UNIVERSITEIT·STELLENBOSCH·UNIVERSITY
jou kennisvennoot • your knowledge partner

Mechanical project 478

Final Report

The Design, Analysis and Investigation into a Baboon Detection System using Machine Learning

Mr DS de Klerk
19249462

Study Leader
Dr W. Smit

November 2020

saam vorentoe · masiye phambili · forward together

Department of Mechanical and Mechatronic Engineering
Departement Meganiese en Megatroniese Ingenieurswese
Privaat Sak X1, Private Bag X1, Matieland, 7602
Tel: +27 21 808 4204 | www.eng.sun.ac.za

ENGINEERING
EZOBUNJINELI
INGENIEURSWESE

Executive Summary

Title of Project
The design, analysis and investigation into a baboon detection system using machine learning.
Objectives
To determine whether current technology is capable of solving the real-life baboon classification problem.
What is current practice and what are its limitations?
There are classifiers that are able to detect baboons however achieving real-time inference on low-cost edge devices need development and testing.
What is new in this project?
The design and implementation of a image classifier that can detect only baboons and be deployed successfully on a low cost edge device.
Was the project successful? How will it make a difference?
The project was a success and achieved the objectives. The success of the project allowed for a further investigation and design into the commercialisation of the image classifier system.
What were the risks to the project being a success? How were these handled?
It is difficult to develop a classifier that does not only detect baboons accurately but does not require much space for storing the trained neural network. This risk will a mitigated by in depth studying on machine learning algorithms and the development of efficient software.
What contributions have/will other students made/make?
There have been no contributions to this topic to date.
What aspects of the project will carry in after completion and why?
Further development and deployment of this technology integrated with a mechatronic alarm system in a fixed surveillance location.
What arrangements have been/will be made to expedite continuation?
Study into modern machine learning techniques in a practical manner and the deployment of these machine learning models on edge device such as a microcontroller.

Plagiarism declaration

I have read and understand the Stellenbosch University Policy on Plagiarism and the definitions of plagiarism and self-plagiarism contained in the Policy [Plagiarism: The use of the ideas or material of others without acknowledgement, or the re-use of one's own previously evaluated or published material without acknowledgement or indication thereof (self-plagiarism or text-recycling)].

I also understand that direct translations are plagiarism, unless accompanied by an appropriate acknowledgement of the source. I also know that verbatim copy that has not been explicitly indicated as such, is plagiarism.

I know that plagiarism is a punishable offence and may be referred to the University's Central Disciplinary Committee (CDC) who has the authority to expel me for such an offence.

I know that plagiarism is harmful for the academic environment and that it has a negative impact on any profession.

Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully (acknowledged); further, all verbatim copies have been expressly indicated as such (e.g. through quotation marks) and the sources are cited fully.

I declare that, except where a source has been cited, the work contained in this assignment is my own work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

I declare that have not allowed, and will not allow, anyone to use my work (in paper, graphics, electronic, verbal or any other format) with the intention of passing it off as his/her own work.

I know that a mark of zero may be awarded to assignments with plagiarism and also that no opportunity be given to submit an improved assignment.

Signature:	[illegible] ....
Name:	Devon Shaine de Klerk
Student no:	19249462
Date:	November 2020

ECSA Outcomes

Outcome	Chapter
ELO 1. Problem solving: Demonstrate competence to identify, assess, formulate and solve convergent and divergent engineering problems creatively and innovatively.	1, 2, 3, 4, 5
ELO 2. Application of scientific and engineering knowledge: Demonstrate competence to apply knowledge of mathematics, basic science and engineering sciences from first principles to solve engineering problems.	2, 4
ELO 3. Engineering Design: Demonstrate competence to perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes.	3, 4, 5
ELO 5. Engineering methods, skills and tools, including Information Technology: Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology.	3, 4
ELO 6. Professional and technical communication: Demonstrate competence to communicate effectively, both orally and in writing, with engineering audiences and the community at large.	Project Proposal; Progress Report; Progress Presentation; First Draft; Final Report; Oral Presentation; Project Poster
ELO 8. Individual, Team and Multidisciplinary Working: Demonstrate competence to work effectively as an individual, in teams and in multi-disciplinary environments.	All aspects of the project
ELO 9. Independent Learning Ability: Demonstrate competence to engage in independent learning through well-developed learning skills.	3, References, Developed code

Acknowledgements

I first and foremost want to thank my study leader, Dr W. Smit for guiding me through this project and sharing valuable knowledge and insight into making a success out of the project and teaching me valuable skills that I can use further in my career as an engineer. I’d also like to thank Dr Smit for having faith in my abilities and pushing me to not give up even though there were times that I thought I’d not be able to complete this project.

I would like to acknowledge and thank all my lecturers during my four years of studying at Stellenbosch University as they helped me equip myself with the skills and knowledge to become a well-rounded engineer.

Thanks to my Parents, Celeste and Quintin for being there for me during my undergraduate studies and always supporting me both financially and personally through these tough and character building years. Your support through it all will never be forgotten and always appreciated.

Thanks to my friends who have helped me along my undergraduate studies and supported me even though I turned down many invites and occasions to do things like dive missions and sleeping out in the mountains.

I would like to say a special thanks to my girlfriend and best friend, Kelly, who has supported me through this project with encouragement and prayer and helped me to keep me going even when I was thinking of giving up. Also, Kelly is the best proof reader I could have asked for.

Lastly and most importantly, I would like to thank God for giving me the strength, direction and wisdom to make good decisions and push through to the other side when I wasn’t able to do so in my own strength.

Table of contents

List of figures

List of tables

Page

List of symbols

A	Wing area
c	Chord length
α	Angle of attack
W	Power in Watt
GB	Storage in Gigabytes

List of Abbreviations

AI	Artificial Intelligence
CCTV	Closed-Circuit Television
RGB	Red Green Blue
DCNN	Deep Convolutional Neural Network
CNN	Convolutional Neural Network
YOYO	You Only Look Once
SIFT	Scale Invariant Feature Transformation
HOG	Histogram of Gradients
SURF	Speed Up Robust Features
ReLU	Rectified linear unit
RPI	Raspberry Pi microcontroller
PCB	Printed Circuit Board
MIPI	Mobile Industry Processor Interface
CSI	Camera Serial Interface
CMOS	Complementary metal–oxide–semiconductor
SD	Secure Digital
SDA	Serial Data
ISO	International Organisation for Standardisation

1 Introduction

The section below outlines the background to this project and delineates the research objectives and motivation.

1.1 Background

Machine learning, a subfield of Artificial intelligence (AI) has had large breakthroughs in the recent century. Machine learning allows cars to be autonomous and has the capacity to detect early stages of lung cancer in human beings with the aid of machine vision and more specifically image classification. The advancement of AI has not only made life easier and more efficient but has the ability to reduce risks and improve safety and well-being.

One of the most fundamental sensory modalities which humans use to process information from their periphery is sight. The ability of the brain to process visual stimuli in the form of frames allows for subconscious learning about one's environment and adaptation through continuous interchange and interaction. This processing of information enables humans to react appropriately to stimuli and contributes to making decisions.

Machine vision is just that. What if machines can see the environment they are in and make educated decisions and take actions accordingly? Cameras have developed over the years to be able to take high resolution photographs however these photographs are simply just pixels on an image and not useful if computers cannot use them to understand what is happening in the picture.

A breakthrough in machine vision was postulated by Fei-Fei Li, the director of Standford's Artificial Intelligence Lab and Vision Lab in the early 2000's. Li developed a machine learning model that could classify 1000 categories using a dataset consisting of 1 million images within Mercari. These powerful machine learning models (see model used in section 3.2.1) were trained on powerful computers and were designed to process images on machines with large processing power.

The Overberg region (see figure 1.1 below) in South Africa is inherently known for their baboon troops. These troops are highly intelligent and are known to raid bins, break into properties and invade kitchens as well as steal food and belongings from tourists. Some areas such as Pringle Bay have dedicated rangers who are solely employed to track these baboon troops and keep them out of town. This is done to ensure there is not a risk of baboon damage (Roos, 2019)to property or residents in the area. The rangers additionally act as a measure to help reduce the impact that baboons can have if they get into the urban area.

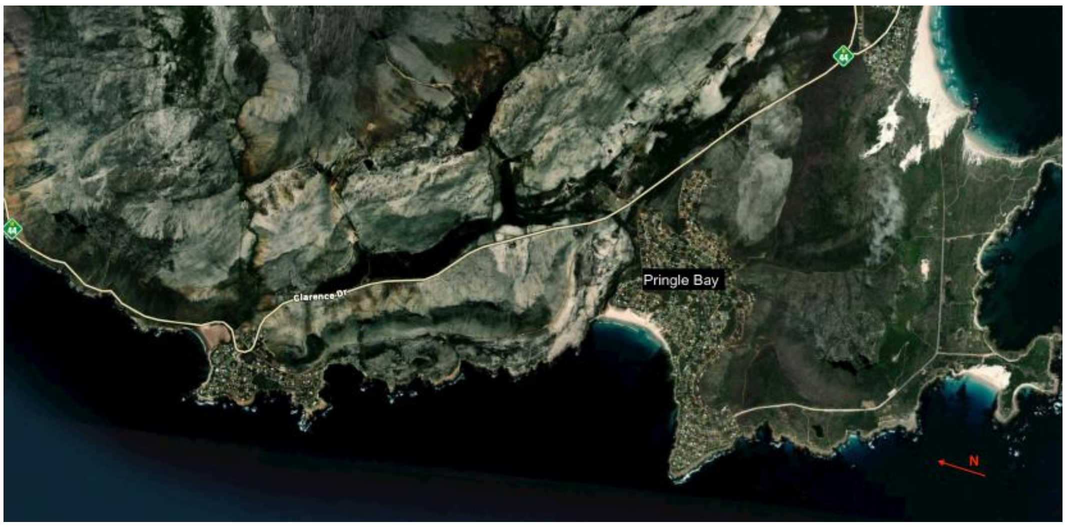


Figure 1.1: Google maps image of Pringle Bay in the Overberg region where this research was done.

Baboons tend to separate from their troop at times and due to limited rangers, it becomes difficult to track individual baboons. The early warning system would be beneficial to notify rangers and residents when baboons are in the vicinity or crossing the perimeters. Integrating machine vision technology with closed-circuit television (CCTV) footage would be an ideal and efficient solution to alleviate the challenges faced in these areas. The system would ideally be able to autonomously classify baboons through live video feed thus removing the need for permanent human surveillance. This technology will continuously process the video footage so that people can receive real time notifications of any baboon activity in the region.

With that being said, running inference on a server could be unsuccessful in the region of interest due to the inherent bandwidth limitations in this region. Thus, the ability for a low cost mechatronic system that can easily be deployed and run image classifications (a term used in computer vision to classify an object/class in an image) locally would overcome this hurdle.

A standalone mechatronic unit that can classify live video footage that sends out an alarm notification when a baboon is detected would be ideal. A microcontroller that runs inference locally with a camera unit that received live footage of the area is recommended. Compared to mainstream computers or servers, microcontrollers are low cost edge-devices. Microcontrollers are inherently known to require less power for computational processes when compared to servers or desktop computers and as such can be powered by a renewable energy source such as solar power.

In order to implement this technology, it is necessary to test whether current technology is capable of solving the real life baboon classification problem as discussed above. This report aims to determine whether it would be viable to implement current machine learning technology and hardware in the region of interest, given the objectives and design specifications. In order to do this, the need for a proof of concept is necessary prior to the potential commercialisation of this technology at a later stage.

1.2 Objectives

The objective of this project was to test whether existing technology is able to solve the real life baboon classification problem in order to implement this technology in a commercialised mechatronic system, at a later stage.

1.3 Motivation

Not only will this technology be able to serve as an early warning system for baboons in the area of interest but will also be beneficial for nature conservation research. Through the use of this technology, the migration paths of baboons can be tracked, and nature conservation specialists will be able to gauge insight into baboons and their lifestyle habits.

Baboon collars are frowned upon by many individuals as they can be perceived to be restrictive to baboons in their natural habitat. This technology allows baboons to roam freely as they do not need to wear collars for tracking purposes.

The further development of this technology to classify other animals as well as identifying poachers in the Overberg region will have a positive overall impact on the environment. By means of early detection and alarm systems, this technology can help to reduce poaching of wildlife both on land and coastal areas.

Due to the low cost and accessibility of the said technology, implementation in nature conservations will be easier as government tenders are less likely to be awarded if the cost implications are too high.

2 Literature Review

This chapter presents literature related to the research topic and will cover all relevant theory applicable to the field of machine learning and more specifically machine vision. The topics discussed will be analysed in order to better understand the field of machine learning, how to implement the technology as well as determining the considerations needed for designing and testing an image classifier.

2.1 Machine Vision

Vision is a complex process that allows humans to classify different visual stimuli appropriately. This process is critical in observing visual data which the brain can use to generate neural networks. These neural pathways determine how we perceive and interact with our environment. When developing machines, it would be favourable for these machines to be able to use the surroundings in order to make informed decisions based on the visual data available.

Humans capture visual stimuli through their eyes which act as a window. Similarly, a machine captures an image through a lens which can be modelled as a window receiving information about the surrounding environment. A typical digital colour image consists of a 3-dimensional grid matrix corresponding to width, height and depth respectively. The width of the image is associated to the sensor used which is usually specified in megapixels. The depth refers to the colour space being used for the specific image. The RGB colour space is most commonly used. Each pixel of the image represents an analogue light signal that was absorbed by the cameras' sensor during the capturing of the image. These light signals are commonly quantised as 8-bit digital values which range from 0-255 (Marais, 2018).

Accounting for all these factors, a typical image with dimensions 64x64x3 at 8-bit resolution would have 12,288 possible values. The image can therefore be represented as a matrix of numbers (see figure 2.1 below) which can be processed using machine learning models. A machine would then use this information in order to determine different characteristics and properties from an image. For many years, specialists have derived mathematical models and methods in order to design machines and generate software code which allow machines to make sense of this data.

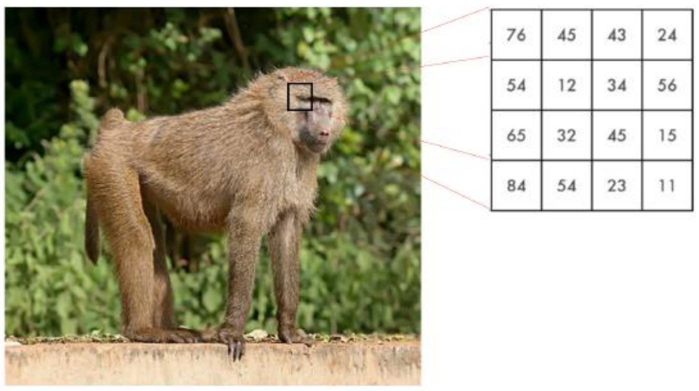


Figure 2.1: Interpretation of how a section in a photograph gets converted into a matrix that can be processed and used for machine learning. (Muhammad, 2010).

