FACULTY OF SCIENCE & TECHNOLOGY

BSc (Hons) [Degree Title]

May 2021

What's Wrong With My Crop? Using Convolutional Neural Networks to Detect Crop Defects

by

Ryan Syme

Bournemouth University, Department of Computing and Informatics, Final Year Project

Faculty of Science & Technology

Department of Computing and Informatics

Final Year Project

Abstract

[The text within the square brackets must be deleted along with the square brackets when finalising your own abstract.

The abstract for an undergraduate dissertation should be between 200 - 350 words.

Arial, Normal, 11pt with 1.2 or 1.5 line spacing should be used. The text in this part has 1.5 line spacing.

An abstract is a brief, accurate and comprehensive summary of the entire dissertation. It is the first thing to be read by your examiners to help them know the brief content of the dissertation. It also serves as a "sales pitch" to form the first impression of your work.

A good abstract should be accurate, self-contained, concise, specific and clear. A quick way to assess the quality of your abstract is to check whether it answers the questions why, how, what and so what.

Researching the efficacy of using CNN's (Convolutional neural networks to identify crop defects) and creating a suitable platform for users to interact with the network.

It is easier to write the Abstract the last.]

Dissertation Declaration

[The text within the square brackets must be deleted along with the square brackets when finalising your declaration.

Note if your project is CONFIDENTIAL because of your client, you will need to adapt this declaration based on the agreement between you and your client accordingly. Do not forget to state the name of your client clearly. You must contact and inform Project Coordinator if your project is CONFIDENTIAL.]

I agree that, should the University wish to retain it for reference purposes, a copy of my dissertation may be held by Bournemouth University normally for a period of 3 academic years. I understand that once the retention period has expired my dissertation will be destroyed.

Confidentiality

I confirm that this dissertation does not contain information of a commercial or confidential nature or include personal information other than that which would normally be in the public domain unless the relevant permissions have been obtained. In particular any information which identifies a particular individual's religious or political beliefs, information relating to their health, ethnicity, criminal history or sex life has been anonymised unless permission has been granted for its publication from the person to whom it relates.

Copyright

The copyright for this dissertation remains with me.

Requests for Information

I agree that this dissertation may be made available as the result of a request for information under the Freedom of Information Act.

Signed: _____

Name: [Your name]

Date: [Date of signing this declaration]

Programme: [Your degree title]

Original Work Declaration

This dissertation and the project that it is based on are my own work, except where stated, in accordance with University regulations.

Signed: _____

Name: [Your name]

Date: [Date of signing this declaration]

Acknowledgements

[The text within the square brackets must be deleted along with the square brackets when finalising your own acknowledgements.

Arial, Normal, 11pt with 1.2 or 1.5 line spacing should be used. The text in this part has 1.5 line spacing.

This is your opportunity to mention individuals who have been particularly helpful. Reading the acknowledgements in the past dissertations in the project library will give you an idea of the ways in which different kinds of help have been appreciated and mentioned.]

Contents

List of Figures

List of Tables

Chapter 1 - Background

1.1 Problem Definition

As it stands there are currently (09/02/2021) No results present in the first 3 pages of a google search for 'crop defect identification' and 'What's wrong with my crop' that show web interfaces for interacting with a crop defect identification service.

Although there exists very capable, publicly available image classification networks (Yan 2021). There is no bespoke application catering soley to crop defect identification, that has the benefit of providing recourse and prevention information to the user.

1.2 Proposed Solution

To provide a web service that interacts with a convolutional neural network (CNN) back-end[1] to diagnose crop defects such as, nurturing problems e.g. lack of water/nitrogen/C02, too hot/cold and external threats such as crop disease/pest infestation. The interface will be simple and intuitive as possible. The User Interface (UI) should minimise points of interaction and streamline the process of uploading a crop image to be analysed. The web service will return information regarding the percentage likelihood of each kind of crop defect, including images that are of similar nature to the one analysed.

1.3 Aims & Objectives

These should be SMART with clear success criteria defined specific, measurable, achievable, realistic, Timebound

1.3.1 Aims

- To aid gardneners and smallholders in identifying crop defects.
- To aid gardeners and smallholders in taking relevant recourse.

[1] see Development Methodology

1.3.2 Objectives

- Provide a way for a user to upload an image to be analysed.
- Display information regarding the likelihood of each kind of defect.
- Display recourse information alongside defect information.
- Have gallery of images filtered by crop and disease type.

1.4 Risk Table

ID	Name	Likelihood	Impact	Control Mechanisms
1	Improper time management	med/low	high	Follow the Gannt chart
2	HDD/storage failure	low	high	All worked backed up to Github
3	Illness/Injury	med	med	Apply for extention of necessary
4	RSI	med	low	Set up workstation correctly
5	Eye strain	med	low	Ensure room is well lit
6	Incorrect task prioritisation	med	med	Iteratively asses work being done.
7	Postural problems	med	low	Set up workstation correctly

1.5 Overview

Firstly the introduction will provide additional context to the problem domain and show relevant research. Next, the methododology and justification for the choice of methods will explored. Including employed technologies, project management methodoglogies and development methodologies.

Chapter 2 - Introduction

2.1 Context

With the increased availability of smartphones (Sta 2021), digital cameras (Ima 2021) and Internet access (Wik 2021) (Glo 2021), coupled with the increased interest in home food cultivation (Goo 2021) and the large number of people reliant on food grown in smallholdings (Walpole and Hutton 2013); the ability to identify defects with crops using technology has potential to be impactful to many people.

2.2 Literature Review

Firstly one will cover general development practice, Secondly, existing research regarding image classification of plants and plant diseases will be explored. Lastly, an analysis of CNN architecture development.

2.2.1 General Software Development Practice

Software, often complex by nature, benefits from being created with a set of best practices in mind. A key principle one has gleaned from the Pragmatic Programmer (Hunt and Thomas 2000) (a key reference guide that has helped one's understanding of best practice) is that of developing for orthogonality. This means creating software from sub modules that operate as independently as possible from the whole. This means that changes to part of the system do not cascasde to unexpected effects elesewhere in the sytem. In practice this involves reducing, or if possible, entirely eliminating the usage of global variables. As this can make it difficult to trace the source of bugs when one does not know which part of the system is modifying the global. It also forces any changes to the modules that interact with the global to be considering the knock on effects it may have to other modules. Additionally it makes switching the code to a multi-threaded design much more difficult as bugs created by 'race conditions' can arrise, whereby both threads of execution only modify the variable in the correct order due to difference in their speed of execution.

Another way of creating orthogonal code involves making interfaces store as little information about their clients as possible, as this allows for the client to be changed more readily without then needing to alter the internal logic of anything it is interfacing with. It also means the interface is able

to be consumed by any client. How to solve it (POLYA 1945) Also reinforces the modular aprach to development by encouriging the problem solver to break apart large problems in to smaller more manageable problems that one knows how to solve, eventually, reaching a point where the known unknown becomes a smaller and smaller peice of the problem until the remaining steps of the solution are trivial or at the very least clearly defined.

Another useful development practice outlined in 'The Pragmatic Programmer', is that of 'tracer-code'. This is much the same idea as opting to acheive a Minimum Viable Product (MVP) to provide a framework to build additional features from. For instance, creating simple client and server applications that are successffully interfacing with one another to send and respond with simple data, can then be expanded to include the capacity to send more complex data and for the server to perform ever more complex operations on said data. Once a basic outline of the software is constructed it can also be extended to include, better response to erronus data and provide a greater number of features. This line of development whereby one creates a simple implementation, goes back to the drawing board, adds new features, and repeats is reinforced by Gall's Law which states "A complex system that works is invariably found to have evolved from a simple system that worked. A complex system designed from scratch never works and cannot be patched up to make it work. You have to start over with a working simple system" (Gall 1977)

Development of software is iterative and cyclical. Involving the creation of features to a set of requirments, and then subsequetly refactoring their implementation to better align with best practices. Some reasons code may need to be refactored as covered in (Hunt and Thomas 2000) are; Quote "Duplication", "Nonorthogonal design", "Outdated Knowledge" and "Performance". As it is the case with software that it can be configured in infinitely many ways to acheive the same outcome. It is inevitable that sometimes features will be implemented in a way that breaches best practices. Or in a way that is not conducive to the overall 'health' of the codebase. A useful analogy shown in the book is to think of the codebase as a garden, that will grow and change over time, with refactoring (see Refactoring) being analagous to weeding and pruning and feature addition being analagous to planting.

SOLID principles

When writing Object Oriented (OO) code. Robert. C. Martin outlined many principles to follow when designing software architecture, architecture in this case refering to the level of interaction between individual classes. From these principles a subset of these principles was selected by Michael Feathers in 2004 to create the SOLID acronym.

S - Single Responsibility Again we see a focus on modularising and de-coupling parts of the software. Making classes that where possible contribute to a single purpose. This makes the class easier to test, and have by virtue of having less functionality, have less classes dependent upon it.

O - Open to Extention, Closed to Modification When new funtionality is to be added to the software, it should be the case that the existing classes are not modified, but inherited from and extended.

L - Liskovs Substitution Principle Tying in to the above principle. If a parent class is changed for a sub-class, the behaviour of the program should not be affected. For instance, one has a parent class Mammal, and a child class Dog. If the program executes properly when it recieves a mammal object, it should still operate properly when receiving a dog object as the dog object should implement the same contract as stipulated in the Mammal interface.

I - Interface Segregation Large interfaces should be made in to seperate smaller interfaces, therefore inheriting classes do not have to inherit methods they will not use.

D - Dependency Inversion This aims to to prevent high-level modules depending on lower level modules. For instance, rather than declaring an instance of class B inside another class A, one should declare an instance of the parent class of class of B. This way, class A is not dependent on the lower level of abstraction.

Rules Of Thumb

Don't repeat yourself (DRY) cited from (Hunt and Thomas 2000). This means creating a single source of truth for information in the system. However a possible pitfall of this aproach is to prematurely astract functionality. For example if two methods contain the same sequence of operations one may make this sequence a seperate method. However later in the development process it becomes apparent that in one of the original methods the sequence must be slightly different based on a condition. Now the new abstracted method must contain branching logic and the code has had uncececarry complexity added, as now branching logic is present at two places in the code, when the decicion could have remained encapsulated inside only the method that needed it.

You Aren't Going To Need It And you aren't going to need it (YAGNI) don't create functionality or levels of abstratction that you don't need. The principle being that it creates unecessarry complexity (Hunt and Thomas 2000).

Avoid Premature Optimization in the words of (Knuth 1974) “premature abstraction is the root of all evil”. This is due to the fact optimizing code often comes with the drawback of making the code more complex and harder to understand, it can also be the case that time is spent optimizing parts of the code that aren’t the real bottlenecks to it’s performance.

2.2.2 Disease Detection Using Machine Vision

Non CNN methods

In 2009 a study (Anthonys and Wickramarachchi 2009) was conducted using deep learning to identify three different disease classes on rice plants. The results showed over 70% classification accuracy on 50 sample images. The method employed used segmentation, followed by image feature extraction using three different algorithms to extract color, shape and texture information from the image. The feature data was input to a classification algorithm whereby the output would be one of the three disease types or no disease present.

In 2015 another experiment (using similar aproach to (Anthonys and Wickramarachchi 2009)). Involved using image segmentation using using K means clustering and other image processing techniques, to find features in the image and create a one dimensional binary feature vector, to be processed by an ANN. (Khirade and Patil 2015) The accuracy of detecting powdery mildew, yellow rust and aphids on wheat were 86.5%, 85.2%, 91.6% and 93.5% respectively[4 vals 3 catagories].

This non CNN technique has subsequently been rendered redundant as this method requires a greater number of computational processes and acheives results that have been surpassed by CNN’s. However, the image segmentation technique (with the purpose of isolating the leaf from the background) Is also sometimes used in CNN aproaches.

CNN Methods

A year later (Mohanty *et al.* 2016) there have been great successes in identifying crop disease with CNN’s. This 2016 paper ran experiments on a 38 class crop disease dataset over 14 crop species and 26 diseases (or absense thereof). Resulting in 99.35% accuracy on a held-out test set (using GoogLeNet). This study utilised two established CNN architectures, namely AlexNet (Krizhevsky *et al.* 2012) & GoogLeNet. (Szegedy *et al.* 2015a) with GoogLeNet achieving a higher F1 score[1] in almost all cases. This study also highlighted the effectiveness of using colour images when training the models. In all experiemnts, the color or segmented image models performed better as appose to grey scale images. A suprising aspect of the results is the fact that the segmented image models almost always performed worse than the colour image models, with

[1] see Evaluation methods

the best performing model being trained on colour, non-segmented images. This may be due to some bias being present in backgrounds of the dataset images. Or it may be more effective to not perform segmentation on the images prior to training.

In 2018 InceptionNetV3 (a later iteration on the GoogLeNet i.e. InceptionNetV1 architecture) is used on a very similar if not the same dataset of 38 class crop diseases (this paper citeps the number of crop species to be 13 appose to 14) and 26 diseases. This resulted in a slight increase in classification accuracy of 0.39%, to 99.74%. (Kulkarni 2018).

Prior to training the models, the training images were segmented to give the crop leaves a black background. Notably, this study began with pre-trained InceptionV3 models and fine tuned them by training a seperate model for each type of crop. This allowed a system whereby the network is fed an image, it determines the crop, then it passes the image to the specific network tailored to that crop species. Unfortunately, there are no results available of experiments with non-segmented data to compare with the Mohanty paper. Interestingly, the author (Omkar Kulkarni) states 'The pre-processing of image is essential for removing noise and segmentation of the image which helps in improving the accuracy of CNN model'. However, the results table [APPENDIX LINK] produced by (Mohanty *et al.* 2016) show non-segmented images acheiving higher accuracy. The increase in accuracy for this paper when comapared to (Mohanty *et al.* 2016) could be explained by the better performing InceptionNet architecture

A study performed in 2015 by Sungbin Choi (Choi) which involved plant species identification from a multi-image observation query, found that an ensemble of CNN's performed with better classification performance. The study utilised an ensemble of fine-tuned[2] GoogLeNet architectures.

A paper (Zhu *et al.* 2018) performed experiments for plant species identification and justified that 'using CNN's can provide better feature representation compared to hand-crafted features.'

From the reviewed sources it is aparent that the best performing architectures have employed the Inception (Szegedy *et al.* 2015b) module. which is consistent with the findings of (Wu *et al.* 2019) for InceptionNets to outperform Resnets. (He *et al.*) and InceptionNet varieties the Inception-ResNet-v2 was the best performing well known architecture. However the researchers crafted a bespoke architecture using inception modules that slightly outperformed the Inception-Resnet-v2.

