

A Highly Accessible Application for Detection and Classification of Maize Foliar Diseases from Leaf Images

By

Joang Adolf Khethisa

Supervised by

Professor Patrick Marais



Minor dissertation submitted in partial fulfilment of the
requirements for the degree of Master of Science in Information
Technology

Department of Computer Science
University of Cape Town

February 2017

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

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

Plagiarism declaration

I know the meaning of plagiarism and declare that all the work in this document, save for which is properly acknowledged is my own.

Signature:

Date: 12th February 2017

Student Name: Joang Adolf Khethisa

Dedication

I would like to dedicate this work to the memory of my late mother, **Malefa Julia Khetisa**. No words are sufficient to describe her contribution to my life. Even though you are gone, your inspiration lives on.

Acknowledgements

I would like to take this opportunity to thank the following people, whom without this thesis would not have been possible:

- Professor Patrick Marais, for your unwavering support, guidance and mentorship throughout the process.
- My family, Khethisa Khethisa, Mosito Khethisa, Manapo Khethisa, Smith Khethisa and Lefa Khethisa, **for your support and understanding when I missed those important family events to study.**
- My fiancée, Boitumelo Lesala, for your patience and support.
- My colleagues at CSG International, for understanding and being patient with me on those days I came into the office half asleep.

Thank you all.

Abstract

Crop diseases are a major impediment to food security in the developing world. The development of cheap and accurate crop disease diagnosis software would thus be of great benefit to the farming community. Many previous studies, utilizing computer vision and machine-learning algorithms, have successfully developed applications that can diagnose crop diseases. However, these studies have primarily focussed either on developing large scale remote sensing applications more suited for large scale farming or on developing desktop/laptop applications and a few others on developing high end smartphone applications. Unfortunately, the attendant hardware requirements and expenses make them inaccessible to many subsistence farmers, especially those in sub-Saharan Africa where both smartphones and personal computers ownership is minimal.

The primary objective of our research was to establish the feasibility of utilizing computer vision and machine learning techniques to develop a crop disease diagnosis application that is not only accessible through personal computers and smartphones but is also accessible through any Internet enabled feature phone.

Leveraging methods established in previous papers, a prototype crop disease diagnosis application capable of diagnosing two maize foliar diseases, Common Rust and Grey Leaf Spot, was successfully developed. This application is accessible through personal computers and high end smartphones as well as through any internet enabled feature phones. The solution is a responsive web based application constructed using open source libraries whose diagnosing engine utilizes an SVM classifier that can be trained using either SIFT or SURF features.

The solution was evaluated to establish classification accuracy, page load times when accessed from different networks and its cross-browser support. The system achieved 73.3% overall accuracy rate when tested using images identical to images end users would upload. Page load times were considerably long on GPRS and 2G network tests. However, they were comparable to average page load times users would experience when accessing google search engine pages from similar networks. Cross-browser support tests indicated that the system is fully compatible with all popular mobile and desktop browsers. Based on the evaluation results, we concluded that it is feasible to develop a crop disease diagnosis application that in addition to being accessible through personal computers and smartphones can also be accessed through any internet enabled feature phones.

Contents

PLAGIARISM DECLARATION	I
DEDICATION	II
ACKNOWLEDGEMENTS	III
ABSTRACT	IV
CONTENTS	V
LIST OF FIGURES	VIII
LIST OF TABLES	X
LIST OF ACRONYMS	XI
1 INTRODUCTION	12
1.1 MOTIVATION	12
1.2 PROBLEM STATEMENT	13
1.3 RESEARCH QUESTIONS.....	13
1.4 SCOPE AND LIMITATIONS	14
1.5 THESIS ORGANISATION	14
2 BACKGROUND	15
2.1 MAIZE	15
2.1.1 <i>Grey leaf spot</i>	15
2.1.2 <i>Common rust</i>	16
2.2 COMPUTER VISION	16
2.2.1 <i>Digital images</i>	17
2.2.2 <i>Digital image processing</i>	17
2.2.3 <i>Machine learning</i>	19
2.2.4 <i>Bag of features</i>	23
2.3 CONCLUSION	23
3 LITERATURE REVIEW	25
3.1 INNOVATIVE TECHNOLOGY APPLICATIONS IN AGRICULTURE.....	25
3.2 CROP DISEASE DIAGNOSIS USING IMAGES	25
3.3 REMOTE-SENSING APPLICATIONS.....	27
3.3.1 <i>Thermography</i>	27
3.3.2 <i>Hyperspectral imaging</i>	27
3.3.3 <i>Chlorophyll fluorescence imaging</i>	28
3.3.4 <i>Studies utilizing remote-sensing techniques</i>	28
3.3.5 <i>Advantages of remote-sensing techniques</i>	28
3.3.6 <i>Disadvantages of remote-sensing techniques</i>	29

3.4	NEAR-RANGE APPLICATIONS.....	29
3.4.1	<i>Studies that utilize near-range techniques</i>	30
3.4.2	<i>Advantages of near-range applications</i>	30
3.4.3	<i>Disadvantages of near-range applications</i>	30
3.5	SMARTPHONE BASED APPLICATIONS.....	31
3.5.1	<i>Advantages of smartphone based applications</i>	31
3.5.2	<i>Disadvantages of current smartphone based applications</i>	31
3.6	CONCLUSION	32
4	DESIGN AND ARCHITECTURAL OVERVIEW	33
4.1.	DESIGN CONSIDERATIONS	33
4.2.	APPROACH	33
4.2.1.	<i>Approach to detecting and classifying diseases from images</i>	33
4.2.2.	<i>Approach to accessibility</i>	34
4.2.3.	<i>Approach to ensuring usability</i>	35
4.3.	ARCHITECTURAL OVERVIEW.....	35
4.3.1.	<i>Use cases</i>	36
4.3.2.	<i>Activity flow charts</i>	37
4.1	CONCLUSION	38
5	IMPLEMENTATION	39
5.1	RESPONSIVE WEB CLIENT	39
5.2	SYSTEM ADMINISTRATOR PORTAL.....	42
5.3	APPLICATION SERVER	43
5.4	CONCLUSION	44
6	EVALUATION AND RESULTS	45
6.1	DATA COLLECTION.....	45
6.1.1	<i>Data preparation</i>	45
6.2	HARDWARE AND SOFTWARE USED	47
6.3	EVALUATION OF THE CROP DISEASE DIAGNOSIS ENGINE.....	47
6.3.1	<i>SURF-SVM tests' results</i>	48
6.3.2	<i>SIFT-SVM tests' results</i>	50
6.3.3	<i>Evaluation of established ideal classifiers</i>	55
6.4	OVERALL SYSTEM EVALUATION	57
6.5.	CONCLUSION	60
7	CONCLUSION.....	61
7.1	ANSWERS TO RESEARCH QUESTIONS	61
7.2	FUTURE WORK	62
8	REFERENCES	63
9	APPENDICES	72

APPENDIX A: BAG OF FEATURES FILE	72
APPENDIX B: SVM CLASSIFIER MODEL	72
APPENDIX C: TRAINING RESULTS	72
APPENDIX D: DIAGNOSIS RESULTS	72