2.2 The Foundations of Neural Networks

The human brain consists of millions of neural pathways meticulously interlinked to create a sophisticated neural network capable of solving complex problems and computational workings. Traditional computers make use of serial communication to process instructions and data. The brain has become a popular model that many scientists and professionals are trying to study and mimic. The main idea around machine learning is to create simple mathematical models based on brain-like systems which can be used to solve these complex problems. An field of study that achieves this is knows as machine vision.

Engineers are interested in using and developing these models in order to solve practical problems by mirroring the biological computational methods of the brain. Some of these practical problems include the classification of objects in an image such as classifying animals for nature conservation as well as signs of disease in a medical x-ray scan. Machine vision and more specifically image classification is useful in this project in determining whether a baboon is in a image.

Similar to the brains neural structure (see figure 2.2.), these machine learning models (see section 2.4 for the appropriate model used in this project) make use of a neuron as the basic processing unit and then replicate it in order to solve complex problems around machine learning such as image classification.

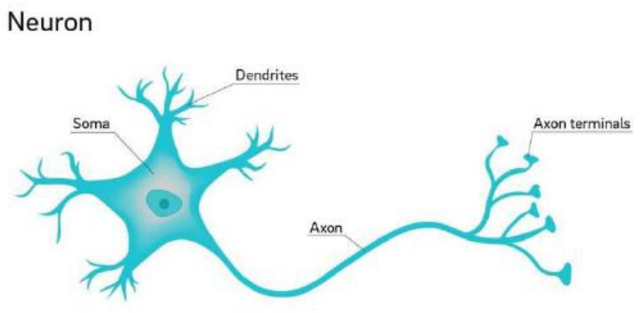


Figure 2.2: Diagram of a basic biological neuron found in the central nervous system (CNS) which the idea of an artificial neural network is based from. (Roos, 2019).

As figure 2.3, the artificial neuron consists of a set of inputs which are typical outputs from other neurons in a neural network. These inputs multiplied by synaptic weights which models the synaptic strengths of biological neurons. The sum of all these inputs multiplied by their respective synaptic weights are then added to the bias of the neuron resulting in the general sensitivity of the neuron. The summed value is then shaped by the activation function which typically limits the minimum and maximum value of the output. A function typically used here for the activation function is a sigmoid function (Glorot & Bengio, 2010).It should however be noted that the process is highly simplified here and complex mathematics models are used to simulate these artificial neurons.

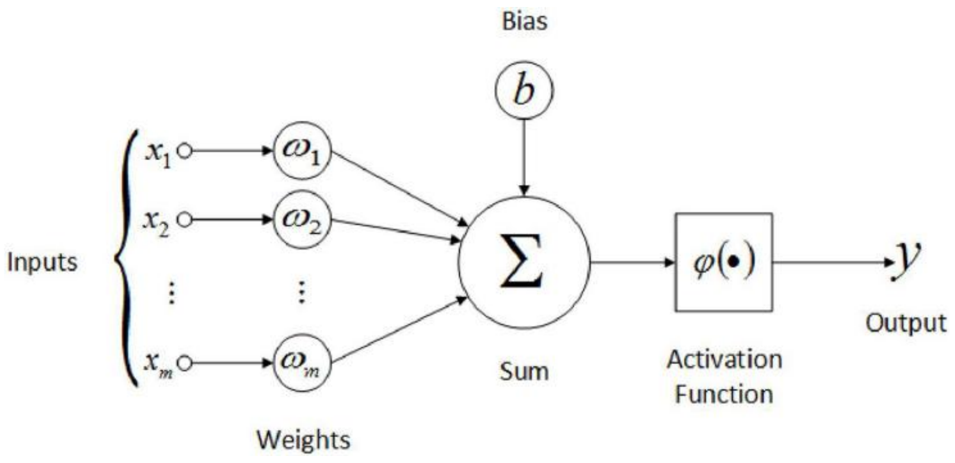


Figure 2.3: Diagram illustrating the architecture of an artificial neuron. (Roos, 2019).

A neural network is a system of these neurons. There are many different types of artificial neural networks available today, where each network is designed for its unique intended purpose. DCNN (Deep convolutional neural networks) have been developed and used mainly for the processing of verbal and visual data. Recent networks such as the YOLO (You Only Look Once) neural network have proven to be more efficient in processing live video feed compared to the slower convolutional networks. This report will however focus on DCNN's as they have proven to be efficient in learning locally shift-invariant features and thus demonstrate their powers in speech and image processing (Zhou, 2020).

2.3 Detection and Classification

In the early stages of AI, before the deep learning era, whole images were scanned intuitively by means of a sliding window. Where images differed in aspect ratios these sliding windows were scaled accordingly and were used to determine unique feature vectors from the image. The encoding of these feature vectors was done by means of low-level visual descriptors such as SIFT (Scale Invariant Feature Transformation), HOG (Histogram of Gradients) or SURF (Speed Up Robust Features). These visual descriptors proved to show a certain level of robustness in terms of scale, illumination and rotation variance (Wu, et al., 2020).

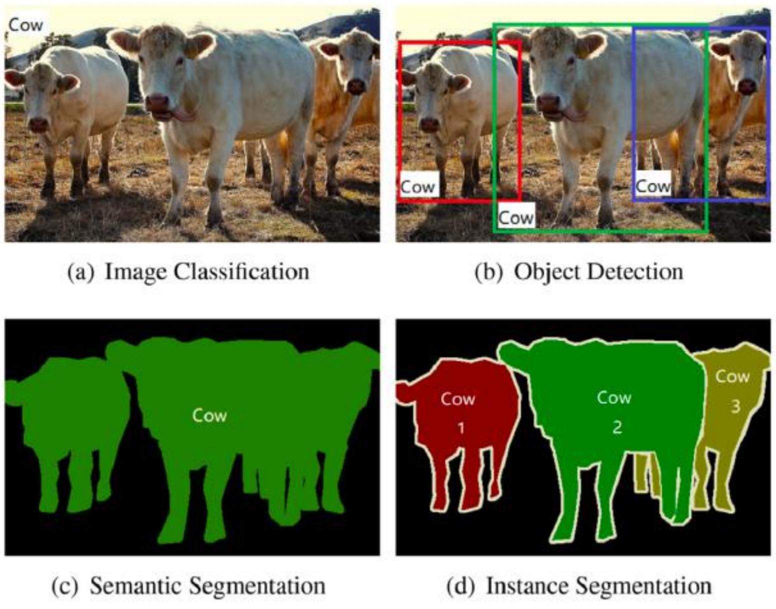


Figure 2.4: Representation of the various functions which neural networks are able to perform. (Wu, et al., 2020).

These regions were then labelled according to their unique features. Machines were used to identify features of different images associated to pre-encoded feature vectors. This method described above is known as Feature Engineering and is robust for small scale training. There were however some major limitations pertaining to this method. These limitations included: (i) feature windows were generated manually which took time and were difficult to generate for large datasets; (ii) features were handcrafted based on visual cues and were thus limited to simple images and proved more difficult once image complexity increased; (iii) each step of the detection process was designed and optimised separately. It can be noted that due to the need to manually generate these feature-vectors, a slight change in an image would not fit these features (Wu, et al., 2020).

As discussed above, the limitations associated with a feature engineering approach can be represented by the following practical engineering problem; if figure 2.5 (left) below were to be labelled as a monkey, the machine would not be able to detect figure 2.5 (right) as being a monkey due to the picture pertaining being comprised of slightly different features. In order to account for all variations in an image, the machine would need to have pre-encoded vector features for all possibilities of a specific label. This could land up being tedious and time consuming, even sometimes impossible when accounting for viewing angles that an object can be seen from.

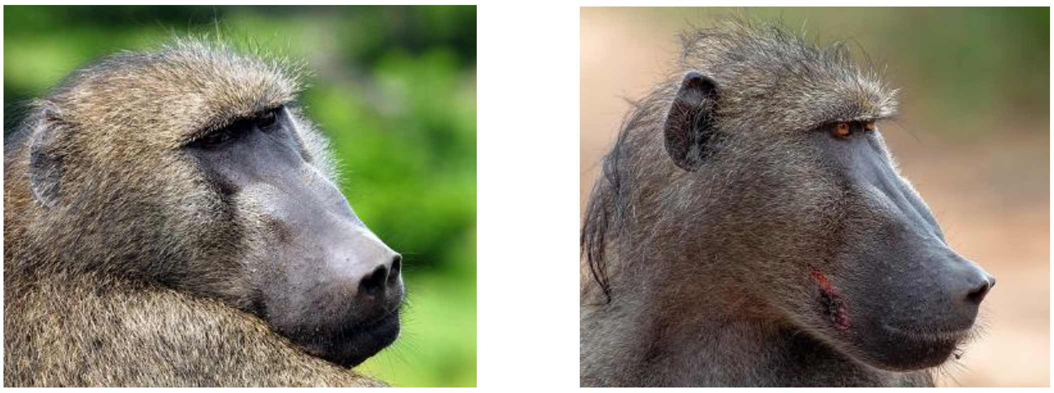


Figure 2.5: Comparison of an individual baboon with a slightly angled profile (left) to a side profile (right) image. This figure represents how a slight change in the class profile would result in classification challenges if feature engineering were used.

Due to the limitations of these traditional image classifiers, deep convolutional networks have been developed and prove more efficient in the generation of hierarchical feature representations from raw pixels (extrapolated from images) to high level semantic information (figure 2.4). These feature representations are learnt automatically from training networks off large datasets.

DCNN’s have the potential to improve feature representations as more data (in our case, big data from images on the internet) becomes available when compared to traditional visual descriptors. Since these traditional visual descriptors are fixed, their feature representations cannot improve even if more data becomes available. Having said that, as data becomes more readily available, the potential which these DCNN's hold for the improvement of feature representations becomes more evident. DCNN's are a more efficient and powerful means of classifying images when considering high-level abstraction, compared to traditional Feature Engineering techniques.

It should be noted that these deep learning methods have been around since traditional feature engineering methods, however due the lack of big data and more powerful processing units such as graphics processing units, these deep learning methods weren't practically useful until the boom of the recent big data era (Marais, 2018).

These deep learning methods have become ever popular and proven very useful in the field of vision and natural language processing. Functions that are able to represent high-level abstractions (in vision and speech), are characterised by non-linear functions (as derived from the neural pathways) which are found in these neural networks (Glorot & Bengio, 2010).

2.4 Detection and Convolutional Neural Networks

When considering object classification and object detection, although the two are commonly confused, object classification is followed by object detection. Object detection entails recognising the unique object in a region/image and entails drawing a boundary around an object. Image classification on the other hand entails assigning a class to the respective objects in an image.

Current modern detectors are categorized into two main categories namely one stage detectors and two stage detectors. Two stage detectors have two main processes. Firstly, the detector generates proposals for sparse sets followed by a second phase whereby feature vectors of these generated proposals are encoded by the deep convolutional neural networks so that the prediction of these object classes are made. This differs from one stage detectors, as these do not have a separate stage for proposal.

One stage detectors (see figure 2.6 below) consider all components of an image as a potential object and thereafter classifies each region as either a background feature or an object. For this reason, one stage detectors are faster at performing real-time object detection but are poorer detectors when compared to two stage detectors (Wu, et al., 2020).

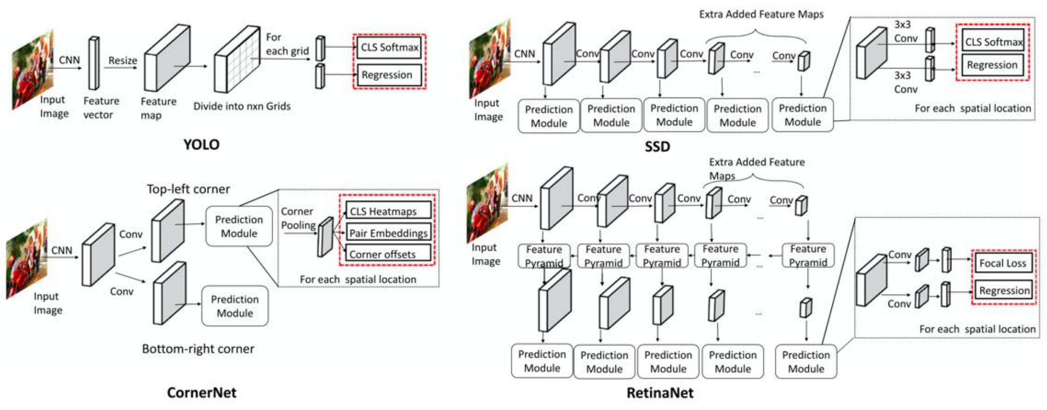


Figure 2.6: Overview of how one-stage detectors are used for generic object detection. (Wu, et al., 2020). (Muhammad, 2010).

2.5 Deep learning and Convolutional Neural Networks

Deep convolutional neural networks are composed of a sequence of convolutional layers, pooling layers (sub-sampling), non-linear activation layers and fully connected layers (classification) as seen in figure 2.7 below. Typical deep convolutional neural networks are comprised of several convolutional, pooling and non-linear activation layers. Each convolutional layer is followed by a non-linear activation function that associated respective weights to each convolution (Kuo, 2016).

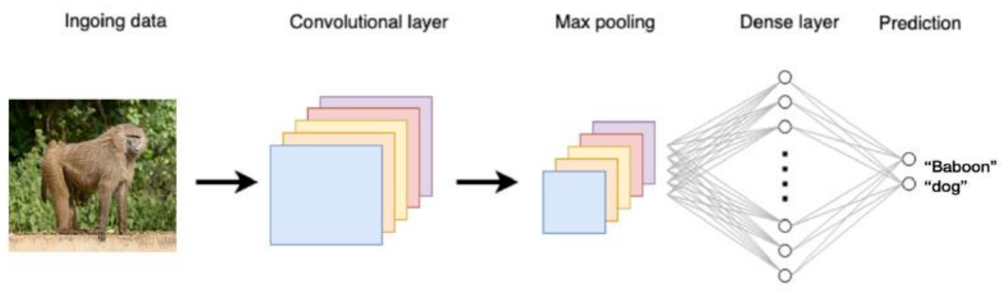


Figure 2.7: A diagrammatic representation of a convolutional neural network illustrating the different stages in a CNN when used to classify ingoing data (an image).

