important that the inspector studies the structure and identifies the most critical zones and the most aggressive local micro-climate prior to inspection. It is these areas that would likely require much of the inspector’s attention. In the case of a bridge subjected to de-icing salts, for example, the inspector may identify the micro-climate conditions shown in Figures 3.1 and 3.2 prior to inspection.

For close observation of defects, inspectors require a suitable means of access to all parts of the bridge. Access ranges from the use of ladders and platforms to specialised mobile inspection units. Simple access equipment can be used to inspect bridges with low deck elevations while heavy mobile equipment may be required for long span high-rise bridges (Rakshit, 2020). In South Africa, specialised access equipment such as Under-bridge Inspection Units (UBIU) are only used when a high-rise bridge has been identified to be in poor condition and in clear need for a more detailed assessment in a project level inspection (COTO, 2016). Mobile access equipment is therefore not used during network-level visual inspections and, as a consequence, the bridge generally
remains open to traffic.

Figure 3.1. Possible micro-climate conditions for a bridge exposed to de-icing salts (Somerville, 2008).

Figure 3.2. Identifying areas exposed to salt spray (Somerville, 2008).

An under-bridge inspection vehicle is shown in Figure 3.3.
3.3. Types of RC Bridge Defects
The purpose of this section is to describe typical RC bridge defects and the ways in which inspectors are able to recognise them during inspections. The focus of this section is on identifying the tell-tale visual characteristics that distinguish different types of defects from each other. It is these visual characteristics that present an opportunity to develop deep learning computer vision models capable of identifying different types of defects using inspection images as input. Although the underlying deterioration mechanisms are briefly described, this section is not intended to provide a detailed treatment (reference can be made to Ballim et al. (2009) and Alexander et al. (2017) where the topic has been covered extensively). Suffice to say, deficiencies in RC bridges are generally due to a combination of one or more aggressive actions (e.g. excessive loading, chloride attack and soft water attack) and construction-related defects (e.g. low cover, inadequate curing and honeycombing) (Somerville, 2008).

3.3.1. Cracking and Spalling
Concrete cracks fall into 2 broad categories: structural and non-structural (Alexander et al., 2017). The former refers to cracks that are load-induced and can be allowed by design in order to mobilise the tensile properties of reinforcing steel. Excessive loading, however, can result in the development of shear and bending cracks that compromise the bridge’s safety.
Non-structural cracks are typically due to restrained volume changes in RC members due to chemical or physical changes occurring within the concrete (Ballim et al., 2009) such as plastic shrinkage, thermal strain and expansive processes (Somerville, 2008). Different non-structural crack patterns are shown in Figures 3.4 and 3.5, while a detailed breakdown of the various underlying causes in hardened concrete is provided in Figure 3.6.

**Crack types:**

- Plastic settlement: A, B, C
- Plastic shrinkage: D, E, F
- Early thermal contraction: G, H
- Long-term drying shrinkage: I
- Crazing: J, K
- Corrosion of reinforcement: L
- Alkali-aggregate reaction (ASR): M

![Figure 3.4. Typical RC cracks (Alexander et al., 2017).](image)

![Figure 3.5. A close-up view of different crack patterns (Rakshit, 2020).](image)
Corrosion of reinforcement is the primary cause of cracks in RC bridges (Rakshit, 2020). It occurs when reinforcing steel reacts with oxygen in a humid atmosphere. This reaction is initiated when the reinforcing steel’s passivating layer is compromised due to chloride attack or carbonation. The expansive nature of the corrosion products generates tensile stresses within the concrete, resulting in the development of cracks parallel to the steel reinforcement.

In the early stages, signs of corrosion come in the form of brown stains along the direction of the main reinforcement. The initial cracks progressively become wider as the corrosion process proceeds until the concrete cover delaminates and spalls (Figure 3.7). Inadequate or porous cover accelerates the corrosion of reinforcing steel (Alexander et al., 2017). It has therefore been observed that handrails are particularly prone to corrosion as they typically have small cover and compaction is inadequate in many cases (Rakshit, 2020).

Figure 3.6. Detailed classification of cracks in hardened reinforced concrete (Alexander et al., 2017).
3.3.2. Leaching
Leaching occurs when soft water percolates through the concrete, dissolving soluble constituent material in the process (Alexander et al., 2017). Since the soluble components of the cement paste can make up to 65% of cement weight (Somerville, 2008), leaching has the effect of increasing the porosity and permeability of the concrete (Alexander et al., 2017). As a result, the concrete becomes vulnerable to other forms of chemical attack and loses some of its strength and stiffness (Somerville, 2008). Excessive leaching has been found to result in strength losses of up to 50% especially in structures exposed to soft water over long periods of time (Alexander et al., 2017). Leached material crystallizes on the surface of the concrete as the water evaporates or by reacting with carbon dioxide in the atmosphere, forming white deposits on the concrete surface as Figure 3.8 shows (Alexander et al., 2017).

Figure 3.8. Efflorescence due to lime leaching (Ryall, 2010).
3.3.3. Disintegration, Abrasion and Scaling of Concrete
Disintegration is when concrete breaks into small pieces that are subsequently removed from the parent body. This can occur due to wetting and drying cycles in the spray and splash zone. In this case, chloride salts penetrate the concrete during wetting and go on to expand and disintegrate the concrete during drying (Rakshit, 2020). Disintegration also occurs when salt-bearing salts rise through the concrete by capillary action and crystallize above high-water level (Alexander et al., 2017; Rakshit, 2020).

Abrasion occurs when the concrete surface wears due to traffic and water-borne objects, leading to mass loss and reduced cover (Alexander et al., 2017), while scaling is said to have taken place when the near surface of concrete peels off leaving the coarse aggregates exposed (Rakshit, 2020).

3.3.4. Excessive Deflection and Vibration
Excessive deflection and vibration can be assessed during visual inspections by observing bridge behavior while a heavy vehicle crosses. Deflections are considered excessive when they can be observed without measurement Ryall (2010).

3.3.5. Accidental Damage
Accidental occurrences such as vehicular impact, scrapping, skidding or overturning can cause damage to bridge structures. The deck and piers are particularly vulnerable to such damage as they are most exposed to traffic (Ryall, 2010).

3.3.6. Defects to Deck Surface Material
Due to direct exposure to the sun, deck surfacing material such as asphalt is susceptible to overheating and rutting. Leaked rainwater can also freeze beneath the surface thereby dislodging the surface material. In the event that its adhesion to the deck is insufficient, it can also be pulled off by traffic (Ryall, 2010).

3.3.7. Foundation Displacement
Foundation damage is often not readily visible and can therefore be difficult to inspect directly (Ryall, 2010). Settlement in foundations can be assessed by visually checking the vertical and horizontal alignment of railings and walls (COTO, 2016). In more detailed inspections, accurate surveying equipment may be used to ascertain the structure’s geometry.
3.3.8. Damage to Bridge Accessories

Bridge accessories refer to expansion joints, bearings, parapets, waterproofing and drainage. The types of damage typical to each are summarised in Table 3.1. During inspection, any such damage is recorded and possible causes are diagnosed.

<table>
<thead>
<tr>
<th>Accessory</th>
<th>Typical Damage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expansion joints</td>
<td>Tracking; cracking; debonding; excessive or limited movement; leakage; misalignment.</td>
</tr>
<tr>
<td>Bearings</td>
<td>Presence of debris, bird dropping or vegetation in bearing shelves; loose bearings; corrosion; spalling; cracking.</td>
</tr>
<tr>
<td>Parapets</td>
<td>Corrosion; accidental damage.</td>
</tr>
<tr>
<td>Waterproofing</td>
<td>Leakage; surface damage due to traction and braking forces.</td>
</tr>
<tr>
<td>Drainage</td>
<td>Blockages; standing water; pipe leakage.</td>
</tr>
</tbody>
</table>

3.3.9. Other Factors that Contribute to RC Bridge Deterioration

As indicated earlier, the inspection of RC bridges must be conducted in the context of each structure’s local macro and micro-climate conditions (Somerville, 2008). For example, the presence of a chemical factory in close proximity to a bridge creates an aggressive environment around the structure where other deterioration mechanisms such as acid or sulphate attack become relevant (Rakshit, 2020). Due consideration must also be given to the fact that such local micro-climates can vary significantly in different locations and may depend on the season of the year (Somerville, 2008). The defects described in this section are therefore not exhaustive and additional types of defects may be encountered depending on the unique circumstances surrounding the bridge in question.

It has also been observed that the quality of concrete and workmanship have a significant bearing on the deterioration of RC bridges. Possible reasons for poor quality concrete include (Alexander et al., 2017; Rakshit, 2020):

i. Poor quality of constituent concrete material – reduces concrete strength.
ii. The presence of detrimental chemicals in constituent concrete material – leads to early corrosion.
iii. Poorly graded aggregates, defective formwork or inadequate compaction – produce porous concrete.
iv. Too high or too low water/cement ratio – the former reduces strength, while the latter reduces workability and increases porosity especially in areas where there is congestion of reinforcement.

v. Poor curing practices – reduces concrete strength, increases porosity and could result in delayed ettringite formation (DEF).

vi. Poor quality control – leads to failure to ensure good workmanship.

Corrosion, being the primary cause of deterioration in RC bridges, requires special attention at the design and construction stages. In chloride contaminated environments, cements rich in aluminate phases that bind chloride ions would be required (Alexander et al., 2017). Likewise, carbonating conditions require the use of cement with a high CaO content (Alexander et al., 2017). Therefore, failure to incorporate such exposure conditions in a bridge’s design and construction can result in accelerated deterioration of the structure.
Chapter 4: Literature Review

4.1. Introduction
Conventional visual inspections typically involve inspectors walking on a bridge deck and making use of different types of access equipment to observe points of interest and allocate ratings. However, in a study involving the deployment of 49 accredited inspectors for the inspection of 6 bridges in the USA, Phares et al. (2001) found that 95% of primary bridge element condition ratings from visual inspections varied within an average of 2 rating points, while 68% varied within 1 rating point due to inspector subjectivity. Nordengen (2012) found that the subjective nature of inspections tends to result in the allocation of conservative defect ratings that may not accurately portray the true condition of the bridge. Nevertheless, most transport authorities still consider visual inspections to be the most reliable network-level inspection method (Dorafshan et al., 2018). It has been shown in other industries, particularly manufacturing and aviation, that inspector subjectivity, bias and human error can be reduced significantly through automation (Dorafshan et al., 2018). This chapter therefore seeks to interrogate state-of-the-art techniques capable of automating the allocation of defect ratings during RC bridge inspections. The use of Unmanned Aerial Vehicles (UAVs) and robots to improve inspector safety and reduce inspection costs is also explored.

4.2. Overview of Bridge Inspection Automation
The International Organisation for Standardisation (ISO 8373: 2012) defines automation as inducing the ability to perform tasks in the absence of human intervention. It can generally be implemented at five key areas of the bridge inspection process, namely navigation, data collection, data processing/analysis, data visualisation and decision making (Agnisarman et al., 2019). The types of automation that can be introduced at each stage are summarised in Figure 4.1.

Much of the research on bridge inspection automation focuses on the navigation, data collection and data processing/analysis stages. At the navigation and data collection stages, multisensory UAVs and robots are the most commonly cited means of automation (Leibbrandt et al., 2012; Gucunski et al., 2015; Dorafshan & Maguire, 2018; Seo et al., 2018; Agnisarman et al., 2019).
Figure 4.1. Types of automation possible at various stages of the inspection process (Agnisarman et al., 2019).
Automation at the data processing/analysis stage is usually conducted using computer vision algorithms (Zhang et al., 2016; Cha et al., 2017; Li & Zhao, 2018; Dorafshan et al., 2018; Dung & Anh, 2019; Agnisarman et al., 2019). Nowadays, UAVs and robots are often embedded with computer vision systems such that the navigation, data collection and data processing/analysis steps are all automated simultaneously. It is important, however, that all parties involved in the inspection process have sufficient trust in the technology and its implementation so that the inspection results obtained from such an automated system have credence (Agnisarman et al. 2019).