[2]meaning pre-trained on generalized data and then improved with domain specific data

2.2.3 Existing CNN Architectures

LeNet5 & Precursor

One of the first major pioneering works in the field of machine vision has been (Cun *et al.* 1989) which designed a CNN architecture to be used for character recognition and printed on a chip. With all of it's 49 templates (today known as filters) being designed by hand. This later goes on to to introduce 'Digit Recognition Using Constrained Automatic Learning' which allows some parameters of the network, including filter values to be updated automatically via backpropogation, removing the problem of having to design all of the filters manually. Manually designing filters can be seen as introducing some *priori* information to the network. Which also aleviates the problem of having a scarce amount of data. A noteworthy step when training their network is the final stages of training whereby they waited for their model to converge at a minima (minimizing loss, i.e. error) which took 23 learning passes, then trained it for a further 5 passes on a dataset that quote "had undergone slightly different preprocessing", resulting finally in a 5% error rate at classifying handwritten digits. Later in the work of creating LeNet5 (LeCun *et al.* 1998) we see the invention of fully automated parameter tuning via backpropogation (for more on backpropogation see (Cun Yann le 1988)) that led to, acheving 0.8% error rate at identifying hand written digits. This was in part made possible by the increased amount of data available to train the network.

AlexNet

Following from LeCun we see the next popular architecture (winner of the 2011 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)) that is often used as a benchmark in later papers AlexNet(Krizhevsky *et al.* 2012). A noteworthy aspect of it's design is the choice to change from using the tanh activation function that we find in (LeCun *et al.* 1998) to the relu activation function, which has been adopted in all state of the art aproaches today. The relu funciton is linear so we see a decrease in training time by virtue of using a function that is less computationally expensive. Another finding was that due to the large number of parameters in the network, it had a tendency to overfit the training set. To mitigate this, data augmentation is used including reflections and zooming in on only parts of the image such as corners/middle. Other augmentation included normalizing the RGB intensity across the image as it stated that 'object identity is invariant to changes in the intensity and color of the illumination'. This network also employs dropout as discussed later. When training the model, stochastic gradient descent is used, whereby the parameters of the network are updated based on the loss of each training example. We also see mention of the concept of 'fine tuning' a network, whereby one takes a network that is pre-trained on a generalized dataset and then further trains it on domain or dataset specific input. Furthermore, we see evidence for

the virtue of using ensembles of CNN's to reduce classification error. Finally the paper concludes stressing the importance of depth for classification accuracy citing a 2% reduction in classification accuracy for top-1 performance if any convolutional layers are removed. However, a later paper (Zagoruyko and Komodakis) has established that 'widening' networks can be just as or more effective than deepening. The concept of widening can also be used to explain the effectiveness of InceptionNets.

VGGNet

Another major improvement in CNN architecture comes in the form of VGGNet (Visual Geometry Group) (Simonyan and Zisserman 2015). In this paper they studied a range of different network depths. They take the aproach of further deepening the network to improve accuracy, citing their use of small receptive field filters (3x3) as the main key to their success. This logically follows as the smaller the receptive field, less weights need to be tuned. This fact coupled with using a moderate amount of filter channels (512 at most) allows for more conv layers. Earlier designs were using larger filters especially in the first layer namely (Krizhevsky *et al.* 2012)&(Sermanet *et al.* 2013). To aid in training their deeper network variations, pre trained layers of shallower networks are used as the initial and final layers, with the middle layers remaining randomly initilaized. When training the network mini-batch training is employed whereby the weights of the network are not updated until a batch of training examples has been seen, as apose to stochastic whereby the weights are updated after a single example. Data augmentation is also important in the aproach; each training image was cropped and zoomed etc. Their best performing ensemble of 2 VGG nets acheived 6.8% top-5 test error on the ILSSVRC test classification dataset. VGG much like it's predecessor AlexNet, employs max pooling layers and Relu activation function.

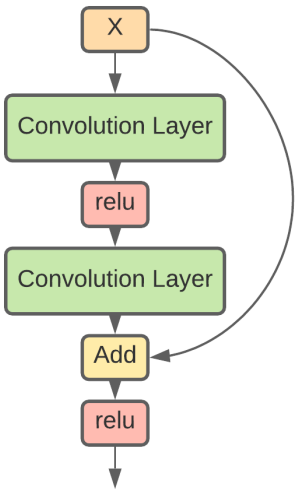


Figure 1: Residual Identity Block

Resnets

Continuing the theme of deepening networks we are introduced to the ResNet (He *et al.*) winner of the ILSVRC 2015 classificaiton task. ResNets introduce the concept of the skip/shortcut/residual connection. Meaning output from an earlier layer is added to the output of a layer some convolutions ahead of it. The emergent result is performance no longer degrading with greater network depths. A reason for this, is that less feature information is lost through the convolution opera-

tions. Due to the fact that every convolution operation reduces the volume of input to the next layer, information is lost after each operation. The skip connection mitigates this by adding earlier information to later layers. With an ensemble of ResNets it's top-5 err on the ILSVRC test set was 3.57%

DenseNets

Taking the residual connection further is the DenseNet. In this architecture we see a stack of dense blocks, in a dense block each layers output is concatenated with each proceeding layer, notably a ResNet adds the output of the previous layer whereas DenseNets concatenate. This has the benefit of preventing the network learning redundant feature maps and preventing the vanishing gradient problem. The network also employs 1x1 convoution layers to reduce input size to the more expensive 3x3 layers, much like the InceptionNet. After each dense block is a transition layer which consists of a 1x1 convolution operation and a 2x2 average pool with stride 2, the transition layer has the purpose of compressing the output volume. It is unusual to see average pooling used as it is typical to see max pooling, and no justification is given for this choice. The authors also claim on the basis that models with over 25m parameters still see increase in accuracy that the DenseNet architecture prevents overfitting. For comparison the original ResNet contains 1.7m parameters.

InceptionNets

An earlier, but more distinctive, when compared to earlier iterations of design, comes in the form of InceptionNet/GoogLeNet (Szegedy *et al.* 2015b). Part of their philosophy when creating the network was to make something that could perform well on hardware that was more widely available quote "the models were designed to keep a computational budget of 1.5 billion multiply-adds at inference time, so that they do not end up to be a purely academic curiosity."(Szegedy *et al.* 2015b). The feature that sets InceptionNet architectures apart is the different varieties of Inception Module. An inception module is multiple convoulutional operations occuring in parallel and finally being concatenated together. This aproach of adding more channels in a single layer is commonly refered to as 'widening' the network. Later iterations of the Inception net (aside from the addition of skip

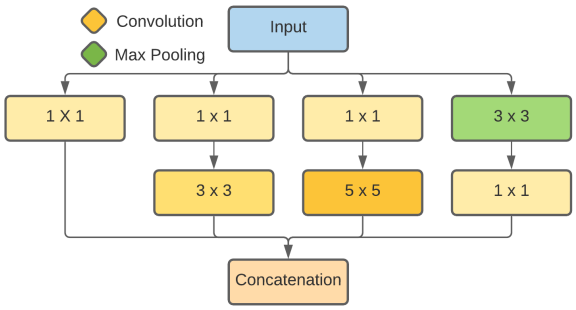


Figure 2: Inception Module Example.

connections) focus on preventing the more expensive computations such as 5x5 convlution operations from bottlenecking the network, this is acheived by using less expensive 1x1 conv operations to reduce the volume of the input prior to the expensive operations.

FractalNet

As it stands, there are few, (if any), papers exploring the efficacy of the novel fractalNet architecture (Larsson *et al.* 2016) for crop disease detection or on any other dataset aside from the commonly used CIFAR10 & 100. The inventors of the fractalNet performed experiments that justified their use over ResNets, demonstrating results that showed improved classification accuracy. Another feature of the fractalNet architecture is the ability for the user to choose between speed of prediction and accuracy of prediciton as it is possible to take longer or shorter paths through the network. Longer paths being more accurate and more time consuming. Drop-out is also applied during training but at a coarser level than individual filters, rather, entire paths are dropped at join layers, leading to the creation of pathways of equally strong predictors. It is said in the paper they see their architecture as a more generic design that is not in it's final form as the best configuration of convulutional module and join layer has not been determined. The architecture can also be seen as a harness for creating an ensemble of networks of different depths.

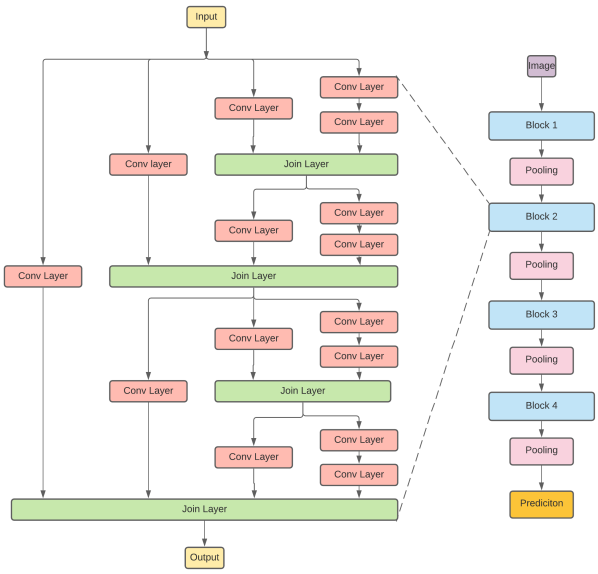


Figure 3: FractalNet Architecture.

Additional CNN research

Research was conducted in 2016 by (Zagoruyko and Komodakis) that determined 'wider' networks perform better than their deep narrow counterparts. They found that their 16-layer deep network had the same accuracy as a 1000 layer thin deep network with a comparable number of parameters. With the wider network being faster to train.

A key feature of CNN design is size and number of filters. In all examples mentioned (Resnet, InceptionNet, AlexNet etc), filter size is always odd and square. For instance 1x1, 3x3, 5x5 and so on. Size of filter will determine the size of the feature the filter will encode for. Number of filters will determine the depth of the convolution output. It is noteworthy that the Resnet50 architecture

(He *et al.*) uses almost entirely 3x3 size filters. Whereas, the Inception module employs a mixture of 1x1 through to 5x5 convoutions.

A technique known as dropout is a feature that has been employed in increasingly deep networks to prevent overfitting and was first introduced by (Srivastava *et al.* 2014). The principle behind its operation is randomly dropping paths between feature activations during training. This ensures that predictions do not become overly reliant on a single (or gorup of) neuronal activation(s), that can correlate to some bias in the training data. And prevents neurons from becoming co-adaptive. This teqnique was employed in AlexNet (Krizhevsky *et al.* 2012) whereby they used dropout in the first two fully connected layers of their model. They found that using dropout prevented overfitting but made training take twice as long. This exact method whereby dropout is used in the first two fully-connected layers is seen again in VGGNet(Simonyan and Zisserman 2015); both methods choosing to employ a dropout chance of 50%. The fully-connectd layers typically occur at the end of the network prior to the final softmax operation that condenses the matrices into a single prediction vector, with each position in the vector containing a prediction value and the position in the vector corresponding to the class.

Another aspect of CNN's is the process by which they progressively narrow down the possibilities until they arrive at their conclusion. This is done by encoding loose abstract forms that relate to groupings of objects in the early layers and gradually arriving at very generalized forms such as horizontal lines or sine waves in the later layers. An example can be drawn from the AI microscope (https://microscope.openai.com/models). In an early layer we may find a neuron that encodes for two semantically unrelated objects, yet objects that have a form in common such as dogs and turnstiles, a middle layer may encode for things under the sea or star shaped objects. Final layers encode for more generic features such as diagonal lines or squares. In the case of the dog and the turnstile, one can observe that a 'branch' of a turnstile is equateable to the leg of a dog and the 'console?' of the turnstile be equateable to the dogs body.

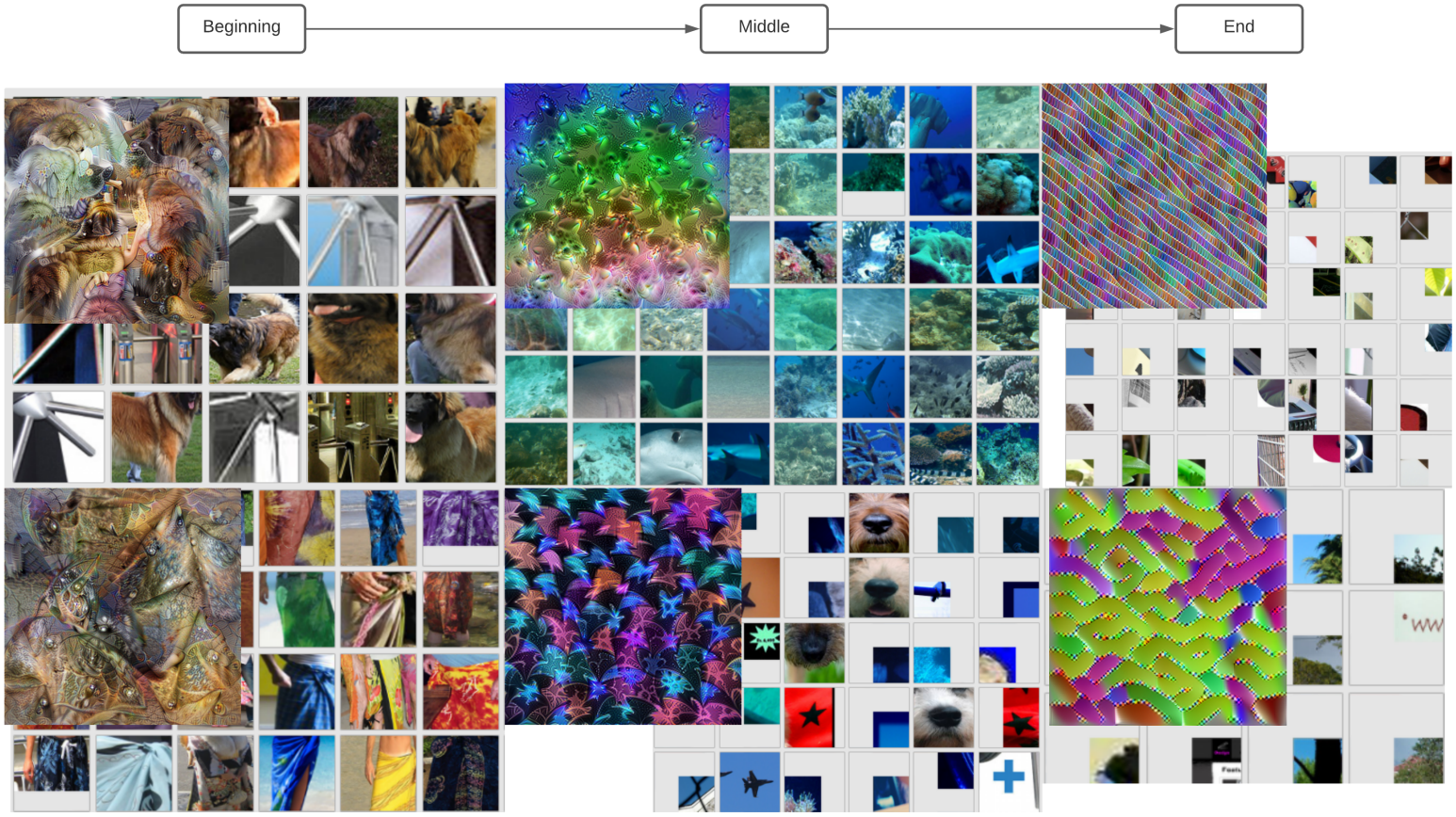


Figure 4: InceptionNet Filter Activation Maps (Examples taken from OpenAI Microscope)

Chapter 3 - Methodology

3.1 Project Management Methodology

3.1.1 Workflow

The workflow one has designed has been inspired by existing 'agile' methods. It will consist of a single cyclical workflow, with two nested "sub workflows" whereby upon completion of a step, it is sometimes necessary to loop back on oneself to perform futher refinement; as illustrated by the diagram below. Throughout the project the focus of the workflow will shift as illustrated by the diagram below.