2.5.1 Convolutional Layers

The convolutional layer generates a feature map by taking n x n kernels, otherwise known as filters, from an image and convoluting it. There are many different types of kernels which are used for various convoluting functions such as edge detection, image sharpening and box blur. The main objective of these kernels are to extract specific information from an image for different types of pattern recognition used for specific convolutional layers. These kernels which are typically smaller matrices slide through the image map which is generated from the original image to produce various convoluted layers, as delineated in figure 2.8 below.

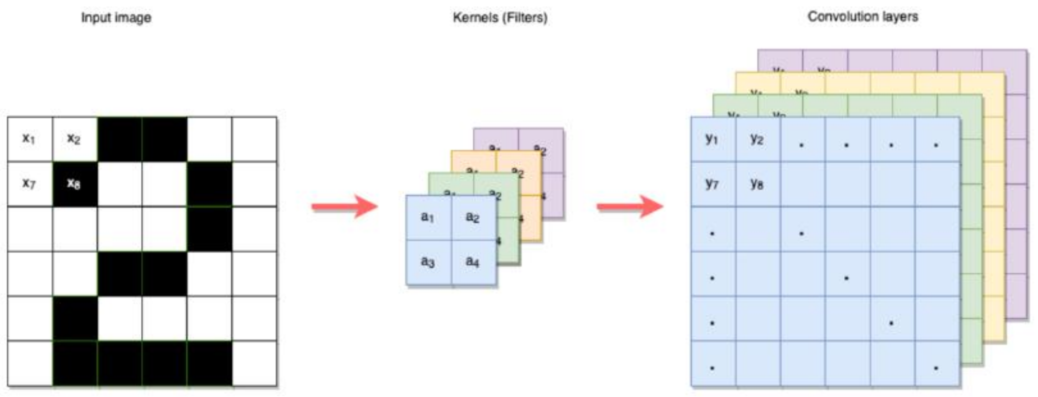


Figure 2.8: Process/Diagrammatic representation of the application of kernels to an input image in order to generate convolutional layers which are then used in the CNN to achieve object classification**.** (Liss, 2020).

2.5.2 Activation Functions

Each pixel of these layers is connected to adjacent pixels from previous layers, which are called the receptive field. A non-linear activation function is then applied to these feature maps.

The main functionality of the activation layer is to create a universal method in the neural network to assess the various levels of intensity. These functions (see figure 2.9 below) represent different characteristics and can thus be used according to the requirement of the network (Liss, 2020).

$$\text{sigmoid: } \sigma(x) = \frac{1}{1+e^{-x}}. \tag{1}$$

$$\text{ReLU: } \sigma(x) = \max(0, x). \tag{2}$$

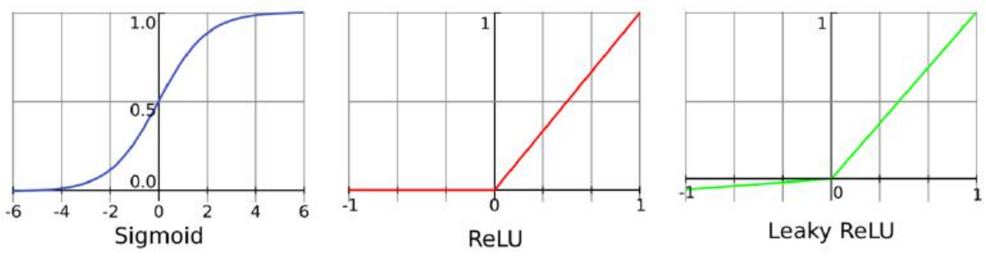


Figure 2.9: Three non-linear activation functions adopted by CNNs. (Liss, 2020).

2.5.3 Pooling Layers

Pooling layers summarize the signals within the receptive fields. The main objective of these pooling layers is to reduce the number of parameters which result from the output of the convolutional layer. By doing so, the pooling layer increases the performance of the network without reducing its accuracy. There are two common ways of pooling namely average pooling and max pooling. An example of max pooling can be found in figure 2.10 below.

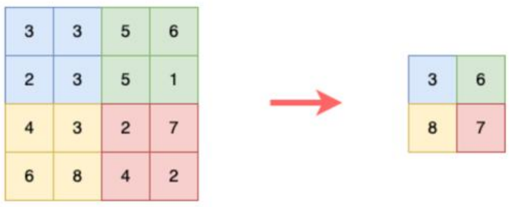


Figure 2.10: Max pooling of a matrix. It is evident that max pooling reduces the number of parameters of the matrix without impeding crucial information about the image(matrix). (Liss, 2020).

2.5.4 Fully Connected Layer

Thereafter, the classification stage occurs whereby the fully connected layer processes the signals. These fully connected layers are comprised of connecting the previous layers outputs to the inputs of the preceding layer. Each connection has an associated weight which is determined by means of training the neural network (Wu, et al., 2020).

3 Design

This section/chapter focuses on the hardware and software design as well as the experimental design investigation used to analyse and test the system. This is necessary to determine whether or not the system design contributes to the achievement of the project objectives.

3.1 Hardware

Many machine learning models run on the cloud where data is stored and applications are processed on servers located in large data centres. Due to bandwidth limitations, these models can be slow when accessing them from a device or sending data to them from a local device for classification or processing. One of the ways to overcome this challenge would be to deploy these models by means of edge computing, which integrates the model on the same hardware as the data source used. This significantly reduces latency, improves energy consumption with data communication and improves security. The deployment of these machine learning models on microcontrollers would mainly be the inference task which requires significantly less computational power than the training phase of the neural network.

In addition, limited memory footprint is one of the most restricting performance limitations when it comes to running a real-time inference classification model on an edge device such as a microcontroller. Most microcontrollers have limited memory space which range from 10-100 KB. This means that the machine learning model would need to run on this limited memory space which could prove challenging as most classifier models can be up to 80 Megabytes large. Furthermore, limited computational resources which these microcontrollers have, result in it being difficult to run classification tasks while achieving high classification performance and accuracy.

3.1.1 Design Requirements

Table A below summarises the hardware design requirements which are necessary to determine which option best suits the outcomes and objectives of this project. This is achieved through comparing alternative hardware options. (see project objectives in section 1.2).

Table 3.1: System hardware requirements

Requirement	Measurable	Description
Memory	Kilobytes	The more ram available, the faster the process speed of the microcontroller. This will correlate to the reference time of the classification process.
Clock speed	Gigahertz	The higher the clock speed, the more calculations the controller can process in a given time period and thus faster inference.
Compact	Size	The microcontroller must be compact in physical size as if the system gets commercialised, it would need to be in the size range of a typical CCTV camera.
Affordable	Rands	The microcontroller would need to be low cost as the need for many of these devices in a given location could be required. There is also a advantage to deploying a high performance classifier on a low cost microcontroller.
Operating system	NA	The software architecture of the microcontroller will determine which software is able to run on it.
Peripherals	UART, GPIO, I2C	The microcontroller should have these main peripherals in the case for future product development and system integration such as external GPS units.
Power Consumption	Watts	This is the power consumed by the microcontroller. The microcontroller would need to draw as little power as possible as a industrialised version would need to run on a battery that would be solar powered.
Storage	NA	Storage capacity or ability to add storage.

3.1.2 Hardware Options

When considering hardware options available, three different microcontrollers were compared. These are popular in industry for machine learning and more specifically image classification.

3.1.2.1 STM32F407/417

The STM32F407 microcontroller is limited to 1 Mbyte of flash memory. The microcontroller would therefore need extra storage space to store the image classification model. An external SD card would be most ideal for this use. In order for the microcontroller to interface with the external memory card, an external card adapter module would be needed. This module has its own circuitry that allows communication between the microcontroller and the SD card.

An external camera module would need to be connected to the microcontroller for data capturing. The CONFIG SIGNALS are signals sent to the camera module that determines the camera setup such as exposure, framerate and other configuration settings. Most of these camera modules are communicated by means of the I2C bus on the microcontroller. The VSYNC and HSYNC signals are data synchronisation signals provided by the camera module. VSYNC and HSYNC are used for frame synchronisation and line synchronisation respectively. The PIX CLOCK is the camera clock and would be provided according to the camera module datasheet accordingly. The hardware connection diagram can be seen in figure 3.1.

The communication between the microcontroller and the external modules are achieved by means of low level C code. This process can be challenging and time consuming due to the fact that the wired connections would need to be soldered by means of a PCB.

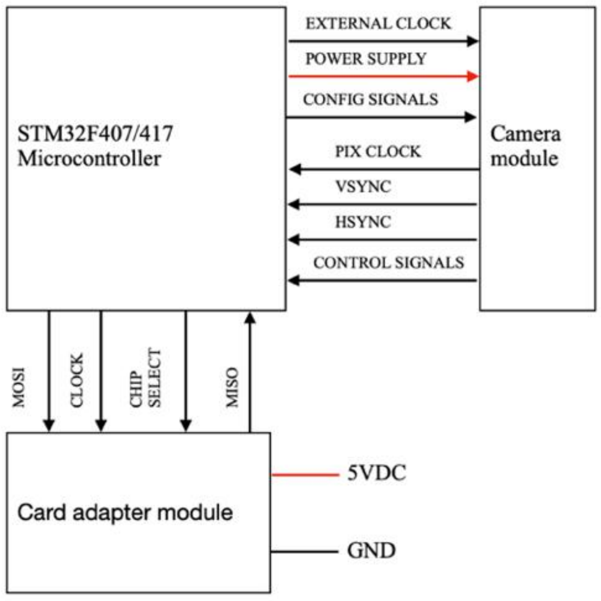


Figure 3.1: Hardware connection diagram representing hardware connections between the STM32F407 microcontroller and the system peripherals such as the camera module and external storage hardware.

3.1.2.2 Raspberry Pi 4 8GB

The raspberry pi has a built-in micro SD card slot which is required for the Raspbian operating system is stored and run from. The highest performing RPI comes with 8GB of RAM and a quad-core CPU. The RPI has additional USB ports which can be used for external storage that allows the user to store much larger files than a micro SD card. The RPI 4 comes standard with a built in wireless module which allows the user to communicate with the RPI wirelessly and/or remotely from his/her laptop. This is ideal if the device needs to be accessed remotely whilst in the field.

The microcontroller communicates with the camera module via a standardised MIPI CSI-2 (Camera Serial Interface) interface (see figure 3.2 below). The camera module can be easily configured by means of the configuration settings once logged into the RPI. The RPI V2 camera module is a 8-megapixel camera capable of achieving a maximum still picture resolution of 3280 x 2464 pixels. It makes use of a Sony IMX 219 PQ CMOS image sensor in a fixed-focus module. The module is connected to the raspberry pi by means of a 15-pin ribbon cable to the RPI's dedicated 15-pin MIPI CSI-2. The camera module is capable of operating in temperatures ranging from -20 to 60 degrees Celsius.

No soldering is required when using the RPI and the camera module, which makes this option seamless and easier to configure. Furthermore, the RPI has built-in commands which can easily be called to either configure or make use of the camera module.

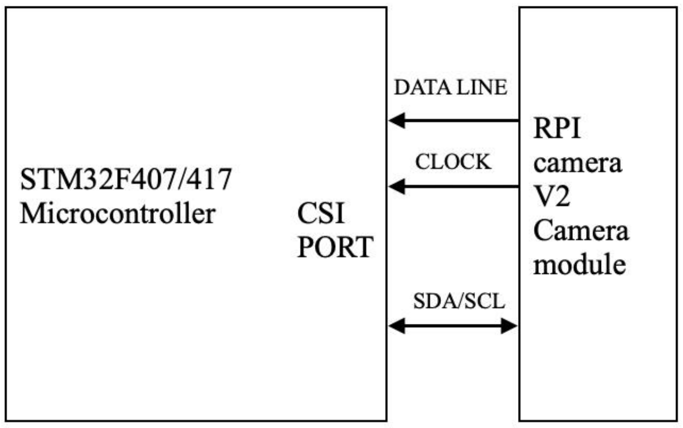


Figure 3.2: Hardware connection diagram representing the hardware connections between Raspberry pi 4 microcontroller and system peripherals such as the camera module.

3.1.2.3 NVIDIA Jetson NANO

Although the NVIDIA Jetson nano is similar to the RPI 4 in terms of hardware configuration and performance, it is a much more powerful microcontroller. This device was designed with machine learning in mind and as a result has a built in 128-core NVIDIA Maxwell^{TM} architecture base GPU. This significantly improves its performance when compared to the RPI, however this comes at a cost. The cost comparison can be seen in table B below. The NIVDIA nano runs on the Linux for Tegra^{R} which allows popular machine learning programming languages such as Python to run seamlessly on the operating system.

The RPI V2 camera module is compatible with the Jetson nano. The Jetson nano has MIPI standard ports which allow for all MIPI camera modules to connect to it (see figure 3.3 below). The availability, performance and low cost of the RPI V2 camera module make it best suited for the Jetson nano.

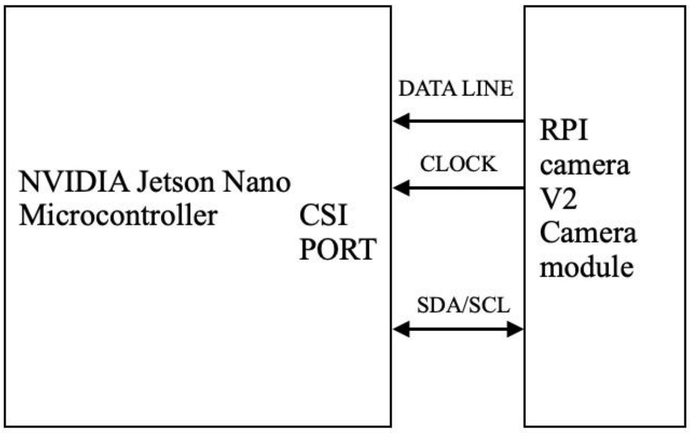

NVIDIA Jetson Nano Microcontroller
CSI PORT
DATA LINE
CLOCK
SDA/SCL
RPI camera V2 Camera module

Figure 3.3: Hardware connection diagram representing the connections between the for NVIDIA Jetson Nano microcontroller and system peripherals such as the camera module.

3.1.2.4 Hardware Options Summary

The specifications of the hardware options discussed above are summarised in table B below in order to quantitively and qualitatively conclude which hardware option is most suited for the project. The hardware chosen for the system will be discussed and outlined in the section to follow

Table 3.2: Summarised comparison of hardware options

Alternative	STM32F407/417	NVIDIA Jetson NANO	Raspberry Pi 4 8GB
Clock Speed	168 MHz	1.43 GHz	1.5 GHz
Cost	R670-00	R2145-00	R1345-00
Operating System	External code deployment onto OS	Linux	Linux
Peripherals	UART, GPIO, I2C	UART, GPIO, I2C	UART, GPIO, I2C
Soldering	YES	NO	NO
Power consumption	0.22 W	5 W	3.4 W

3.1.3 Implemented Hardware

With any hardware implementation one needs to keep in mind the application and requirements of the system. Even though the STM microcontroller draws little power, it is not as powerful as other options. The STM microcontroller would be useful for mechatronic systems but is complex and has significant limitations when considering and implementing machine learning compared to the other options.