The degrees of automation at each stage of the inspection process can be described using the scale of human-machine interaction presented in Table 4.1 (Sheridan, 2002).

### Table 4.1. Sheridan’s scale of human-machine interaction (Sheridan, 2002; Agnisarman et al., 2019)

<table>
<thead>
<tr>
<th>Level of automation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Automation offers no aid; Human in complete control.</td>
</tr>
<tr>
<td>2</td>
<td>Automation suggests multiple alternatives, filters and highlights what it considers to be the best alternatives.</td>
</tr>
<tr>
<td>3</td>
<td>Automation selects an alternative, one set of information, or a way to do the task and suggests it to the person.</td>
</tr>
<tr>
<td>4</td>
<td>Automation carries out the action if the person approves.</td>
</tr>
<tr>
<td>5</td>
<td>Automation provides the person with limited time to veto the action before it carries out the action.</td>
</tr>
<tr>
<td>6</td>
<td>Automation carries out an action and then informs the person.</td>
</tr>
<tr>
<td>7</td>
<td>Automation carries out an action and informs the person only if asked.</td>
</tr>
<tr>
<td>8</td>
<td>Automation selects method, executed task, and ignores the human (the human has no veto power and is not informed).</td>
</tr>
</tbody>
</table>

An automation system that combines UAVs or robots with computer vision algorithms for bridge inspections is a Level 3 system (Table 4.1). This is because the system collects data (i.e. inspection images), processes it and provides the operator with one best solution (e.g. defect type or rating output).

The technologies typically used to automate the various stages of the bridge inspection process, namely UAVs, robots and computer vision algorithms, are reviewed in
Sections 4.3, 4.4 and 4.5, respectively.

4.3. UAV-Enabled Bridge Inspections
In an effort to overcome challenges associated with the need for access equipment, there has been a growing interest in the use of UAVs during bridge inspections. Research in this area has been motivated by rapid advances in the capabilities of UAV and digital camera technologies, and the falling costs of each (Adams et al., 2013). The drive has been to use UAVs to inspect bridge elements that are particularly difficult or expensive to access through traditional means. While several studies have been conducted to assess their effectiveness, the study by Wells & Lovelace (2018) is perhaps the largest and most comprehensive research on the topic to date. Together with the Minnesota Department of Transportation (MnDOT), they administered the inspection of over 60 bridges in Minnesota, USA, using UAVs (Wells & Lovelace, 2018). Some of the conclusions from this study were:

i. The use of UAVs can lead to a reduction of inspection costs by approximately 40% on average, without compromising inspection quality.

ii. Cost savings due to UAV use are greatly increased in cases where under-bridge inspection vehicles would otherwise be required.

iii. The use of UAVs greatly reduces safety risks associated with working at heights and in close proximity to traffic.

4.3.1. UAV Capabilities Required for Bridge Inspections
Various types of UAVs that cater for a wide-range of users are currently available on the market. Examples of these are shown in Figure 4.2. Rotorcrafts are most suitable for UAV-enabled inspections as they are highly maneuverable and have hovering capabilities (Floreano & Wood, 2015). Collision-tolerant rotorcrafts such as that shown in Figure 4.2b (e.g. the ‘Flyability Elios’) are surrounded by a protective cage that allows the UAV to safely come in contact with the bridge while in flight. Such rotorcrafts are also able to roll along the structure during inspections. This has been found to be particularly useful when inspecting very tight areas or confined spaces as Figure 4.3 illustrates (Wells & Lovelace, 2018). However, the protective cage of these UAVs are always partially visible in inspection photographs and videos (Wells & Lovelace, 2018).
Figure 4.2. Examples of different UAV types: a, Fixed-wing UAVs; b, Rotorcraft; c, Flapping-wing UAVs; d, Coordinate system of a generic UAV (Floreano & Wood, 2015).

Figure 4.3. A collision-tolerant rotorcraft rolling inside a steel box beam (left) and on top of an abutment bearing seat (right) (Wells & Lovelace, 2018).

Overall, UAVs must possess the following capabilities in order to perform effective UAV-enabled inspections (Seo et al., 2018; Wells & Lovelace, 2018):

i. They must be able to stay airborne for at least 20 minutes. This minimises interruptions to the inspection process when changing batteries.

ii. They must have long-range remote control capabilities. This feature is useful in the event that the bridge structure in question is physically inaccessible to inspectors.

iii. They must either have an additional camera on top or a single camera with vertical rotation capabilities so as to allow the inspection of soffits.

iv. They must be able to carry the required payload e.g. cameras and flashlights.
v. They must be able to fly without the need for a GPS signal as is sometimes the case underneath bridges and in confined spaces.

vi. They must be collision tolerant when used to inspect confined spaces and in cases where the camera needs to be within a few meters from the inspection element under observation.

vii. They must be able to carry out pre-programmed flights when required. According to Wells & Lovelace (2018), this saves time on site and allows pilots to better prepare for site conditions.

viii. When required, they must be capable of sensing and measuring distance from nearby objects (above, below and ahead) during flights.

ix. They may ideally have distance-lock and cruise control features so that photographs of vertical and horizontal surfaces can be taken at a fixed predefined working distance when required.

x. The camera must be capable of capturing geo-tagged, high resolution images and videos in low illumination conditions e.g. underneath the deck (Wells & Lovelace, 2018). It must also be capable of capturing narrow cracks with sufficient clarity from the chosen working distance. For example, the in-built camera aboard the senseFly Albris UAV is capable of capturing 0.1mm cracks from a maximum working distance of 0.6m (senseFly, 2017).

A limited number of off-the-shelf inspection-grade UAVs that possess a majority of the above requirements are available at the time of writing, e.g. senseFly Albris, Intel Falcon 8+, Aeyron Skyranger and the DJI M200 series. Consumer-grade UAVs are typically deficient of many of the advanced inspection-specific features (Wells & Lovelace, 2018). Some state-of-the-art inspection-grade UAVs possess computer vision features for the detection of defects (e.g. senseFly Albris).

4.3.2. Levels of Available UAV Autonomy
The automation of UAVs can be described at three levels, namely sensory-motor autonomy, reactive autonomy and cognitive autonomy (Floreano & Wood, 2015):

i. Sensory-motor autonomy: the ability to convert high-level human commands (e.g. to move to specified GPS coordinates) into combinations of control signals for the UAV (e.g. pitch, roll or speed). Human supervision is required during operations.
ii. Reactive autonomy: the ability to react to or interact with external disturbances, obstacles or other moving objects. This requires the UAV to also possess sensory-motor autonomy. UAVs with reactive autonomy require little human supervision.

iii. Cognitive autonomy: the ability to conduct simultaneous localisation and mapping, plan operations, address conflicting information, recognise objects or people, and/or learn. To have cognitive autonomy, the UAV must also possess reactive autonomy. No human supervision is required.

At present, state-of-the-art inspection rotorcrafts have, to a limited degree, achieved reactive autonomy (Floreano & Wood, 2015). The senseFly Albris UAV, for example, is capable of automatically avoiding obstacles and hovering at a fixed predefined position all in the presence of external disturbances such as wind (Wells & Lovelace, 2018). UAVs capable of cognitive autonomy are still subject of research in the field of artificial intelligence (Kersandt et al., 2018; Wu et al., 2019).

4.3.3. **Deliverables from UAV-enabled Inspections**
A major advantage of using UAVs for bridge inspections relates to how the collected inspection data can later be processed. A number of photogrammetry software packages (e.g. Pix4D and Recap) are available to process high resolution UAV imagery to produce, among others, 3D photorealistic models, orthomosaics and orthoplanes (Wells & Lovelace, 2018). These are very useful when it comes to effective communicating of inspection results to bridge owners and a non-technical audience. Just as importantly, they supplement inspection and inventory data for storage in a BMS.

Unlike traditional inspection photologs, 3D photorealistic models are made up of very precise georeferenced point clouds and triangular meshes that allow deficiencies to be measured and annotated (Wells & Lovelace, 2018). This also allows photographs taken during an inspection to be located and referenced to the bridge model. The model is also interactive, allowing inspection images captured at a particular area or on a specific element to be displayed on-screen when clicked. In addition, the model captures bridge condition at a specific point in time. The condition of the bridge can therefore be monitored more closely as the bridge ages. For comparison, an example
of a traditional and a 3D inspection photolog is presented in Figure 4.4 and Figure 4.5, respectively.

Figure 4.4. Example of a traditional inspection photolog (Wells & Lovelace, 2018).

Figure 4.5. Example of a Pix4D 3D photolog (Wells & Lovelace, 2018).

2D orthomosaics and orthoplanes can also be produced from UAV-enabled inspections. These are large top-down and planar images, respectively, that are obtained by combining multiple UAV photographs (Wells & Lovelace, 2018). Examples of these are provided in Figures 4.6 and 4.7.
Inspection results in the form of 3D models, orthomosaics and orthoplanes are often difficult to share to interested parties using traditional methods (e.g. email) owing to their large file sizes. To circumvent this, they are typically shared using cloud-based platforms that allow users to access the inspection results without first downloading and storing large amounts of data (Wells & Lovelace, 2018).

4.3.4. Aviation Authority Regulations
While UAV use in bridge inspections has several benefits, a common barrier to implementation reflected in literature relates to strict rules and regulations set out by aviation authorities (Chan et al., 2015; Zink, 2016; Wells & Lovelace, 2018). The aviation industry generally has a conservative safety culture which has resulted in aviation authority rules and regulations being slow to adapt to new opportunities (Wells & Lovelace, 2018). For this reason, a number of approvals from the relevant aviation authority are required before any UAV-enabled inspections can take place. These approvals, however, can take significant amounts of time to obtain and are often costly (Zink, 2016). Some authorities, such as the American Federal Aviation Administration (FAA), are actively seeking to eliminate these obstacles so that UAVs can be adopted...
more widely in future (Zink, 2016). In South Africa, the Commercial Unmanned Aviation Association of Southern Africa (CUAASA) has been making efforts to ensure that the South African Civil Aviation Authority (SACAA) does the same. One major challenge the SACAA is looking to overcome pertains to significant backlogs in processed Remotely Piloted Aircraft System Operators Certificate (ROC) applications. These backlogs amounted to approximately 400 applications as of 2018 (CUAASA, 2018).

Aviation authority regulations generally differ between countries. While countries such as the UK prohibit the operation of UAVs beyond line of sight near congested areas, for example, France allows their use in this scenario under certain conditions (Floreano & Wood, 2015). By default, South Africa prohibits the use of UAVs in this manner unless the responsible organisation provides sufficient documentation that demonstrates their ability to conduct such operations safely and to SACAA’s satisfaction (Kock, 2015). Other activities pertinent to UAV-enabled inspections that are prohibited by SACAA (under Part 101 of the Civil Aviation Act) without special approvals include:

i. Conducting operations within restricted airspace;
ii. Conducting operations directly overhead or within a 50 m lateral distance from any person apart from persons involved in the UAV operations; and,
iii. Conducting operations within a 50 m lateral distance from any structure or public road.

It is therefore important to take Aviation Authority regulations into account prior to deploying UAVs for bridge inspection purposes.