The paper (Highsmith and Cockburn 2001) makes a compelling argurment for the usage of 'agile' methodologies over more linear project management methodology (PMM) styles such as waterfall. 'agile' focuses on design being done 'on an ongoing bases in smaller chunks' (Highsmith and Cockburn 2001). It is also highlighted that 'agile' allows development to respond to feedback and change. In fact 'agile' has customer, designer and developer feedback baked in to it's formula in the form of regular team meetings, especially if using the SCRUM 'flavour'.

For a lone developer 'agile' with Kanban is the PMM of choice for this project as it allows the flexebility to iterate on designs as one learns more about the technologies being used and gives the freedom to modify one's requirments in light of newly found research and technical limitations. Aditionally, the nature of 'agile' is to have short lifecycles which has the virtue of frequent milestones. This allows one to better track progress of the project by being able to see how many tasks have been completed on the Kanban board during the development cycle[4].

Using Kanban to track tasks makes sense for a single developer as appose to using feature driven development. As this allows for non-programming tasks to be tracked in the same way as programming tasks.

Furthermore, 'agile' is a worthwhile methodology to use and familiarise oneself with as it is becoming the new norm in industry; as shown by a study conducted by Hewlett Packard.

[4]Granted tasks alone are not the be-all, end-all metric but it does give some idea of progress

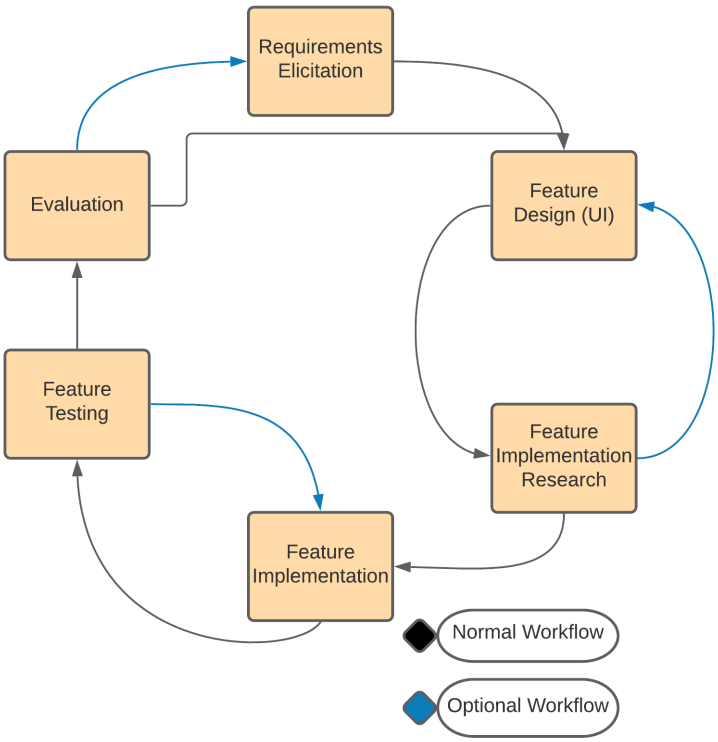


Figure 5: Development Lifecycle

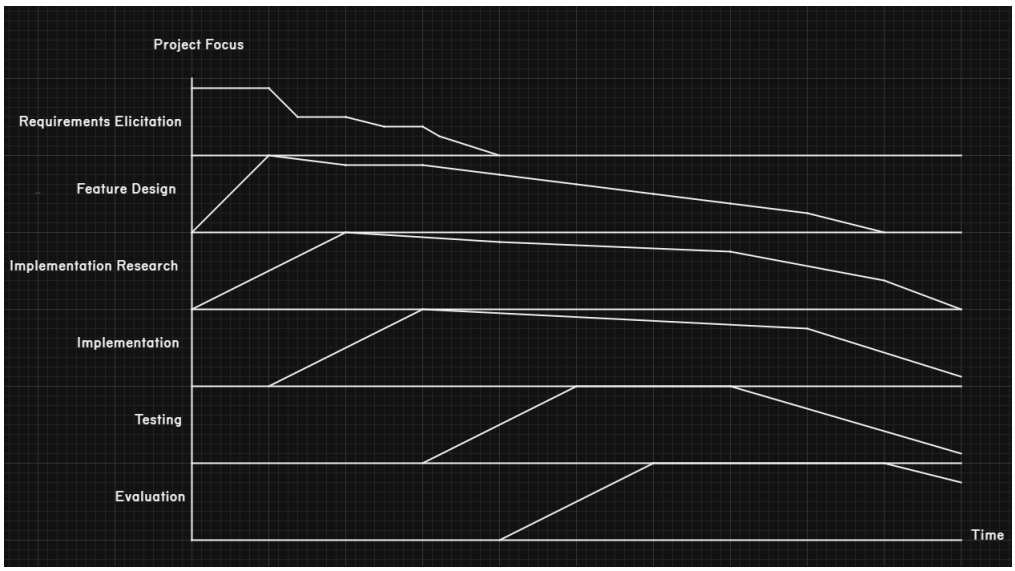


Figure 6: Project Focus Over Time

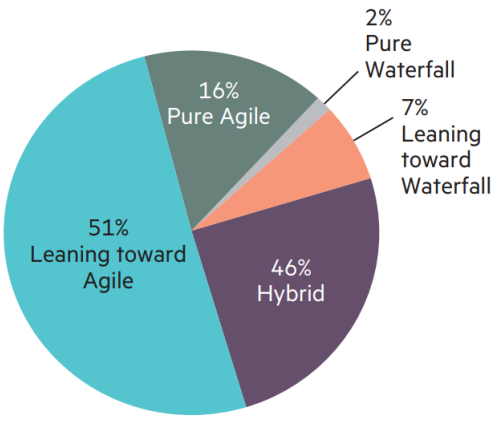


Figure 7: Primary development method used in organization across projects (601 respondents) diagram created by Hewlett Packard

3.1.2 Requirements Elicitation

This involves determening the needs of the user and defining requirements to meet those needs. As the requirements will be determined by the developer in this case, reason being, the scope of the problem domain is narrow in terms of interaction points for the user. The following rationale will be used. One will create requirments that contribute towards providing information about crop defects to the user. Also, as the system is intented for use by non-technical persons, the interaction should be minimal, and clear. This means not cluttering the UI with too many points of interaction, making it clear what the application is doing at any given time. And using non-technical language, especially avoiding jargon.

3.1.3 Feature management

To track the creation and completion of features, a Kanban board will be used. This will include columns for 'To do', 'Doing' and 'Done'. To determine which features will be prioritised one will employ the MOSCOW method (See Requirements).

3.1.4 Evaluation Methods

This will be conducted by linking users to the website with a link to an online survey alongside a link to a google drive folder containing relevant test images. With the questionaire mostly consisting of System Usability Survey (SUS) questions. And some aditional bespoke questions regarding accuracy of prediction. User feedback aside from the data collected from the SUS survey will also be valuable for this project, it is a good way to be allerted to any usability issues; such as bugs, difficulty understanding how to use the application (what icons mean, order to carry out steps etc)

and difficulty seeing UI elements due to poor colour choice.

3.2 Development Methodology

3.2.1 Feature Design

UI

Wireframes will be uitilized to establish interface element placement i.e. layout. Then more mock-ups that align better with the possible implementation will be created superseding the original prototype wireframes. A colour picker that takes predetermined segments of the colour wheel, will be utilized to define the colour scheme. In later iterations of the design, once there is a functioning UI, usability will continue to be refined with the help of existing usability research, to guide the usage of font/colour/highlight on hover/font size etc. Additionally once a desktop friendly layout has been established, work will begin on optimizing a version for mobile.

Backend

To better conceptualize the needs of the user. Use case diagrams and activity diagrams will be utilized. Then the main types of code to help with design are psuedocode which is written loosley in a programming style but does not excecute. Prototype code, which runs but is intented to be discarded. And tracer code, this is a kind of MVP but on the individual feature level, which becomes the starting point for further development.

Design & Implementation Coupling

An iterative aproach has been chosen because the design of the software is affected by factors of the implementation. Before implementing a feature it is not always clear how the solution will look and which possibilites will be simplest to meet the requirements. Therefore, an initial design will be created to guide the first steps of implementation, then as more information is learned and perhaps an easier to make design is realised, the new design can be checked against the requirments. If satisfactory, the new design can become canon and development on the updated design can continue.

This coupling brings about the necessity of Won't have catagory in the MOSCOW method. This prevents future iterations of design and additional requirements encroaching on what the program will not be able to do. This can arise as an implementation choice may rule out the possibility of other features being included. For example the choice to use a sub-optimized javascript library may rule out the possibility of the application running on mobile.

3.2.2 Implementation

The first step will involve determining the apropriate technologies and libraries to achieve the design. This is necessary to realize the constraints that are imposed by the implementation method and know to what extent the design is feasible.

The process to create the feature will then be carried out iteratively. When development begins we first arrive at a base-case often known as a Minimum Viable Product (MVP), often hard-coded solution and subsequently make the code more dynamic, flexible, and robust to erronus data. Then if necessary optimize the perfomance of the code to improve run-time. Ensuring to adhere to the software development practices outlined in the literature review.

The ideal MVP will be one that fulfils the 'must have' requirements. Then it will be possible to expand it's features to include 'could have' features etc. Aditionally, once the MVP is acheived, design considerations such as ease of extensability and addition of new features will be more heavily focused on. Sometimes with major code refactor occuring at this point (see Refactoring), as to acheive the MVP quickly it is sometimes necessary to implement features in a way that may not be conducive to easy maintinence, efficency, readability or serving content dynamically. As noted in the literature review, this process of gradual addition of complexity is supported by many authors.

3.2.3 Testing methods

Testing will be conducted iterativley as the functionality expands, with unit tests being introduced for some components depending on time constraints. Testing will consist of firstly confirming that when interfaces are interogated with sound data, the responses are consistent with requirments. Secondly stress testing the interfaces with extraneous data will be carried out to ensure that apropriate error responses are given and the application does not simply crash. We ensure that if in the case of crashing, it is able to automatically re-start.

On a larger project, especially if working with multiple developers, developing to test and having unit tests for each module is imperative. These test the module with valid and extraneous data against it's interface 'contract' i.e. what it promises to produce upon interaction. Secondly, integration tests would be created that test the interatction between modules. As integration is the most likely point for bugs to occur. When discovering a bug, it is good practice to create a test to reproduce it, firstly to cut down time checking if the bug is fixed, and to ensure it does not occur again.

On a sufficiently large project it is also good practice to test the tests, by creating branches of the

software that deliberatley fail, to ensure the tests are functioning correctly, and catching the errors you expect them to catch.

3.2.4 Refactoring

an example to illustrate the process of refactoring that will be utilized during development. The first peice code was arrived at because the base case was to retrieve the most likely defect. The extended requirment was to find the top three most likely defects. This has resulted in two methods being created for each case as the base case was coded first and extended requirment added in later. Although this is still functional, it's unnecessary and the code can be simplified, by using the result produced by the newly created method, the base-case method and any calls to it can be deleted. (note the method is declared in Model class which has been ommitted for clarity) Refactor example one, three, is entirely debateable as to whether or not it is an improvement. It is certainly more terse, which makes it arguably more difficult to read. The counterpoint to this being, one is not allocating a computationally redundant variable at runtime. This being moreso noteable in the language of Python. Given it's interpreted nature, the interpreter will garunteed cause the machine to execute this operation (granted it's a fairly inexpensive one). If it were a compiled however, it is likely the compiler would ignore this redundancy and it would have no bearing on the speed of the program. Finally, comments are either added or updated.

```
def determineClass(self, predictionArray):
    return np.argmax(predictionArray,axis=1)

def determineTopClasses(self, predictionArray):
    predictDict = {}
    topClasses = []
    for idx, prediction in enumerate(predictionArray):
        predictDict[idx] = prediction
    predictDict = sorted(predictDict.items(),key=operator.itemgetter(1),reverse=True)

    for entries in list(predictDict)[0:3]:
        topClasses.append(entries[0])

    return topClasses

predictedClass = determineClass(results)
topClasses = model.determineTopClasses(results)

return{
"classNo" : predictedClas,
"topClasses": topClasses,
}
```

Figure 8: Example 1 : Initial

```
def determineTopClasses(self, predictionArray):
    predictDict = {}
    topClasses = []
    for idx, prediction in enumerate(predictionArray):
        predictDict[idx] = prediction
    predictDict = sorted(predictDict.items(),key=operator.itemgetter(1),reverse=True)

    for entries in list(predictDict)[0:3]:
        topClasses.append(entries[0])

    return topClasses

topClasses = model.determineTopClasses(results)

return{
"classNo" : topClasses[0],
"topClasses": topClasses,
}
```

Figure 9: Example 1 : Refactor 1

```
def determineTopClasses(self, predictionArray):
    predictDict = {}
    topClasses = []
    for idx, prediction in enumerate(predictionArray):
        predictDict[idx] = prediction
    for entries in list(sorted(predictDict.items(),
    key=operator.itemgetter(1),
    reverse=True))[0:3]:
        topClasses.append(entries[0])
    return topClasses

topClasses = model.determineTopClasses(results)

return{
"classNo" : topClasses[0],
"topClasses": topClasses,
  }
```

Figure 10: Example 1 : Refactor 3

```
#Used to find X most likely predictions from a prediction array created by keras
#CNN model
def topClasses(self, predictionArray, xClasses=3):
    predictDict = {} #key = original array position & therefore class number
                     #value = prediction likelehood
    topClasses = [] #values to return from the function
    #we zip the array into a dictionary to keep track of the values original position
    #as this correlates to the predicted class.
    for idx, prediction in enumerate(predictionArray):
        predictDict[idx] = predictionArray
    #we sort the dictionary in ascending order by value.
    #then we take the first X entries of the sorted dictionary.
    #and take the virst value i.e. class-number from the entries, as each entry
    #is a tuple, key,value pair.
    for entries in list(sorted(predictDict.items(),
    key=operator.itemgetter(1),
    reverse=True))[0:xClasses]:
        topClasses.append(entries[0])
    return topClasses
```

Figure 11: Example 1 : Refactor 4

In this example of refactoring we see how multiple redundant variable allocations can be condensed into a single line.

```
results = results[0]
results = results.tolist()
jsonResults = json.dumps(results)

return{
"results" : jsonResults,
}
```

Figure 12: Example 2 : Initial

```
return{
"results" : json.dumps(results[0].tolist()),
}
```

Figure 13: Example 2 : Refactor

Given these examples it's easy to see why lines of code created is a rather ineffective way (in isolation) of determening programmer productivity.