List of Figures

<i>Figure 2-1. A maize leaf infected by Grey Leaf Spot</i>	<i>16</i>
<i>Figure 2-2. Maize leaf infected by common rust.</i>	<i>16</i>
<i>Figure 2-3. A simplified model of an artificial neural network indicating input layer, hidden layers and output layer</i>	<i>21</i>
<i>Figure 4-1 A high level architectural overview of the system showing main modules. Pointed arrows indicate communication between different modules</i>	<i>36</i>
<i>Figure 4-2 System use cases from both the system administrator and system user perspectives.</i>	<i>36</i>
<i>Figure 4-3 Flow diagram of system administrator using the administrator panel to set up a classifier</i>	<i>37</i>
<i>Figure 4-4 Flow diagram of a system using when diagnosing images using the responsive web client.....</i>	<i>37</i>
<i>Figure 5-1 An image of the application’s home is rendered by a windows 10 desktop chrome browser.</i>	<i>39</i>
<i>Figure 5-2 The application’s results page as rendered by desktop version of Chrome browser. The image on the right is an example of a diagnosed disease and the image in the “Your Image Column” is the image the user uploaded. The other columns show the diagnosed di.....</i>	<i>39</i>
<i>Figure 5-3 Application’s home page showing preview of images the user is about to submit for diagnosing.....</i>	<i>40</i>
<i>Figure 5-4 Results page showing diagnosis results of images shown in figure 5-3. Only the top two results are shown as other results were beyond screen print’s reach.....</i>	<i>40</i>
<i>Figure 5-5 Home page rendered on a Nokia C200 feature phone emulator’s opera mobile browser</i>	<i>41</i>
<i>Figure 5-6 Home page rendered on a Google Nexus 5 smartphone emulator’s mobile chrome browser.</i>	<i>41</i>
<i>Figure 5-7 Admin Panel Interface showing required fields when SURF is selected as a feature algorithm.....</i>	<i>42</i>
<i>Figure 5-8 Admin Panel Interface showing required fields when SURF is selected as a feature algorithm.....</i>	<i>42</i>
<i>Figure 5-9 A pop up used for configuring data classes for the model. The system admin has to specify the full path to training data folder, full path to cross validation data and the class name.</i>	<i>43</i>
<i>Figure 6-1 Typical images in our testing data sets. No background noise is removed so that images resemble images our system users are likely to upload for diagnosis.</i>	<i>46</i>
<i>Figure 6-2 Effects of varying number of clusters while keeping Hessian Threshold fixed at 200 and training data at 100 images per class when training SURF-SVM classifier.</i>	<i>48</i>
<i>Figure 6-3 Effects of varying the hessian threshold clusters while keeping number of clusters fixed at 100 and training data at 100 images per class when training a SURF-SVM classifier. .</i>	<i>49</i>

<i>Figure 6-4 Effects of varying training data quantity while keeping number of clusters fixed at 100 and Hessian Threshold at 100 when training a SURF-SVM classifier.</i>	<i>50</i>
<i>Figure 6-5 Effects of varying number of clusters when training a SIFT-SVM classifier.....</i>	<i>51</i>
<i>Figure 6-6 Effect of varying number of octave layer when training a SIFT-SVM classifier.....</i>	<i>52</i>
<i>Figure 6-7 Effects of varying contrast threshold when training a SIFT-SVM classifier.....</i>	<i>52</i>
<i>Figure 6-8 Effects of varying edge threshold when training a SIFT-SVM classifier.</i>	<i>53</i>
<i>Figure 6-9 Effects of varying Sigma when training a SIFT-SVM classifier.....</i>	<i>54</i>
<i>Figure 6-10 Effects of varying quantity of training data when training a SIFT-SVM classifier... </i>	<i>54</i>
<i>Figure 6-11. Example of misclassified Grey Leaf Spot image.</i>	<i>57</i>
<i>Figure 6-12. Example of Common Rust validation set images.</i>	<i>57</i>
<i>Figure 6-13. Example of misclassified Common Rust image.....</i>	<i>57</i>
<i>Figure 6-14. Example of Grey Leaf Spot validation set images.</i>	<i>57</i>

List of Tables

<i>Table 2-1. An example of a confusion matrix of a binary classifier with yes and no classes.....</i>	<i>22</i>
<i>Table 2-2 An example of a confusion matrix for a binary classifier indicating terms often used in calculations of classifier's performance measures.</i>	<i>22</i>
<i>Table 6-1 Data collected from www.plantvillage.org before any pre-processing.....</i>	<i>45</i>
<i>Table 6-2 Data after cropping training set images to create more training images.....</i>	<i>46</i>
<i>Table 6-3 Confusion Matrix of the best SURF-SVM classifier.....</i>	<i>55</i>
<i>Table 6-4 Confusion matrix of the SIFT-SVM classifier</i>	<i>56</i>
<i>Table 6-5 Average load times of system pages compared to Google Search Engine load times</i>	<i>58</i>
<i>Table 6-6 Compatibility test of system's web pages on popular browsers.....</i>	<i>59</i>

List of Acronyms

3G	In networking, it refers to third generation of wireless mobile telecommunications technology.
4G	In networking, it refers to fourth generation of wireless mobile telecommunications technology
ANN	Artificial Neural Network
API	Application Programming Interface
CSS	Cascading Style Sheet
DTO	Data Transfer Object
GPRS	General Packet Radio Service
HCI	Human–computer interaction
HTML	Hyper-Text Mark-up Language
MNF	Minimum Noise Fraction
MVC	Model-View-Controller
RWD	Responsive Web Design
SVM	Support Vector Machines
XAML	Extensible Application Mark-up Language

1 Introduction

It is estimated that the world's population will grow by an average of 1.8% in 2016, resulting in about 83 million more people to feed [1]. This population growth will add an extra weight to an age-old problem; ensuring that the world is food secure. Looking at our current food security status and considering the future, this problem becomes particularly significant for those of us living in sub-Saharan Africa. Sub-Saharan Africa's population as per 2015's estimates stood at over 800 million people [2]. Per World Food Program's 2015 report [3], a massive 220 million people were undernourished. To be food secure in 2050 when our population is estimated to reach 1.5 billion people, it is estimated sub-Saharan Africa will require 360% as much food production as in 2006 [2].

Since time immemorial, political leaders, business leaders, farmers, academics and the population at large, each in their respective ways have been trying to solve the issue of food security. Globally, considerable progress has been achieved over the years. Reports indicate that about 795 million people were undernourished in 2015, down 167 million over the last decade, and 216 million less than in 1990-1992 [3]. However, as indicated by sub-Saharan numbers, the challenge persists.

Due to the multifaceted nature of the food security problem, a wide range of solutions by different sectors have been proposed and implemented. One novel approach by computer scientists that is in active research is the use of computer vision and machine-learning techniques to develop applications that diagnose crop diseases from images of host plants; thus, allowing earlier treatment of diseases, consequently improving agricultural yields.

Leveraging on this approach, we have developed a more accessible prototype application for detection of two of the most common maize foliar diseases, Common Rust and Grey Leaf Spot. The developed application is accessible through multiple devices; ranging from feature phones which are prevalent in Sub-Saharan African to smartphones and personal computers, more common in the Western Countries. Indeed, the only requirements for accessing the developed application are internet connectivity and a browser that can render HTML and CSS. When implemented at a larger scale, the developed prototype can empower farmers, especially those based in rural Sub-Saharan regions whom most previous solutions have overlooked, to conveniently diagnose their crops and act accordingly.

1.1 Motivation

A considerable amount of research on utilizing image-processing and computer-vision algorithms to detect and classify crop diseases has been carried out before. Unfortunately, the studies carried out over the years have primarily focused on developing traditional desktop/laptop applications and a few others on high end smartphone applications. Thus, making them inaccessible to many subsistence farmers, especially those in sub-Saharan Africa where ownership of personal computers and smartphones is still not as prevalent as in developed economies [5]. As per Ericsson 2015 mobility report [6], there were 690 million

mobile subscriptions in sub-Saharan Africa by end of 2015 and only a mere 170 million of these subscriptions were smartphones.