Although the NVIDIA Jetson is the most powerful device for machine learning, it is significantly more expensive. The RPI community offers more support than the other options which is useful in designing software for these hardware components and solving problems that might arise. Many programs such as MATLAB also support RPI systems which make the integration of software onto the hardware seamless and reduces the amount of complexity that can come along with the migration of software on these standalone devices.

When considering the three options, the NVIDIA Jetson NANO is the most powerful device, which contributes to a faster inference speed during classification. The RPI is however still more than capable of achieving fast inference times. If the image classifier needed to be trained on the microcontroller, the NVIDIA Jetson would be a better option. Training nevertheless can be done on a more powerful machine such as a laptop and thereafter the trained model migrated onto the microcontroller.

Therefore, due to the RPI community support, low cost and reduced complexity of software migration on the microcontroller, the RPI 4 8GB with the RPI V2 camera module was chosen for this application. It is worth nothing that if the project does expand in future development, the Jetson nano package would become the favourable choice even though it comes at a higher cost.

3.2 Software

In this section, the software orientated scope of the project will be discussed. Firstly, the CNN used for the classifier will be introduced followed by the design requirements which are necessary to achieve the project objectives. The various software platforms are then argued and the implementation of ResNet50 is presented.

3.2.1 Neural Network Used

ResNet50 is a pretrained 50 layer deep convolutional neural network. This network is able to classify 1000 different categories/classes of images. One of the classes ResNet50 can classify are baboons which is ideal for this project as the need to retrain a new network is not necessary. The CNN has an image input size of 224-by-224 pixels. The image from the camera module needs to be resized before being passed into the CNN for classification. Due to the network being a deep neural network, powerful processing is required for high inference speed. This reference speed is a design requirement that can be seen in section 3.2.2 below. This inference speed is dependent on the classifier density as well as the hardware performance.

3.2.2 Design Requirements

Software design requirements are necessary for the classifier to perform successfully given the requirements, and to determine whether the alternative hardware options ultimately achieve the project objectives. These requirements are tested by means of an experimental design investigation in section 3.3. The experimental design investigation findings are evaluated and deduced in chapter 4.

3.2.2.1 Image Size

When considering image size, the size of a baboon in an image frame will significantly affect the classifiers certainty of a class classification. It is therefore necessary to determine a baseline ratio of baboon size to image size in order to evaluate the image classifier. A more practical method of measuring this specification would be to consider the distance between a baboon in relation to the camera lens.

It is assumed that the image classifier must only need to classify adult baboons correctly. This is because baboons travel in troops and as such there is always an adult present. Therefore, the image classifier must be able to detect an adult baboon in a photo that is a certain distance from the camera.

Baboons typically travel across frequent trails which makes it easier to install a classifier system along the trail. Noting that these systems would typically be placed on the perimeter of a town such as Pringle Bay in the Western Cape, the camera should be able to correctly classify a baboon from a minimum distance of 10 meters from the camera. The stationary camera used should classify an area such as a trail road or pathway where baboons commonly cross and should be able to classify a baboon when one crosses the trail road or pathway under surveillance.

3.2.2.2 Robustness

The robustness of the image classifier model deals with the ability for the classifier to perform well under various conditions. These conditions refer to the natural conditions which affect the quality of an image.

Gaussian noise is noise introduced to an image based on electrical characteristics from hardware such as amplifiers and detectors. The magnitude of the noise is measured by the standard deviation of a standard bell curve with a mean of zero and a standard deviation of one. Although this noise can be removed from an image, the image's edges become blurred as a result which is not optimal for classification. This noise will need to be reduced by means of keeping the ISO of the camera module as low as possible as high ISO settings typically introduce edge blur.

The RPI V2 camera module should therefore be left on the default ISO setting of 50 in order to reduce the effect of Gaussian noise to the input images.

Shot noise occurs when the image sensor in the camera reads varying intensities of radiation. The radiation that plays a role in this project would be thermal radiation, in the form of heat intensity. The robustness of the image classifier should therefore be tested in environments with varying temperatures. This can be done by testing the system on different days and within different weather conditions. As such it can be assumed/predicted that the temperature during the day will play a role in the type of noise produced. It should be noted that different surfaces radiate heat differently and thus the system should not only be tested in different temperatures but also within different environments with different surfaces such as in the presence of manmade features such as roads, at the seaside and within the bush.

Blurred images may occur when an image of a baboon is captured whilst the animal is moving across the frame. This is difficult to avoid, however a few software settings can be incorporated to minimise the effects if this were to happen. The camera shutter speed has a significant impact on this, and thus reducing the camera shutter speed can reduce the chances of eliciting a blurred image. By changing and adapting the camera shutter speed, the brightness of the image will be impacted as increasing the shutter speed will reduce the exposure of the image. The classifier should therefore be able to detect baboons in both well exposed and poorly exposed photos as the system needs detect and classify baboons in low light conditions.

3.2.2.3 Weather Conditions

The quality and exposure of an image is influenced not only by the weather conditions but also by the presence of noise in the environment. The ability for the baboon image classifier to classify baboons correctly in different weather conditions in necessary for the system to be beneficial and robust thus the system should be tested in various weather conditions including rain, sunshine and cloud cover. The reason for this is that baboons are not seasonal and are active and ambulant in both clear and overcast weather. In order to test this, the system must prove that changes in weather, such as when it is raining, do not have a significant effect on the classification accuracy of the classifier.

3.2.2.4 Inference Time

The inference speed for this system is critical as the system would need to classify images at a high frequency when baboons are in the location. Taking into account that baboons generally travel in large groups, an inference time of 10 seconds would be sufficient as this would allow the system to process a new image every 10 seconds. Less inference time than this would be better, however a inference time longer than 10 seconds is not acceptable as the system could potentially miss a baboon passing by the frame.

3.2.3 Software Options

The software design requirements should be considered whilst keeping its integration with hardware in mind. The reason for this is that the operating system and software architecture of the RPI should support the software written and developed for the project. Furthermore, the software should be able to run efficiently and compute timelessly on the microcontroller.

3.2.3.1 FastAI

FastAI is an open source machine learning platform that has only recently been developed. The drive behind FastAI is to simplify the training of fast and accurate neural networks using modern best practices. FastAI is a powerful platform that uses the Python programming language. Even though FastAI uses powerful algorithms and competes with most world leading machine learning platforms, it is a relatively new platform and is being updated frequently. This reduces the stability of the platform because functions and dependencies can change on a monthly and even daily basis. In August 2020, FastAI updated their whole library to FastAI V2 which no longer supports functions that were available in FastAI V1.

RPI does not have standard packages which support the FastAI library to date and as such open source files and functions need to be manually configured and setup on the RPI environment. This is necessary for FastAI python source codes to run on the RPI. The FastAI libraries that are compatible to run on the RPI ARM environment are available from open source platforms such as Gitbub, however they are not always stable. The process of ensuring the correct environment is setup on the RPI to run FastAI can prove challenging and time consuming if the user is not experienced in the field of software development and engineering.

FastAI is capable of training and using most modern image classifier models including the powerful ResNet models. The training of models can be done using GPU's such as Google Colab which is free to use and runs off the cloud. This allows for model training on Googles powerful servers and thereafter exporting the trained models onto a device such as a microcontroller where inference can be done.

3.2.3.2 OpenCV

OpenCV is a popular open source computer vision and machine learning software library which is used throughout companies, research groups and governmental bodies. It runs on most operating systems including Mac OS, Windows, Linux and Android. OpenCV has MATLAB interfaces which allow users to run OpenCV functions and operations through MATLAB.

The stability and reliability of OpenCV is attributed to its standardised libraries and high popularity in machine learning worldwide.

Installing OpenCV on RPI hardware is less challenging when using the PIP commands from the terminal and dependency errors are less common when running OpenCV functions on RPI software. Having said that, dependency issues may still arise if the wrong software versions are installed. Furthermore, when using OpenCV, trained models need to be in a specific format which are readable and capable of being processed on the RPI operating system.

3.2.3.3 MATLAB

MATLAB is an extremely powerful closed source platform utilized by engineers and scientists worldwide. MATLAB has powerful and specialised machine learning and computer vision toolboxes that allow for seamless training and the testing of modern machine learning models on desktop computers.

MATLAB supports RPI hardware and thus allows the user to train and test models in the MATLAB environment and thereafter export the developed program onto a RPI as a standalone executable application. This is useful as the RPI software does not require any environmental setup in order to run the classifier or program. Furthermore, MATLAB uses various OpenCV functions and libraries which are well developed and used throughout in industry as discussed in section 3.2.1.2. MATLAB is therefore ideal for testing machine learning models in edge devices such as that of the RPI, before commercialising the software or system.

3.2.3.4 Software Used

Table 3.3 below is a summary of the main features of the various software options according to the specified requirements.

Table 3.3: Software Platform Summary

Requirement	FastAI	OpenCV	MATLAB
Platform Stability	Unstable	Stable	Stable
Support	Moderately supported	Well supported	Well supported
Proprietary	Open Source	Open Source	Closed Source
Software migration ease onto RPI	Low	Medium	High
Programming Language	Python	C++	C/C++

MATLAB was used to build the image classifier and deploy it onto the RPI microcontroller. Although MATLAB is expensive compared to the other software options, this project is at its conceptual phase and thus has not yet needed to be commercialised.

MATLAB is a stable platform and accompanies well supported documentation and professional help. Although OpenCV is more ideal for commercialisation, MALTAB is already available through the University of Stellenbosch to individuals associated to the organisation and as such its use poses no additional cost implications on the project. The implementation of MATLAB allows research to be focused on evaluating the main objective of the project and reduce additional time spent on problems such as runtime issues that are often found using the OpenCV and FastAI platforms.

Furthermore, ResNet50 is available on MATLAB as well as OpenCV and therefore further development or commercialisation of the technology can be successfully implemented using OpenCV.

3.2.4 Software Program

Figure 3.4 below shows the main flow of the system at its highest functional level. The main program starts on bootup by means of built-in raspberry pi crontab software. For the real world tests, the main program was started remotely by means of a laptop in order to ensure that the processes were running correctly and that inference was executing correctly. Following this, the program either gets terminated by means of manually terminating the program remotely via SSH protocol or by unplugging the power supply from the raspberry pi. The main program flow can be seen below in figure 3.4.

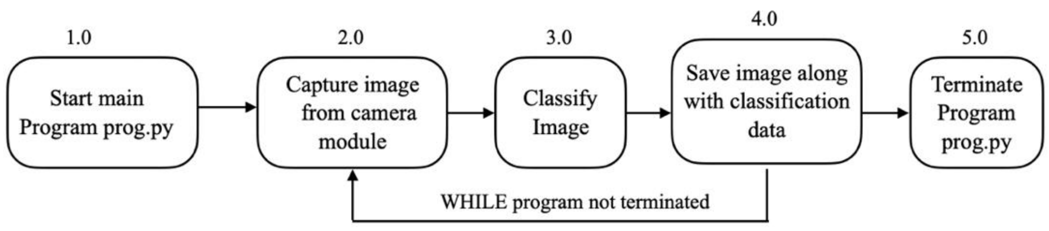

1.0
Start main Program prog.py
2.0
Capture image from camera module
3.0
Classify Image
4.0
Save image along with classification data
5.0
Terminate Program prog.py
WHILE program not terminated

Figure 3.4 Classifier flow logic.

The source code below is the main python program which runs on the RPI. The code for the main program for inference and data capturing are represented as follows:

```
import subprocess                 #import subprocess library
import time                       #import time library for inference time
from time import sleep            #import sleep library
from picamera import PiCamera     #import RPI camera library
import os                         #import os library
import re                         #import re library
import shutil                     #import shutil library
camera = PiCamera()               #setup the camera module
counter = 0
while counter < 1:                #loop for live inference
    camera.start_preview()        #start up camera module
    camera.capture("image1.jpg")  #capture live image
    timenow=time.strftime("%s")   #captures the current time
    camera.stop_preview()         #stop camera
    p1 =
subprocess.run(['sudo','./raspi_fileRead_resnet.elf','image1.jpg'],capture_output=T
rue)                              #calls classifier program for classification
    program_output = str(p1.stdout)  #assigns classification probability
    totaltime= int(time.strftime("%s"))-int(timenow)  #computes inference time
    new_filename = '/home/pi/ClassifierProgram/classifiedimages/' + totaltime +
program_output[-47:-39] + '.jpg'     #saves image with prediction percentage and
inference time
    shutil.copy("image1.jpg",new_filename)  #move image into file storage
    os.remove("image1.jpg")                 #delete image in current directory
```

3.2.4.1 Capturing Real-time Images from Camera Module

The raspberry pi has a built-in library called PiCamera which includes high level python functions and allows image capturing using the PiCamera module V2. Once assigning a class to the camera, the camera.capture() function is called which captures the live image and stores it onto the current directory. This function is called on every classification.

3.2.4.2 Classifying the Image

The classification process in software is complex and is summarised in figure 3.5. The classifier was built and developed in MATLAB and thereafter deployed as a standalone program onto the Raspberry Pi microcontroller.

The following diagram in figure 3.5 represents the flow of the program followed when classifying an image. The program that gets deployed onto the Raspberry Pi is a .elf file which is then called from the python script prog.py each time a new image needs to be classified.

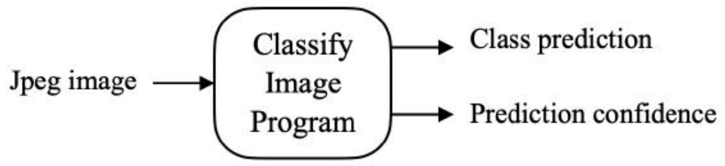


Figure 3.5: Classifier function.

The diagram in figure 3.6 represents a more detailed description of the classifier program.