3.2.5 CNN Model Creation & Integration

To create the CNN it is possible to perform modifications to an 'off the shelf' model included in the Keras library. The modifications involve changing the first and last layers to match the input image size and number of output classes respectively, we see this aproach used in several of the papers cited in the literature review.

A fast to train model will be essential due to the limitations of using Google Colab, which requires frequent user interaction to prevent from stopping compuation or entirely disconnecting the runtime. Therefore a model with low-number of parameters will be chosen.

The weights of the CNN will need to be saved after each epoch to ensure that if a disconnect occurs, less training progress will be lost. Once the model is finished training it will have it's class list extracted and the model weights will be saved into a .h5 file to be loaded on the API. The JSON class list will also be added to the API.

When training the CNN one will utilize a pre-trained (on the ILSVRC dataset) to further cut down training time, as this will mean the CNN already has a set of generic feature maps. The focus on reducing training time is based on the fact that training large networks with millions of parameters can take days or weeks, even when utilizing high end GPU's.

For the tradeoff in training time it is unlikely the model will acheive state of the art performance, but it is likely to achieve within a few percent. This has driven the requriement that the model must be easily replaced.

3.2.6 Version control

I will be using Git and Github. This will allow the creation of branches to explore experimental parts of the soloution space without disrupting the progress of the main branch. If the experimental implementation is successfull it will be merged with the main branch. It also allows the

development of features in paralel, with any conflicts in their implementation being resolved at the merge stage. The inclusion of a remote repository allows for work to continue on a seperate machine if necessary and later be synced with the local main branch.

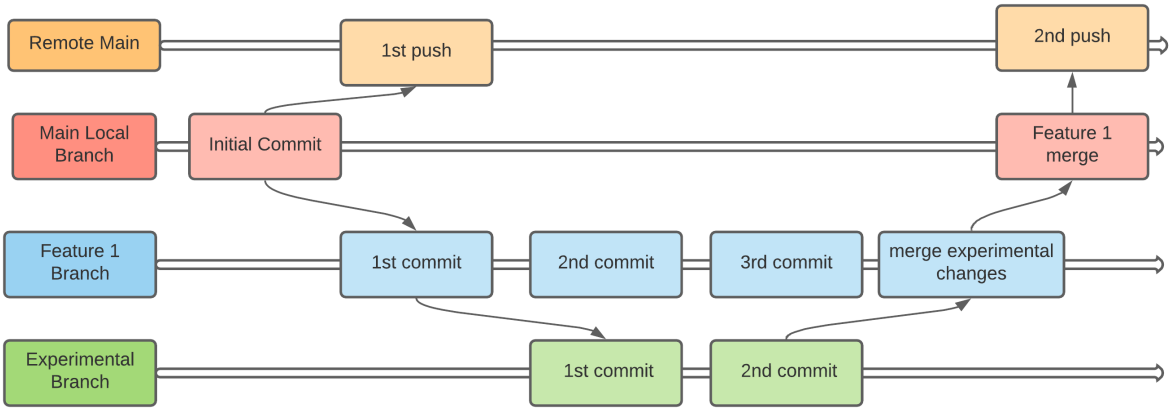


Figure 14: Example Workflow To Highlight Branch Usage

3.3 Evaluation methods

3.3.1 User Interface

The main method that will be utilized to determine the quality of the user interface will be the System Usability Scale (SUS) which can be seen here. (CITATION) The evaluator will be given remote access to the webservice. They will also be provided with some sample images to test the perfomance of the CNN incase they do not have suitable images of their own. The opinion data will be collected via online questionare.

3.3.2 Convolutional Neural Network (CNN)

Metrics for the evaluation of the CNN will be:

- Time to train the network on available hardware
  - The constraint here being if the network cannot be trained on the available hardware in under sixteen hours. Purely for practical considerations.
- Accuracy of CNN predictions. (which will be most effective when there are equal numbers of samples belonging to each class) $Accuracy = \frac{CorrectPredicitons}{TotalPredictions}$ else if the samples are skewed, the network could be a faliure at detecting a specific under-represented class, yet still score high accuracy.

- Precision. This is the number of correctly predicted images out of all predictions of that class. $Precision = \frac{CorrectlyPredictedforClass}{TotalPredictedforClass}$ The network is precise for a class when the predictions it does make for the measured class are correct. Precicion cannot be used in isolation due to the fact that the network can have a high prediction for a class but still fail to identify the majority of images for that class. Succeeding soley on the fact that the images it has classified as the class being measured are correct.
- Recall. Is the correct number of predictions for a class out of the number present of that class. $Recall = \frac{CorrectPredictedforClass}{No.PresentForClass}$ This metric can also not be used in isolation due to the fact it does not take in to account the number of false positives. i.e. The number of images incorectly classified as the class in question. For example, if an image dataset contained three classes A, B, C, and the classfiier labeled all images A. The recall for A would be 100 percent.
- F1 score. This metric tries to find the balance between precision and recall and can be expressed as $F1 = 2 \times \frac{1}{\frac{1}{precicion} + \frac{1}{recall}}$

3.4 Initial Designs

Firstly I have created a wireframe UI

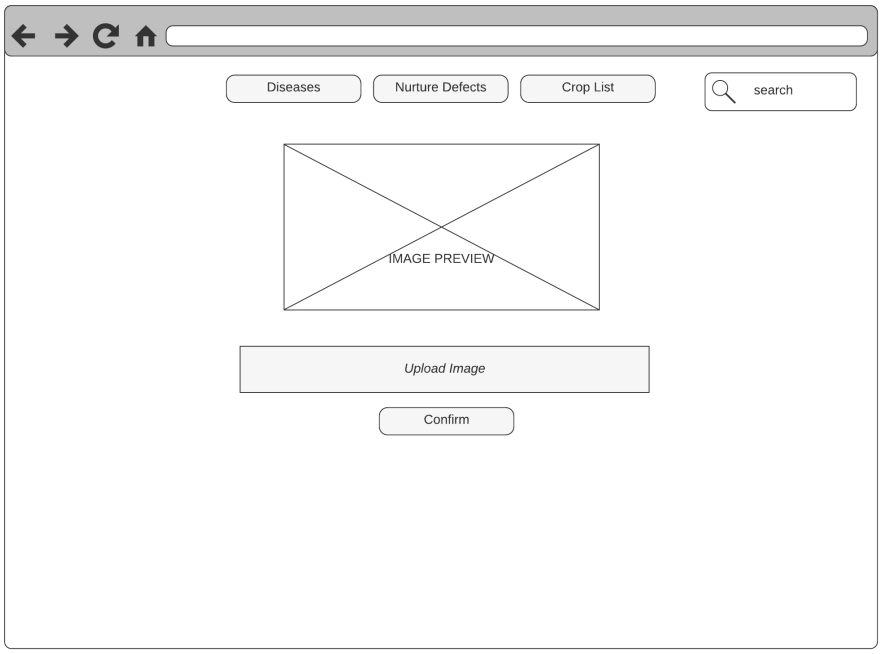


Figure 15: Homepage Wireframe

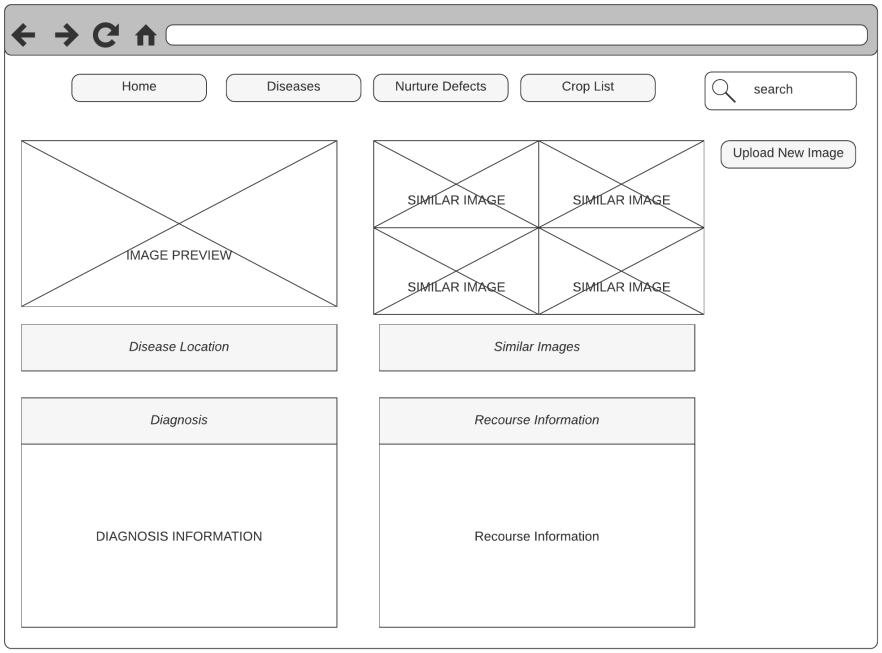


Figure 16: Defect Information Wireframe

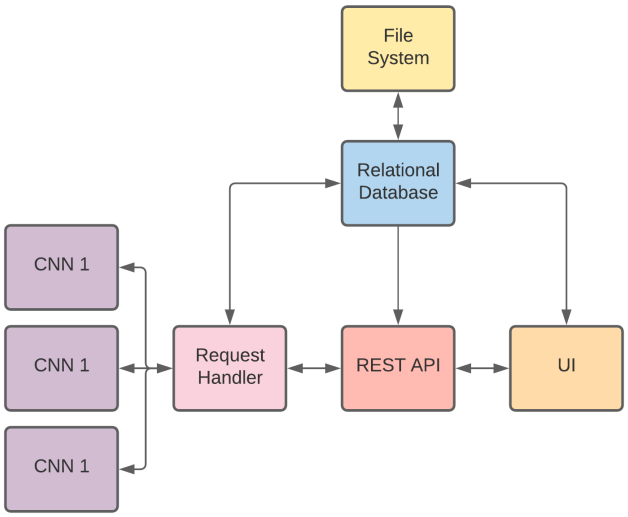


Figure 17: System Overview

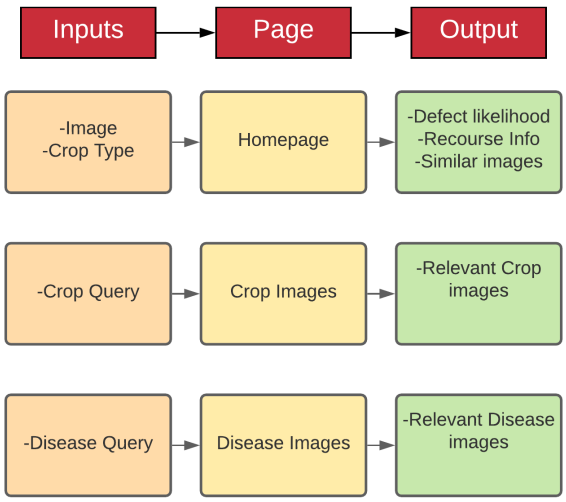


Figure 18: Input/Output overview

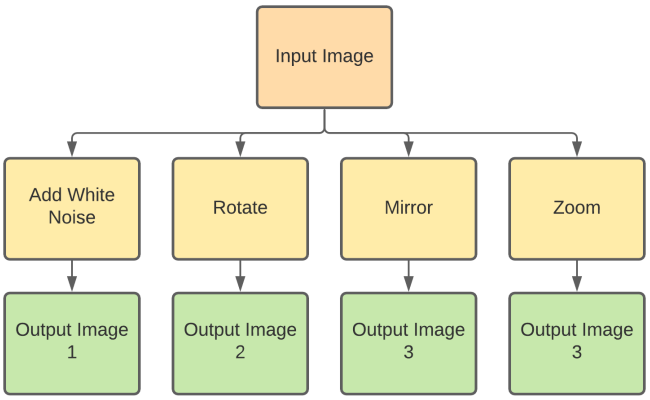


Figure 19: Input/Data Augmentation Methods

See diagrams Apendix for additional diagrams.

3.5 Employed Technologies & Justifications

3.5.1 Software Structure

Back-End

The design aims to seprate the functionality of the CNN and the User Interface (UI) through use of a Representational State Transfer (RESTFul) Aplication Interface (API). The central idea of this aproach is to allow a seperation of distinct components which allows for flexebility in managing updates and changes to each component. The 'backend', which in this context, refers to the CNN

and it's related functionality such as processing it's input's and outputs; will both receive input and serve it's output through the interface of the RESTFul API. each 'endpoint' or in other words internal class of the API will all inherit from the Resource interface which means it will recieve the main four HTTP protocol functions. POST, GET, PUT, DELETE. All interogations of the CNN will be conducted through these requests.

A key feature of A RESTFul interface is the fact that no client information is stored between requests, this aids in making the interface more scalable and iteroperable. That is, scalable in the sense that increasing clients does not increase the amount of information the backend will have to store and be interoperable in the sense that any client, be it mobile appliation, CLI program or webpage can use the interface.

To increase orthogonality of the system, defect images are stored standalone on an image server. With the API serving the name of the defect which gives the information necessary for the webpage to retrieve the correct images. This is because the directory structure of the image server is set up such that directory names are defect classes and all image names are numbers. Having the images stored on a seperate server rather than bundled with the front-end makes the service more useful. As it means any other interface can also choose to obtain the defect images (although image server endpoint information is not currently served by the API).