4.4. Robotic Bridge Inspections
The earliest form of robotic bridge inspections were ground vehicles used for deck inspections (Dorafshan & Maguire, 2018). An example of such a vehicle is the multi-sensor RABIT Bridge Deck Assessment Tool used to detect bridge deck defects (Figure 4.8) (Gucunski et al., 2015). The RABIT was fitted with a high resolution digital camera and several on-board sensors; namely impact echo, ultrasonic, ground penetrating radar and resistivity sensors. Further advances in the field of robotics led to the development of climbing robots capable of reaching difficult-to-access bridge elements. An example of this is the climbing robot developed by Nguyen et al. (2020)
for the inspection of steel bridges.

![The RABIT bridge deck inspection vehicle (Gucunski et al., 2015).](image)

**Figure 4.8.** The RABIT bridge deck inspection vehicle (Gucunski et al., 2015).

![An example of a climbing inspection robot (Nguyen et al., 2020).](image)

**Figure 4.9.** An example of a climbing inspection robot (Nguyen et al., 2020).

The development of inspection robots is currently an emerging area subject of considerable research. The main motivation for robot-aided inspections is to negate the need for access equipment while carrying out non-destructive tests on bridge structures. They are therefore suitable for application during detailed project-level inspections of long span high-rise bridges.
4.5. Computer-Vision Based Bridge Inspections

Computer vision (CV) is a broad field that deals with the means by which computers can gain a high level understanding of scenes or objects depicted in digital images or videos (Sonka et al., 2014). Advances in this field have led to the development of several techniques that seek to automatically extract information from digital imagery or video data. Some of these techniques, particularly deep learning, have shown promising results with regards to automating the detection of visible defects during RC bridge inspections. Indeed, several studies exist that involve the development of computer-vision based systems for concrete defect detection. Much of these studies, however, focus on the detection of cracks. Examples of research in this area include studies by Cha et al. (2017), Li & Zhao (2018), Zhang et al. (2016), Dorafshan et al. (2018) and Dung & Anh (2019). In these studies, deep learning algorithms were used to develop RC crack-detection models with high accuracies (> 95%). The detection of non-crack defect types, however, has received relatively little attention largely due to the tediousness of image acquisition (Mundt et al., 2019). Few studies, including those by Mundt et al. (2019) and Yang et al. (2017), sought to develop deep learning models capable of detecting other defect categories such as spalling, corrosion and efflorescence. Models with test accuracies of 72% and 70% were attained in these two studies, respectively. Makantasis et al. (2015) developed a similar, 90% accurate defect-detection deep learning model for use in tunnel inspections.

From the aforementioned studies, the following trends were observed:

i. Convolutional neural network (CNN) algorithms were used to develop models for automated crack/defect detection. This was done by training CNN algorithms on large quantities of annotated inspection images so that they learn to accurately recognise visible defects.

ii. A number of high-resolution defect images captured by a handheld digital camera, smartphone or UAV were initially collected. Prior to training, these images were cropped to a number of smaller images (typically 256 x 256 resolution or less) to increase dataset size and reduce computational processing requirements.

iii. The distance between the camera and point of interest on the concrete surface while compiling training datasets varied across the studies. For example, the
SDNET2018 dataset compiled by Dorafshan et al. (2018) comprised images captured at a fixed 0.5m distance using a 16MP camera. In contrast, Cha et al. (2017) opted to vary this distance between 1.0 and 1.5m using a 24.1MP camera.

CNNs have mainly been used to build 3 types of systems for RC defect detection, namely image classification, image segmentation and object detection systems. Image classification is the process of taking an image with one or more objects of interest (e.g. a crack) as input and automatically assigning a suitable class label to it (Russakovsky et al., 2015). This process involves building a CNN model capable of automatically specifying which of \( k \) categories an object in an input image belongs to (Goodfellow et al., 2016). This process is illustrated in Figure 4.10.

![Figure 4.10. Deep learning for image classification (Perez et al., 2019).](image)

Li & Zhao (2018) developed an image classification model for concrete crack detection and integrated it into a smartphone application (Figure 4.11).

Unlike image classification, image segmentation is the process of partitioning images into multiple segments, each segment representing a particular object (Minaee et al., 2020). It is concerned with identifying groups of pixels that ‘go together’ in an image (Szeliski, 2010). Makantasis et al. (2015) used this technique to develop a defect detection system for tunnel inspections (Figure 4.12).

Object detection is an extension of image classification whereby a border (or bounding box) is produced around detected defects to mark their location in an image
(Russakovsky et al., 2015). An illustrative example of this process during a UAV-aided windmill inspection is provided in Figure 4.13. At the time of writing, there have been no studies that deploy deep learning models to rate the severity and extent of RC bridge defects once they are identified.

Figure 4.11. Image classification as applied to crack detection (Li & Zhao, 2018).

Figure 4.12. Application of image segmentation algorithms for tunnel defect detection: (a) Original image (b) Detected defects (Makantasis et al., 2015).
4.6. **Summary**

The highly subjective nature of visual inspections has been shown to result in inaccurate prioritisation of bridge repair and rehabilitation activities in spite of extensive inspector training (Phares et al., 2001; Nordengen & Nell, 2005). Furthermore, access to difficult-to-reach elements often requires the use of costly access equipment and exposes inspectors to various safety hazards. By automating the defect rating allocation process, inspections become less observer-dependent, reducing the effect of human error, bias and subjectivity on inspection results and improving consistency. It also has the effect of improving inspector safety during inspections and reducing overall inspection costs in the case of long span high-rise bridges (Wells & Lovelace, 2018).

The automation of RC bridge inspections has largely focused on onsite navigation, data collection and data processing/analysis. Multisensory UAVs and robots are capable of effectively automating navigation and data collection, while data-driven computer vision algorithms, particularly CNNs, have proven useful in terms of automating the data processing and analysis steps of the inspection process. State-of-the-art inspection UAVs and robots are often equipped with computer vision algorithms such that onsite navigation, data collection and data processing/analysis can be conducted simultaneously and in real-time.

---

*Figure 4.13.* Application of object detection algorithms for automated windmill inspections (Google, 2019).
While computer vision algorithms have proven capable of automatically detecting and characterizing visible defects, no studies exploring their usefulness in the allocation of Severity and Extent ratings have been published at the time of writing.
Chapter 5: Overview of Deep Learning and Convolutional Neural Networks

5.1. Introduction
Pre-2012, feature-based learning was prominently used for image classification tasks (Koul et al., 2019). This method involved first applying a feature extraction algorithm to image pixels to produce a vector that quantifies the contents of the image. These vector outputs were subsequently used to train traditional machine learning models (e.g. Support Vector Machines) for image classification (Rosebrock, 2017). The feature extraction algorithms (e.g. Histogram of Oriented Gradients or HOG) used are manually developed/handcrafted to quantify specific features of interest in an image e.g. colour, shape or texture. Deep learning (DL) algorithms, in contrast, extract features from images automatically. Deep learning, a subset of artificial intelligence (AI) and machine learning (ML), is a class of algorithms that makes use of multiple stages (or layers) of non-linear computations to progressively and automatically extract/identify high level features from raw input (Deng & Yu, 2014). This technique has particularly excelled in complex applications, including computer vision, speech recognition and natural language understanding (Koul et al., 2019). The family of algorithms used in such applications are called neural networks (NNs). For computer vision, in particular, convolutional neural networks (CNNs) are the specific class of neural networks used. These algorithms take raw image pixels as direct inputs and automatically learn features through a training process to produce image classification models (Rosebrock, 2017). During training, the CNN learns features hierarchically, with its earlier layers detecting lower-level features (e.g. edges) while the later layers detect more abstract features (e.g. complex shapes) (Koul et al., 2019). Since this process is automatic, the development and application of feature extraction algorithms is not required. Figure 5.1 highlights the differences between feature-based learning and deep learning.

In 2012, CNNs outperformed traditional feature-based methods for the first time (Krizhevsky et al., 2012) and have since become state-of-the-art for image classification, segmentation and detection tasks.
5.2. A Brief History of Deep Learning

Due to rapid improvements in computer processing hardware such as Graphical Processing Units (GPUs) and the proliferation of cheap sensing devices in the 2000s, significant research went towards the practical application of CNNs for computer vision tasks in the last decade. Russakovsky et al. (2015) developed the ImageNet dataset consisting of 14 million images and launched the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2010. The ILSVRC challenged the research community to continuously develop improved techniques to classify ImageNet images. The competition has since become a benchmark for evaluating the effectiveness of various classification and detection algorithms. Figure 5.2 shows the annual classification error (i.e. proportion of incorrectly classified images) attained by different techniques between 2010 and 2017. It can be seen that traditional feature learning-based methods (with 0 layers) were state-of-the-art in 2010 and 2011, achieving classification errors of 28% and 25% respectively. In 2012, a CNN model won the competition for the first time, nearly halving classification error from 25% to 15%. Since then, traditional feature learning-based approaches were abandoned in favor of deep

![Figure 5.1. Traditional feature learning versus deep learning for image classification (Rosebrock, 2017).](image)
learning methods for computer vision applications.

![ImageNet LSVRC Winning Entries](image)

**Figure 5.2.** LSVRC results between 2010 and 2017 (Koul et al., 2019).

As computer processing speeds continued to increase, deeper CNNs (i.e. with increasing number of layers) were developed resulting in a year-on-year decrease in classification error. According to He et al. (2015), CNNs surpassed human-level performance in image classification for the first time in 2015, further lowering error rates to approximately 4%. By 2017, a 2% error rate had been achieved by a 200-layer deep CNN.

### 5.3. Influence of Network Depth

As Figure 5.2 shows, the ability to make use of multiple layers has been key to the success of deep learning models. Ng (2018) notes that the performance (i.e. classification accuracy) of traditional algorithms tends to eventually plateau even when the quantity of training data is increased (Figure 5.3). In the case of deep learning algorithms, however, the number of layers can be manipulated to increase the algorithm’s capacity for more data and ensure a corresponding improvement in classification accuracy (Aggarwal, 2018). The ability to increase the number of layers to handle extensive amounts of data ensure that CNNs do not reach a saturation point. However, increasing the depth of CNNs increases computational costs of network training (Goodfellow et al., 2016). As a result, computational resources available for
training can impose a ceiling on the accuracy that a CNN can attain in practice.

![Graph showing the effect of amount of training data on performance of CNNs and traditional machine learning algorithms](image)

**Figure 5.3.** Effect of amount of training data on performance of CNNs and traditional machine learning algorithms (Aggarwal, 2018).

Another benefit of the deep learning approach stems from the Universal Approximation Theorem which states that “A neural network with a single hidden layer can represent any continuous function within a specific range, acting as a universal approximator” (Koul et al., 2019:579). This implies that a CNN with a single layer could, in theory, model any machine learning problem. However, using a single layer for complex tasks is usually infeasible due to computational constraints. It is therefore more common to use multiple layers in practice (Koul et al., 2019).

### 5.4. A High-level Overview of CNNs and their Inputs

The inputs for CNNs are image pixel intensity values. Different layers of the network act on these values to extract features and ultimately classify objects. A colour image can be represented as an array of integer pixel values in three dimensions (Koul et al., 2019). The width (W) and height (H) of the image are defined by rows and columns of pixel values, while depth (D) is described by the number of Red, Green and Blue (RGB) channels in the image. Each pixel value ranges from 0 to 255, representing the intensity of red, green and blue at any one position in the image. Grayscale images, however, are made up of a grid of pixel values in one channel. Unlike colour images, their pixel values represent varying shades of gray with values closer to 0 being progressively darker than those closer to 255 (Rosebrock, 2017). For image classification, the array of numbers representing an input image is sometimes referred to as an input volume.