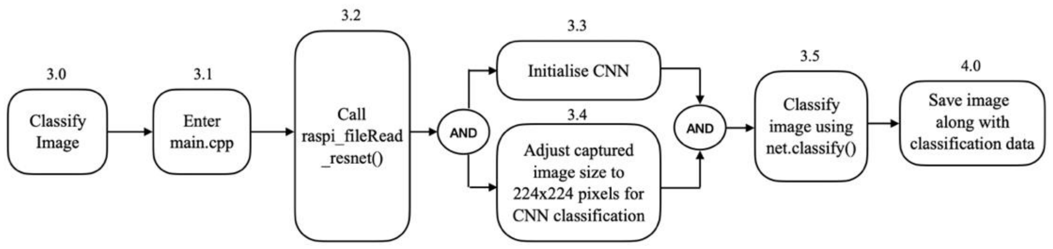

3.0
Classify Image
3.1
Enter main.cpp
3.2
Call raspi_fileRead _resnet()
AND
3.3
Initialise CNN
3.4
Adjust captured image size to 224x224 pixels for CNN classification
AND
3.5
Classify image using net.classify()
4.0
Save image along with classification data

Figure 3.6: Classifier program.

The resizing of the captured image to 224x224 pixels was necessary as it is required by ResNet50 for classification. A function was used for this which made use of the OpenCV library for image resizing.

Once the image was resized, it was passed into the deep neural network for classification. The classify function net.classify() is a built-in MATLAB function which was used. This function returned the label (prediction) and score (confidence) of the classified image. Section 2 discusses the general process of classification.

3.2.4.3 MATLAB Software Deployment onto Raspberry Pi

Firstly, the host computer as well as the RPI were configured into the same wireless network in order to wirelessly link the two hardware devices. A command was then executed in MATLAB that allowed for communication between MATLAB and the RPI. MATLAB's built in target Hardware function was then used in order to create a hardware configuration target. This defines the type of hardware that MATLAB must generate C++ code for image classification on the RPI. RPI runs on ARMv7 architecture which was then defined and assigned to the hardware configuration target. This ensured that when the program was deployed to the RPI, the file architecture was written for the correct arm architecture. Finally, the program written in MATLAB was deployed onto the RPI as a .elf standalone executable program using the deploy() built in MATLAB function.

3.3 Experimental Design

The following section will focus on the experimental design that was used to successfully complete field tests in order to achieve the objective of the project by testing the various system specifications depicted in section 3.2.2.

3.3.1 Research question and Variables

The research question and variables were used to guide and direct the experimental design and ensure that the experimental treatment results are useful in determining whether the experimental hypothesis is justified.

3.3.1.1 Research Question

The type of research methodology conveyed in this project is research in application. Even though the ResNet50 image classifier has already been developed and is able to classify baboons, it is has only been trained to validated and classify baboon in images that are controlled and mainly of high resolution and quality. Although these images are useful in training the classifier, they are not very efficient in testing the robustness of the classifier in a real world scenario where variables such as noise and poor lighting conditions may be present (see section 3.2.1).

As such, validating the classifiers performance in a real-life context by means of real-life images specifically in an uncontrolled environment plays a key role in the development of the classifier. An experimental design needs to be formulated and conducted which can test the various design specifications of the classifier as well as identifying limitations that may arise.

The results attained from the experimental design will be crucial in evaluating the baboon classifier. The research question for the experimental design is therefore whether Resnet50 running inference on a RPI 4 is capable of classifying baboons correctly in a real-world situation.

It should be noted that the probability output from the classifier will be the main variable used to formulate conclusions and findings about the performance of the classifier. This probability output is the returned probability that a baboon class is in the image that the classifier runs inference on.

3.3.1.2 Dependent Variables

The dependent variables can be seen as the output of the process. The main dependent variable for this experimental research is the classifiers output probability to determine whether there is a baboon in the image. It can be anticipated that this variable will be largely influenced by all independent variables and the relationship that exists between them. This relationship will be discussed further in the forthcoming section. Table 3.4 below summarises the dependent variable in this experiment.

Table 3.4: Dependent variable from the Experimental design investigation.

Dependent Variable	Description
Probability	This is the returned probability (output) from the classifier that represents the certainty that a specific class (baboon) is classified in the image (input).

3.3.1.3 Independent Variables

Table 3.5 below summarises the independent variables of the experimental design.

Table 3.5: Independent variable for experimental design

Independent Variable	Description
Distance	The distance that the baboon is from the camera.
Weather conditions	The weather condition at the time image is captured. Three different weather conditions are relevant to investigate in the region of interest namely clear, overcast/cloudy and rain.
Baboon orientation	The orientation of the baboon as seen from the camera lens.
Environment	The environment that the camera captures an image in. These are bush, manmade(road) and Sea/Rocks.

3.3.1.4 Confounding Variables

The confounding variables of this experiment refer mainly to the noise which affects the quality of an image and thus impacts the overall performance of a classifier.

3.3.2 Hypothesis

Current machine learning technology is capable of solving the real-life baboon classification problem.

Technology in this case refers to testing a baboon classifier using ResNet50 CNN running real time inference in a RPI 4 capturing data whilst using a RPI Camera V2.

3.3.3 Experimental Treatments

The experiment needs to be tested in various environments which represent and simulate the natural environment where the system will be implemented for future use. In order to obtain useful data (a large dataset) from the investigation, the classifier unit needs to be moved and tested in different locations in order to assess the classifier's performance in these various environments and conditions. Once in a location, the RPI was powered up by a power bank and then inference was done on the RPI autonomously. While the RPI classifier was operating, it was accessed remotely by means of wireless connection with a laptop from the safety of a vehicle in order to ensure the unit was operating correctly.

The location of the RPI unit was strategically chosen to access the unit for all independent variables in order to ultimately test the hypothesis of the experiment.

The following table depicts the varying conditions in which the unit was tested in order to obtain the valuable results.

Table 3.6: Experimental test locations matrix

	Clear Weather	Overcast/Cloudy	Rain
Bush	Clear and Bush	Overcast/Cloudy and Bush	Rain and Bush
Manmade (Road)	Clear and Manmade (Road)	Overcast/Cloudy and Manmade (Road)	Rain and Manmade (Road)
Sea/Rocks	Clear and Sea/Rocks	Overcast/Cloudy and Sea/Rocks	Rain and Sea/Rocks

Each location and weather condition stated in table above was then tested with various distances (2m-50m) and baboon orientations. The orientation of the baboon was difficult to account for in all setup locations as this variable in less controllable that the others. This is attributed to the fact that one is unable to determine where a baboon is going to be looking at a certain point in time.

The probability can therefore be represented by the following equation:

$$P = \text{function } (W, E, D, BO) \tag{3}$$

Where,

P: Classifier output probability

W: Weather condition

D: Distance

BO: Baboon orientation

The baboon orientation was tested in order to determine whether the classifier was able to detect and classify a baboon correctly, irrespective of its orientation in the image frame. This is due to the fact that most classifiers are trained using ideal images which in this case would be an image of a baboon looking directly at the camera, however in real life, this is not always possible.

3.3.4 How these Experimental Design Results Fit into the Bigger Picture

The results from this experiment were used to test whether the hypothesis is true. If the hypothesis is proven to be true, further development and investigation into a baboon alarm system can be considered and ultimately implemented in real life as a fixed permanent system. If the hypothesis cannot be supported, this will serve as a clear indication that further development into the classifier or hardware needs to be done. These experimental results can save not only time and money but can also serve beneficial with regards to the safety and wellbeing of individuals that are affected by baboons.

4 Results

As discussed in section 3.3 above, the results were obtained from the experimental design with the purpose to determining whether the classifier system successfully meets the objectives set out in section 3.2.2 namely; robustness, image size, weather conditions and inference time.

Due to time constraint of the project and the unpredictable behaviour of baboons, it proved challenging to gather an ideal amount of data (classified images) that cover all spectrums of the classifier's performance. This is typical of research in application as the ideal outcome of an investigation is difficult to achieve due to unforeseeable variables and challenges. These challenges include the limited time available for the project timeline and the difficulty associated with tracking baboons.

4.1 Results from Experiments

The data was gathered by means of testing the classifier unit in the environment of interest, being the Pringle Bay area.

The tests performed followed the experimental treatments discussed in section 3.3.3 as closely as possible. The data captured included 82 images with baboons and 255 images with no baboons. The data captured was restricted to the region of interest in order to maintain the boundary for the experimental design investigation in section 3.3.

The distance which the baboons were from the camera was predicted by eye and was reported in a excel sheet for every image where a baboon was present. It should be noted that this is not the most accurate method of measuring distance, however it was the most suited method when considering safety and the environment.

The orientation of the baboon was manually decided upon by each individual image classifier in order to determine whether the classifier performance is affected by the orientation of the baboon in the image.

The inference time for the concept system was found to be at most 9 seconds and least 8 seconds. This satisfies the design specification of an inference time less than 10 seconds. This is positive as it allows the system to classify a steam of images at 10 second periods which is useful for animals that move quickly across the camera frame and thus allows for many images to be processed in a short period of time if there is high activity in the area where the classifier system in situated.

4.1.1 Reliability of Classification Probability

Results from the two images classified below in figure 4.1 indicate that when a baboon was present in an image, the classifier was able to detect features of there being a baboon in the image as seen by the increased classification probability. Due to the background being the same for these two images, it can be proven that the classifier clearly increases in probability when a baboon is present in an image. Although the classifier was able to show signs of detection, this low-class probability indicates that the classifier is still uncertain that there is a baboon in the classified image as seen by the low baboon class probability.

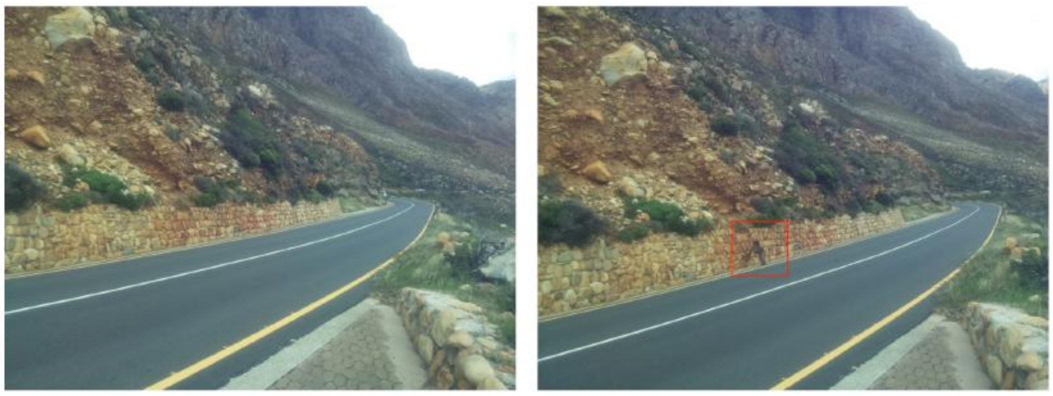


Figure 4.1: Classification Probability of 0.08% (left) and 0.18% (right) indicating that when a baboon is introduced into the image (right), the classifier clearly picks it up which is seen by the increase in class probability.

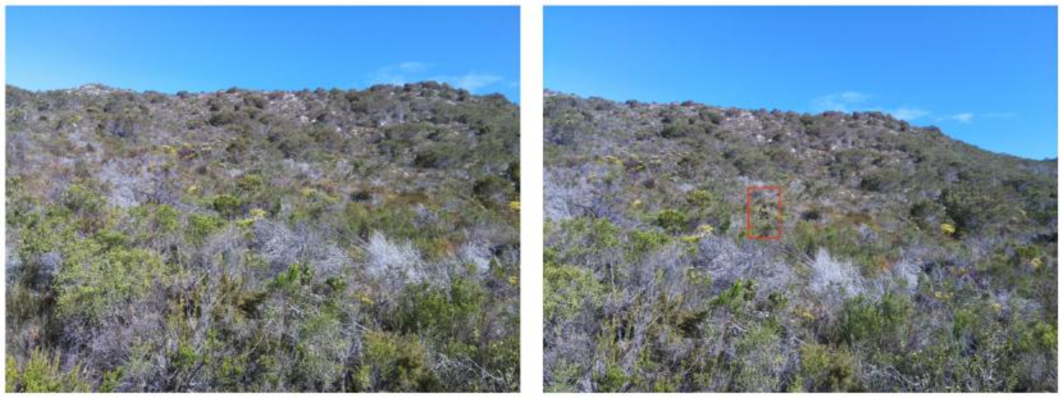


Figure 4.2: Classification Probability of 0.34% (left) and 0.21% (right) showing that non-baboon features in an image can influence the classifier class prediction probability.

With that being said, the above argument is no longer supported when comparing the classification probability of Figure 4.2. This raises the question of whether the images in figure 4.1 and 4.2 play a role in the varying classification probability. This is the challenge faced when it comes to the interoperability of machine learning models. In terms of class probability, it is difficult to understand which image features have the largest impact. This is due to the fact that the classifier is seen as a black box and thus we are unable to easily understand how the model assigns a probability of a certain class to a certain image.

With that being said, the above images prove that the classifier is indeed robust and reliable since small changes in the input image do not led to large changes in the output probability of the certain class. This shows that the classifier is not easily affected by the introduction of noise to the image. However, it can concluded that the probabilities within these ranges cannot reliably indicate that there is a baboon in the image.

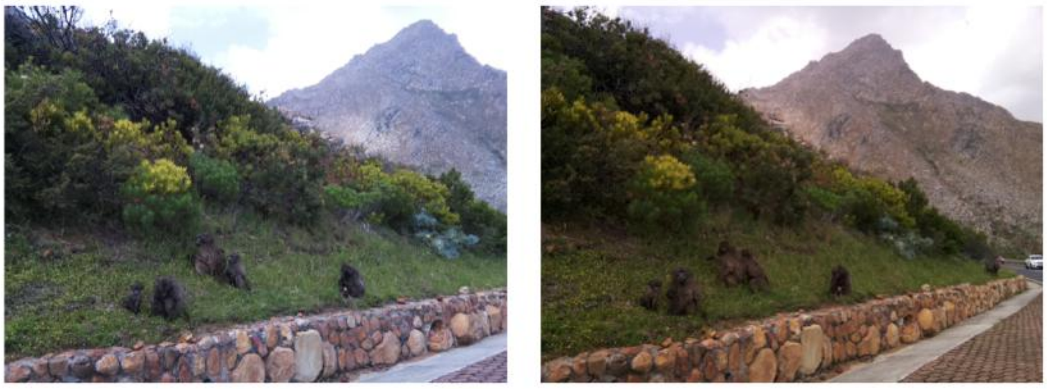


Figure 4.3: Classification Probability of 0.66% (left) and 0.25% (right) representing the classifiers sensitivity to light intensity.

Research has shown that the light intensity of an image can affect the classification accuracy of an image classifier (Prijatna, et al., 2018). The above Figure 4.3 proves this. As seen by the left hand image in figure 4.3, it is apparent that an image with better light exposure (sunny) improves the classification abilities/probability when compared to the same image taken in poorer lighting conditions due to cloud cover. This shows that the classifier is indeed sensitive to lighting conditions in the image. It shows that the design specification referring to the classifiers performance under different weather conditions is indeed affected.