Despite these low numbers of smartphones and personal computers, it is not all gloom and doom when it comes to making such novel applications accessible to these populations. Shipment of feature phones, which already amounted for 20% to 30% of mobile subscriptions across Sub-Saharan region by 2014 [7] grew by an average of 31% between 2014 and 2016 while smartphone shipments fell by 5.2% within the same period [8].

A crop disease diagnosis application that can also be accessed through feature phones would be accessible to a significantly higher population. Consequently, it would be more effective in ensuring world food security than most previous solutions that have primarily targeted personal computers and smartphones.

1.2 Problem statement

Several previous studies, utilizing different combinations of computer vision and machine-learning algorithms, have successfully developed applications that can satisfactorily detect and classify crop diseases. However, these studies have primarily developed applications that are exclusively accessible through either personal computers or high end smartphones making them inaccessible to many subsistence farmers, especially those in sub-Saharan Africa where both smartphones and personal computers ownership is minimal.

With successes of the previous studies in mind, we wish to establish a possibility of developing a crop disease diagnosis application utilizing computer vision and machine learning techniques, that in addition to being accessible through personal computers and smartphones, can also be accessed through internet enabled feature phones thus making it accessible to a larger population.

1.3 Research Questions

To address the problem statement stated above, the following questions were posed:

- 1) Can all algorithms necessary for diagnosing images successfully execute and return the results before connection time-out for clients?

Algorithms often used for detection and classifying of crop diseases are resource intensive and can execute for lengthy periods. To prevent a device from endlessly waiting for a server response, browsers such as Internet explorer have a default time limit after which they time out and show an error. For our solution to be usable, all the necessary processing should be completed within this time out period and results sent back to users.

- 2) Can the system still be accessible and usable to users connecting from slow networks such as GPRS?

Although coverage of modern networks such as 3G and 4G has increased significantly over the last few years, there are still areas, especially in rural Sub Saharan Africa,

where prevalent networks are still the older and much slower GPRS and Edge networks. The primary objective of our study is to develop a highly accessible solution and our target users are mainly rural subsistence farmers. It therefore is crucial that our application can still function properly even when connecting from such slow networks.

3) What is the highest classification score we can get?

Given a limitation that our system must execute all necessary algorithms for diagnosing crop diseases and return results within limited time periods, will this necessity of speed conflict with accuracy of results?

1.4 Scope and limitations

As outlined, the primary focus of this study is determining the feasibility of developing a highly accessible crop disease diagnosis system based on approaches taken by previous successful studies. Our definition of highly accessibility is that in addition to be accessible through devices that previous studies have targeted, desktops, laptops and smartphones, our system also must be accessible through feature phones which are more prevalent in Sub-Saharan region.

Two commonly occurring maize foliar diseases, Common Rust and Grey Leaf Spot were selected as a case study. The reason maize was selected is that it is the most widely grown and consumed staple crop in Africa. The two diseases were chosen because they have very identical symptoms, especially during early stages of infection, and we believe a system that can successfully differentiate between the two will most likely also be able to differentiate diseases that have non-identical symptoms.

1.5 Thesis organisation

The rest of this thesis is organised as follows. Chapter 2 presents the background concepts relevant to understanding the rest of this study. Chapter 3 puts the study into perspective by underlining the background to the study and the context under which it occurs. Chapter 4 discusses design considerations and provides an architectural overview of the developed prototype. Chapter 5 follows up on chapter 4 by providing more implementation details of different components of the system. Chapter 6 gives details of the experimental setup we used to evaluate how well the system meets its objectives. Finally, chapter 7 concludes the study by providing answers to the posed research questions and suggests possible extensions to this research.

2 Background

This chapter introduces some core concepts underlying the study. It starts with a brief discussion of maize and an overview of the diseases selected for this study. In the subsequent sections, an overview of the main concepts of computer vision and machine learning used in crop disease detection and classification is given.

2.1 Maize

Maize (*Zea mays*) is a plant that is grown widely throughout the globe in a range of agro-ecological environments. It belongs to a large family of monocotyledonous flowering plants referred to as “poaceae” or grasses [9]. It is used as food, animals’ feed and as a source of industrial raw materials such as corn-starch, maltodextrins, corn oil and corn syrup.

Believed to have been introduced into Africa by the Portuguese in the 16th Century [10], maize today is the most widely grown and consumed staple crop in Africa [11]. It currently covers over 25 million hectares in sub-Saharan Africa and is mainly grown on smallholder farms [12]. It accounts for about 20% of the calorie intake of 50% of the sub-Saharan population.

Below, two of the most commonly occurring disease pathogens affecting maize in sub-Saharan African are discussed.

2.1.1 Grey leaf spot

Grey leaf spot, figure 2-1, is a fungal disease caused by a pathogen called *Cercospora Zeina* and is recognized as one of the most significant yield limiting maize diseases. In Africa, it was first observed causing economic losses in maize fields in South Africa during the 1990/91 growing season and has since been reported as being widespread in Ethiopia, Kenya, Malawi, Mozambique and Zimbabwe and to a lesser extent in the Congo, Nigeria, Tanzania and Zambia [12].

Its symptoms are initially observed on the lower leaves of the host maize plant. The immature lesions are like lesions caused by other foliar maize pathogens and appear as small irregular shaped tan spots with yellow or chlorotic borders of about 1 to 3 mm, while mature lesions are more distinct with rectangular shape (5 to 70 mm long and 2 to 4 mm wide) and run parallel with leaf veins [13].

Losses associated with grey leaf spot occur when photosynthetic tissue is rendered non-functional due to lesions or the blighting of entire leaves [14]. Most sub-Saharan countries suffer considerable yield losses due to grey leaf spot yearly. Losses of 29% to 69% have been reported in Malawi [15] and in Tanzania. It can cause yield losses of up to 40% [16]. In South Africa, losses usually vary between 30% to 40% [17] annually.

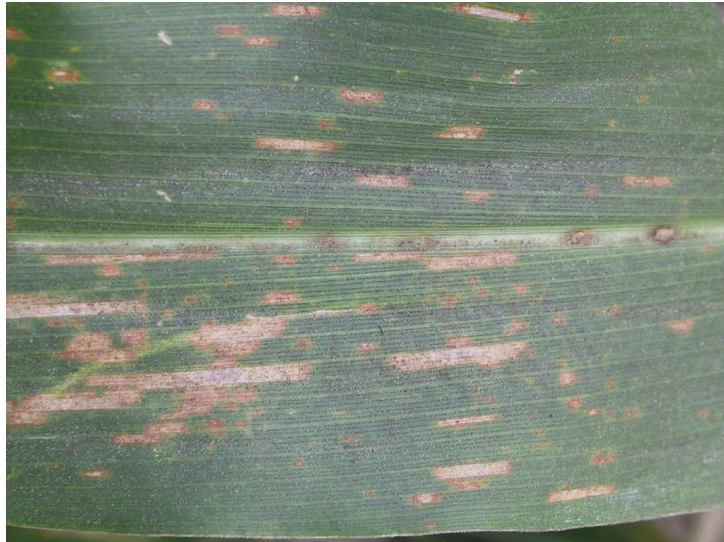


Figure 2-1. A maize leaf infected by Grey Leaf Spot

2.1.2 Common rust

Common rust, shown in figure 2-2, is the most frequently occurring maize rust [18]. It is caused by a fungus called *Puccinia sorghi* and occurs in most maize growing regions.

Its symptoms begin as light yellow spots about the size of a pinhole. As it advances, these chlorotic spots develop into reddish brown pustules, which rupture to reveal the presence of small, cinnamon-brown powdery spores. With time, the pustules become darker brown to black [19].