A CNN can be described as a complex non-linear function composed of a hierarchical
and interconnected assembly of simple linear and non-linear functions (Wu, 2017). They are composed of 2 main parts: convolutional (Conv) layers and fully connected (FC) layers. The role of convolutional layers is to convert a large number of pixel values of an input image into a smaller array of values that represent image features. The fully connected layers subsequently convert these features into probabilities. The object class with the highest probability becomes the main output (or prediction) of the network. Convolutional and fully connected layers therefore act as feature extractors and classifiers, respectively (Koul et al., 2019). The earlier layers of the network are used to extract basic visual features (e.g. edges), while deeper Conv layers extract more complex patterns (Aggarwal, 2018). A high-level overview of this process is shown in Figure 5.4. A more detailed description of the feature extraction process is provided in Section 5.6.

In addition to convolutional and fully connected layers, CNNs are also made up of other layer types; namely the loss, pooling and ReLu layers. These layers are not shown in Figure 5.4 for ease of illustration. The functions of loss, pooling and ReLU layers are described in Section 5.5.1, Section 5.7.1 and Section 5.7.2, respectively.

5.5. The Training Process
A high-level overview of the training process is shown in Figure 5.5. Image classification models are trained using image inputs and a set of variables called weights (or parameters). These weights are initially random and define how the model operates. During training, the model and its weights act on image inputs to make
predictions. The predictions are subsequently compared to ground truth (i.e. correct image classes) to evaluate the model’s performance. Weights are then updated such that the performance of the model improves. This process is repeated until weights that achieve maximum model performance are found. (Gugger & Howard, 2020)

![Figure 5.5. An overview of the model training process (Gugger & Howard, 2020).](image)

A more detailed overview of the training process is provided in the following sections.

### 5.5.1. Forward Pass and Back Propagation

The process of accepting an input image and subjecting it to a sequence of CNN layers to obtain an output is called a forward pass (Wu, 2017). An abstract description of this process is provided in Equation 5.1.

$$x^1 \rightarrow \theta^1 \rightarrow x^2 \rightarrow \ldots \rightarrow x^{L-1} \rightarrow \theta^{L-1} \rightarrow x^L \rightarrow J(\theta)$$  \hspace{1cm} (5.1)

Conceptually, the following computations take place during a forward pass (Wu, 2017; Koul et al., 2019):

i. The input image matrix (an array of pixel values), $x^1$, is subjected to the first layer (represented by the box) using weights $\theta^1$ to produce the output $x^2$ which becomes the input to the second layer. This process continues through to the final fully connected layer.

ii. The fully connected layer gives the prediction of the network in the form of a vector with probabilities for each object class, $x^L$. The probabilities denote the network’s estimation of the likelihood of features in the image belonging to a particular object class.

iii. Vector $x^L$ goes through an additional layer called the loss layer. This layer computes the distance $J(\theta)$ between a target vector $t$ (with correct probability values) and the predicted vector $x^L$. This is typically performed using a loss function that computes the cross-entropy loss of the prediction.
The cross-entropy loss $J(\theta)$ initiates a process called back propagation whereby the weights of each layer are updated to improve model performance (Gugger & Howard, 2020). Another forward pass is subsequently performed using the new weights. Forward passes and back propagations are repeated until weights that minimise the cross-entropy loss function are found.

### 5.5.2. Estimation of Class Probabilities

Minimising the loss function ensures that the resulting model estimates high probabilities for instances where a particular object is present in an input image, and low probabilities for negative cases. This is done using a softmax regression model and applying the softmax function (Géron, 2017). For each image $x$, the softmax regression model computes a comparative value (or score) $s_k(x)$ for each class $k$ according to Equation 5.2.

$$s_k(x) = \theta_k^T \cdot x$$  \hspace{1cm} (5.2)

where $\theta$ is a parameter matrix.

The softmax function is subsequently applied to each class score to estimate the probability $\hat{p}_k$ of the instance belonging to class $k$ (Géron, 2017). As Equation 5.3 shows, the softmax function computes the exponential of each score and divides them by the sum of all the exponentials.

$$\hat{p}_k = \frac{\exp(s_k(x))}{\sum_j \exp(s_j(x))}$$  \hspace{1cm} (5.3)

where $K$ is the total number of classes.

The cross-entropy loss function is then applied to the computed probabilities to measure how well they match the target classes (Géron, 2017). This function is formally described in Equation 5.4.

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} y_i \log(\hat{p}_k)$$  \hspace{1cm} (5.4)
where \( y_k(i) = 1 \) if the target class for the \( i \)th instance is \( k \), or \( y_k(i) = 0 \) if otherwise.

A softmax regression classifier can then be used to predict the class with the highest probability as shown in Equation 5.5 (Géron, 2017).

\[
\hat{y} = \arg \max_k s_k(x) \tag{5.5}
\]

5.5.3. Readjustment of Weights

During training, the first forward pass makes use of initially random weights in each layer (Koul et al., 2019). The process of updating these weights during back propagation is then performed through a process called Stochastic Gradient Descent (SGD) (Wu, 2017). A mathematic description of SGD is provided in Equation 5.6.

\[
(\theta(t+1))^i = (\theta(t))^i - \eta \frac{\partial J(\theta)}{\partial (\theta(t))^i} \tag{5.6}
\]

where \( \eta \) is the learning rate.

Equation 5.6 shows that SGD takes a point on the error surface at time \( t \) and moves it in the direction opposite that of the gradient at that point, towards the global minimum (Figure 5.6). This downward movement is described by the negative partial derivative \( \frac{\partial J(\theta)}{\partial (\theta(t))^i} \).

![Figure 5.6. Minimising the loss function through SGD (Géron, 2017).](image)

The learning rate (\( \eta \)) controls the amount by which the point shifts at each learning
step. \( \eta \) is typically a small number (e.g. \( 10^{-3} \)) to ensure that the parameters are incrementally updated by a small amount at a time (Géron, 2017). The incremental updates continue until the lowest possible \( J(\theta) \) (i.e. the global minimum) is achieved (Wu, 2017).

When all the training examples have been used for a round of weight updates at a particular learning rate, an epoch is said to have been processed (Wu, 2017).

### 5.6. The Feature Extraction Process

Convolutional layers consist of a number of filters (or kernels) that slide across the width and height of the input image. This is a stepwise process in which filters shift vertically and horizontally by a fixed magnitude called a stride. For example, a stride value of 1 indicates that the filters move one pixel at a time across the image. At each position during this process, a dot-product is computed between the filter and local areas of the input image. This process, called convolution, produces a set of 2-dimensional activation (or feature) maps that are indicative of each filter’s response at every spatial position. Each convolutional layer aims to learn a group of filters during training, with each filter dedicated to identifying a particular visual feature in the input volume. The activation maps produced by the first convolutional layer are subsequently stacked together to produce an output volume that is transmitted to the following layer. A high-level illustration of this process is provided in Figure 5.7. (Wu, 2017)

![Figure 5.7](image.png)

**Figure 5.7.** Conceptual illustration of the convolution process (Rosebrock, 2017).

The convolution operation is illustrated in Figure 5.8.
Filters in deeper convolutional layers are able to identify more complex patterns by hierarchically building on the features detected by earlier layers (Aggarwal, 2018). An example of this process is shown in Figure 5.9.

Since activation maps are obtained by computing dot-products at each spatial location of the input volume, the spatial size of the feature maps is always smaller than that of the input image (Wu, 2017). The output volume depth, however, is equivalent to the
number of filters used in the convolution process. In cases where it is desirable for the output volume to have the same width and height as the input volume, a process called zero-padding with a stride of 1 is often used. The zero-padding technique involves adding \((F - 1)/2\) pixel values of zero around the border of an input volume to preserve its spatial footprint (Figure 5.10); where \(F\) represents the size of one side of the square-shaped filter being applied (Aggarwal, 2018). This way, information loss at the border is minimised during successive convolutions.

![Figure 5.10. An example of zero-padding (Aggarwal, 2018).](image)

The stride (\(S\)) alongside the input size (\(W_1 \times H_1\)), filter size (\(F\)) and amount of zero-padding (\(P\)) has a bearing on the spatial size of the output volume. This is formally captured in Equations 5.7 and 5.8, while the depth (\(D_2\)) of the output volume is equal to the number of filters (\(K\)) used (Equation 5.9). (Li et al., 2019)

\[
\begin{align*}
Output \ Width, \ W_2 &= \frac{W_1-F+2P}{S} + 1 \\
Output \ Height, \ H_2 &= \frac{H_1-F+2P}{S} + 1 \\
Output \ Depth, \ D_2 &= K
\end{align*}
\]

**5.7. Pooling and ReLU Layers**

In addition to convolutional and fully connected layers, CNNs also consist of pooling and Rectified Linear Unit (ReLU) layers.
5.7.1. The Pooling Layer
The Pooling layer is used to effect downsampling (i.e. reduce the spatial size of an input volume) in order to reduce the number of parameters in the network and, consequently, the amount of computation required (Li et al., 2019). The downsampling operation is illustrated in Figure 5.11. The figure shows that the pooling layer reduces spatial image size while preserving volume depth.

![downsampling](image)

**Figure 5.11.** An example of the downsampling operation (Li et al., 2019).

During a pooling operation, filters (often 2 x 2) are applied at every slice of the input throughout the depth of the input (D) using a predefined stride (typically $S = 2$). Each input slice is spatially resized either through max pooling (i.e. mapping a subregion of the input slice to its maximum value) or average pooling (i.e. mapping a subregion of the input slice to its average value). Max pooling, the most widely used downsampling operation, is illustrated in Figure 5.12 (Li et al., 2019). It is also evident from Figure 5.12 that different stride values result in different spatial sizes of the output volume.

![max pooling](image)

**Figure 5.12.** Examples of max pooling operations using different stride values (Aggarwal, 2018).
5.7.2. The Rectified Linear Unit (ReLU) Layer
The mapping of semantic information in an image from input pixel values is a highly non-linear operation (Wu, 2017). CNNs therefore need to be non-linear functions to undertake this task. The purpose of the ReLU layer is to introduce non-linearity to the CNN architecture. It is governed by the ReLU activation function defined as \( f(x) = \max(0,x) \) (Romanuke, 2017). In the ReLU layer, this function is applied to input activation maps to set all negative values to zero.

5.8. CNN Architectures and Transfer Learning
CNNs architectures are built by stacking convolutional (Conv), pooling (Pool), ReLU and fully connected (FC) layers together. There is no consensus in literature regarding the best layer sequence for computer vision applications although the following arrangement is commonly used (Li et al., 2019):

\[
INPUT \rightarrow [(CONV→RELU)*N→POOL?)*M→[FC→RELU]*K→FC
\]

where ‘POOL?’ is an optional pooling layer, and \( N, M \) and \( K \) are integers representing number of successive repetitions.

Large quantities of data (i.e. images) are typically necessary to sufficiently train all the layers in the network to achieve accurate models (Koul et al., 2019). However, the use of pretrained models and transfer learning is often recommended as opposed to designing entire CNN architectures from scratch (Géron, 2017; Rosebrock, 2017; Koul et al., 2019). This involves adapting pretrained models to new tasks. The use of pretrained models has the effect of significantly reducing training time from several hours/days to a few hours/minutes (Koul et al., 2019). The mechanism of transfer learning involves the reuse the earlier layers of a pretrained model and only retraining later layers for relevant task-specific tasks. This is typically done by removing the final fully connected layers of a pretrained model and replacing them with those tailored to the present task as shown in Figure 5.13 (Koul et al., 2019). Since training is only conducted on a few later layers, transfer learning reduces the amount of training data necessary to produce accurate models.
Fine tuning is an extension of transfer learning whereby more layers of a pretrained model (in addition to fully connected layers) are trained for the new task (Figure 5.14) (Géron, 2017; Rosebrock, 2017; Koul et al., 2019). Layers of the pretrained model that are not fine-tuned are called frozen layers.