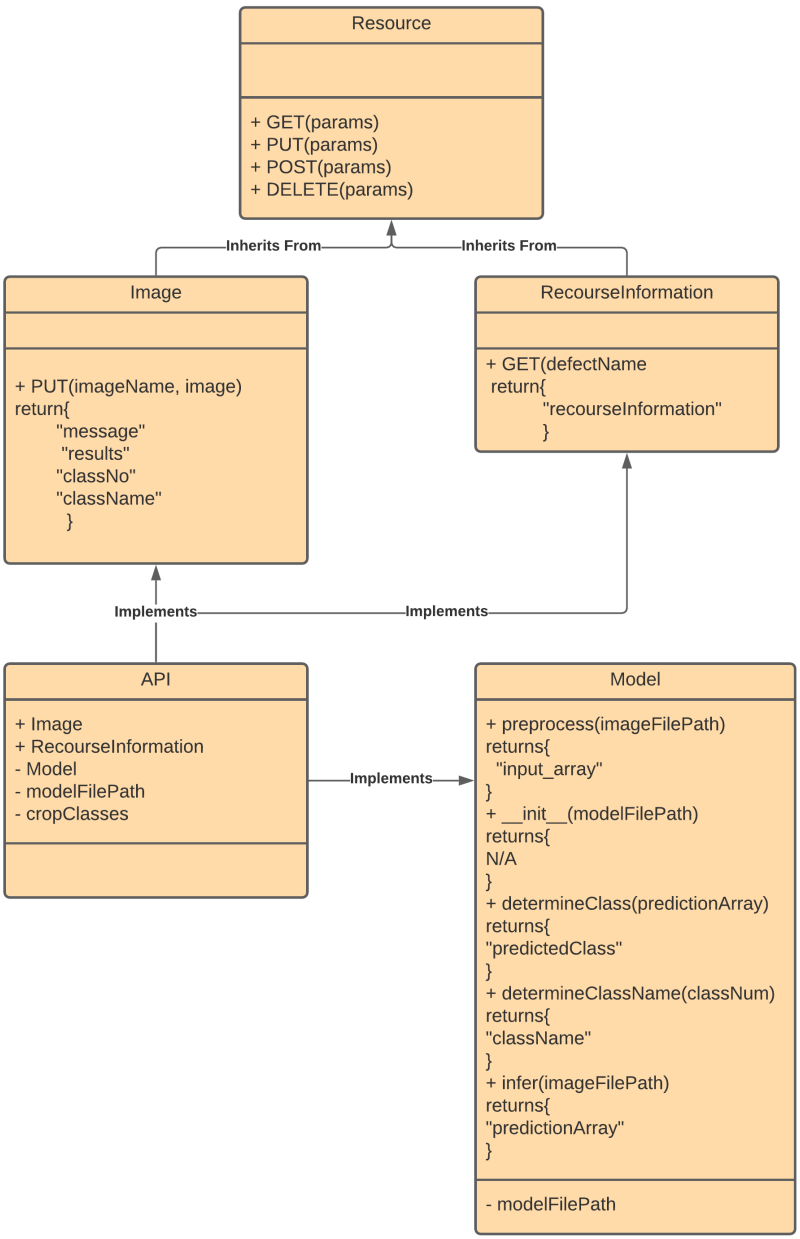


Figure 20: Backend Class Diagram

Front-End

Using the Vue.js webpack framework means the front-end will be split in to components. A component acting much like an endpoint in an API. Each component will have it's own set of functions and data, with the purpose of seprerating functionality of the UI in to manageable subsets of functionality and styling with the objective being to decouple dependancies between compoonents as much as possible. This helps simplify the development process as it seperates the problem in to distinct parts and it helps with maintinence, allowing them to be tested and developed on in isolation.

UI components will be defined using Hypertext Markup language (HTML) and Cascading Style Sheets (CSS). Interativity will be scripted in Javascript. All three extremely common in web development.

The main functionality of the web-app will be uploading an image to be analysed by the CNN and recieving a prediciton of which defect is/isn't present. This will be conducted by a single component. Once a prediction has been made, the user can choose to see more information (consisting of images and recourse information) about the defect, which will be handled by two further components.

Responses from the API will come in JSON format. JSON is a structured data language and stores data heirachically. This leads to dictonaries and lists of arbitrary depth in a tree structure. Much the same as XML or a filesystem. JSON is a very common language when dealing with webservices and is considered the norm to be the response of an API.

The application is made with the intention that as much data processing, such as determining which class is predicted, image preprocessing and determining of defect image URL's, defect names etc. is handled on the back-end, meaning that any client wishing to interact with the API does not have to implement these features themselves. Therefore, all functionality implemented in the web app will concern the consumption of user input and display of API response.

This means the navigation drop-downs, defect images and recourse/prevention information are all served dynamically through the API and can be scaled more easily than if the information was stored on the frontend. Because the content is dynamic, any client consuming the information won't have to manually update their defect list or defect names if the CNN model changes. This is in-line with the don't repeat yourself (DRY) principle. Having two places where what is supposed to be the same information is stored is an immediate 'code smell' and can easily lead to bugs. This also stands true for methods which enact some kind of bisuness logic. For instance it would not be best practice to implement the algorithm which re-scales the image both on the front and back end, as the size the algorithm needs to re-scale to may change, or a better method for re-scaling the image may become apparent, this would cascade to changing two parts of the system.

3.5.2 Hosting

To make the application accessible on the internet one will utilize virtual machines (VM's) running linux, hosted in the cloud. There will be three VM's, one to host the front-end, back-end and the other to host the defect images. In all cases nginx will be used to create HTTP servers that will handle all functionality regarding the HTTP protocol and the socket layer.

Aside from creating backups of work, version control allows production environments (in this case the VM's) to be easily updated with new features, simply by pulling the latest changes from the repo, installing any new dependencies and re-building the application.

3.5.3 Choosing The CNN Architecture

The initial architecture one experimented with was the InceptionNetV3, the reason being that one already had a familiarity with the architecture and therefore could rationalise about what may need to change in order to get better performance on the target dataset if initial experiments were not satisfactory. One opted to use a model pre-trained on the ILSSVRC dataset and fine-tune it to suit the crop dataset. This resulted in the model taking around six hours to train over an epoch (using Google Colab GPU runtime) and also drastically overtiffing the training data. One reasoned that the vast number of weights could be to blame for both problems, firstly the more weights the greater the training time, secondly, more filters leads to the network making less generalisations and being able to store many bespoke feature maps to target subsets or perhaps even individual images. Hence when it comes to an unseen example it's feature maps are not generic enough to aid in identifying the unseen image. Therefore one decided to use a a smaller network, 'EfficentNetB0', a network available as a pre-trained[1] [1] CNN through the Keras library. For comparison the InceptionV3 architecture is 92mb and contains 23,851,784 parameters (weights) whereas the EfficientNetB1 is 29mb and contains 5,330,571 parameters (approx 1/3 the parameters). Fine-tuning this network proved very successfull, acheiving a 98.6% accuracy. For this dataset, due to the fact all classes the network is classifiying for are relatively similar, less feature maps are needed to generalise about the forms present. To contrast, a network that must classify 100 different classes ranging from a baseball to a shark. with all objects present in different contexts, for instance a closeup of the baseball sat on a table, and a long range shot of it mid flight across the stadium. And it becomes easy to see why so many filters would be necessary. If we take the crop-example we see all images are closeups of leaves with a neutral background. Therefore it is possible to generalise about this smaller subset of possible images with fewer filters. In fact few parameters becomes a desirable quality of the network as it forces the network to generalise as it does not have the ability to 'memorise' all the individual training images.

CNN Data and Training

One of the most important parts when creating a CNN is the data it is trained on (Halevy *et al.* 2009). The more properly labled data the network sees, the better. The most comprehensive

[1] on the ILSSVRC dataset

dataset available (and the one which will mostly be used) is the Plant Village dataset. it contains 54303 images of healthy and unhealthy leaf images, in 38 categories divided by species and disease. This dataset also contains pre-segmented[2][1] versions of all the images. This dataset was used in two seperate studies cited in the literature review.

Other datasets that may be used to pre-train the network before are the CIFAR10, CIFAR100 & ILSVRC datasets. These datasets are not specifically plant based, but may benefit the network to build initial feature maps. In (Choi) the authors used networks pre-trained on generic data such as the datasets mentioned.

One unfortunate aspect was the fact that one was unable to obtain a dataset that reflected plant defects other than diseases, such as nitrogen/water/heat deficiencies.

3.5.4 Technologies & Libraries

Vue.js

Vue.js allows for rapid creation of usable websites thanks to its webpack framework, which handles most of the boilerplate needed to handle URL routing and overall structure. It is also considered easy to learn when compared to React and Angular (im Studiengang Bachelor *et al.*) which for this project is optimal. Although Vue is currently less popular than React and Angular its popularity is rising, which means its not entirely redundant to learn from an employability standpoint. Vue is a component based framework, meaning that it creates a structure to allow the programmer to utilize component technologies to build the website. The component technologies are; custom elements, which allow the programmer to embed javascript code in to either existing HTML elemnts by inheriting from them, or creating entirely new ones. HTML templates, which allows for sections of HTML to be re-used throuought the application, for instance navigation bars. Also, Shadow DOM's, which allow for DOM elements to contain sub-trees of elements, this is to aid in encapsulating the scope of the contained scripting and styling, meaning individual elements can have their own layers of encapsulation.

Bootstrap

Bootstrap is a CSS framework that has predefined classes for styling HTML elements, this helps with the asthetics of the webpage. For a solo developer this can allow for more time to spent on making the functionality more robust and cut down time spent on appearance.

[2]meaning the background has been blacked out

Jupyter Notebook / Google Colab

The Jupyter Python environment allows for easy prototyping of code, it allows the programmer to write code in cells that can be run independently and not necessarily in the same order each time. In the case of Colab, it allows for the usage of google servers and Graphics Processing Units (GPUS) which can be used to train neural networks. With the benefit of being able to then save the trained weights for usage elsewhere.

Tensorflow & Keras

These Python libraries allow for the creation of complex CNN architectures by abstracting away the underlying matrix/tensor multiplication of deep learning. This allows the programmer to declare relationships between CNN features such as convolutional layers, batch normalisation, dropout etc. Without having to construct their inner workings, it also allows for the creation of user defined layers, so adding in ones own bespoke functionality for a layer of the CNN is possible. These libraries are ubiquitous in the machine vision sector, with cutting edge models being created

Nginx

NginX is a web server. This software, installed on the virtual machines (VM's) hosting, front/back end and files. Handles incoming HTTP protocol requests on ports it is configured to listen to, and serves content accordingly. Additionally, if the need arises, NginX can be used as a load balancer. Taking incoming requests and distributing them amongst the back-end servers via a reverse-proxy. With the reverse-proxy aproach it is also possible to cache response data. Reducing load on the back-end. It can also be used to create IP address white/black lists, set file upload limits, keep access logs and more.

Gunicorn

Gunicorn is another technology used to host the application. This time it is used to interface with the Flask API. It is a Web Server Gateway Interface (WSGI) HTTP server. This gives an interface to the API to allow it to send data back and forth between the web server, NginX.

Flask/FLask-Restful

These libraries provide a framework for creating RESTFul API's in Python and gives the Resource class to inherit from for each endpoint. Flask also provides a built in development server for testing the appliation on your local machine.

Numpy

This Python library is essential when performing mathematical operations with Python, especially matrix multiplication, as in this case it is over 10,000 times faster than vanilla python. Other mathematical operations are also often orders of magnitude faster than vanilla. Aditionally it adds the ability to use 'proper' arrays and not lists which are extensible, this means less memory is needed.(Wickramarachchi 2020)

Node & Node Package Manager

Node creates an environment for JavaScript to be run outside of the browser environment. This is especially useful during development as it allows the developer to create a local Node server to test the code. Node's package manager checks for any dependencies the code may have and automatically installs apropriate versions, ensuring that the project does not have incompatible versions of dependencies installed simultaneously. As this can easily be a source of frustration when trying to handle multiple dependencies. This also means that whenever a new dependency is added in development, the package manager will automatically add install it in production during the build process.

3.6 Requirements

For assessing requirement priority one will employ the MOSCOW method (Clegg and Barker 1994). This creates subcategory of requirments with the headings Must/Should/Could/Won't - have. The contents of the categories can change during development in light of development progress. Items placed in the Won't-have category prevent the process of 'scope-creep'.

1. Must Have
    (a) CNN that is capable of classifying at least 2 different defects across 2 different plant species.
    (b) An API that allows communication from the UI to the CNN
    (c) API must be able to receive images.
        - Accepted formats being .jpg & .png
    (d) API must return defect information, which will be an array of probability values for each defect class
    (e) API must return recourse information.

(f) Application must display images that show the predicted defect.

- These images may be stored either on a seperate server to the front-end. Perhaps in the API servers 'static folder'. Alternatively they will be bundled with the front end.

(g) The API will be robust enough to handle the receipt of erroneous requests.

(h) A python backend that will handle image classification using a CNN.

(i) A UI that will allow the user to upload an image to be analysed.

- The user will be able to choose an image file from their local storage using a file explorer popup.

(j) The UI will display information regarding the likelihood of each kind of possible defect.

(k) To display the relevant images that fit the description of the most likely defects.

(l) To display recourse information to rectify the defect.

(m) Collecting, cleaning and pre-processing the image data.

(n) Artificially grow the dataset by performing translations/rotations/adding noise to the images to make the training data more comprehensive.

2. Should Have

(a) A page to allow users to see a gallery of images sorted by defect type.

(b) The CNN should be able to classify at least 7 different defects across at least two different plant species.

(c) The CNN should acheive at least 80% accuracy at classifying all different classes of defect in a held out test set that contains an equal number of each class.

(d) Regularisation techniques to prvent the NN overfitting.

3. Could Have

(a) Mobile device support.

(b) Ability for users to add additional information about the crop to determine the defect.

4. Won't Have

Chapter 4 - Results and Discussion

4.1 Main Results

4.1.1 CNN Performance

The Performance of the CNN given the available hardware (Google Colab GPU Runtime), is good. That is, with a 98.6% classification accuracy on a held out validation set. With current cutting edge, being 99.74% on the same dataset (as covered in the literature review). The CNN is capable of identifying 38 unique classes of crop images. Unfortunately due to the lack of data, one was unable to include additional defects such as lack of water/nitrogen/sunlight etc.