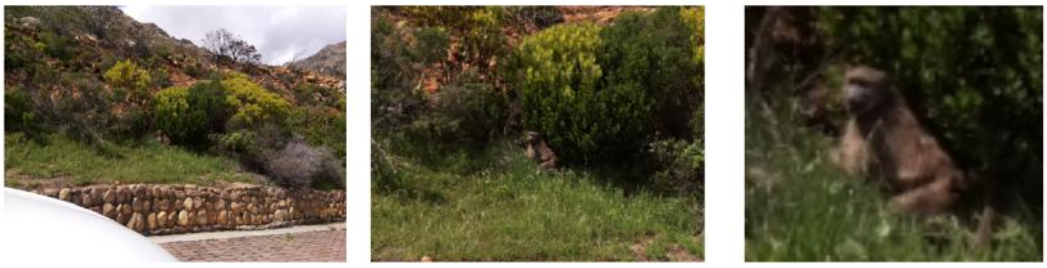


Figure 4.4: Classification Probability of 0.0029% (left), 9.56% (middle) and 44.65% (right) showing how background noise can affect the performance of a classifier.

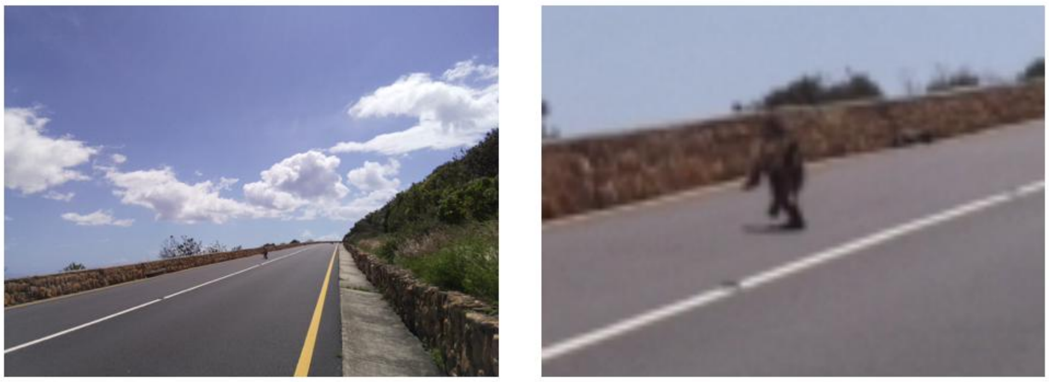


Figure 4.5: Classification Probability of 0.0041% (left) and 1.61% (right) showing how background noise can affect the performance of a classifier.

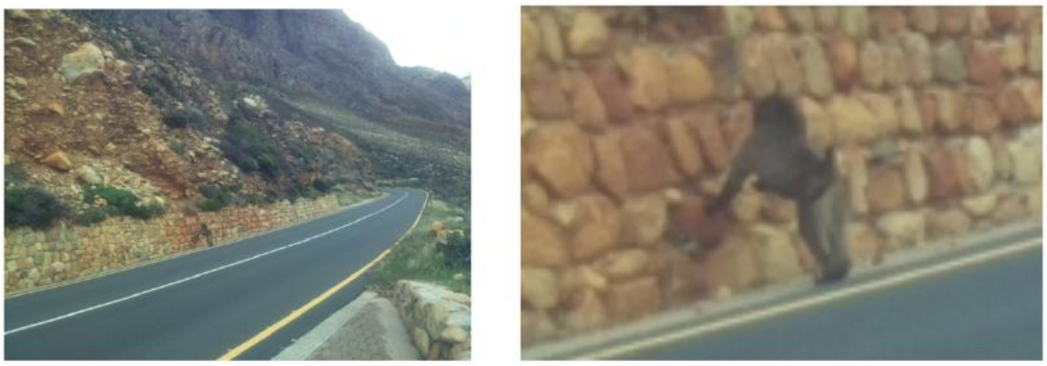


Figure 4.6: Classification Probability of 0.185% (left) and 0.023% (right) showing that other features present in an image can affect the probability of the classifier being able to classify a baboon class.

When an image is classified, the whole image is processed through the CNN. This means that the background as well as the baboon is processed through the abovementioned CNN. The question is then raised if a baboon was in an image, but relatively small in relation to the overall size of the said image, would zooming into the image frame with the baboon improve the probability of the baboon classification? This can be correlated to the design specification of image size. As previously discussed, the scope of machine learning that detects various objects in an image refers to object detection. Although object detection is not within the scope of this project, it was incorporated manually once the images were classified. By looking at figure 4.4 and figure 4.5, it is evident that when cropping the background, the probability significantly increased. This would mean that an uncropped image resulting in a false negative classification could be classified as a true positive once the background is removed.

It should however be noted that removing the background will not always result in the classifier increasing the output probability. This is evident in Figure 4.6, as the classifier returned a decreased probability of there being a baboon in the image even though the background was removed. The reason for this could be that the original image had more baboon class features when compared to the cropped image and thus returns higher classification probability in the original image which has more baboon-like features. This includes features such as the colour of the mountains in the background which are a similar shade to that if a baboon. Once cropping was done, the returned classification probability of the image was reduced. As a classifier could detect features that are not of a baboon which has the potential to increase the output probability of there being a baboon in the image. It proves that the classification probability threshold is critical in the process of object detection and classification.

4.1.2 Limitations of the Classifier

Although the classifier performed well under various weather conditions, limitations were evident. A well trained ranger can spot a baboon easily in its environment. This is because the rangers have been trained to spot baboons in their natural environments, even though the baboon is difficult to spot with an untrained eye. This is one of the challenges faced with computer vision, as a computer is unable to perform well when presented with foreign conditions. This can be seen by figure 4.7 below. This furthermore presents the challenge regarding a machines limited ability to adapt. Humans can adapt to their surroundings more readily than that of a machine due to the brains plasticity and various perceptual abilities. This can however be overcome by adaptive machine learning which is not in the scope of this project.

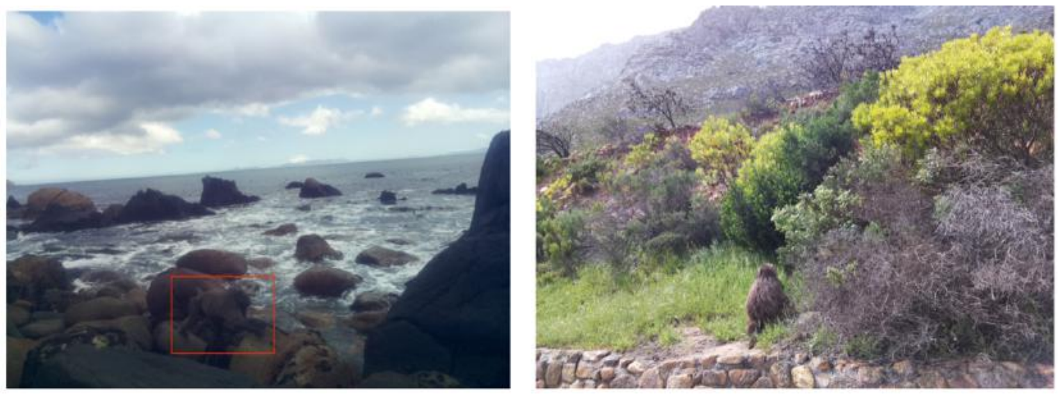


Figure 4.7: Classification Probability of 0.0026% (left) and 36.16% (right) representing the classifiers limited ability to classify a baboon in its natural environment when the baboon is well camouflaged.

Baboons can protect themselves from threats such as predators, as the natural environment allows them to camouflage and blend into their surroundings. It was found that a baboon's ability to camouflage directing impacts the classifiers classification probability. This is evident in figure 4.7 where even though the baboons in both images are of the same size and similar orientation, the classification probability of each image varied based on environmental differences. It can be seen that there is a significant difference in the classification probability when comparing the two images which shows that this is a large limitation to the classifiers performance in different lighting conditions and robustness to changes in the environment. This conclusion can be expanded upon is future research as the ability for an animal to camouflage and blend into their surroundings can be a large obstacle and challenge to overcome when designing modern CNN. This is specifically challenging when it comes to machine vision as a machine cannot be trained on every possible image that includes a baboon.

In figure 4.7, it is likely that the image with the higher classification probability has features of which the classifier has previously been trained on and is thus familiar with that image, whereas the image on the right is more complex and contains features which the classifier has probably never seen or been trained on before.

In light of this, new data such as that produced from left image of figure 4.7 is invaluable and can be used to train the CNN further. This can assist with improving the classifiers ability to classify baboons in their natural environments correctly and with higher certainties. Transfer learning is the method of retraining an already trained network on new data in order to improve its ability to classify a certain class. Transfer learning would thus be a great approach to improve the classifiers performance in a wider range of environments (Bird & Faria, 2018).

4.1.3 Data Analysis

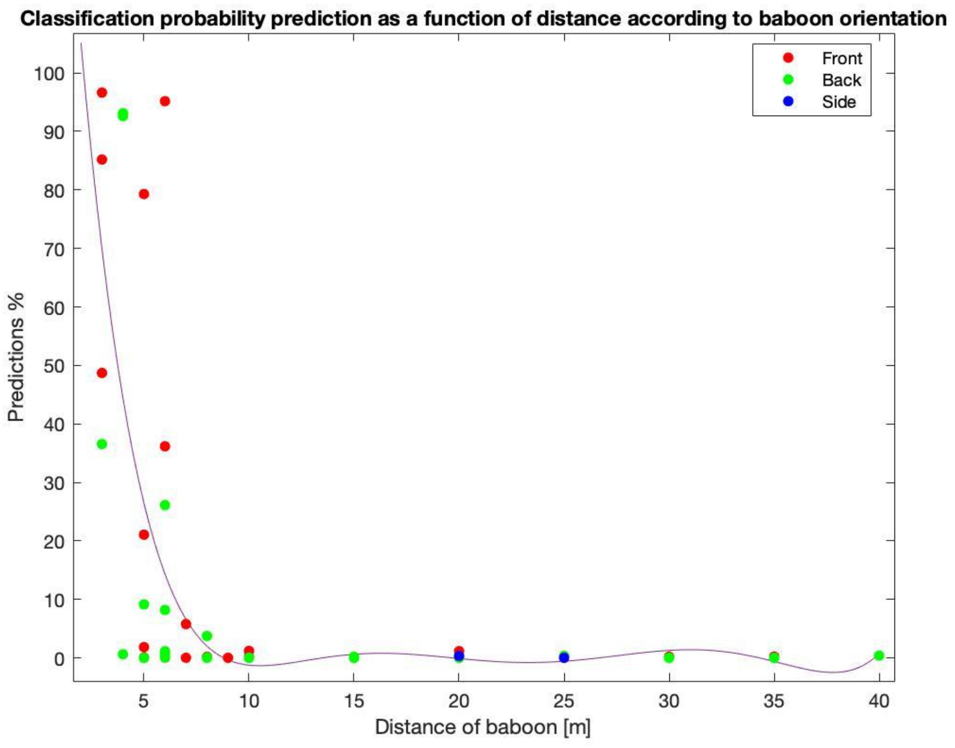


Figure 4.8: Line graph representing the classifiers output probability of images that have a baboon present, as a function of the distance the baboon is to the image, grouped according to baboon orientation in the image.

Figure 4.8 was constructed by means of MATLAB data analysis software. The baboon orientation and distance variables were decided upon by means of the method described under heading 4.1 which was determined during the processing of the data gathered from the field tests. The reason for separating the orientation of the baboon as a different colour plot is to determine whether there is any significant change in the returned probability of the classifier when the orientation of the baboon in the image changed.

The line of best fit was constructed using the method fitting a polynomial using least squares method. The maximum order of polynomial is limited to the amount of data points that are present in the dataset. It is however best practice to use the least degree that most accurately presents the dataset, as a too high degree polynomial may lead to a mathematical phenomenon called polynomial wiggle. This phenomenon may influence the representation of the said mentioned dataset.

As seen in figure 4.4 above, the classifiers returned probability significantly increases when the baboon is within 10 meters from the camera. This is a large limitation to the system as the ability for the classifier to classify a baboon beyond the 10 meter range significantly reduces. A camera can be placed strategically to increase the likelihood of capturing images of baboons within 10 meters. This can be done by placing the system close to an entrance or fence where baboon activity is known to be high. Additionally, if object detection is implemented at a later stage, the classifier will be able to classify baboons with a high probability when placed further than 10 meters. This can be seen in figure 4.4.

The change in weather conditions (as illustrated by figure 4.9 below) did not significantly affect the accuracy of the classifier as presented by the classification probability. This proves that the classifier is able to perform well under different weather conditions. Typical images that were classified in these three weather conditions can be seen in figure 4.9 below.

It is however notable that the classifier performs more optimally when there is clearer skies such as when the weather in not cloudy and rainy. The data captured in rainy conditions was light to moderate rainfall. It should therefore be noted that in the case of heavy rainfall, it is expected for the classifier to perform as well by returning a lower classification probability.