By increasing the number of layers repurposed to the new task, fine-tuning ensures greater classification accuracies in resulting models (Koul et al., 2019).

**5.9. Overfitting and Underfitting**

CNN models are trained through SGD using labelled training data (Goodfellow et al.,
The performance of the model on the training data is measured in terms of training error, i.e. the fraction of training examples for which incorrect predictions are made by the model. However, the central objective in deep learning is to build models that perform well in the real world on new and previously unseen inputs. The ability to perform well on previously unseen inputs is referred to as a model’s generalisation (Goodfellow et al., 2016). A model’s generalisation is assessed using a dataset called a test set. The test set is fed into a trained CNN model which goes on to classify each test image. The output of this process is a performance metric called a generalisation (or test) error which is defined as the fraction of test examples for which incorrect predictions are made by the model. How well a model generalises is controlled by its capacity, i.e. its ability to fit a wide range of different types of functions.

Underfitting occurs when a trained CNN model attains an unacceptably high training and test error (Goodfellow et al., 2016). It is indicative of a low capacity model which is unable to fit the training set adequately. Conversely, overfitting occurs when the training error is low but the gap between it and the test error is significant. It indicates that the model became too familiar with the training data and adapted too closely to it such that it fails to generalise, i.e. its capacity is too high such that it was able to memorize properties of the training set that do not apply to the test set (e.g. noise) during training.

It is important to ensure that the capacity of a CNN model reflects the complexity of the task to be performed and the quantity of training data used (Goodfellow et al., 2016). The ideal capacity is achieved when a model is neither unsophisticated (i.e. underfits) nor too complex (i.e. overfits) (Figure 5.15).

![Figure 5.15](image.png)

**Figure 5.15.** Conceptual illustration of underfitting, overfitting and ideal fit of models (Koul et al., 2019).
The typical relationship between model capacity and error is shown in Figure 5.16. It further emphasizes that the capacity of CNN models must neither be too low nor too high in order to achieve an ideal fit.

**Figure 5.16.** The typical relationship between model capacity and error (Goodfellow et al., 2016).

Overfitting is often encountered when a CNN is trained using small quantities of data (Koul et al., 2019). The network easily memorizes features in all the training images which results in extremely high accuracies on the training set but performs poorly when applied to the test set. In such instances, the likelihood of overfitting can be reduced by (Koul et al., 2019):

i. Using transfer learning and only fine-tuning a few layers.

ii. Artificially increasing the size of the training set through data augmentation. This involves generating additional images that are duplicates of the original images subjected to a combination of rotations, random shifts, zooms, flips, warps, illumination adjustments, etc.

### 5.10. Hyperparameter Tuning

Prior to training, a number of settings called hyperparameters are manually pre-specified (Claesen & De Moor, 2015). Some of these were introduced in earlier sections; namely: filter size (F), number of filters (K), stride (S), zero-padding (P), learning rate (\(\eta\)), number of epochs and overall depth of CNN architecture. Hyperparameter selection has a significant impact on the performance of the model and is influenced by contextual computational cost constraints (Goodfellow et al., 2016). The appropriate selection of hyperparameters can either be done manually or through the use of automatic hyperparameter-search algorithms (Goodfellow et al., 2016). Goodfellow et al. (2016) recommends selecting hyperparameters based on their...
effects on model capacity. Examples of such effects are outlined in Table 5.1.

Goodfellow et al. (2016) argues that the learning rate is the most important hyperparameter. It determines the rate at which the network converges to the global minimum on the error surface through SGD. When the learning rate is too high, the network tends to diverge thereby increasing cross-entropy loss and training error as Figures 5.17 shows (Géron, 2017).

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Increases capacity when...</th>
<th>Reason</th>
<th>Caveats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of network</td>
<td>increased</td>
<td>Increasing the number of layers increases the representational capacity of the model, i.e. deeper networks produce models that fit the training data more closely than shallow networks.</td>
<td>Increasing number of layers increases computational costs.</td>
</tr>
<tr>
<td>Learning rate</td>
<td>tuned optimally</td>
<td>Inappropriate learning rates (too high or too low) results in optimisation failure.</td>
<td>-</td>
</tr>
<tr>
<td>Filter size</td>
<td>increased</td>
<td>Increasing filter size increases the number of parameters in the model.</td>
<td>A wider filter results in a narrower output dimension, reducing model capacity unless implicit zero padding is used to reduce this effect. Wider filters require more memory for parameter storage and increase runtime, but a narrower output reduces memory cost.</td>
</tr>
<tr>
<td>Zero-padding</td>
<td>increased</td>
<td>Adding implicit zeros before convolution keeps the representation size large.</td>
<td>Increases time and memory cost of most operations.</td>
</tr>
</tbody>
</table>

If the model does not diverge, it may randomly carom off the error surface without converging as Figure 5.18 shows. A learning rate that is too low is also undesirable as it results in a very slow training process (Figure 5.19) (Gugger & Howard, 2020).
In reality, error surfaces are non-convex functions often characterized by several saddle points and local minima as Figure 5.20 shows. When a very low learning rate is used during training, CNNs tend to converge into the nearest local minimum or saddle
point which could have an unacceptable training error (Goodfellow et al., 2016). It is therefore important to carefully tune the leaning rate so that convergence occurs nearer to the global minimum.

![Figure 5.20. The non-convex nature of error surfaces (Goodfellow et al., 2016).](image)

One method of preventing premature convergence is to use dynamic learning rates whereby an initially high value is used to bypass saddle points/local minima (Figure 5.21). The learning rate is then monotonically decreased during training as the network gets in the general vicinity of the global minimum (Smith, 2017). Smith (2017) recommends the use of Cyclic Learning Rates (CLR) that allow the learning rate to oscillate between prespecified minimum and maximum values. Weights are then saved when the network finally converges.

![Figure 5.21. The effect of cyclic learning rates on model convergence (Huang et al., 2017).](image)
Huang et al. (2017) proposed a variation of the CLR technique called Snapshot Ensembling whereby weights are saved each time convergence occurs. After each convergence, the learning rate is abruptly increased and quickly lowered to escape the current local minimum or saddle point. Convergence via the Snapshot Ensemble method is shown in Figure 5.22.

When transfer learning and fine-tuning are used, Gugger & Howard (2020) recommend the use of discriminative learning rates. This entails applying a low learning rate for the earliest layer of the network, and a high value for the final layer. The layers in-between are trained using learning rates that are multiplicatively equidistant between the low and high values. This approach is based on the fact that the earlier layers of the network belong to a pretrained model and are therefore already capable of recognizing basic features in images. The final layers, however, initially contain random parameters that require significant amounts of training so that they can be tailored to identify more complex task-specific features (Gugger & Howard, 2020).

![Figure 5.22. Snapshot Ensembling (Huang et al., 2017).](image)

5.11. Performance Metrics
During training, the training set is used to find weights that minimise the loss function. These weights are then applied to a separate dataset called a validation set which
consists of previously unseen images intended to evaluate the performance of the model. The proportion of validation images for which the model makes correct predictions is referred to as the validation accuracy. Hyperparameters are intentionally selected in such a way that high validation accuracies are attained, i.e. models are trained to learn weights that are intended to perform very well on the validation set. For this reason, the validation accuracy alone does not adequately reflect how the model will perform in real-world applications. Rather, once a model has achieved an acceptable validation accuracy during training, it is applied to a test set that would have been kept completely separate from the training process. This produces a performance metric called the test accuracy which becomes the basis of final model performance evaluation (Gugger & Howard, 2020)

The final trained model consists of a CNN architecture and weights (Gugger & Howard, 2020). When in use, it behaves similarly to regular computer programs as Figure 5.23 shows.

![Figure 5.23. Application of trained models (Gugger & Howard, 2020).](image)

### 5.12. Object Detection

Once objects in an image are classified using trained CNN models, it is sometimes necessary to mark the location of each object. One method of achieving this is through a process called object detection. It is an extension of image classification that is concerned with obtaining the following from an input image (Rosebrock, 2017):

i. \((x, y)\)-coordinates or bounding boxes for each object in an image.

ii. A class label for each bounding box.

iii. A confidence score for each bounding box and class label.

Object detection is usually implemented using state-of-the-art region-based CNNs (i.e. R-CNNs, Fast R-CNNs and Faster R-CNNs), You Only Look Once (YOLO) algorithms and Single Shot MultiBox Detectors (SSDs) (Zhao et al., 2019). A detailed description of these frameworks is beyond the scope of this study and reference can be made to Girshick et al. (2014), Girshick (2015), Ren et al. (2015), Liu et al. (2016), Redmon &
Farhadi (2017) and Redmon & Farhadi (2018) for a more extensive discussion.

5.13. **Summary**

This chapter provided an overview of deep learning CNNs and various object detection techniques. The following key observations were made:

i. CNNs are state-of-the-art for image classification applications and outperform alternative feature-based methods.

ii. CNNs learn by being trained on several labelled images. During training, they extract patterns from the training data and produce models to fit the data. The models are then tested on new data to assess their performance in real-world conditions.

iii. Training CNN involves setting hyperparameters and running a series of forward passes and back propagation throughout the network, using SGD to minimise the cross-entropy loss function. The training process is a global optimisation problem that aims to find weights or parameters for which the lowest cross-entropy loss is attained. The learning rate, in particular, must be carefully selected to ensure reasonable training speeds and avoid model divergence.

iv. The capacity of a CNN model must reflect the complexity of the task to be performed, as well as the quantity of training data available to avoid underfitting or overfitting.

v. Object detection is an extension of image classification that is used whenever there is need to mark the location of objects in an image, in addition to classifying them. It can be done using frameworks such as Faster R-CNNs, SSD and YOLO.
Chapter 6: Methodology

6.1. Introduction
The highly subjective nature of visual bridge inspections can significantly reduce the accuracy of inspection results and BCI calculations. Augmenting human judgement with an automated system during inspections can go a long way towards reducing the effect of inspector subjectivity, human error and bias on the accuracy of bridge repair and rehabilitation prioritisation. CNN algorithms have shown the most potential with regards to automating the detection and classification of defects during RC bridge inspections. However, much of the research in this area focuses on detecting the presence of cracks. None of the studies attempted to use CNNs to identify RC crack-type once cracks were detected by their models. Research that took other defect types into account largely focused on broad binary classification problems, i.e. classifying images in 2 broad classes (e.g. ‘defective’/’not defective’, ‘corroding’/’not corroding’, etc.). In addition, no research has been done on rating the severity and extent of defects with the aid of CNNs at the time of writing.

This chapter outlines a methodology that was used to build proof-of-concept CNN models that have received little to no attention in literature, namely:

i. A model that identifies the type of defect depicted in an image among multiple defect-type classes (defect-type classifier).
ii. A model that identifies the type of crack (if any) is depicted in an image (crack-type classifier).
iii. A model capable of detecting exposed reinforcement (exposed rebar detector).
iv. A model capable of assigning DERU Degree ratings to shrinkage cracks (D-rating classifier).

It was envisaged that these models would interact with each other during inspections in one end-to-end process to detect, classify and rate defects according to Figure 6.1.
Proof-of-concept model selection was further motivated by the following:

- The DERU rating system was chosen as a case study since it conveniently separates ratings that are entirely visual (Degree and Extent) from those that rely on engineering judgement (Relevancy and Urgency). Automation of Degree (D) and Extent (E) ratings is therefore a computer vision task suitable for CNN implementation.
- In order to train E-rating models, images that simultaneously capture entire inspection items and all their defects were required. However, no such images were available in the public datasets that were used during this study for model training. The manual on-site collection of these images was not possible within the research timeframe. E-rating models were therefore not built during the study. Nevertheless, the methodology described in this section is also applicable for the development of such models provided the necessary training data is made available.
- Shrinkage cracks were selected as a case study for automated D ratings. D-
rating models for other defect types were not built due to time constraints.