Yield losses associated with common rust are not as devastating as those of Grey Leaf Spot. Estimates of reduction in grain weight vary from about 3%-8% for each 10% of leaf area infected. However, yield losses as high as 25% have been measured and in some extremely susceptible hybrids', yields may be reduced by as much as 75% [18].



Figure 2-2. Maize leaf infected by common rust.

2.2 Computer vision

Computer vision is a science discipline whose focus is the automated extraction of information from digital images [20]. It aims at using cameras for analysing or understanding scenes in the

real world [21]. Its recorded history can be traced back to a summer in 1966 at Massachusetts Institute of Technology when Marvin Minsky asked his undergraduate student Gerald Jay Sussman to spend the summer vacation linking a camera to a computer and getting the computer to describe what it saw [22]. Since then, computer vision has developed exponentially and now incorporates a diverse range of concepts from other disciplines of science such as computer graphics, image processing and machine learning [23].

In the next subsections, an overview of computer vision concepts often applied in the diagnosis of crop diseases is provided. First things first, what is a digital image and what is digital image processing?

2.2.1 Digital images

In everyday language, an image refers to a visual representation of something. Mathematically, an image can be defined as a two-dimensional function, $f(x, y)$, where x and y are spatial coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or grey level of the image at that point [24]. A digital image is an image $f(x, y)$ that is discretized both in spatial coordinates and in brightness: i.e. x , y and the amplitude values of f are all finite, discrete quantities. It is composed of a finite number of elements, each of which has a particular location and value, commonly referred to as pixels.

2.2.2 Digital image processing

Digital image processing is a well-developed field of digital signal processing that deals with systems that perform operations on digital images. It includes topics such as image enhancement, image compression, and correcting blurred or out of focus images. One of its earliest successful applications was in the newspaper industry when pictures were first sent by submarine cable between London and New York back in 1921 [24].

Digital image processing techniques and concepts commonly used in crop disease diagnosis and those relevant to understand the rest of this dissertation are introduced below. A more detailed introduction to the field of digital image processing field can be found in [23].

Image features

Pratt describes an image feature as a distinguishing primitive characteristic or attribute of an image [25]. Features can also be thought as significant properties of an object than can be used as part of input to processes that lead to distinguish the object from other objects or to recognize it [26]. Features can be based on intensity, colour, texture or a combination of these attributes.

There are two types of features that can be extracted from an image content; namely, global and local features. Global features describe the image as a whole and can be interpreted as particular properties of the image involving all pixels, while local features are those that aim to detect key-points within the image and describe regions around these key-points.

Good features are those that are distinctive and whose extraction is repeatable and precise. In addition, good features are reasonably invariant to illumination changes, and ideally, also to scaling, rotation and minor changes in viewing direction.

Due to the significant role they play in most computer-vision applications, the topic of image features has received considerable attention in the computer-vision research community, and as a result various successful algorithms have been developed. Two of the most frequently used image feature algorithms in diagnosis of crop disease are Scale Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF). Both are introduced below:

Scale-invariant feature transform (SIFT)

Scale-invariant feature transform, known as simply SIFT, is a popular image feature detector and descriptor first introduced by Lowe in 1999 [27]. SIFT algorithm's approach to feature detection is to only select key features that are invariant to translation, scale, rotation, and partially invariant to illumination changes as well. The detected features can be stored in a 128 members' vector referred to as a SIFT descriptor which can then be used for further analysis. The detail of the operation is best left to the source material [27] as it is beyond the scope of this dissertation.

SIFT has several advantages when compared to other feature detectors and descriptors. Not only are SIFT features reasonably invariant to rotation, scaling and to an extent illumination, SIFT features are also robust to occlusion and clutter. However, compared to other approaches that give comparable results such as SURF which is discussed next, SIFT is computationally expensive.

Speeded Up Robust Features (SURF)

Speeded Up Robust Features algorithm, commonly referred to as just SURF, is another popular feature detector and descriptor. It was first presented at the European Conference on Computer Vision in 2006 by Bay et al [28]. It is based on the same principles and follows similar steps to SIFT, however, it uses a different scheme and its resulting descriptor can consist of either 64 elements or 128 elements for more feature distinctiveness. Full details of SURF can be found on the source material [28], and a detailed comparison of SURF and SIFT can be found in [29].

Like SIFT features, SURF features are also invariant to scale, rotation and to an extent illumination. In addition, SURF inventors claim that it is considerably faster than SIFT while achieving comparable results [28]. However, Panchal et al in their comparison study found not only does SURF produces less features than SIFT but that is more sensitive to changes in illumination [30].

Image segmentation

Image segmentation refers to the task of partitioning an image into regions of similar attribute or attributes. It divides an image into regions that correspond to different objects or different parts of an object. It simplifies and often increases the efficiency of the subsequent steps of analysis. Typical attributes for segmentation include colour intensities, image edges, and image texture [25].

The most commonly used segmentation techniques in crop disease diagnosis applications are Otsu’s thresholding method and k-means clustering. Both are briefly introduced below.

Otsu’s thresholding method

Published in 1979 by Nobuyuki Otsu [31], Otsu’s method assumes that the image contains two classes of pixels, foreground pixels and background pixels. It then calculates the optimum threshold separating the two classes so that their intra-class variance is minimal or alternatively, so that their inter-class variance is maximal.

It has been successfully applied on different segmentation problems and is one of the most popular methods for segmenting grey level images. However, it has some drawbacks. It assumes the histogram of an image is bimodal, does not work well with variable illumination, breaks down when the two classes have extremely different sizes and makes no use of spatial coherence, nor any other notion of object structure.

K-means clustering

K-means clustering groups objects based on attributes into k number of clusters where k is a positive integer. Although the term “k-means” was first used by James MacQueen in 1967 in his paper titled “Some methods for classification and analysis of multivariate observations” [32], the idea can be traced back to Hugo Steinhaus in 1957 [33]. It is one of the most commonly used algorithms to solve clustering problems.

It aims to partition n data points into k clusters in which each data point belongs to the cluster with the nearest mean, serving as a prototype of the cluster. It achieves this by iteratively assigning each data point to one of k clusters based on the features that are provided. To understand k-means, one can assume there are n data points $x_i, i = 1 \dots n$ that must be partitioned into k -clusters and the goal is to assign a cluster to each data point. K-means aims to find the positions $\mu_i, i = 1 \dots k$ of the clusters, $\mathbf{S} = \{S_1, S_2, \dots S_k\}$, that minimize the square of the distance from data points to the cluster: i.e. it aims to find:

$$\arg \min_s \sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|^2 \quad (1)$$

where μ_i , is the mean of points in S_i . More details on k-means clustering can be found in [32].

K-means approach to segmentation has several benefits that include its fast-computational speed and the fact that most open-source computer vision libraries are distributed with its implementation. However, it suffers from a major drawback; the number of desired clusters, k , needs to be set beforehand.

2.2.3 Machine learning

To solve a problem on a computer, an algorithm, a sequence of repeatable instructions that should be carried out to transform input to output is needed. In conventional computing, this logic is often explicitly coded by a programmer after carefully learning the problem and determining logical steps to solve it. However, not all computer vision problems can be explicitly coded. Machine learning is a subfield of computer science that explores the study and

construction of algorithms that can learn from and make predictions based on data. Using algorithms that iteratively learn from data, machine learning allows computers to find hidden insights without being explicitly programmed where and what to look for. In crop diseases detection and classification applications, machine learning algorithms are often used to perform classification.