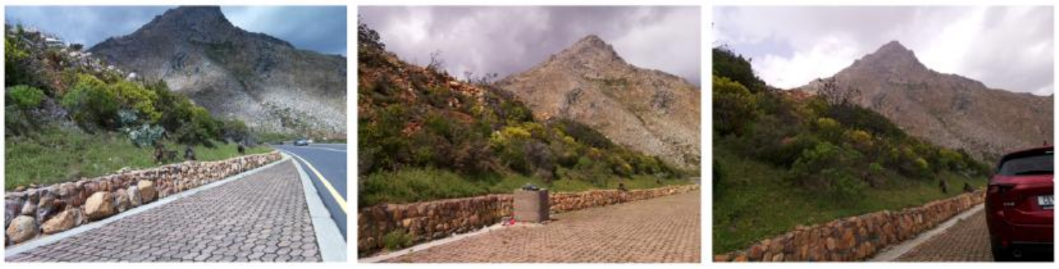


Figure 4.9: Various weather conditions from left to right; Clear, Cloudy/Overcast and Rainy returning a classification probability of 0.13%, 0.0091% and 0.052% respectively. This represents how lighting caused by weather conditions can affect the performance of a classifier

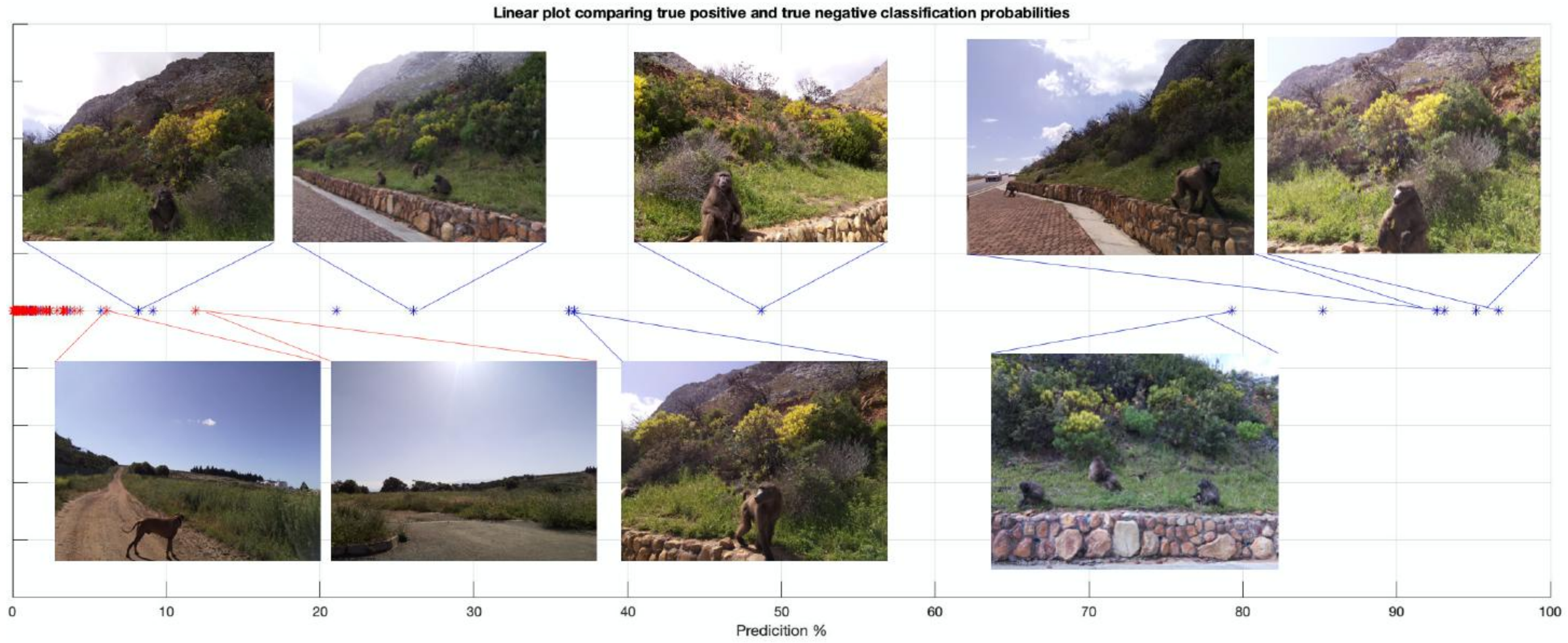


Figure 4.10: A line plot presenting the data captured with their respective classification probabilities. The blue datapoints are images that have a baboon present whereas the red datapoints represent an image that does not have a baboon present.

Figure 4.10 was developed in order to gauge insight into what classification certainty is required to ascertain whether a baboon is present in an image. The plot was developed by two groups of data namely true positive baboon classes and true negative classes. All the data was captured in the region under investigation in order to ensure that the results most correctly represent the classifiers performance in the region of study.

This is a critical point as drawing the boundaries of what data is included in the true negative data classes can be difficult if all regions were to be accounted for. If all regions were included, true negative class data would need to be generated for every case and type of environment along with all possible objects that can be found in the respective environments. In the region of interest, with the exclusion of human beings, the animals most are dogs and wild leopards. A photograph of a leopard was classified as it was difficult to get real life images that had a leopard present. From figure 4.11, it is evident that these two animals return a maximum classification probability of 6.107% which is rather low. As such it can be concluded that if the system had to be deployed in future, the classifier would not easily return a false positive classification.

F**igure 4.11:** Classification Probability of 6.107% (left) and 0.0004% (right) from CAPE LEOPARD TRUST showing that the likelihood of a false positive classification for an animal other than baboons found in the region of interest is low.

From analysis, it can be assumed that a baboon is most likely in an image if the classifier returns a probability of more than 20%. This means that there is a 50% certainty that a classified image will detect the presence of a baboon in an image when the baboon class probability is more than 20%. Despite this, it should be noted that the probability threshold is only appropriate for the system deployed in the Overberg region and other regions such as Cape Town, South Africa would require further data analysis to determine a new probability threshold for the baboon class.

4.1.4 Risk Matrix

The risk matrix delineated below has been developed in order to gain an understanding of whether the implications of certain outcomes are resulting from the classifier predicting a false positive or true negative classification.

Table 4.1: Risk Matrix assessing the risks associated to false negative classifications of baboons.

	Negligible	Marginal	Critical	Catastrophic
Certain	🟧	🟧	🟥	🟥
Likely	🟨 Baboon raids bins	🟧	🟧	🟥
Possible	🟩 Baboon raids household	🟨	🟧	🟥
Unlikely	🟩 Baboon injured by vehicle	🟩 Baboon killed by residents	🟨 Person mauled by baboon	🟥 Group of persons mauled by baboons
Rare	🟩	🟩	🟨	🟧

Table 4.2: Risk severity associating the colours in the above table 4.1 to their respective levels of risk severity.

Risk Severity	Low	Moderate	High	Extreme
Colour	Green	Yellow	Orange	Red

Table 4.2 above represent the resulting severity of a case where a baboon is not classified correctly and a consequence thereof may arise.

The case where a Group of persons is mauled by baboons is highly unlikely to happen in the region under investigation. This case falls into the extreme risk category, however the risk can be ignored due to the unlikelihood of the occurrence.

As human beings encroach more on the natural habitat of these baboons due to development and population growth, the occurrences where people have contact with baboons will increase. In terms of risk, it must be assumed that human beings are not educated regarding the typical behaviour of baboons. In other words, it is assumed that most human beings do not know how to behave and respond appropriately when baboons are present. An example of such is that an alpha male baboon will attack if he perceives the troop to be in danger or when a baby baboon is threatened. This could happen when a human being finds him/herself between a baby baboon and its mother. Therefore, a false negative classification could lead to severe injuries to human beings. This is however unlikely to happen, but the risk is worth noting.

With that being said, a classifier could classify many images of a baboon troop at any given time, due to the fact that baboons travel slowly and predominantly in large troops. It could therefore be assumed that at least one image will be classified correctly when baboons are passing through an area. This means that the moderate risks associated to table 4.2 will most likely not occur as the classifier is expected to have a high chance of correctly classifying a baboon in a passing troop. This will lead to the rangers in the area being notified and ultimately the risk being avoided.

5 Conclusion

The objective of this project was to determine whether current technology is capable of solving the real-life baboon classification problem faced in the Overberg region, South Africa. The following steps were taken in order to achieve this objective.

Three hardware configurations were considered for the classification system. The RPI 4 8GB microcontroller with the RPI V2 camera module was chosen to be the best option for this system. This decision was made due to its low cost, high performance and support through the Raspberry Pi community. It was however seen that if more demanding computational tasks need to be implemented at a later stage for the development of this system, the NVIDIA Jetson might be a better option due to its better performance when compared to the RPI 4 8GB microcontroller.

MATLAB, the software platform that was chosen for the development of the classifier proved to be a successful option. MATLAB is a widely used platform in research and free to Stellenbosch University students and therefore was a suitable tool for this project during its conceptual phase. MATLAB will assist with testing and validating the classifier. MATLAB supports RPI hardware and the migration of the classifier onto the RPI was effective.

ResNet50 was chosen to be the CNN used for the system as it had already been trained to classify the baboon class. ResNet50 was further supported on the MATLAB platform which made it suitable for building a baboon classifier from.

In order to prove that this classifier system was viable for further development and commercialisation, the concept needed to be tested in the real world by testing certain design requirements. The system was tested for robustness, performance under various weather conditions and inference time.

Analysis on the data obtained from the field tests were insightful and successfully achieved the goal of testing the classifier system against the design requirements. This was achieved by conducting field tests and setting up an experimental design, which proved that the classification system was in fact able to meet the design requirements stipulated in the report.

It was evident that the classifier was able to perform well and accurately classify a baboon when the baboon was within 10 meters of the camera. The reason for this was that the environment introduced a significant amount of noise to the images when a baboon was further than 10 meters away from the camera. This could have been improved by introducing object detection as a first step to the classification process.

Given different weather conditions the classifier still performed well within a 10 meter range. It was seen that an image with different lighting conditions significantly affected the performance of the classifier. This was due to the weather changing during the field tests and allowing for the system to be tested under various weather conditions. The classifier performed best when there was good quality lighting and most poorly during overcast conditions.

It was found that the inference time for classification was never longer than 9 seconds. This was said to be sufficient given the maximum allowable inference time of 10 seconds, which was argued for the application of the system.

Field tests were successfully completed that determined the accuracy of the classifier given various environments. The classifier was performed best when a baboon was in a manmade region such as the side of the road. This suggests that if the system gets implemented at a later stage, it should be placed in an area with manmade surfaces such as at the entrance to a trail or the side of the road. One of the main limitations of the system was its ability to classify baboons in areas where they were well camouflaged. The classifier was unable to detect a baboon when a baboon was present on a rock. This was due to the baboon being extremely well camouflaged in its natural environment. This could be improved by training the CNN further from this new data.

Comparing true positive and true negative classifications, the probability threshold was chosen to be 20% probability. This means that there is a 50% certainty that a classified image will detect the presence of a baboon in an image when the baboon class probability is more than 20%.

The classification system would correctly classify at least one image of a baboon when a troop passes by as the classification system would capture many images of that baboon troop. It can therefore be assumed that people would be notified when baboons pass the classifier system, as it would be able to classify a true positive classification out of a number of images taken at any given time while baboon/s are in the area. A risk matrix was compiled to determine the severity of a risk when a baboon could not be identified which could lead to potential negative outcomes. This could include a group of persons getting mauled by baboons. Despite the risk having a low likelihood of occurrence, the risk was further reduced by deciding on a low probability threshold of 20%.

Considering the system as a whole, it is positive to see that the conceptual system was able to prove that current technology is capable of solving the real-life baboon classification problem which was the objective of the project.

6 List of References

Prijatna, D. et al., 2018. A Study of Light Level Effect on the Accuracy of Image Processing-based Tomato Grading. *Earth Environ,* p. 147.

Bird, J. J. & Faria, R. D., 2018. *A study on CNN Transfer Learning for Image Classification,* Birmingham: School of Engineering and Applied Science Aston University, Birmingham, B4 7ET, UK.

Bajenescu, 2018. PROGRESS OF ARTIFICIAL INTELLIGENCE (AI). *Journal of Engineering Science,* 25(2), pp. 57-64.

Cheng, F., Zhang, H. & Harris, B., 2018. Image Recognition Technology Based on Deep Learning. *Springer Link,* Volume 102, pp. 1917-1933.

Schofield, D. et al., 2019. Chimpanzee face recognition from videos in the wild using deep learning. *Science,* Volume 5.

Favorskava & Pakhirka, 2019. Animal Species Recognition In The Wildlife Based In Muzzle And Shape Features Using Joint CNN Shape Features Using Joint CNN. *Science Direct,* Volume 159, pp. 933-942.

Glorot, X. & Bengio, Y., 2010. Understanding the difficulty of training deep forward neural networks. *JMLR,* Volume 9, pp. 249-256.

Kuo, 2016. Understanding convolutional neural networks with a mathematical model. *Journal of Visual Communication and Image Representation,* Volume 41, pp. 406-413.

Liang, J., Wenzhao, F., Chen, C. & Junguo, Z., 2010. *Neural architecture search based on model pool for wildlife identification.* [Online]
Available at: https://doi.org/10.1016/j.neucom.2020.03.035
[Accessed 20 April 2020].

Liss, A., 2020. *Implementation Of a Neural Network On A Microcontroller For Recognition Of Warning Signals,* s.l.: APPSALA University.

Marais, J., 2018. *Automated Elephant Detection And Classification from Aerial Infrared and Colour Images Using Deep Learning,* Stellenbosch: Stellenbosch University.

Norouzzadeh, et al., 2018. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences,* 115(25), pp. E5716-E5725.

Pons, P., Jaen, J. & Catamala, A., 2017. Assessing machine learning classifiers for the detection of animals. *Expert Systems with Applications,* Volume 86, pp. 235-246.

Rumerhart, D., Widrow, B. & Lehr, M., 1994. The baisc ideas of netural networks. *Communications of the ACM,* 37(3), p. 87+.

Wu, X., Sahoo, D. & Hoi, S., 2020. Recent advances in deep learning for object detection. *Neurocomputing,* Volume 396, pp. 39-64.

Zhou, D., 2020. Theory of deep convolutional neural networks. *International Neural Network Society, European Neural Network Society & Japanese Neural Network Scoiety,* pp. 319-327.

Muhammad, K., 2010. *An Olive baboon (Papio anubis) in the Ngorongoro Conservation Area in Tanzania.* Tanzania: Wikipedia.

Roos, M., 2019. *Deep Learning Neurons versus Biological Neurons.* s.l.:Towards data science.

Appendix A Techno-Economic Analysis

The techno-economic analyses covers the project budget, time management, technical impact, return on investment as well as potential for commercialisation.

A.1 Time management and Gantt chart

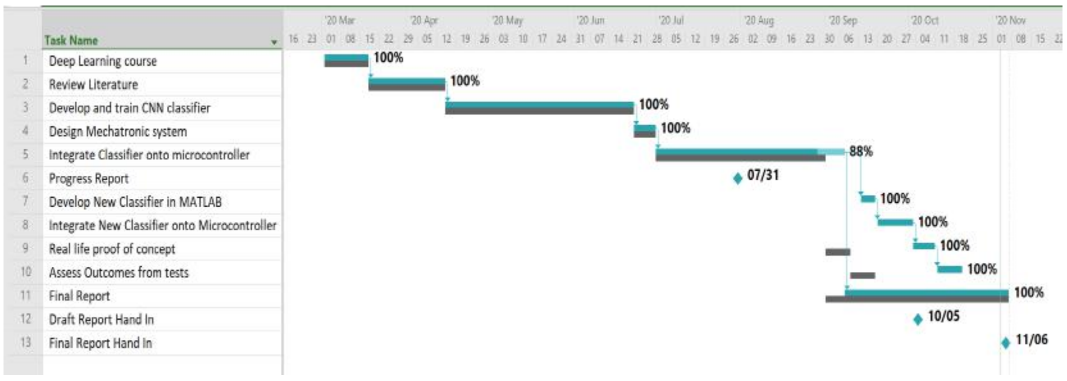


Figure A 1.1: Project Gant Chart

Figure A.2.1 above depicts the estimated project activity duration (grey) compared to the actual project activity duration (blue). The deep learning course along with the literature review were completed during the first semester as estimated. The training of a new classifier went well and as estimated. Most activities were completed successfully and on time with the exception of task 5 as discussed below.