6.2. Resources Used for Development
The development of CNN models typically requires the following tools (Koul et al., 2019):

i. A python-based deep learning library such as TensorFlow or PyTorch.

ii. An Integrated Development Environment (IDE) such as PyCharm, or notebook-oriented development systems such as Jupyter Notebook for writing code.

iii. Graphical Processing Units (GPUs) to drastically reduce training time.

In this study, a relatively new PyTorch-based deep learning library called fastai was used for development. This library was selected over the more widely used TensorFlow due to its capacity to build models relatively quickly using significantly fewer lines of code (Howard, 2018). Jupyter Notebook was preferred over traditional IDEs as it allows more rapid and interactive training and experimentation with hyperparameters. The Google Cloud Platform (GCP) was used to set-up a cloud GPU to accelerate training. Suitable hardware GPUs were not used in this study owing to their high costs relative to cloud computing options.

6.3. The Model Development Steps
Each model was built following the steps presented in Figure 6.2:

![Figure 6.2. Methodology flowchart.](image-url)
6.3.1. Dataset Compilation
The first step towards developing the models was to compile images for model training and evaluation. The data was collected from the following publicly available sources:

i. The Concrete Defect Bridge Image dataset (CODEBRIM) (Mundt et al., 2019);
ii. The SDNET2018 dataset (Maguire et al., 2018);
iii. The CCNY Concrete Structure Spalling and Crack database (CSSC) (Yang et al., 2017); and,
iv. A dataset of concrete bridge defects compiled by Hüthwohl et al. (2019).

Table 6.1 shows the quantity of images that were collected for each of the four models. Each dataset had the same number of images per class (i.e. was balanced) to avoid inducing bias in the resulting models (Rosebrock, 2017). The exact number of images compiled per model was selected on a trial-and-error basis, i.e. an arbitrary number of images was used to train each model, only to be experimentally increased by arbitrary quantities in the event of poor model performance due to overfitting.

Table 6.1. Attributes of compiled datasets.

<table>
<thead>
<tr>
<th>Model</th>
<th>Image contents</th>
<th>Classes</th>
<th>No. of images per class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect-type classifier</td>
<td>Concrete defects</td>
<td>Cracking; Spalling &amp; delamination; Efflorescence; Background (no defect)</td>
<td>320</td>
</tr>
<tr>
<td>Crack-type classifier</td>
<td>Concrete cracks</td>
<td>ASR; Corrosion; Shrinkage</td>
<td>20</td>
</tr>
<tr>
<td>Exposed rebar detector</td>
<td>Rust-stained concrete</td>
<td>Exposed rebar; Unexposed rebar</td>
<td>200</td>
</tr>
<tr>
<td>D-rating classifier</td>
<td>Shrinkage cracks</td>
<td>Degree 1; Degree 2; Degree 3; Degree 4</td>
<td>124</td>
</tr>
</tbody>
</table>
Each image was manually labelled/annotated according defect or rating class. This was done by placing each image file in a folder whose name corresponds to the relevant class category. For example, image files of concrete cracks were accordingly placed into folders labelled as ‘shrinkage’, ‘ASR’ or ‘corrosion’ to build the training dataset for the crack-type classifier.

Shrinkage crack images from the SDNET2018 dataset were used to build the D-rating dataset. All the images in this dataset were conveniently captured by the same camera at a fixed working distance from the defect (500mm). This allowed direct comparison between images and visual discernment of the relative severity of the pictured cracks so that D-rating labels could be appropriated to each image (Figure 6.3).

![Figure 6.3. Sample labelling of SDNET2018 images based on relative crack widths.](image)

### 6.3.2. Image Pre-processing
The following pre-processing steps were performed on each labelled dataset:

i. 10% of the collected data was set aside to constitute a test set. A training and validation set was created from the remaining data at an 80-20% split based on common practice (Rosebrock, 2017). The images constituting these sets were selected automatically and randomly to minimise human bias in the creation of these sets.

ii. All the images were reduced to a 224 x 224 resolution to ensure that the resulting models would fit into GPU memory (by shrinking or cropping each image). This specific resolution was selected as it corresponds to that of images used to train the pretrained model architecture elected in the next step (Section 6.3.3). Despite the small resolution, defects depicted in the images were still clearly visible.

iii. The training set was randomly split into mini-batches to ensure that training occurs using batches of data that the GPU can manage (Géron, 2017). This was
done by specifying a hyperparameter called batch size (bs) which represents the number of training images in each mini-batch. Through trial-and-error, a batch size of 64 was found sufficient for the GPU that was used in this study.

iv. Since the training datasets were small, data augmentation was used to artificially increase their sizes and reduce the probability of overfitting. The random transformations that were used include rotations, flips, warps and illumination adjustments. However, care was taken to ensure that the transformations applied to the shrinkage crack dataset did not distort crack widths so as not to affect D-ratings.

An example of fastai code used to implement the above steps is shown in Figure 6.4.

```python
# Step 1: Specify the location of the image dataset for training.
path = '/home/jupyter/D_and_E_Ratings/Degree_imgs/Cracks/

# Step 2: Specify desired data augmentation transforms (tfms). In this example, vertical
# flips are applied on the shrinkage crack dataset (please note that vertical flips are
turned off by default and need to be manually specified as shown below).
tfms = get_transforms(flip_vert = True)

# Step 3: Reserve 20% of the images for use as the validation set (the remaining 80% would
# make up the training set). Resize all the images into squares. A standard image size of 224
# x 224 pixels was used in this example.
.data = (ImageList.from_folder(path)
  .split_by_rand_pct(0.2)
  .label_from_folder()
  .transform(tfms, size = 224, resize_method = ResizeMethod.SQUISH)
  .databunch(path = path))

# Step 4: Randomise the selection of validation set images.
np.random.seed(42)
```

Figure 6.4. Example of dataset pre-processing using the fastai python library.

Once pre-processed, the data was ready for model training.

6.3.3. Selection of a Pretrained Model
Transfer learning was necessitated by the small sizes of the compiled datasets. State-
of-the-art pretrained architectures called residual neural networks (ResNets) were
selected for model training (Figure 6.5). This selection was largely informed by
ResNets’ domination of the Stanford DAWNBench image classification benchmarks
(Mattson et al., 2019).
6.3.4. Selection of Suitable Learning Rates

Cyclic learning rates were used to ensure that each model converges as close to the global minimum as possible, at reasonable speeds (Smith, 2017). To identify suitable values, the fastai library was used to conduct mock training on each dataset while gradually increasing the learning rate and plotting the corresponding cross-entropy loss. A Loss versus Learning Rate graph such as that shown in Figure 6.6 was produced. All the Loss versus Learning Rate graphs attained for each model have been provided in Appendix B.

Figure 6.6 shows that as the learning rate increased, the graphs eventually reached a point beyond which the cross-entropy loss abruptly started to increase. This was because the learning rates at and beyond that point were too high and, therefore, caused model divergence (see Section 5.10). On this basis, learning rates well before the point of divergence were selected for model training. In the case presented in
Figure 6.6, a learning rate of $10^{-4}$ was selected.

Discriminative learning rates were used to maximize the accuracies of the resulting models (Gugger & Howard, 2020). They were applied as follows:

i. The final layer of the network was trained at the learning rate identified from the Loss vs Learning Rate graph. For illustrative purposes, this value would be $10^{-4}$ for Figure 6.6.

ii. The first layer of the network was trained at a learning rate 10 times smaller than the value identified from the Loss vs Learning Rate graph as recommended by Gugger (2018) (e.g. $10^{-5}$ for Figure 6.6).

iii. The layers in-between were trained using learning rates that were multiplicatively equidistant throughout this range of values (i.e. between $10^{-5}$ and $10^{-4}$ for Figure 6.6) (Gugger & Howard, 2020).

It may be noted that the computation of Loss vs Learning Rate graphs is only intended to aid in the selection of suitable learning rates that will later be used for model training. This is therefore an intermediary step and the graphs produced do not constitute the results of the investigation.

6.3.5. Network Training

Using the learning rates identified in the previous step, each model was trained iteratively over several epochs. During the training process, the cross-entropy loss on the training set (i.e. training loss) and validation set (i.e. validation loss), as well as model accuracy on the validation set (i.e. validation accuracy) were computed and displayed after each epoch to monitor training progress (see Figure 6.7). The weights at each epoch were saved for later reference using the callback function available in the fastai library. The validation accuracy was the metric used to judge the performance of each model during training and guide the tuning of hyperparameters.

For the purpose of illustration, the fastai code that was used to train the Defect-type Classifier is shown in Figure 6.7. The code used to train the Crack-type Classifier, D-rating Classifier and Exposed Rebar Detector was similar to that shown in Figure 6.7. The only difference in the code used to train each classifier lay in the number of training epochs and learning rates that were required to develop accurate models.
Following each epoch, weights were updated through SGD based on the training set. The updated weights were subsequently applied to the validation set to compute the validation accuracy. This process was repeated until all the pre-specified number of epochs were complete. The number of epochs used were determined through trial-and-error until that which results in high accuracies was found. Fine-tuning was also used to ensure that the accuracies attained by the resulting models were as high as possible.

A general decrease in train_loss and valid_loss, as well as an increase in validation accuracy was generally observed during training (Figures 6.8 and 6.9). Validation accuracy graphs for each model have been provided in Appendix C.

A large number of epochs (>20) was deliberately selected to ensure that each model was sufficiently trained. The possibility of overfitting due to a high number of epochs was of no consequence because the callback function allowed suitable models not
prone to overfitting to be selected for evaluation once training was complete. The depths of the CNNs used to train the models were selected by trial-and-error until those that resulted in accurate models were found.

A 34-layered ResNet (ResNet34) architecture was used to train the defect-type, crack-type and D-rating classifiers. Accordingly, an 18-layered ResNet (ResNet18) architecture was used to train the rebar detection model. Trained models with the highest validation accuracies were ultimately selected for final evaluation.

![Image](image.png)

**Figure 6.8.** Typical downward trend of train_loss and valid_loss during training.

![Image](image.png)

**Figure 6.9.** Example showing improvement of model accuracy during training.

### 6.3.6. Model Evaluation
The evaluation of selected models was done primarily by assessing the characteristics of validation images that were most poorly classified by each model. A function in the
fastai library called ‘plot_top_losses’ was used to display these images (Figures 7.1-7.4). Each image was displayed alongside the model’s prediction, the actual image class, the cross-entropy loss and the model’s confidence in the prediction. This information was assessed in order to understand the most probable reasons for each model’s poor predictions.

In the event that a model failed to make obvious predictions, the labels of misclassified images and the data augmentation transforms that were applied were investigated. For example, certain image transforms (e.g. warps) distorted crack images such that the labels of the image files no longer corresponded to the crack features. In such instances, augmentations were altered accordingly before retraining and re-evaluating the model in question. This process was repeated until the number of inaccurate predictions was minimised. Whenever poor predictions were due to environmental noise, however, noisy images were not removed and no alterations were made to the datasets. This was because doing so would have reduced the training data’s ability to represent real-world inspection conditions, thereby inducing representation bias to the resulting models (Suresh & Guttag, 2019).

The final models were ultimately applied to the relevant test sets. Test images were passed into each model, one at a time, and the test accuracy was computed as the proportion of correct predictions.