Machine learning algorithms can be categorized into two general groups: supervised learning and unsupervised learning. Supervised learning algorithms are trained using a set or sets of inputs along with corresponding correct outputs, and the algorithms' task is then to generate a general rule that maps inputs to outputs. The generated rule can then be used in similar applications to classify additional unlabelled data. Examples of supervised learning algorithms include Support Vector Machines and K-Nearest Neighbours algorithms. Supervised learning is commonly used in applications where historical data predicts likely future events and where there are well known outputs to given inputs. In unsupervised learning, the algorithm is only given input data without any corresponding data and its task is then to explore the data and find some structure within. K-means algorithm introduced in section 2.2.2 is an example of unsupervised learning algorithm. Other examples include Self-Organising Maps and Independent Component Analysis algorithms. Unsupervised learning can be used to discover image features and to determine their classes as well.

Crop disease diagnosis applications often make use of supervised learning algorithms; two of the most commonly used algorithms are introduced below.

Artificial Neural Networks (ANN)

An artificial network (ANN), is a model of computation inspired by the neural structure of the mammalian cerebral cortex. Natural neurons receive signals through synapses located on the dendrites or neuron membranes. When the received signal surpasses a certain threshold, the neuron is activated and emits a signal. The emitted signal might be passed on to another synapse and might activate other neurons. This complexity is highly abstracted in modelling of Artificial Neural Networks. ANNs are typically organized in layers. Layers are made up of several interconnected nodes that contain activation functions. Patterns are presented to the network through the input layer, which communicates to one or more hidden layers where the actual processing is done through a system of weighted connections. The hidden layers then link to an output layer where the answer is output. Most ANNs contain some sort of learning rule that modifies the weights of the connections according to the input patterns presented to it. A simplified example of a typical ANN is shown in figure 3.

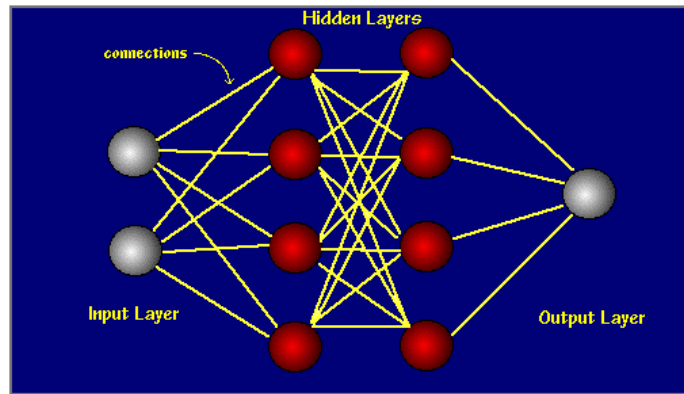


Figure 2-3. A simplified model of an artificial neural network indicating input layer, hidden layers and output layer

Since what is generally considered the first artificial neural model was introduced by McCulloch et al in 1943 [34], numerous different models that are accepted as ANNs have been developed. Though they may vary in functions, learning rules, topology and other attributes, they all share one main advantage: they allow modelling of physical phenomena in complex systems without requiring explicit mathematical representations or without requiring exhaustive experiments. ANNs are best suited for capturing associations or discovering irregularities within a set of patterns; deriving solutions for problems where the volume, number of variables or diversity of data varies greatly; and for deriving solutions of problems where relationships between variables are vaguely understood or too difficult to describe adequately with conventional algorithms.

They however have some drawbacks. ANNs are in a sense black boxes. Apart from defining the general structure of a network, the user has no other role except to feed it with training data and watch it train and await the output model. They are also comparatively slower to train than more modern machine learning algorithms such as Support Vector Machines (SVMs). In addition, ANNs sometimes overfit if training goes on too long, i.e. they consider noise as part of the pattern.

Support Vector Machines (SVM)

Support Vector Machines (SVMs), are another popular machine learning algorithm commonly used for classification and regression analysis. The original SVM algorithm can be traced back to 1963 by Vapnik and Chervonenkis. In 1992, Boser et al proposed a non-linear variant, and the current standard incarnation was published by Cortes and Vapnik in 1995 [35] [36].

SVMs are based on the concept of decision planes that define decision boundaries. A decision plane can be defined as a plane that separates between a set of objects having different class memberships. Support Vector Machines are primarily classifier methods that perform classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. The success of SVMs can be attributed to their ease of use, good generalization performance and the fact that the same algorithm can solve a variety of problems with little tuning. Several studies have been carried to compare SVMs and ANNs on various classification problems and each study has come to its own conclusions that are not necessarily in agreement with others. However, most acknowledge that in most classification problems,

SVMs and ANNs produce comparatively similar results and that SVMs’ results better reflect the nature of input data [37] [38].

Evaluating a machine learning classifier

When using a machine learning algorithm to classify data, one often needs to establish how reliable it is. One clean and unambiguous way to present such an evaluation that consequently is also used in this study is the use of a confusion matrix.

In simple terms a confusion matrix is a table that is used to describe the performance of a classifier on a set of test data for which the true values are known. Table 2-1 is an example of a confusion matrix for a binary classifier that classifies inputs as either yes or no.

Table 2-1. An example of a confusion matrix of a binary classifier with yes and no classes

N=120	Predicted: No	Predicted: Yes
Actual No	39	21
Actual Yes	13	47

In the above example, a sample of 120 inputs was put into a classifier to determine if they were yes or no. Out of the 60 known “no” inputs, the classifier correctly classified 39 and out of the 60 known “yes” inputs, it correctly classified 47. In statistical terms, 39 is referred to as True Negative (TN), 21 as False Positive (FP), 13 as False Negative (FN) and 47 as True Positive (TP) as presented in table 2-2.

Table 2-2 An example of a confusion matrix for a binary classifier indicating terms often used in calculations of classifier’s performance measures.

120	Predicted: No	Predicted: Yes	
Actual No	TN = 39	FP = 21	60
Actual No	FN = 13	TP = 47	60
	52	68	

From the confusion matrix, measures often used in evaluating classifiers can be calculated. The measures, their formulae and calculations as per above example are as follows:

- **Accuracy**

This measure refers to the overall correctness of the model and is calculated as the sum of correct classifications divided by the total number of classifications.

$$\frac{TP+TN}{Total} = \frac{47+39}{120} = 0.7167 \quad (2)$$

- **Precision**

This is a measure which is evaluates the accuracy provided that a specific class has been predicted.

$$\frac{TP}{TP+FP} = \frac{47}{47+21} = 0.6912 \quad (3)$$

- **Recall**

This a measure of the ability of a classifier to select instances of a certain class from a data set. It is commonly also called sensitivity, and corresponds to the true positive rate.

$$\frac{TP}{TP+FN} = \frac{47}{47+13} = 0.783 \quad (4)$$

- **Specificity**

This measure corresponds to recall measure explained above. It is commonly used in two class problems where one is more interested in a particular class and it corresponds to the true-negative rate.

$$\frac{TN}{TN+FP} = \frac{39}{39+21} = 0.65 \quad (5)$$

2.2.4 Bag of features

Bag of Features, BoF, is a popular technique for representing images as unordered collections of local features for use as input into classifying algorithms. It is inspired by another popular concept called Bag of Words that is used in textual information retrieval.

In Bag of Words model, one represents a document as a normalized histogram of word counts. Commonly, one counts all words from a dictionary that appear in the document, and presents them in a sparse vector where each element is a term in the dictionary and the value of that element is the number of times the term appears in the document divided by the total number of dictionary words in the document. This dictionary may exclude non-informative words such as articles and may represent a set of synonyms with a single term. The Bag of Features image representation is analogous. A visual vocabulary is constructed to represent the dictionary by clustering features extracted from a set of training images. The image features represent local areas of the image, just as words are local features of a document. To generate a discrete vocabulary from a large number of local features sampled from the training data, clustering is often used.