4.1.2 Evaluation Results

System Usability

Strongly agree = 5, Strongly disagree = 1. In the case of odd questions, agreeing is positive sentiment. For even questions agreeing is negative sentiment. The calculations are as follows. X is the sum of odd-numbered questions -5. Y is 25-sum of all even-numbered questions. The SUS score is $(X + Y) * 2.5$ The mean average of all SUS scores is then calculated. With the best possible score being 100.

1. I think	2. I fou	3. I tho	4. I th	5. I fc	6. I th	7. I wc	8. I fc	9. I fe	10. I n	X	Y	SUS Score
5	1	5	1	4	1	5	1	5	1	19	20	97.5
3	2	4	2	4	2	5	1	4	3	15	15	75
4	2	2	3	4	2	4	2	4	1	13	15	70
3	1	5	1	5	1	4	1	4	1	16	20	90
4	2	4	1	4	1	5	2	4	1	16	18	85
3	1	4	2	4	3	5	2	4	1	15	16	77.5
4	2	4	1	4	3	4	2	4	2	15	15	75
5	1	4	2	4	2	5	1	4	2	17	17	85
												81.875

Figure 21: SUS scores table

In this case the system scored 81.875 which is above the average of 68 (Jeff Sauro 2011). However with the limited number of questionaire respondents (8) the evidence is not wholly conclusive.

4.1.3 Requirements Checklist

1. Must Have

(a) CNN that is capable of classifying at least 2 different defects across 2 different plant species.✓

(b) An API that allows communication from the UI to the CNN ✓

(c) API must be able to receive images. ✓

- Accepted formats being .jpg & .png

(d) API must return defect information, which will be an array of probability values for each defect class ✓

(e) API must return recourse information. ✓

(f) Application must display images that show the predicted defect. ✓

- These images may be stored either on a seperate server to the front-end. Perhaps in the API servers 'static folder'. Alternatively they will be bundled with the front end.

(g) The API will be robust enough to handle the receipt of erroneous requests. ✓

(h) A python backend that will handle image classification using a CNN. ✓

(i) A UI that will allow the user to upload an image to be analysed. ✓

- The user will be able to choose an image file from their local storage using a file explorer popup.

(j) The UI will display information regarding the likelihood of each kind of possible defect. ✓

(k) To display the relevant images that fit the description of the most likely defects.✓

(l) To display recourse information to rectify the defect.✓

(m) Collecting, cleaning and pre-processing the image data. ✓

(n) Artificially grow the dataset by performing translations/rotations/adding noise to the images to make the training data more comprehensive.✓

2. Should Have

(a) A page to allow users to see a gallery of images sorted by defect type. ✓

(b) The CNN should be able to classify at least 7 different defects across at least two different plant species. ✓

(c) The CNN should acheive at least 80% accuracy at classifying all different classes of defect in a held out test set that contains an equal number of each class. ✓

(d) Regularisation techniques to prvent the NN overfitting. ✓

3. Could Have

(a) Mobile device support. ✓

(b) Ability for users to add additional information about the crop to determine the defect.

4. Won't Have

4.2 Discussion

Missed/Partial Requirements

Requirement 3b (Ability for users to add additional information about the crop) was not implemented. A possible method would have been to ask the user via a drop-down menu which crop was to be analysed. This could have allowed the back-end to select a tailored CNN which classifies only the classes relevant to the selected crop. The reason for the absense of this implementation is the time nececarry to train the extra CNN's.

Requirement 1e (API must return recourse information). This requirment is only partially fulfilled, because at this stage one was unable to obtain the information nececarry to populate these values. However, once the relevant information is obtained, only JSON files on the API side need be updated to properly fulfill this requirement.

Architecture

The separation of responsibility between the API and front-end has created two orthogonal systems that can be upgraded and interchanged easily. The front end does not need to know any details about the CNN(model) it is interacting with. The classes the model is able to identify are dynamically updated on the front-end and prediction data also remains consistent between model implementations. Recourse and prevention information are similarly easy to update and can be changed by updating a JSON file on the back-end. Images that the front-end display are also easily changed and are hosted on a seperate image server, further seperating the systems interdependence. A call for modularity and separation of responsibility was a key theme occuring in many works cited in the literature review.

Database

A major change made to the design of the application was the choice to drop the relational database in favor of using heirachical storage as it was noted that the application did not need to handle complex relational data. Initially the purpose was to have the database handle filepaths to images and recourse/defect information - this was easily handled with consistent directory naming on the image server and JSON files.

Move to Production

Some of the main issues that arose during development were during the move to production. This involved learning new technologies such as NginX and Gunicorn to deploy the project as the configuration can be obtuse upon first interaction. Some API code also did not work the same way as on the development server and so, had to be modified to work in the production environment.

Version Control

Through the use of version control it has been straightforward to update the code on the production servers simply by pulling the latest changes from the git repository. Version control has been helpful throughout the lifecycle thanks to the ability to branch the codebase (see version control).

Vue.js

Upon reflection, using the Vue.js framework standalone may not have been the best choice for the sole reason that one did not find a way of caching page states. This has led to the inability to go back to the results display state of the Upload_Image page after navigating away from it. It is possible that using the additional Vuex library would have made this possible. Because the Upload_Image component is currently implemented with the results display state being arrived at by a series of function calls, there does not seem to be a vanilla Vue.js way to cache the page state to allow it to be navigated back to without starting at the initial state of the component.

Reliability

Since first deploying the system to production around the start of April 2021, it has been robust enough to handle all requests made of it without crashing for over a month. The only downtime the system has had is during updates to it's functionality. This is evidence that the configuration of all hosting technology and error handling in the code is good enough to operate in a production environment; albeit with low user traffic. Noteably the site is frequenlty served requests, from bots running common vulnerability checks, such as attempting to open a debugging shell. None of

which seem to have been successful

CNN

Modularity Making the CNN straightforward to change was chosen as a requirement given the limited ability to train a state of the art model. However, once a new model is trained and better performance is realised, changing the model, can be done by changing a single filepath on the API server. Although if one wishes to extend the number of defects (classes) the model is able to predict; three steps must take place. Firstly, the filepath to the JSON file defining all classes must be updated on the API to reflect the new set of classes, the model .h5 file needs updating. And lastly the images for the new defect must be added to the image server. Making the process, simply, changing two filepaths and uploading new images (using Secure File Transfer Protocol (SFTP)).

Model Choice The final model, EfiicientNetB0 was not originally covered in the literature and came about through comparing the number of parameters in the 'off the shelf' models in an attempt to reduce number of parameters and therefore training time. Fortunatley The model proved very quick to train, only taking around three hours. One's original intention was to create a fractalNet model, however this proved to be absorbing more time than orignially anticipated. As can be seen in appendix C, many diagrams to better conceptualize the model were created. Ultimately the decicion was made to use a pre-made architecture.

Chapter 5 - Conclusion

5.0.1 Further Work

The design of the API lends itself to easily being extended to include more endpoints. This can enable the API in the future to handle requests to different models for different classification problems. One possible usage of this extensibility is that of automated pest/weed contol.

References

2021. URL https://yandex.com/images/.

ArabSat 5C - Internet by Satellite in Africa, 2021. URL https://www.globaltt.com/en/coverages-Arabsat5C_C.html.

best vegetables to grow - Explore - Google Trends, 2021. URL https://trends.google.com/trends/explore?q=bestvegetablestogrow&date=all&geo=US.

Digital Camera Market Share, Size, Trends and Forecast 2021-2026, 2021. URL https://www.imarcgroup.com/digital-camera-market.

File:Internet users per 100 inhabitants ITU.svg - Wikipedia, 2021. URL https://en.wikipedia.org/wiki/File:Internet_users_per_100_inhabitants_ITU.svg.

• Smartphone users 2020 — Statista, 2021. URL https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/.

G. Anthonys and N. Wickramarachchi. An image recognition system for crop disease identification of paddy fields in Sri Lanka. In *ICIIS 2009 - 4th International Conference on Industrial and Information Systems 2009, Conference Proceedings*, pages 403–407, 2009. ISBN 9781424448371.

Sungbin Choi. Plant identification with deep convolutional neural network: SNUMedinfo at LifeCLEF plant identification task 2015. Technical report.

Dai Clegg and Richard Barker. *Case Method Fast-Track: A Rad Approach*. Addison-Wesley Longman Publishing Co., Inc., USA, 1994. ISBN 020162432X.

Y. Le Cun, I. Guyon, L. D. Jackel, D. Henderson, B. Boser, R. E. Howard, J. S. Denker, W. Hubbard and H. P. Graf. 1989. Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning. *IEEE Communications Magazine*, **27**, 41–46. ISSN 01636804. (doi:10.1109/35.41400)

Cun Yann le. A Theoretical Framework for Back-Propagation, 1988. URL http://new.math.uiuc.edu/MathMLseminar/seminarPapers/LeCunBackprop1988.pdf.

John Gall. 1977.

Alon Halevy, Peter Norvig and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, **24**, 8–12. ISSN 15411672. (doi:10.1109/MIS.2009.36)

Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition. Technical report. URL `http://image-net.org/challenges/LSVRC/2015/`.

Jim Highsmith and Alistair Cockburn. Agile software development: The business of innovation, sep 2001. ISSN 00189162. (doi:10.1109/2.947100).

Andrew Hunt and David Thomas. *The Pragmatic programmer : from journeyman to master*. Addison-Wesley, Boston [etc.], 2000. ISBN 020161622X 9780201616224. URL `http://www.amazon.com/The-Pragmatic-Programmer-Journeyman-Master/dp/020161622X`.

im Studiengang Bachelor, Betreuende Prüferin, Ulrike Steeens Zweitgutachter, Martin Behrmann geb Knoblauch and Eric Wohlgethan. Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung. Technical report.

PHD Jeff Sauro. Measuring usability with the system usability scale, 2011. URL `https://measuringu.com/sus/`.

Sachin D. Khirade and A. B. Patil. Plant disease detection using image processing. In *Proceedings - 1st International Conference on Computing, Communication, Control and Automation, ICCUBEA 2015*, pages 768–771. Institute of Electrical and Electronics Engineers Inc., jul 2015. ISBN 9781479968923.

Donald E. Knuth. 1974. Structured Programming with go to Statements. page 208.

Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Technical report, 2012. URL `http://code.google.com/p/cuda-convnet/`.

Omkar Kulkarni. Crop Disease Detection Using Deep Learning. In *Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018*. Institute of Electrical and Electronics Engineers Inc., jul 2018. ISBN 9781538652572.

Gustav Larsson, Michael Maire and Gregory Shakhnarovich. may 2016. FractalNet: Ultra-Deep Neural Networks without Residuals. URL `http://arxiv.org/abs/1605.07648`.

Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**, 2278–2323. ISSN 00189219. (doi:10.1109/5.726791)

Sharada P. Mohanty, David P. Hughes and Marcel Salathé. sep 2016. Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, **7**, 1419. ISSN 1664-462X. URL http://journal.frontiersin.org/article/10.3389/fpls.2016.01419/full. (doi:10.3389/fpls.2016.01419)

G. POLYA. *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press, 1945. ISBN 9780691164076. URL http://www.jstor.org/stable/j.ctvc773pk.

Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus and Yann LeCun. dec 2013. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*. URL http://arxiv.org/abs/1312.6229.

Karen Simonyan and Andrew Zisserman. VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION. Technical report, 2015. URL http://www.robots.ox.ac.uk/.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Technical report, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015a.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. Going Deeper with Convolutions. Technical report, 2015b.

Smith J. Rosser A. Brown C. Schulte-Herbruggen B. Booth H. Sassen M. Mapendembe A. Fancourt M. Bieri M. Glaser S. Corrigan C. Narloch U. Runsten L. Jenkins M. Gomera M. Walpole, M. and J. Hutton. Enabling poor rural people to overcome poverty Smallholders, food security, and the environment. Technical report, 2013. URL https://www.ifad.org/documents/38714170/39135645/smallholders_report.pdf/133e8903-0204-4e7d-a780-bca847933f2e.

Anuradha Wickramarachchi. Advantages of using numpy over python lists, 2020. URL https://medium.com/swlh/why-use-numpy-d06c573fbcda.

Zifeng Wu, Chunhua Shen and Anton van den Hengel. jun 2019. Wider or Deeper: Revisiting the ResNet Model for Visual Recognition. *Pattern Recognition*, **90**, 119–133. ISSN 00313203. (doi:10.1016/j.patcog.2019.01.006)

Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. Technical report.

Heyan Zhu, Qinglin Liu, Yuankai Qi, Xinyuan Huang, Feng Jiang and Shengping Zhang. nov 2018. Plant identification based on very deep convolutional neural networks. *Multimedia Tools and Applications*, **77**, 29779–29797. ISSN 15737721. URL https://doi.org/10.1007/s11042-017-5578-9. (doi:10.1007/s11042-017-5578-9)

Appendix A - Project Proposal & Plan

Bournemouth University

Department of Computing and Informatics

2020-21AY Undergraduate Final Year Project

Project Proposal Form

Please refer to the **Project Handbook Section 4** when completing this form

Degree Title: Computing	**Student's Name:** Ryan Syme
	Supervisor's Name: Dr Vegard Engen
	Project Title/Area: What's wrong with my crop? Using convolutional neural networks to detect crop defects.

Section 1: Project Overview

1.1 Problem definition - use <u>one sentence</u> to summarise the problem:

Crop defects threatening food security in the developing world and food supplementation in the developed world.

1.2 Project description - briefly explain your project:

Creating a REST (representational state transfer) API (application programming interface) to allow crop pictures to be uploaded and analysed by a CNN (convolutional neural network). Which will feed back percentage likelihood of each kind of crop defect and images that the network sees as a good match to the input.

1.3 Background - please provide brief background information, e.g., client, problem domain:

This web service could be of aid to small scale gardeners/farmers for identifying problems that may arise with their crop. For example, diseases, pest infestation, lack of water, lack of nitrogen, too hot/cold, humid/dry. With global food security still a major concern in both the developing and developed world, diseases and pests become an ever more prevalent threat. In the context of a developing country, communities are reliant on the crop produced by smallholders for survival as "Smallholders manage over 80 per cent of the world's estimated 500 million small farms and provide over 80 per cent of the food consumed in a large part of the developing world" (Walpole et al., 2013). In the developed world namely

the US, more people that are not expert horticulturalists are taking to producing food. Especially in recent times (2020), there has been a huge increase in interest for home food production (Google Trends, 2021). Therefore, having a quick way to get an idea of what is causing crop failure/defects and information on what is appropriate recourse, will allow producers to act against the problem and as a result, improve their yield. Current examples of research into image classification for aiding crop production include, crop disease identification of 3 different diseases in paddy fields (Anthonys and Wickramarachchi, 2009) whereby they obtained over 70% classification accuracy. And a research project in 2016 produced a model that achieved 99.35% accuracy at identifying 26 diseases (or lack thereof) on over 14 different crop species on a held-out test set (Mohanty, Hughes and Salathé, 2016)

1.4 Aims and objectives – what are the aims and objectives of your project?

Aims and Objectives:

- have a working REST API. The API will provide information regarding the likelihood of each kind of crop defect, when served an image via a link to a relational database. In addition to other metrics such as similar images and time to compute. The API will be robust enough to handle the receipt of erroneous requests.
- A python backend that will handle image classification using a CNN.
- A UI that will allow the user to upload an image to be analysed.
- The UI will display information regarding the likelihood of each kind of possible defect.
- To display the relevant images that fit the description of the most likely defects
- To display recourse information to rectify the defect
- Collecting, cleaning and pre-processing the image data
- Artificially growing the dataset by performing translations/rotations/adding noise to the images to make the training data more comprehensive.
- Include regularisation techniques to the NN to prevent overfitting.