Chapter 7: Model Investigations and Results

7.1. Introduction
A defect-type, crack-type, exposed rebar detection and shrinkage crack D-rating image classification model was produced in this study. This chapter discusses the performance of these models and draws insights with respect to their generalisation capabilities for practical RC bridge inspections.

7.2. Model Performance Results
The final performance of each model was assessed based on:

i. The model’s accuracy on the test set;
ii. The model’s robustness to noise.
The performance metrics for each model are summarised in Table 7.1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Validation accuracy (%)</th>
<th>Test accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect-type classifier</td>
<td>97.3</td>
<td>92.1</td>
</tr>
<tr>
<td>Crack-type classifier</td>
<td>96.4</td>
<td>92.6</td>
</tr>
<tr>
<td>Exposed rebar detector</td>
<td>96.4</td>
<td>90.0</td>
</tr>
<tr>
<td>D-rating classifier</td>
<td>93.9</td>
<td>87.0</td>
</tr>
</tbody>
</table>

7.2.1. Evaluating Model Accuracies
Table 7.1 shows that high test accuracies (≥ 87%) were attained by the final models. This indicates that they neither underfit nor overfit the training data, i.e. optimal generalisation capacity was achieved. However, it is likely that increasing the sizes of training datasets, applying more advanced training techniques and using more complex CNN architectures would have the effect of improving model accuracies. As this was a proof-of-concept study, advanced training methods to further improve model performance were not explored.

7.2.2. Investigating Model Robustness to Noise
Validation images most poorly classified by the models are shown in Figures 7.1-7.4:

![Figure 7.1](image1.png)

**Figure 7.1.** Images most poorly predicted by the defect-type classifier.

![Figure 7.2](image2.png)

**Figure 7.2.** Images most poorly predicted by the rebar detection classifier.
Based on the poorly predicted images, the following observations were made:

i. Images (a) and (b) in Figure 7.1 show that the defect-type classifier is prone to vegetative noise, while Figure 7.1 (c) suggests that the model is incapable of correctly identifying the presence of multiple defect-types in a single image (spalling and efflorescence in this instance).

ii. Figure 7.2 (a) suggests that the rebar detection model mistook the rust stains for exposed reinforcement and classified the image accordingly. Likewise, the model mistook the red markings in Figure 7.2 (b) for exposed rebar. Figure 7.2 (c) was misclassified on account of a black feature visible at the bottom-left corner of the image.

iii. The crack-type classifier failed to make a fairly straightforward prediction (Figure 7.3). This can be attributed to the small training dataset size that did not sufficiently represent all class categories.

iv. Figure 7.4 shows that the D-rating model confused classes that were close to each other by a 1 degree magnitude, i.e. a Degree 1 crack was wrongly assigned to a Degree 2 class, while a Degree 4 crack was wrongly predicted as
a Degree 3 defect. This suggests that the D-rating model is unlikely to confuse drastically dissimilar D-ratings in the event that incorrect predictions are made. This is desirable as it minimises the effect of misclassifications on subsequent Condition Index calculations.

The assessment of the most poorly classified images led to the following conclusions:

i. The models built were susceptible to noise. This was attributed to the training datasets being relatively small and largely devoid of noise such that training images under-represented real-world inspection conditions. In practice, training data must therefore include a number images with non-defect features such as vegetation, joints, graffiti, shadows and occlusions, among other types of noise to improve model robustness. The images must also reflect the different illumination conditions under which inspections are conducted, taking different weather conditions and the use of flashlights into account (Rosebrock, 2017).

ii. The models built were incapable of correctly detecting the presence of multiple defect types in a single image. This was because the models were trained for single-label classification tasks, i.e. they were tailored to identify one category of defects per image. For practical applications, multi-label image classification models are therefore necessary (Gugger & Howard, 2020). Such models were not developed in this proof-of-concept study due to time limitations.

It follows that human-in-the-Loop (HITL) computer vision systems (i.e. requiring human interaction) would be necessary so that inspectors are able to override incorrect model predictions in real-time (DoD, 1998).

7.2.3. Confusion Matrices

To better visualize the performance of the models, confusion matrices for each model’s predictions on the validation sets were plotted (Figure 7.5). These matrices summarise the prediction results of each model, depicting the number of correct and incorrect predictions for each class.

Figure 7.5 shows that there was little confusion between classes by each of the four models that were developed. This suggests that the visual features of defects that differentiate each class from the other classes are learnable, i.e. CNNs are able to
extract patterns in the training images that sufficiently distinguishes one class from another.

The results demonstrate that CNNs are capable of accurately undertaking the constrained task of identifying RC corrosion, spalling, ASR, cracking and efflorescence defects, and assigning appropriate D ratings to shrinkage cracks. With more training data, it can be assumed that CNN models can be built to identify and allocate D ratings to any other visible defect type. Likewise, E-rating models can be built provided the necessary training data can be acquired. An efficient method of collecting data for training E-rating models is proposed in Section 8.2. In practice, multi-label image classification or object detection models would be necessary in order to detect the presence of multiple defect types in a single image.

![Confusion matrices of final models](image)

**Figure 7.5.** Confusion matrices of final models.
7.3. Summary
Proof-of-concept models with accuracies ≥ 87% were developed for the automatic detection of RC corrosion, spalling, ASR, cracking and efflorescence defects, and D-rating assignment to shrinkage cracks. However, the models were found to be susceptible to representation bias as the training datasets under-represented real-world inspection conditions. It was determined that more representative training data would make the models more robust to environmental noise for practical applications. It was also found that there would be need for the development of multi-label image classification models so that the presence of multiple defect types in each image can be detected in practice. Nevertheless, the ability of the four models to accurately identify RC corrosion, spalling, ASR, cracking and efflorescence, and allocate D-ratings to shrinkage cracks, means that similar models for other visible RC defect types can be developed with more training data. E-rating models for various defect types can also be developed, provided the requisite training data is acquired (Section 8.2). Given that CNN models sometimes make poor predictions, a human-in-the-loop system that allows inspectors to override incorrect model predictions in real-time would be necessary in practice as discussed in the deep learning-based RC bridge inspection framework proposed in Chapter 8.
Chapter 8: Proposed Deep Learning-Based RC Bridge Inspection Framework

8.1. Introduction
The results of this study showed that CNN algorithms are capable of generating models that can recognise image contents and assign appropriate class labels with high degrees of accuracy. More specifically, it was shown that accurate image classification models can be built to recognise different types of concrete defects and assign D-ratings. With more data, it was assumed that similarly accurate E-rating models can be built as the Universal Approximation Theorem suggests. This chapter proposes a practical DERU-based framework that aims to automate RC bridge inspections with the aid of UAVs and image classification models.

8.2. Preliminary Requirements
The successful implementation of UAV-enabled and deep learning-based DERU RC bridge inspections depends on the availability of inspection-grade UAVs and sufficiently trained human-in-the-loop image classification models with minimal representation bias. The specific requirements are as follows:

i. UAVs with the capabilities outlined in Section 4.3.1 are required and licenses necessary for UAV-aided inspections must be obtained to meet aviation authority regulations.

ii. Training images must be accurately labelled by a team of accredited bridge inspectors to ensure credibility of the resulting models. Such a team-oriented approach prevents individualistic inspector bias in the labelling process itself. It is proposed that the labelling be done in a workshop format, with oversight from organisations such as SANRAL and the Council for Scientific and Industrial Research (CSIR). These organisations have each produced visual inspection reference guides (i.e. TMH19 Part B and Part C of CSIR’s Visual Assessment Manual) documenting several examples of defects and corresponding DERU ratings. Inspectors currently use these examples as benchmarks when allocating DERU ratings during inspections (COTO, 2016). The idea, therefore, is to involve these organisations in the data labelling process so that their required standards of DERU rating allocation become a built-in feature of the computer vision system itself.
iii. Training images must accurately capture the variability of onsite inspection conditions. To ensure that defects are clearly visible in each training image and avoid perspective distortion, they must be captured using a high-resolution camera in perpendicular alignment to the concrete surface of interest. In confined spaces, this can be achieved using collision-tolerant rotocraft UAVs such as that shown in Figure 4.3. In the event that inspection images are captured at an angle due to restrictive site conditions, such images can still be stored in the BMS’ inspection module. They will, however, be unsuitable for Degree and Extent model training.

iv. The training images must be on a fixed scale so that the relative severity of defects can accurately be discerned when inspection photographs are visually compared to each other. This can be achieved by ensuring that photographs are taken using a fixed camera resolution at a fixed distance $x$ from the concrete surface. The choice of $x$ and camera resolution are both inconsequential provided the defects of interest are clearly visible in each image.

v. More capable accelerators would be required if high resolution images are to be used directly for training (e.g. Field-Programmable Gate Arrays (FPGAs) or Application-Specific Integrated Circuits (ASIC)) (Kou et al., 2019). This is because training on high resolution images produces large models that do not fit in the memory of cheaper GPUs like that used in this study (Le et al., 2019).

vi. To reduce computational requirements, it is proposed that high resolution images captured onsite be preprocessed to produce low resolution ‘defect map’ images that can then be used for E-rating model training. This can be done by taking large images of an inspection item (e.g. deck) and running them through object recognition algorithms trained for defect detection (e.g. Faster R-CNNs, YOLO or SSD). These algorithms would automatically produce bounding boxes around defects of interest. Low resolution images that only show bounding box locations can then be used to cost-effectively train E-rating models without compromising classification accuracy.

vii. The performance of trained models must be benchmarked against that of human inspectors. In the event that their performance is unsatisfactory, they can be retrained using larger datasets and/or more complex CNN architectures. Models must ideally perform at the same level as (or better than) experienced human inspectors to reliably automate the allocation of Degree and Extent ratings.
8.3. Proposed Inspection Methodology

No formal practical methodology currently exists for the implementation of CNN algorithms and UAVs to automate DERU-based RC bridge inspections. This section proposes such a methodology, with a particular focus on the real-time automation of D and E defect ratings, on-site. An overview of the proposed methodology is shown in Figure 8.1.

![Figure 8.1. Overview of proposed RC visual bridge inspection methodology.](image)

The inspector first selects an inspection item and uses an inspection-grade UAV to methodically photograph the entire element’s surface from a fixed perpendicular working distance $x$. Distance $x$ must be approximately equivalent to the working distance that would have been used to train the CNN models mounted on the UAV. This ensures that the images used in training and those collected during the inspection have a fixed scale. Resolution and general camera settings used must also be comparable to those used to train the models to ensure high classification accuracies and standardize the quality of images stored in the inspection module of the BMS.

To maintain a fixed working distance $x$ during flight, the distance-lock feature available in some inspection UAVs (e.g. senseFly Albris) can be utilised. Flight paths and imaging positions can be pre-programmed such that the entire surface is photographed systematically (e.g. in a grid-like pattern). Images are then stitched together to form one large high-resolution image of the inspection item. This image is subsequently subjected to the following processes:
i. The stitched image is passed to an object-detection algorithm (Faster R-CNN, YOLO or SSD) that identifies and produces bounding boxes around each defect.

ii. The content of each bounding box is cropped out and passed to the D-rating model that corresponds to the defect type identified by the object detection algorithm. The D-rating models go on to allocate D ratings to all identified defects.

iii. The system identifies the defect types with the highest D-rating (i.e. the most severe defects) and prompts the inspector to select the defect-type(s) to be assigned an E rating in the next step of the process. The inspector exercises engineering judgement to make this selection (e.g. based on perceived Relevancy ratings).

iv. A defect map with bounding boxes marking locations of instances of the most severe defect type (identified in the previous step) is produced and passed to an E-rating model.

v. The E-rating model allocates an E rating to the most severe defect based on the distribution of bounding boxes on the inspection item.

vi. The D and E ratings of the most severe defect are automatically (or manually) entered in the relevant fields of a standard inspection form. The inspector reviews model outputs and, in case of perceived inaccuracies in the predictions, manually edits them with values he/she deems to be more accurate. To avoid inducing inspector bias to the inspection results, however, manual edits by the inspector are saved for later review and moderation by CSIR or SANRAL-appointed persons.

vii. Relevancy (R) and Urgency (U) ratings are manually allocated to the defect by the inspector based on his/her engineering judgement.

viii. Each image file (and the defect-type and ratings assigned to it) is stored in the inspection module of the BMS. The number of images in this module would grow over time as more bridge inspections are conducted. These images and their labels would eventually be used to retrain the models in future in order to further improve their accuracies.