Bag of Features is often used to present images in problems where position and orientation of features within an image are not of primary significance and fixed vector lengths are required irrespective of number of detections. It is very successful when used in combination with machine learning algorithms to classify images according to objects they contain.

2.3 Conclusion

This chapter has introduced major concepts relevant to understanding this thesis. We introduced maize and two of its diseases selected as a case study for this research. We introduced the significant role maize plays in Sub-Saharan Africa and outlined the average

losses incurred annually due to Common Rust and Grey Leaf Spot diseases. In addition, we provided theoretic knowledge of computer vision and machine learning concepts applied in this research. Where applicable, we introduced the advantages and disadvantages of the introduced concepts.

In the next chapter, we aim to highlight the current status quo of utilizing computer vision and machine learning concepts in fighting crop diseases.

3 Literature review

This chapter puts the study into perspective by discussing the background to the study and the context in which the research occurs.

3.1 Innovative technology applications in Agriculture

Agriculture has become much more than a means to feed the ever-growing global population. Plants have become an important alternative source of energy for our energy-hungry economies and are of vital importance in our fight against global warming; hence, there is a need to manage worldwide production of agricultural commodities more efficiently. Consequently, there has been an increase in the use of innovative technologies in agriculture to meet this need; a trend often referred to as “Precision Agriculture”.

As Crookston puts it, Precision Agriculture is one of the most significant revolutions to ever come about in agriculture [39]. It involves better management of farm inputs such as fertilizers, herbicides, seeds, fuel, to mention but a few. It offers to improve crop productivity through improved resources’ management [40] [41] while taking into consideration environmental impact [42] [43] [44] [45]. Its origins can be traced back to the mid 1980’s as a means to improve the application of fertilizers by varying rates and blends as needed within fields. It surprisingly began with two contrasting philosophies [46]. The first was exemplified by the “Farming by soil” school [47]. This school’s philosophy advocated the use of soil sampling and customized management of farm inputs by soil mapping unit. The second was exemplified by the “Soil Sampling Management Zone” school, later known as “site-specific crop management” [48] [49] [50]. This school proposed that management zones are relatively homogeneous sub-units of farm fields that can each be managed with a different, but uniform customized management practice.

Since then, the field has grown exponentially and attracted professionals from various, traditionally non-farming related fields. As a result, additional practices have evolved. Amongst, chief contributors to the field are computer vision practitioners. Over the past decades, they have proposed several solutions for some of the most pressing issues in agriculture. Their contributions include systems for automatic guidance of agricultural vehicles and machinery; systems for sorting and grading agricultural yields [51] [52] ; systems that allow for targeted application of herbicides [53] [54] [55]; and last but not least, computer vision based crop disease diagnosis systems. In this thesis, it is the latter that we are interested in.

3.2 Crop disease diagnosis using images

The potential yield of agricultural and horticultural crops worldwide is significantly hampered by innumerable kinds of pathogens. Roughly, crop diseases are directly responsible for losses ranging between 18% and 20% of global agricultural productivity [56] [57]. Accurate, accessible and timely diagnosis of crop diseases is essential to offset these losses.

Crop pathogens often produce some distinct visual cues in various parts of the plant; roots, kernels, fruits, stems and leaves. Historically, farmers have relied on human raters, trained and untrained, to observe any visual cues present on host plants and diagnose what pathogens may have infected their crops. This diagnosis method is still very popular even in this modern day age, especially in developing regions such Sub-Saharan Africa. However, there are some inherent disadvantages associated with visual assessment. By virtue, human visual assessment is a subjective task and prone to psychological and cognitive phenomena that may lead to bias, optical illusions and, ultimately, to error. Bock et al [58] elaborate more on these and other related disadvantages.

Computer vision researchers have for the past decades been investigating ways to mitigate this dependency on human raters. Numerous studies have been carried out resulting in various automated crop disease diagnosis systems. In this thesis, we are particularly interested in those systems that apply digital image processing and computer vision techniques in their architectures.

Computer vision based crop disease diagnosis systems typically involve five main procedures: image acquisition whereby images of host plants are taken or obtained by other means; image pre-processing which involves a series of image operations that enhance the quality of images in order to remove defects such as geometric distortion, noise, and non-uniform lighting; segmentation for subdividing images into their constituent regions of interest; feature extraction whereby distinguishing features that can be used for discriminating patterns in different categories are selected; and finally detection and classification where patterns of interest are detected and classified accordingly.

For the above mentioned main procedures, there are innumerable possible techniques that can be used to complete each. As an example; with algorithm options ranging from basic ones like Harris Corner Detector to scale invariant ones like SIFT to invariant and speed optimized ones like SURF, researchers are truly spoiled for choice when it comes to selecting what feature extraction algorithm or combination to use in their studies. The same applies to all other main procedures. Our aim is not to introduce alternative algorithms for any procedure nor provide supporting literature on why others are better than others; but rather, we aim to build onto previous studies to architect a more accessible solution. We therefore will not discuss much technical details of the underlying algorithms in our discussions. There is plenty of published literature on what algorithms are best suited for what tasks. An interested reader can refer to [59] [58] [60] [61] for more information. We have introduced some which are most relevant to this study in the preceding background chapter. This review rather aims to critique the underlying approaches and suggest what needs to change in order to architect more accessible solutions.

Previous studies that have employed computer vision techniques to architect crop disease diagnosis systems can be classified into two main categories, remote-sensing applications and near-range applications. More recently, a few studies have also explored the use of smartphone applications. We discuss all the three categories in detail and mention their benefits and shortcomings over the subsequent sections.

3.3 Remote-sensing applications

De Jong et al [62] define remote sensing as obtaining information about an object without having direct physical contact with it. In the context of this thesis, remote sensing applications are those that passively monitor crops, constantly acquiring images of the target crops/fields and actively analysing the acquired images to diagnose diseases based on symptoms that develop overtime without any human having to be physically present at the target fields. Thus, this category includes even those applications that may require some sort of physical apparatus to be set up at the target field prior to diagnosis as long as they don't require any human physical presence in the field after initial setup.

The idea of using remote-sensing technologies to diagnose crop diseases precedes digital photography. Its roots can be traced back to the late 1920's with aerial photography at the Texas Agricultural Experiment Station in College Station, Texas, when army pilots stopping at the station reported that cotton rot spots were readily visible from air even at high altitudes. Following up on the report, Neblette [63] hung a film camera over the side of the aircraft and photographed the fields at varying altitudes between 76m and 152m. The stark contrast between healthy cotton plants, bare soil where *Phymatotrichum* had killed the plants and infected regions made the spots stand out in the photographs and the vertical angle at which the images were taken allowed for comparative measures of healthy and diseased acreage.

Over the decades, as a direct result of technological advancements in photography, computer vision, machine learning and other technologies; newer, more accurate and faster approaches have been adopted. Thermography, hyperspectral imaging and chlorophyll fluorescence imaging are some of the most popular approaches to image acquisition for remote-sensing applications now used in place of the 1920's aerial photography, and the analysis and interpretation of images are now performed mainly using digital image processing and computer vision techniques. In the next subsections, we briefly introduce each of the above techniques; and state advantages and disadvantages associated with remote sensing remote sensing techniques.

3.3.1 Thermography

Plant temperature is negatively correlated to transpiration rate [64]. Depending on the nature of infection, pathogens have different effects on the temperature of the infected plant tissues. Pathogens that induce stomatal closure in plants lead to decreased transpiration rates and increased leaf temperature. The opposite is also true. The infrared radiation emitted by plants can be detected by thermographic cameras and the captured images analysed further using computer vision techniques to detect and classify pathogens.