The focus of the project will be building and optimising the CNN that will perform the image classification. A stretch goal will be to include features may that will enable the analysis of the NN's decision making process, perhaps leveraging open source libraries such as lucid (tensorflow/lucid, 2021)

Section 2: Artefact

2.1 What is the artefact that you intend to produce?

The artefact will consist of two main parts. Firstly, a REST API integrated with a CNN (convolutional neural network) that I will build largely from scratch and will perform image classification of crop defects. Secondly, a web interface that sends/receives relevant parameters from the user/client in the form of a RESTful API and a front end that visualises the returned data with the help of JavaScript.

2.2 How is your artefact actionable (i.e., routes to exploitation in the technology domain)?

With "90.7% of U.S. residents age 15–44 having Internet access" (Couper et al., 2018) and the number of people with internet access from all backgrounds steadily increasing year on year (Internet users per 100 inhabitants ITU.svg, 2021) with this number set to continue to increase due to the proliferation of satellite based internet access. With a single company claiming an Africa-wide satellite internet coverage (ArabSat 5C - Internet by Satellite in Africa, 2021). Therefore, services that can be accessed via the internet are becoming an ever more viable solution for the majority of people around the world. Hence, the web service will also continue to become more available to those in developing countries where it will be most impactful when aiding the identification of crop defect. And at present will be easily accessible to those in developed countries. Additional requirements will be a method of taking pictures and a device to access the internet. Smartphone use is increasing around the world (Smartphone users 2020 | Statista, 2021) which satisfies both these requirements. Furthermore, digital cameras have reached a market value of 14.7 billion (Digital Camera Market: Global Industry Trends, 2021) and home computer access has reached 49.7% of the world's population in 2019 (How many people have access to a computer 2018 | Statista, 2021) a statistic that has been increasing year on year.

Section 3: Evaluation

3.1 How are you going to evaluate your work?

Firstly, were the aims and objectives met? This can be evaluated through observation of a working and stable artefact that has the features mentioned in the proposal. This will be formalised by comparing the final software artefact with a list of requirements that will be iteratively created during the SDL, ensuring inclusion the features mentioned in my initial proposal.
Secondly, providing the application to potential users with an attached questionnaire to gather feedback. My questionnaire feedback will be evaluated against the System Usability Scale (SUS) which will include the generic questions and method of interpreting scores outlined on (System Usability Scale (SUS) | Usability.gov, 2021)

3.2 Why is this project honours worthy?

The project is honours worthy as it will be a clear demonstration of one's ability to carry out the production of a piece of software, start to finish, back-end and UI. Which will show one's planning and problem-solving skills as well as the domain knowledge one has acquired during their years of academic study and additional year in industry. The project will be an addition to a growing body of research into image classification and how its usefulness can be incorporated into the domain of food production.

3.3 How does this project relate to your degree title outcomes?

With the degree title being Computing, the main degree title outcomes that will be covered are computer algorithms, software development and some amount of creating a web application. The project will also be strongly building upon one's final year units, being; machine intelligence, deep learning and data mining. With the creation of a CNN being a direct extension of the deep learning module and an example of employing complex computer algorithms. The need to pre-process data being an extension of the data mining module. Creating a REST API will further build upon the web development skills of lvl5. The coupling and design of the front/back end will be a demonstration of one's software development ability.

3.4 How does your project meet the BCS Undergraduate Project Requirements?

The project builds upon one's final year modules, it contains objectives for the creation of a relevant software artefact, it will highlight one's main skills under the computing remit. Furthermore, the project will be created using current technologies and methodologies to show current best practice as determined by existing successful software applications and literature.

3.5 What are the risks in this project and how are you going to manage them?

- Improper time management – this mitigated by staying on track to the Gantt chart
- HDD/storage failure – any work done for the project will be backed up to cloud storage such as GitHub and Google Drive.
- Illness/injury – Should the need arise I will apply for an extension to the due date.

Section 4: References

4.1 Please provide references if you have used any.

[1] https://www.ifad.org/documents/38714170/39135645/smallholders_report.pdf/133e8903-0204-4e7d-a780-bca847933f2e (p6)

Anthonys, G. and Wickramarachchi, N., 2009. An image recognition system for crop disease identification of paddy fields in Sri Lanka. 2009 International Conference on Industrial and Information Systems (ICIIS),.

Globaltt.com. 2021. ArabSat 5C - Internet by Satellite in Africa. [online] Available at: <https://www.globaltt.com/en/coverages-Arabsat%205C_C.html> [Accessed 1 February 2021].

Couper, M., Gremel, G., Axinn, W., Guyer, H., Wagner, J. and West, B., 2018. New options for national population surveys: The implications of internet and smartphone coverage. Social Science Research, 73, pp.221-235.

Digital Camera Market: Global Industry Trends, O., 2021. Digital Camera Market Share, Size, Trends and Forecast 2020-2025. [online] Imarcgroup.com. Available at: <https://www.imarcgroup.com/digital-camera-market#:~:text=The%20global%20digital%20camera%20market,viewfinder%20or%20live%20preview%20screen.> [Accessed 1 February 2021].

Google Trends. 2021. Google Trends. [online] Available at: <https://trends.google.com/trends/explore?q=best%20vegetables%20to%20grow&date=all&geo=US> [Accessed 1 February 2021].

Statista. 2021. How many people have access to a computer 2018 | Statista. [online] Available at: <https://www.statista.com/statistics/748551/worldwide-households-with-computer/> [Accessed 1 February 2021].

En.wikipedia.org. 2021. Internet users per 100 inhabitants ITU.svg. [online] Available at: <https://en.wikipedia.org/wiki/File:Internet_users_per_100_inhabitants_ITU.svg> [Accessed 1 February 2021].

Mohanty, S., Hughes, D. and Salathé, M., 2016. Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science, 7.

Statista. 2021. Smartphone users 2020 | Statista. [online] Available at:

<https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/> [Accessed 1 February 2021].

GitHub. 2021. tensorflow/lucid. [online] Available at: <https://github.com/tensorflow/lucid> [Accessed 1 February 2021].

Walpole, M., Smith, J., Rosser, A., Brown, C., Schulte-Herbruggen, B., Booth, H., Sassen, M., Mapendembe, A., Fancourt, M., Bieri, M., Glaser, S., Corrigan, C., Narloch, U., Runsten, L., Jenkins, M., Gomera, M. and Hutton, J., 2013. Smallholders, food security, and the environment. [ebook] Available at: <https://www.ifad.org/documents/38714170/39135645/smallholders_report.pdf/133e8903-0204-4e7d-a780-bca847933f2e> [Accessed 1 February 2021].

Usability.gov. 2021. *System Usability Scale (SUS) | Usability.gov*. [online] Available at: <https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html> [Accessed 1 February 2021].

Section 5: Ethics (please delete as appropriate)

5.1 Have you submitted online ethics checklist to your supervisor? **Yes / No**

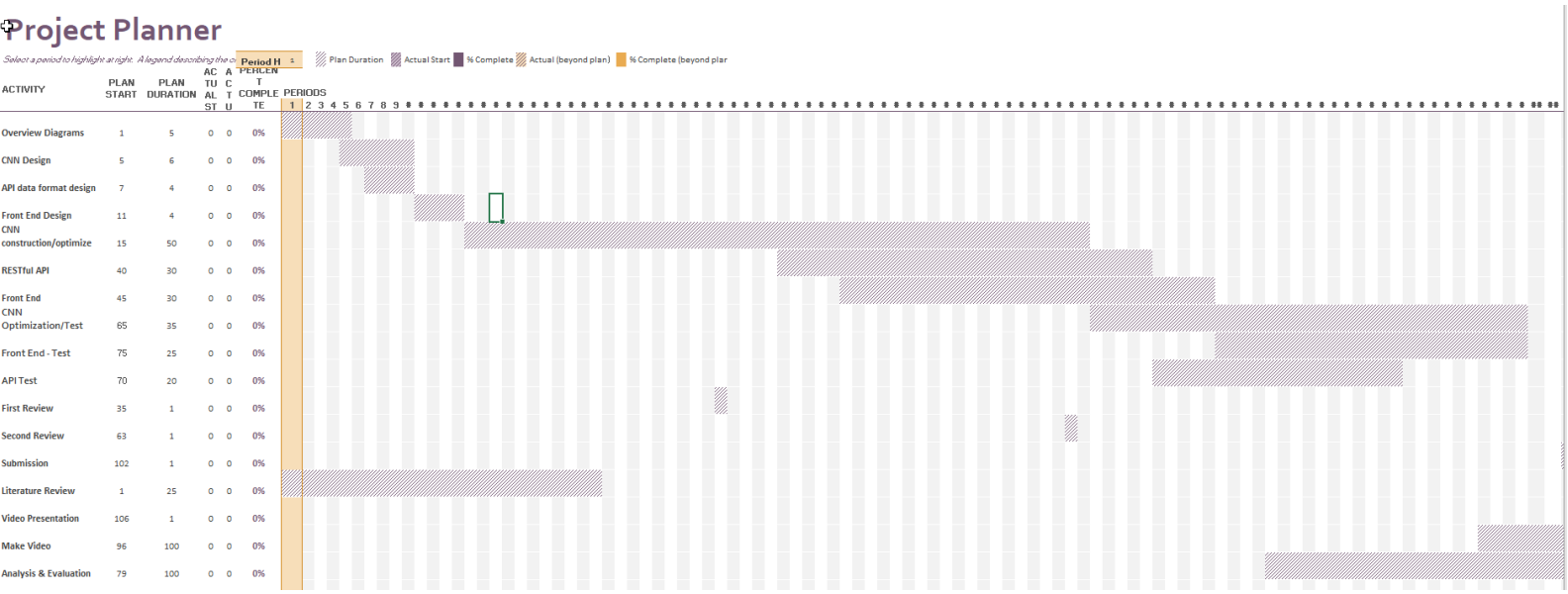
5.2 Has the checklist been approved by your supervisor? **Yes / No**

Section 6: Proposed Plan (please attach your Gantt chart below)

Bournemouth University

Department of Computing and Informatics

2020-21AY Undergraduate Final Year Project



Appendix B - Ethics Checklist

Research Ethics Checklist

About Your Checklist	
Ethics ID	36142
Date Created	04/02/2021 12:34:37
Status	Approved
Date Approved	04/03/2021 14:45:56
Date Submitted	01/03/2021 11:27:03
Risk	Low

Researcher Details	
Name	Ryan Syme
Faculty	Faculty of Science & Technology
Status	Undergraduate (BA, BSc)
Course	BSc (Hons) Computing
Have you received funding to support this research project?	

Project Details	
Title	What's wrong with my crop? Using convolutional neural networks to detect crop defects.
Start Date of Project	01/02/2021
End Date of Project	14/05/2021
Proposed Start Date of Data Collection	25/04/2021
Supervisor	Vegard Engen
Approver	Vegard Engen

Summary - no more than 600 words (including detail on background methodology, sample, outcomes, etc.)

Creating a website to allow crop pictures to be uploaded and analyzed by a computer program to determine what if any defects are present.

Filter Question: Does your study involve Human Participants?

Participants
Describe the number of participants and specify any inclusion/exclusion criteria to be used

Number of participants will depend solely on external factors, however I will aim to have at least 10 people complete the questionnaire. Participants will be 18 or over.

Do your participants include minors (under 16)?	No
Are your participants considered adults who are competent to give consent but considered vulnerable?	No
Is a Disclosure and Barring Service (DBS) check required for the research activity?	No

Recruitment

Please provide details on intended recruitment methods, include copies of any advertisements.

Posts on social media and word of mouth.

Example Advert:

Hello, I am gathering feedback for my dissertation project What's Wrong With My Crop. It's function is to identify crop defects from an uploaded image of the crop. Click the link -> <link> to be taken to the website. Click here -> <link> to be taken to the survey.

Do you need a Gatekeeper to access your participants?	No

Data Collection Activity

Will the research involve questionnaire/online survey? If yes, don't forget to attach a copy of the questionnaire/survey or sample of questions.	Yes

How do you intend to distribute the questionnaire?

online

If online, do you intend to use a survey company to host and collect responses?	Yes

If yes, please provide details of survey company.

TBD

Will the research involve interviews? If Yes, don’t forget to attach a copy of the interview questions or sample of questions	Yes

Please provide details e.g. where will the interviews take place. Will you be conducting the interviews or someone else?

I will be conducting interviews. Wherever suits the interviewee.

Will the research involve a focus group? If yes, don't forget to attach a copy of the focus group questions or sample of questions.	No
Will the research involve the collection of audio materials?	No
Will your research involve the collection of photographic materials?	No
Will your research involve the collection of video materials/film?	No
Will the study involve discussions of sensitive topics (e.g. sexual activity, drug use, criminal activity)?	No
Will any drugs, placebos or other substances (e.g. food substances, vitamins) be administered to the participants?	No

Will the study involve invasive, intrusive or potential harmful procedures of any kind?	No
Could your research induce psychological stress or anxiety, cause harm or have negative consequences for the participants or researchers (beyond the risks encountered in normal life)?	No
Will your research involve prolonged or repetitive testing?	No

Consent

Describe the process that you will be using to obtain valid consent for participation in the research activities. If consent is not to be obtained explain why.

Prior to taking the survey the participant will not be asked any personal information and will only be asked if they consent to having their answers included as part of an anonymous agregate.

Do your participants include adults who lack/may lack capacity to give consent (at any point in the study)?	No
Will it be necessary for participants to take part in your study without their knowledge and consent?	No

Participant Withdrawal

At what point and how will it be possible for participants to exercise their rights to withdraw from the study?	They will be able to close the survey window at any point during their evaluation of the web service.
If a participant withdraws from the study, what will be done with their data?	It will not be stored until the participant has explicitly completely the survey and confirmed as such. Due to the data being stored without personal information, once the survey has been submitted it will not be possible to remove the answers given by the participant.