Once the inspection and remedial work sheets are completed, the inspector proceeds to select the next element and repeats the process until inspection is complete. For
partial inspections where only certain inspection items or structurally vulnerable locations (e.g. pier supports and midspans) are assessed, the same steps may be conducted on the specific elements or areas of interest. A sample inspection sheet for DERU-based inspections has been provided in Appendix A.

### 8.4. Proposed Equipment Setup

All image classification and object detection models can be deployed onboard UAVs using embedded devices (e.g. Raspberry Pi, NVIDIA Jetson Nano or PYNQ-Z2) and USB accelerators (Koul et al., 2019). Edge computing, whereby model outputs are transmitted to a handheld device/laptop in real-time, is proposed (Chen & Ran, 2019). Real-time transmission can be facilitated by a Wi-Fi, Bluetooth Low Energy (BLE), or Long Range Wide Area Network (LoRaWAN) connection (Kang & Eom, 2019). Recent studies have found LoRaWAN to be a particularly cost-effective option for data transmission (machine-to-machine communication) in rural or remote areas with little to no wireless coverage (Myagmardulam et al., 2021; Behjati et al., 2021). Real-time transmission allows the inspector to review model outputs and swiftly make any necessary corrections while still on-site. Inspection photographs and ratings obtained can ultimately be entered into the BMS inspection module manually or through the cloud. The proposed setup is shown in Figure 8.2.

![Figure 8.2. Proposed hardware setup.](image)

### 8.5. Continuous Improvement of Models

The proposed methodology automates the data collection and labelling process for future model training. Continuous improvement of existing models is achieved by updating training datasets through the addition of new labelled data stored in the inspection module of the BMS (Figure 8.3). To achieve noticeable improvements in
classification accuracies and minimise training costs, models must be retrained once significant quantities of new training images (e.g. 500 per class) are collected (Géron, 2017).

![Figure 8.3. Continuous improvement of image classification models (Géron, 2017).](image)

**8.6. Limitations of Proposed Framework**

The proposed methodology makes use of CNN models that specifically detect and classify visible concrete defects according to defect-type and D and E rating. In addition to not being able to detect hidden defects, these models are also unable to identify other deficiencies such as blocked waterways or drains and the presence of unwanted vegetation or debris. In these instances and in cases where contextual site conditions do not allow the practical implementation of this framework, conventional inspection methods must be used.

The proposed UAV-computer vision bridge inspection system is a Level 3 system on Sheridan’s scale of human-machine interaction (Table 4.1). This means that the system takes image inputs and suggests defect-type and rating outputs to the inspector. The inspector remains responsible for the inspection results and it is incumbent on him/her to review the suggested model outputs and manually override them, if necessary, by exercising engineering judgement. To minimise introducing inspector bias/human error to the results, the framework proposes that such interventions by the inspector be subject to later review and moderation overseen by agencies such as CSIR or SANRAL (see Section 8.3 (vi)).

It can be expected that the computer vision system will frequently make inaccurate predictions in the early years of its implementation and progressively improve with further model training on larger datasets. As a consequence, trust in the system is also
expected to increase over time and the early years of practical implementation can be considered part of the system’s development. However, as trust in the system improves, it is important that inspectors do not become over-reliant on it to the extent of either refraining from exercising engineering judgement (i.e. automation complacency) or favoring the system’s suggestions while ignoring correct contradictory engineering judgement (i.e. automation bias) (Cummings, 2004; Goddard et al., 2012). Automation bias and automation complacency would be considered forms of misuse of the automation system (Parasuraman & Manzey, 2010).

The development of the needed hardware and software infrastructure is likely to be a costly enterprise requiring the specialised expertise of computer vision practitioners, software developers, cloud specialists, etc. The collection and labelling of data to train the first model iterations would be a time consuming process. Model training can also be costly and time consuming depending on the GPU used. Additional costs may include acquiring inspection-grade UAVs, maintaining the UAVs and other hardware infrastructure and ensuring that inspectors possess UAV piloting licenses or are accompanied by licensed persons during inspections. Inspectors will also need to be trained in the use of the computer vision system to supplement traditional inspector training requirements. Therefore, the decision to implement this framework should be preceded by weighing these costs against those associated with the status quo (i.e. the continued use of specialised access equipment and effects of inspector subjectivity) in the long run. Please note that a cost analysis was not conducted in this study as the primary focus was to investigate the technical capabilities of UAVs and deep learning algorithms for visual RC bridge inspections. In the event that the project is ultimately commissioned, there would be need to solicit long-term commitment from transport authorities and all relevant stakeholders for successful implementation.

As the proposed framework could not be field-tested within the research timeframe, there remains room for further revisions. With technology rapidly evolving, there is also the possibility that the techniques proposed in this study will become outdated in the short to medium-term. Therefore, it is important that the framework be continuously refined using latest techniques and hardware.
Chapter 9: Conclusions and Recommendations

The potential synergistic exploitation of UAV technology and deep learning algorithms in defect-based RC bridge inspections was explored in this study to test the following hypotheses:

i. The use of state-of-the-art inspection-grade UAVs can significantly reduce inspection costs and enhance inspector safety.

ii. State-of-the-art deep learning models are capable of automating the defect detection and rating allocation process during visual inspections, thereby reducing the effect of inspector subjectivity on inspection results.

A survey of the UAV landscape was conducted to establish UAV capabilities necessary for UAV-enabled inspections and the legal requirements thereof. Proof-of-concept deep learning models capable of detecting RC bridge defects and allocating D-ratings from image inputs with accuracies ≥ 87% were then built. Based on the findings and the literature review, the following conclusions were drawn:

i. State-of-the-art inspection UAVs possess a number of features that enable safe navigation and data collection during visual bridge inspections. By eliminating the need for specialised access equipment, the use of UAVs can also reduce project-level visual inspection costs by approximately 40%. Strict rules and regulations set out by aviation authorities and delays in the issuing of UAV licenses, however, have been identified as potential barriers to implementation.

ii. CNN algorithms, as the proof-of-concept models show, are capable of automating the defect detection and rating process during visual RC bridge inspections. The development of these models is, however, highly dependent on the availability of requisite training data that is representative of real-world inspection site conditions. Since these models automate the allocation of Severity and Extent ratings, the effect of inspector subjectivity during visual inspections can be minimised.

Based on the literature review and findings, a practical framework for UAV-enabled and deep learning-based visual RC bridge inspections was ultimately proposed. The main principle behind the proposed framework was to introduce automation to the inspection process while allowing the inspector to still exercise engineering judgement and remain
responsible for the inspection results.

Since implementation of the proposed framework would likely be capital and labor-intensive, the long-term commitment of transport agencies and other stakeholders would be necessary. Oversight from organisations such as SANRAL and CSIR during the automated system’s development and implementation would also be necessary to ensure the system’s credibility. Furthermore, inspectors would need to be trained on how to operate the system and it is important that they are well informed about the system’s underlying principles and limitations.

It must be emphasized that the system’s predictions are to be regarded as being suggestive and not authoritative. It would be the responsibility of the inspector to review the system’s predictions and exercise engineering judgement to make alterations where necessary. Significant alterations can be later reviewed and moderated by a team of inspectors appointed by the transport authority for that purpose to ensure that such edits would not have been a result of inspector bias or error. The inspector is ultimately responsible for the inspection results and failure to exercise engineering judgement due to ‘automation bias’ or ‘automation complacency’ would be considered a form of misuse of the system.

Although the proposed framework was primarily developed for visual RC bridge inspections, it could be adapted for other related applications. Examples of these include:

i. Deploying CNN models for quality control during conventional inspections by using model predictions as benchmarks for inspector-generated ratings.

ii. Using CNN models and UAV image geo-tagging features to monitor the evolution of Degree and Extent ratings at specific locations on a bridge structure.

iii. Adapting the framework for the inspection of culverts and prestressed bridges.

It is necessary to field-test this framework in future studies. Since this was not done in this study, there are opportunities for further improvements and refinement. Further research arising out of this study may also be considered for the following topics:

i. The use of unsupervised machine learning algorithms to label training data and possibly develop new Degree and Extent rating scales.
ii. The use of real-time width-measuring algorithms during inspections.

iii. The use of image classification and other machine learning techniques to aid in the interpretation of NDT results for project-level inspections (e.g. ground-penetrating radar images).

iv. The application of deep learning algorithms to enable the automatic classification of bridge-type and subsequent detection of vulnerable zone coordinates that can be used as waypoints to guide UAVs to locations of interest with minimal human input.

v. The potential emergence of fully autonomous (‘self-flying’) UAVs and prospective implications to UAV-enabled bridge inspections.
References


JRA. 2017. *State of the City’s Infrastructure*.


Wells, J. & Lovelace, B. 2018. Improving the Quality of Bridge Inspections Using Unmanned Aircraft Systems (UAS).


## Appendix A: Sample Inspection Sheet (DERU)

<table>
<thead>
<tr>
<th>Structure Type</th>
<th>Location Sketch</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROAD AUTHORITY</td>
<td>BRIDGE - GENERAL</td>
</tr>
<tr>
<td>Structure Type</td>
<td>Bridge Number</td>
</tr>
<tr>
<td>Inspection Information</td>
<td>Bridge Name</td>
</tr>
<tr>
<td>Inspection Type</td>
<td>Inspector Name</td>
</tr>
<tr>
<td>GPS COORDINATES - START</td>
<td>Latitude (South)</td>
</tr>
<tr>
<td>GPS COORDINATES - END</td>
<td>Latitude (South)</td>
</tr>
<tr>
<td>Vertical Clearances (road-over-road bridges only)</td>
<td></td>
</tr>
<tr>
<td>Location Detail</td>
<td>Road No.</td>
</tr>
<tr>
<td>Structure Information</td>
<td>No. of Spans</td>
</tr>
</tbody>
</table>

### Inspection Ratings

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
<tr>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
<td>Abut</td>
</tr>
</tbody>
</table>

### Further Inspection Required

### Further Inspection Required

<table>
<thead>
<tr>
<th>Item</th>
<th>Position</th>
<th>Activity Code or Description</th>
<th>Qty</th>
<th>Unit</th>
<th>Make</th>
<th>Safe</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please provide any additional comments or recommendations.
Appendix B: Loss vs earning Rate Graphs

The figures show the Loss vs Learning Rate graphs attained for the Defect-type Classifier (a), Crack-type Classifier (b), Crack D-Rating Classifier (c) and Exposed Rebar Detector (d). These graphs were used to select appropriate learning rates for each model as described in Section 6.3.4.
Appendix C: Validation Accuracy Graphs

These figures show the evolution of each model’s accuracy on the validation set during training at the selected learning rates and epochs. ((a), (b), (c) and (d) correspond to accuracy graphs for the Defect-type Classifier, Crack-type Classifier, Crack D-Rating Classifier and Exposed Rebar Detector, respectively). Using fastai’s callback function, models with the highest accuracy were selected for final evaluation (Section 6.3.5).