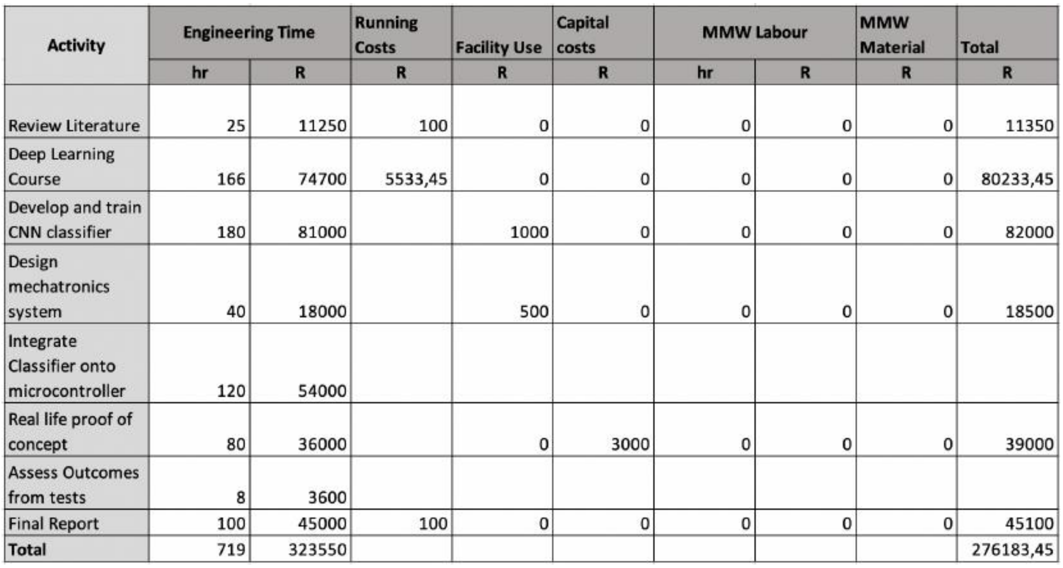
Task 5 proved time consuming and inherited some challenges steered the project in a slightly different direction but keeping to the main project objective.

Task 5 was not completed and instead, MATLAB was used to develop a new classifier (Task 7). and thereafter implement this onto the mechatronic unit (Task 8). Due to these additional tasks, the real like proof of concept was delayed as it is can only be done once task 8 was complete. Similarly, task 10 was pushed forward but still completed successfully.

In order to keep track of the project and the deadline, project milestones were implemented. Furthermore, the project report was compiled while the project was under way in order to be reach the final deadline.

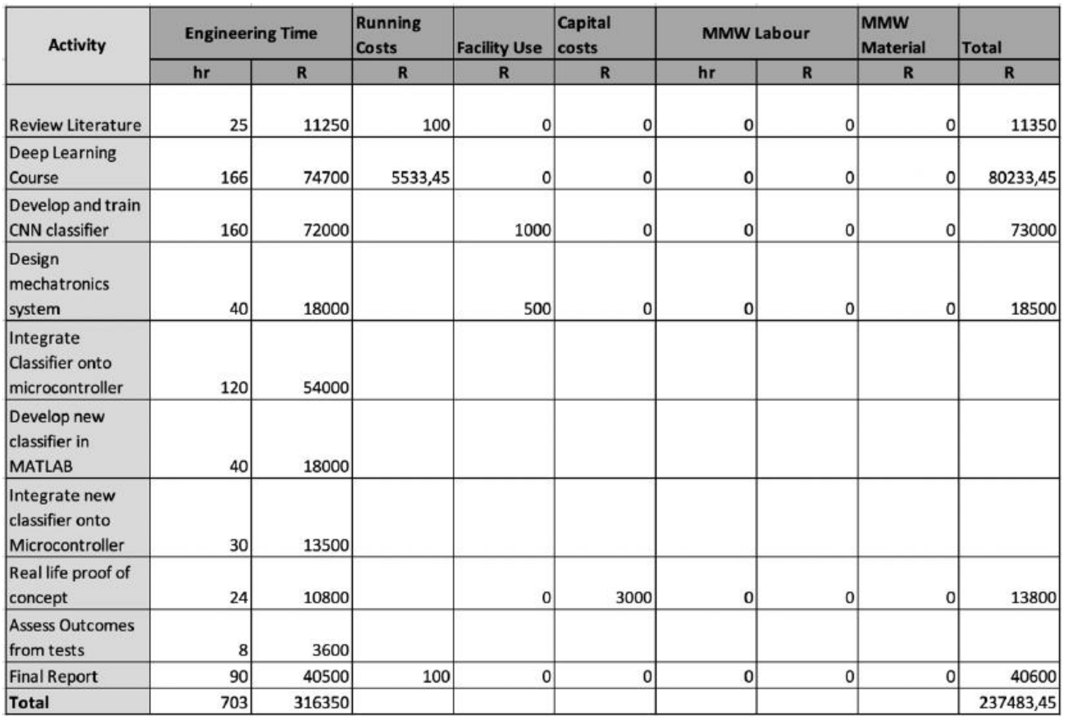
A.2 Budget

The proposed budget and project cost given in figure A2.1 and A2.2 respectfully.



Activity	Engineering Time		Running Costs	Facility Use	Capital costs	MMW Labour		MMW Material	Total
	hr	R	R	R	R	hr	R	R	R
Review Literature	25	11250	100	0	0	0	0	0	11350
Deep Learning Course	166	74700	5533,45	0	0	0	0	0	80233,45
Develop and train CNN classifier	180	81000		1000	0	0	0	0	82000
Design mechatronics system	40	18000		500	0	0	0	0	18500
Integrate Classifier onto microcontroller	120	54000							
Real life proof of concept	80	36000		0	3000	0	0	0	39000
Assess Outcomes from tests	8	3600							
Final Report	100	45000	100	0	0	0	0	0	45100
Total	719	323550							276183,45

Figure A2. 1: Initial Budget



Activity	Engineering Time		Running Costs	Facility Use	Capital costs	MMW Labour		MMW Material	Total
	hr	R	R	R	R	hr	R	R	R
Review Literature	25	11250	100	0	0	0	0	0	11350
Deep Learning Course	166	74700	5533,45	0	0	0	0	0	80233,45
Develop and train CNN classifier	160	72000		1000	0	0	0	0	73000
Design mechatronics system	40	18000		500	0	0	0	0	18500
Integrate Classifier onto microcontroller	120	54000							
Develop new classifier in MATLAB	40	18000							
Integrate new classifier onto Microcontroller	30	13500							
Real life proof of concept	24	10800		0	3000	0	0	0	13800
Assess Outcomes from tests	8	3600							
Final Report	90	40500	100	0	0	0	0	0	40600
Total	703	316350							237483,45

Figure A2. 2: Final Cost

The final cost of the project was R38700.00 under budget. This was largely due to the real life proof of concepts taking much shorter than expected. This was achieved by contacting relevant wildlife professionals and rangers which directed field tests towards baboons locations. Using MATLAB largely reduced costs that could have been incurred as it is free to use by University staff and students. The main cost incurred on the project is engineering time which costs a rate of R450 per hour.

A.3 Technical Impact

This project has given great insight into the capabilities of modern image classifiers such as ResNet50 which was used in this project. This project contributes into the greater field of machine learning by determining testing the performance of existing technologies (CNN) and stipulating where further development is required for this technology to be commercialised.

The technology proved to be useful in the real world environment which could lead to a largely positive impact on nature conservation and our natural environment. The correct implementation of this technology can be used to further protect and gain a better understanding of wild animals who are inherently hard to track. This is because this technology can be developed and used on other animal species apart from baboons.

The technical value of this project thus warrants the financial unput as it draws insightful conclusions that have not been thoroughly research regarding baboons and exiting image classifiers.

A.4 Return of Investment

The time and money spent on this project was useful in determining whether existing technology (machine learning based classifiers) can classify baboons in their natural environment using edge hardware such as a microcontroller in real time. The project draws useful conclusions that forms a solid foundation from where further development can continue from.

Further development in this field of focus would not require and additional hardware as the hardware used has been proven to be able to successfully run real time inference. Furthering research in this field would be beneficial compared to the required development still needed for this technology to be ready for commercialisation.

A.5 Potential for Commercialization

South Africa is rich in wildlife. This technology would be beneficial to nature conservation if further developed. It can be used for tracking animals as well as protecting endangered animals from concerns such as poaching. Poaching is a major challenge in South Africa and the positive impact that this technology can bring to the table would be greatly beneficial and attractive to commercial and private markets.

Appendix B Safety Report

This section summarises the safety report compiled for the experimental design.

Table B 1: Emergency contacts

Contact	Work nr.	Email address
Cape Nature	(028) 314 0062	reservation.alert@capenature.co.za
Human Wildlife Solutions	(072) 028 0008	info@hwsolutions.co.za
Emergency contact (Robey de Klerk)	(082) 786 4354	robey.deklerk@gmail.com

Although this experiment was conveyed in nature in areas where pedestrians were allowed to walk, there are still inherent dangers and risks associated to being in the areas.

Refer to section 3 for an outline of the experiment and the people involved. The following general safety rules apply.

The following general safety instructions are applicable:

- Wear closed shoes while in nature
- Wear long pants while in nature
- Keep safety report available at all times while conducting experiments
- Plan evacuation procedure in case of emergency
- If uncertain, ask or call for assistance
- Do not interact with wild animals
- Do not litter or feed wild animals

An activity-based risk assessment was followed as seen in Table B followed by the risk types below:

- P Personal
- E Equipment

Table B 2: Activity-based risk assessment

Activity	Risk	Risk Type	Mitigating steps
Tracking Baboons by vehicle	Motor vehicle collision	P	1. Be aware of surroundings. 2. Ensure separate person tracking and driving vehicle.
Tracking baboons by foot	Body injuries	P	1. Be aware of your surroundings. 2. Keep to the track or dedicated walk ways. 3. Wear closed shoes and reflective clothing to deter animals or be visible to motor vehicles. 4. Remain close to rangers and equip yourself with knowledge needed regarding animal behaviour
	Lost/damaged equipment	E	1. Ensure all equipment is stewed away in a bag when not in use. 2. Run through checklist of equipment used before leaving a site.
Setting Up Unit for data capture	Damage to individual and equipment	P & E	1. Remain safe distance from baboons or wild animals 2. Keep equipment at a safe distance from animals

Declaration

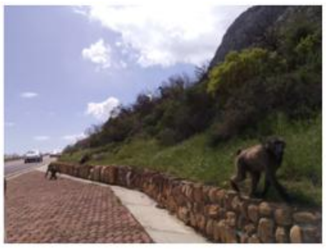
I, the undersigned, declare that I have read the Safety Report and understand the risks involved in this project and working in Nature conveying tests.

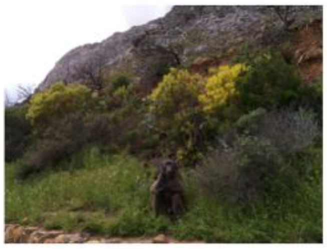
[illegible]
____

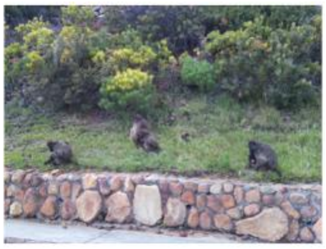
DS de Klerk

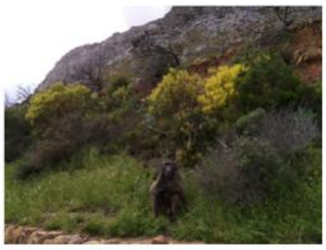
Appendix C Data obtained from field tests

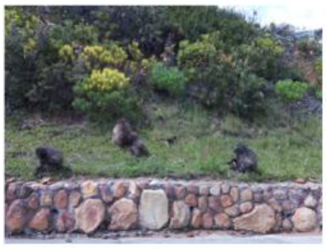
C.1 Top 10 Images that Include Baboons and Their Classification Probabilities

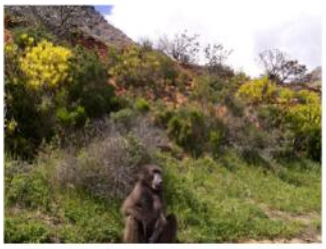

92.61%

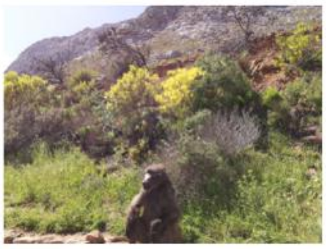

93.12%

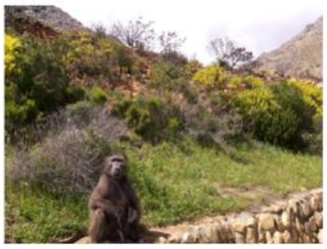

95.17%

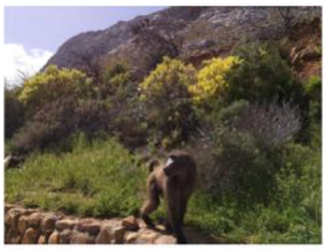

93.11%

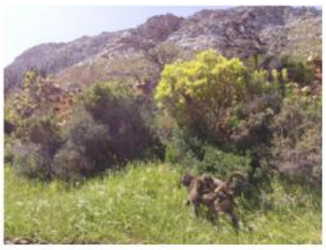

79.29%

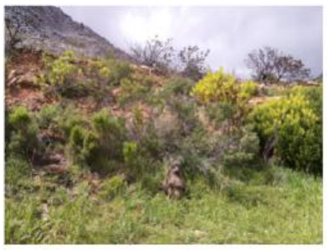

85.19%

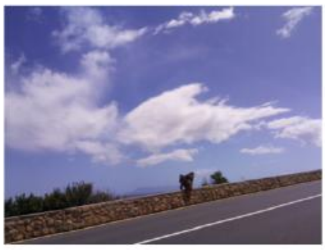

96.65%


48.71%


36.54%


21.05%


3.731%


0.0083%

C.2 Top 10 Images that Do Not Include Baboons and Their Classification Probabilities

11.92%

6.107%

4.368%

4.008%

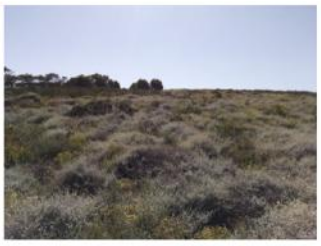
3.998%

3.572%

3.492%

3.313%

2.898%


2.893%

2.023%

1.082%