Participant Compensation

Will participants receive financial compensation (or course credits) for their participation?	No
Will financial or other inducements (other than reasonable expenses) be offered to participants?	No

Research Data

Will identifiable personal information be collected, i.e. at an individualised level in a form that identifies or could enable identification of the participant?	No
Will research outputs include any identifiable personal information i.e. data at an individualised level in a form which identifies or could enable identification of the individual?	No

Storage, Access and Disposal of Research Data

Where will your research data be stored and who will have access during and after the study has finished.

It will be stored on my local machine, and in a private Github repository.

Once your project completes, will any anonymised research data be stored on BU's Online Research Data Repository "BORDaR"?	No

Please explain why you do not intend to deposit your research data on BORDaR? E.g. do you intend to deposit your research data in another data repository (discipline or funder specific)? If so, please provide details.

NOT APPLICABLE

Final Review

Are there any other ethical considerations relating to your project which have not been covered above?	No

Risk Assessment

Have you undertaken an appropriate Risk Assessment?	Yes

Filter Question: Does your study involve the use or re-use of data which will be obtained from a source other than directly from a Research Participant?

Additional Details

Please describe the data, its source and how you are permitted to use it	Non applicable.

Attached documents

SUS.pdf - attached on 09/02/2021 15:05:47
participantInformationSheetV1.docx - attached on 01/03/2021 11:25:22

Appendix C - Additional Diagrams

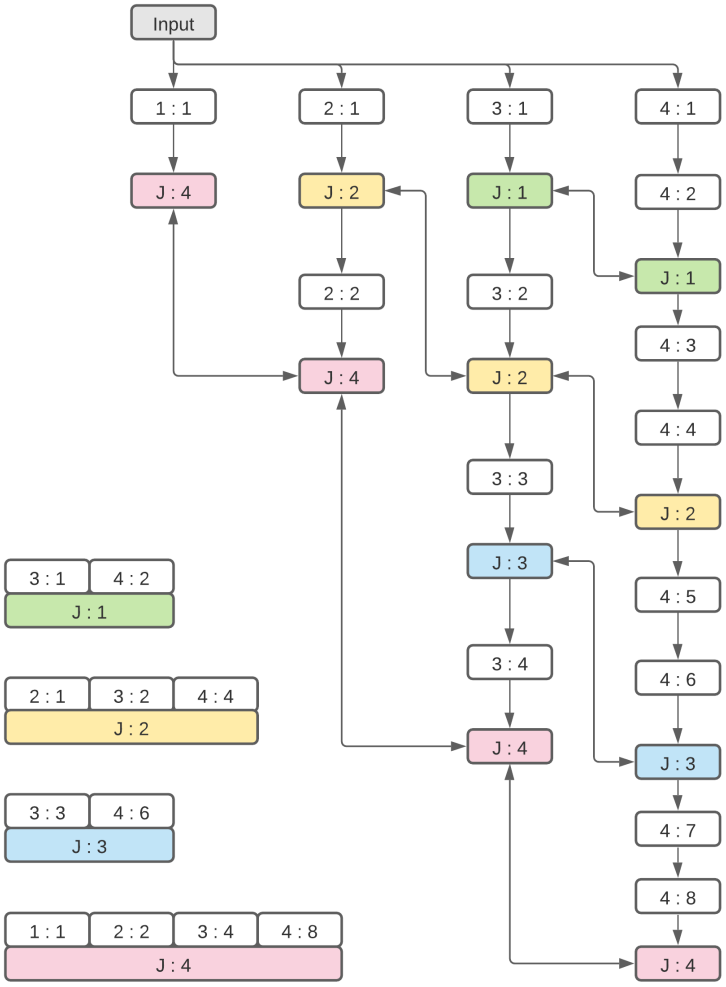


Figure 22: FractalNet With Coloured Join Layers

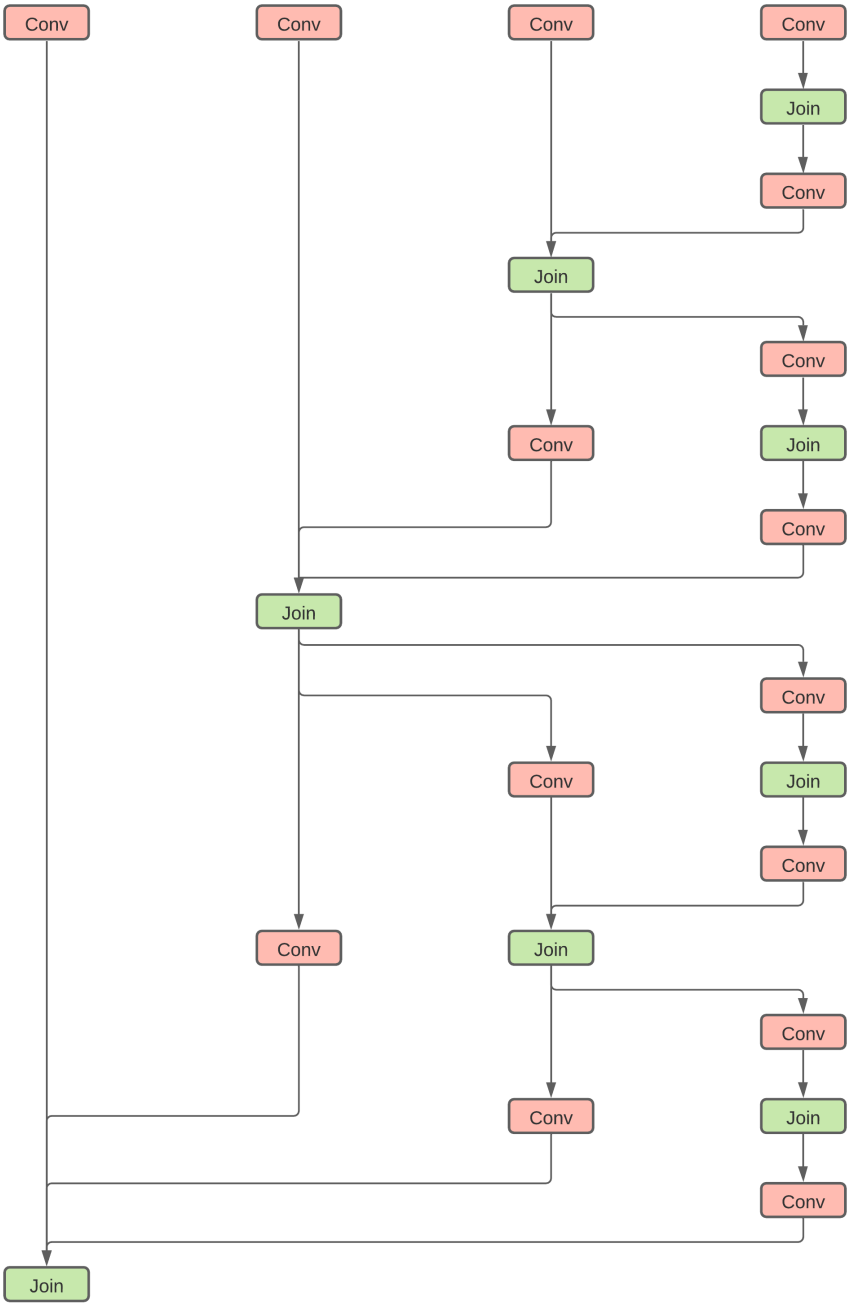


Figure 23: FractalNet Expanded

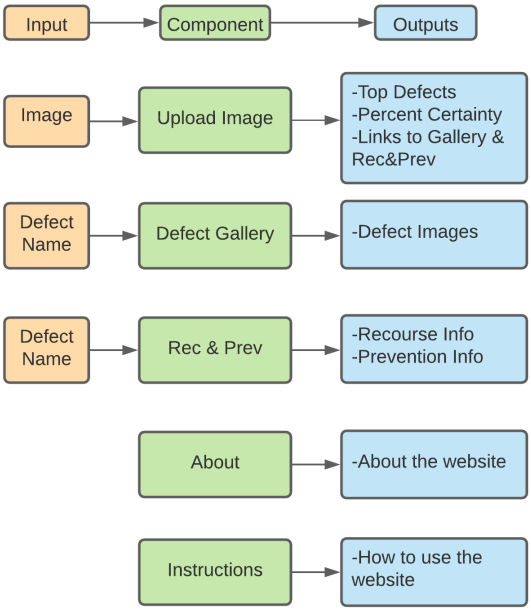


Figure 24: Frontend Overview

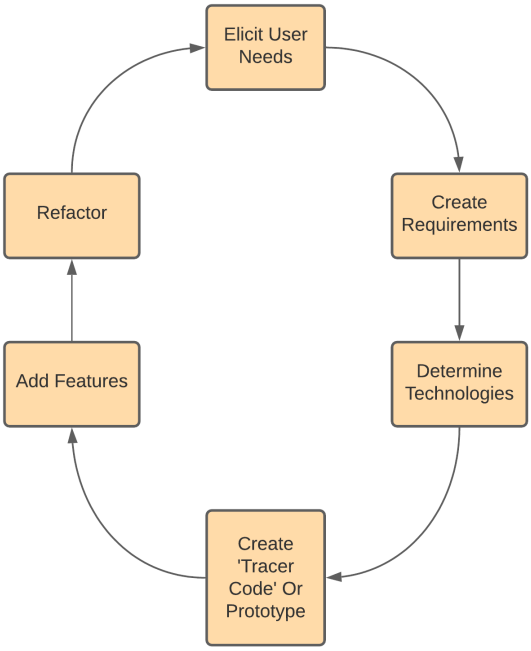


Figure 25: workflow

Appendix D - Questionaire

What's_Wrong_With_My_Crop

Page 1: Privacy Notice

Website can be found at this URL:

http://165.22.190.72/#/Upload_Image

This survey will not require the collection of any personal data.
For this reason, once the questionaire has been submitted, answers cannot be withdrawn. As the answers will not be linked with a participant and will be treated as part of an aggregate.

For more participant information. See the Participant Information Sheet at the link below.

https://static.onlinesurveys.ac.uk/media/account/97/survey/708608/question/participantinformationsheetv2.docx

For images to test the system with.

See this google drive link:

https://drive.google.com/drive/folders/1mrfrCFVjxCIGTbYTMfS2AbvS5m_rwYit?usp=sharing

Alternatively all crop images are also hosted at this URL

http://178.62.60.223/static/

1. I have read and understood the Participant Information Sheet and information above and consent to take part in this questionnaire ✱ *Required*

- YES

2. I give permission for members of the Research Team to have access to my anonymised responses. I understand that my anonymised responses may be reproduced in reports, academic publications and presentations but I will not be identified or identifiable. ✱ *Required*

- Yes

Page 2: Page 1

3. I think that I would like to use this system frequently.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

4. I found the system unnecessarily complex.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

5. I thought the system was easy to use.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

Page 3

6. I think that I would need the support of a technical person to be able to use this system.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

7. I found the various functions in this system were well integrated.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

8. I thought there was too much inconsistency in this system.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

9. I would imagine that most people would learn to use this system very quickly.

- Strongly Disagree

- Disagree
- Neither
- Agree
- Strongly Agree

Page 4

10. I found the system very cumbersome to use.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

11. I felt very confident using the system.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

12. I needed to learn a lot of things before I could get going with this system.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

Page 5

13. I used the system with my own images and found it to be successfull.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

14. I used the system with the images provided and found it to be successfull.

- Strongly Disagree
- Disagree
- Neither
- Agree
- Strongly Agree

Page 6: Final page

Thank you for completing the survey!

I plan to continue to add information to the website regarding crop defects and expand the number of defects the service is able to detect.

If you have any sugestions for features or notice any bugs. Feel free to email me at:

s5122732@bournemouth.ac.uk

Appendix E - Ethics Docs

Ref & Version: 1
Ethics ID: 36142
Date: 22/02/2021

Bournemouth University

Participant Information Sheet

The title of the research project

What's wrong with my crop. Using convolutional neural networks to identify crop defects.

What is the purpose of the questionnaire?

To establish the usability of the web service created to identify crop defects.

Why have I been chosen?

Anybody over the age of 18 is a suitable candidate.

Do I have to take part?

It is up to you to decide whether to take part. If you do decide to take part, you will be given this information sheet to read. You can withdraw from participation at any time and without giving a reason, simply by closing the browser page. Please note that once you have completed and submitted your survey responses, we are unable to remove your anonymised responses from the study. Deciding to take part or not will not impact upon you in any way.

How long will the questionnaire/online survey take to complete?

Less than 20 mins.

What are the advantages and possible disadvantages or risks of taking part?

Whilst there are no immediate benefits for those people participating in the project, it is hoped that this work will aid in making the 'What's Wrong With My Crop' website easier to use.

What type of information will be sought from me and why is the collection of this information relevant for achieving the research project's objectives?

The information gathered will be your opinions on the features and usability of the website.
This will be relevant for improving the usability and adding additional features.

Use of my information

Participation in this study is on the basis of consent: you do not have to complete the survey, and you can change your mind at any point before submitting the survey responses. We will use your data on the basis that it is necessary for the conduct of research, which is an activity in the public interest. We put safeguards in place to ensure that your responses are kept secure and only used as necessary for this research study and associated activities such as a research audit. Once you have submitted your survey response it will not be possible for us to remove it from the study analysis because you will not be identifiable.

The anonymous information collected may be used to support other research projects in the future and access to it in this form will not be restricted. It will not be possible for you to be identified from this data.

Contact for further information

If you have any questions or would like further information, please contact Ryan Syme at s5122732@bournemouth.ac.uk

In case of complaints
Any concerns about the study should be directed to contact Ryan Syme at s5122732@bournemouth.ac.uk
.If your concerns have not been answered by Ryan you should contact Professor Tiantian Zhang, Bournemouth University by email to researchgovernance@bournemouth.ac.uk.

Consent to Participate (to be included as a consent statement as part of the online questionnaire):

Please indicate that you have read and understood the Participant Information Sheet for this research project and that you consent to take part in this questionnaire before continuing:

☐ I have read and understood the Participant Information Sheet and consent to take part in this questionnaire

☐ I do not consent to take part in this questionnaire [exit at this point]

Please indicate your agreement for the Research Team to access and use your recorded responses to this questionnaire before continuing:

☐ I give permission for members of the Research Team to have access to my anonymised responses. I understand that my anonymised responses may be reproduced in reports, academic publications and presentations but I will not be identified or identifiable.