3.3.2 Hyperspectral imaging

Hyperspectral imaging belongs to a class of techniques referred to as spectral imaging or spectral analysis. Its roots can be traced as far back as the early 1980's in National Aeronautics and Space Administration (NASA)'s Jet Propulsion Laboratory with the development of Airborne Imaging Spectrometer (AIS) [65]. A simple analogy to understand the concept better

is: the human eye sees visible light in three bands, red, green and blue, whereas hyperspectral imaging divides the spectrum into many more bands.

3.3.3 Chlorophyll fluorescence imaging.

Chlorophyll fluorescence is light re-emitted by plant chlorophyll molecules during return from excited to non-excited states. It has long been known that chlorophyll fluorescence emissions from plants provide indications of plants' photosynthesis performance [66] [67]. More recently, it has been proven that plants experiencing both biotic and abiotic stresses exhibit changes in their chlorophyll fluorescence emission patterns [68] [69], which can be captured through fluorescence imaging.

3.3.4 Studies utilizing remote-sensing techniques

Several studies have utilized the above explained image acquisition techniques in combination with computer vision techniques to successfully develop crop disease diagnosis systems. Examples of successful studies that utilize thermography to detect crop diseases include a study by Stoll et al [70] to detect *Plasmopara viticola* pathogen in a grapevine and a study by Oerke et al [71] to detect effects of downy mildew on cucumber leaves. Studies that have successfully utilized fluorescence imaging for crop disease diagnosis include that by Lins et al [72] to detect citrus canker in citrus plants using laser induced fluorescence spectroscopy. Similarly, multiple studies have successfully made use of hyperspectral imaging to detect and classify crop diseases. Examples include a study by Zhang et al [73] in which Significance Analysis of Microarrays (SAM) technique in combination with minimum noise fraction (MNF) transformation on hyperspectral images was used to detect late blight disease on tomatoes, and a study by Rumpf et al [74] in which Support Vector Machines were used to classify *Cercospora* leaf spot, sugar beet rust and powdery mildew on sugar beet plants based on hyperspectral images.

3.3.5 Advantages of remote-sensing techniques

There are several advantages of utilizing the remote sensing approach for crop disease diagnosis hence the interest from a large community of researchers. Remote sensing applications are generally non-invasive, non-contact and therefore non-destructive. They also require minimal human interference once initialized. In addition, they have an added advantage of economies of scale. Thus, when well designed and implemented, one such application can be used at an industrial scale to monitor several fields of interest at once. Moreover, there are several other advantages related to specific approaches of remote sensing. Thermography and fluorescence imaging have successfully been utilized to detect presence of diseases before any visual symptoms formed, paving way for them to be used as part of preventive measures to avoid major outbreaks of pathogens. Lindethal et al [75] and Oerke et al [76] in different studies successfully demonstrated pre-symptomatic thermographic detection of cucumber downy mildew. Stoll et al [77] also found characteristic thermal responses of grapevine leaves infected with *Plasmopara viticola* before any visible symptoms appeared. Similarly, Cséfalvay et al [78] successfully detected presence of *Plasmopara viticola* pathogen before any visual symptoms formed based on chlorophyll fluorescence images of the target grapevine.

Due to their multidimensional nature, hyperspectral images enable better characterization and identification of targets. Spectroscopic data analysis techniques can be used to extract chemical composition from each or an aggregate of pixels. Because of these combined features, hyperspectral imaging can greatly enhance our capability to identify materials and detect subtle features in an object. Thus, making detection and classification based on hyperspectral imaging fairly accurate.

3.3.6 Disadvantages of remote-sensing techniques.

It is common knowledge that each rose has a thorn: Each of the above introduced remote sensing techniques faces its own formidable challenges. Use of thermal imaging to detect pathogens has proved to be hard beyond laboratory settings. It is sensitive to environmental variations such as cloud cover and solar orientation. It is also difficult to identify plots of interest from thermal images during further processing: i.e. to separate crop canopy from soil in the background, and to adjust for different temperatures that may result from varying plant heights and environmental conditions [79]. Likewise, despite a number of successful laboratory studies that have used fluorescence imaging, this approach too still has major technical challenges outside laboratory settings, the main one being that its very sensitive to other abnormalities in photosynthesis. Papers by Scholes and Rolfe [80] and Chaerle et al [81] provide more detailed discussions on the technical challenges facing the use of chlorophyll fluorescence imaging for disease detection.

Even hyperspectral imaging which offers the most potential compared to other approaches discussed above, still has hurdles to overcome. One of the major challenges facing the use of hyperspectral imaging to detect pathogens is the selection of disease-specific spectral bands to use for further analysis. For example, Lu [82], Xing and Baerdemaeker [83], Nicolai et al [84], and ElMasry et al [85] all used hyperspectral imaging for detecting bruises in apples and all reported different findings. Lu [82] reported that 1000 nm to 1340 nm were best for bruise detection, while Xing and Baerdemaeker [83], Nicolai et al [84], and ElMasry et al [85] reported bands within range 558-960 nm were more suited.

Based on the research interest on the subject, there is no doubt that most of the above-mentioned challenges facing the use of remote sensing technologies to detect crop diseases will be resolved in time. However, regardless of whether these methods will overcome all the major technical hurdles and finally get implemented commercially, one major problem is likely to remain; accessibility to the poorest communities of the world which arguably need such solutions the most. The technology used in remote sensing applications is not easily accessible as the required devices are expensive and require special training to use. It is this challenge of accessibility that we aim to address in this study.

3.4 Near-range applications

In the context of this study, near range applications refer to those applications whose data acquisition techniques require physical presence of a human at the target field. In principle, this includes studies that quantify pathogens using molecular or immunological techniques and those that use hand held digital cameras for data acquisition. However, in this study we are

interested only in the latter. Our definition of hand held digital cameras does not include mobile phone cameras as those are discussed separately.

Digital camera technology has become relatively inexpensive and ubiquitous over the years, and the technical parameters of these simple, handheld devices such as the light sensitivity of the photo sensor, spatial resolution, or optical and digital focus have also improved significantly [86]. These advantages have consequently led to some researchers investigating ways in which to use consumer-level digital cameras as cheaper and more accessible alternative data sources for crop disease diagnosis systems.

3.4.1 Studies that utilize near-range techniques

Even though the idea of using consumer level cameras for plant disease identification has only gained attention from researchers in the last two decades, considerable progress has already been made. Several studies that range from those that focus on detecting a single disease of interest amidst other diseases to those that detect and discriminate different diseases have successfully utilized data acquired using consumer level cameras to detect crop diseases. Abdullah et al [87] successfully detected corynespora from rubber tree leaf images taken using a Fujifilm FinePix 6900 digital camera. Similarly, Huang [88] successfully detected and discriminated three diseases affecting Phalaenopsis seedlings namely, bacterial soft rot, bacterial brown spot and phytophthora black rot, using data acquired using a commercially available Sony XC-711 camera. A comprehensive discussion on more of these studies can be found in a review by Barbedo [89].

3.4.2 Advantages of near-range applications

Perhaps the greatest advantage of the approach of utilizing hand held digital cameras as data sources for crop disease diagnosis systems is their accessibility. As briefly indicated in the introduction of this section, digital cameras are relatively inexpensive; thus, making systems that use them as data sources accessible to a larger proportion of the population when compared to remote sensing applications.

In addition, compared to remote sensing technologies, digital cameras are easier to operate, reducing the effort required for training consumers to utilize their applications. Moreover, maintenance of digital cameras is relatively lower when compared to remote sensing technologies.

3.4.3 Disadvantages of near-range applications

The solutions discussed above require the images acquired using digital cameras to be uploaded on desktop or laptop computers for further processing and diagnosis. Very few households in Sub-Saharan Africa own such devices. In addition, to get to know what diseases have affected their crops, farmers would still have to travel to nearest centres where they can access computers; thus, resulting in potentially delayed responses to disease outbreaks.

Moreover, reports reveal that globally, ownership of stand-alone digital cameras is on the decline mainly due to the increasing popularity of smartphones. According to a report by

Mintel [90], sales of digital cameras in the United Kingdom fell by 29% between 2006 and 2011. The same report predicted by 2016, the market values would have decreased from the value of £843 million in 2006 to £523 million. A similar downward trend is also visible in the United States of America whereby according to Statista.com [91], only 124.42 million people owned digital cameras in the Spring of 2015 compared to 151.95 million in Autumn 2010. We could not find any specific reports on Sub-Saharan Africa, but it is safe to assume it has also followed the global trend as it too has witnessed a rapid increase in the ownership of mobile phones capable of taking photos.

3.5 Smartphone based applications

The successful uptake of the mobile phones throughout the world has significantly impacted economic development initiatives. In 2014, 1.57 billion people owned smartphones worldwide and predictions estimate the number will almost double by 2020, reaching 2.87 billion [92]. These coupled with the decreasing popularity of stand-alone digital cameras and other factors have inspired a growing number of researchers to investigate ways to build up on the success of systems that utilize digital cameras' images to architect solutions that utilize mobile phones images and are accessible through mobile phones.

The most popular amongst mobile phone based crop disease diagnosis application is undoubtedly, Plantix, an android based commercial application by a German based Progressive Environmental and Agricultural Technologies (PEAT) [93]. It allows anyone with an android device to upload images of infected crops and instantly get diagnosis results. In addition to providing diagnosis, it also provides steps to mitigate diseases and information on preventing diseases in the next growing season. It currently covers an impressive 60 crops worldwide and has prescriptions for over 200 crop diseases [94]. A comprehensive discussion of other studies that have explored the use of mobile phones for crop disease diagnosis can be found in a review by Pongnumkul et al [95].

3.5.1 Advantages of smartphone based applications

The biggest advantage of smartphone based solutions is their accessibility, anyone with a smartphone or access to one can have their crops diagnosed. In addition, smartphone applications provide diagnosis results almost instantaneously, all farmers must do is take images, upload them using the respective applications and within a matter of minutes have results back. This spares farmers' unnecessary trips to have their crop diagnosed and allows ample time to react to disease outbreaks.

3.5.2 Disadvantages of current smartphone based applications

Undoubtedly, mobile phone based solutions are the most accessible and cost effective amongst the solutions discussed thus far. However, they are still a few hurdles facing current solutions. Current solutions have mainly focused on developing applications exclusively for smartphones. Only about 34% of South Africans and 27% of Nigerians', two of the largest economies in Sub Saharan Africa, citizens owned smartphones in 2015 [96]. To make matters worse, those that do own smartphones are mostly young urban professionals, a demographic that is less likely to

participate in agricultural activities: thus, making the solutions inaccessible to those that undoubtedly need them the most, the rural subsistence farmers.

3.6 Conclusion

As evidenced in this chapter, there are already plenty of successful computers vision based crop disease diagnosis solutions; ranging from remote-sensing technology based systems which are well suited for commercial farmers, to smartphone solutions which are well suited for small scale farmers. However, as we have indicated, there is still a problem of accessibility, especially for rural subsistence farmers who can neither afford remote-sensing technologies nor have purchasing power to acquire smartphones and whom unfortunately, farming is most likely their only source of income.

In the next chapter, we provide an overview of the approach we took in this dissertation to address the issue of accessibility.

4 Design and architectural overview

As discussed in the preceding literature review chapter, a considerable number of successful studies have been carried out to develop systems for diagnosing crop diseases using computer vision techniques, some of which have enjoyed commercial success. However, as identified, there is still need to develop solutions that are more accessible, especially for small scale farmers in Sub Saharan Africa for most of which farming is their only livelihood.

This chapter provides an overview of the methodology we adopted in designing a crop disease diagnosis system that attempts to resolve this accessibility problem. It discusses factors we took into consideration and our approach to solving them, use cases and activity flows we envisioned when designing the system, and finally provides an architectural overview of the system.

4.1. Design considerations

When designing the system, we took into consideration many factors that would make the solution more accessible. Chief amongst is that the solution should be accessible through many devices with varying computational power. In addition to being accessible through various devices, it must not require any specialized hardware to access, thus even feature phones with minimal capabilities must be able to access the core functionality of the system.

Another important factor we considered was that the solution must be conveniently accessible. That is, users must not have to travel distances to access it. Ideally, a user must be able to take picture while at the farm, upload it to the system and almost instantaneously get diagnosis results.

In addition to the above considerations, we also considered other factors which we however decided not put too much emphasis on them due to scope limitations. These include that the system should be scalable enough to cater for diagnosing more crop diseases with minimal redesign in the future. Finally, given our target demographic, the system also must have minimal learning curve.

4.2. Approach

Bearing in mind the main objective of this study, developing a highly accessible crop disease diagnosis application, and having taken into consideration the factors discussed in the design considerations section, we decided on the following approach:

4.2.1. Approach to detecting and classifying diseases from images

As emphasised in the preceding chapters, our aim is not to come up with new techniques to detecting and classifying crop diseases from images, but rather, to adopt already proven techniques and architect a solution that is more accessible.

From our literature review, we identified one approach that seems to be the de' facto choice of recent studies and adopted it for our solution. This approach involves the of use of a

combination of Bag of Features and machine learning algorithms and is often referred to as just Bag of Features classification model. We introduced fundamentals of both Bag of Features and Machine Learning Algorithms in the background chapter. When implementing this model, local features are extracted from all training images and a dictionary referred to as a bag of features is created as explained in [section 2.3](#). Afterwards, local features of images in individual classes are matched against the dictionary and matches stored in individual class matrices. The resulting class matrices are then used as input into a machine learning algorithm that then creates a model that is used as a classifier.

There are many factors behind the popularity of this model. Chief amongst them which consequently makes it our preferred approach is that it is easier to understand and implement while still giving highly accurate results. Its use of machine learning algorithms for classification ensures that the results are more reliable and repeatable as machine learning algorithms take into consideration patterns and insights humans are likely to miss due to them either being too complex to explicitly code or not obvious to the human eye. Moreover, the model is highly scalable and adoptable. It has successfully been implemented to solve the time series classification problem [97]; to classify Alzheimer's disease in magnetic resonance images [98]; and to solve Histopathology image classification problem [99]; to mention but a few.

From our literature review, we identified that SIFT and SURF features are preferred algorithms for feature extraction while SVMs and variations of ANNs are preferred classifiers amongst studies applying the Bag of Features classification model. Both SIFT and SURF algorithms are ideal for such applications because they are both robust to image transformations such as scale, rotation, noise and affine transformations. In addition, all major open source image processing libraries ship with them, saving researchers time. Though the underlying theories of ANNs and SVMs vary significantly, they are well suited for deriving solutions of problems where relationships between variables are vaguely understood or too difficult to describe adequately with conventional algorithms hence their popularity in image classification problems.

For our solution, we decided that the system should allow training using either SIFT or SURF and SVMs. While there is no clear consensus on which is better between SVMs and ANNs from papers we surveyed [100] [101] [102] [103] [104], there seems to be a general agreement that SVMs are considerably more efficient on computational time, less sensitive to parameter settings and provide accurate results even with less training data. It is because of these reasons that we selected SVMs over ANNs.

4.2.2. Approach to accessibility

To ensure that our solution is accessible to many end users utilizing different devices with varying computational power, we decided to adopt client-server architecture [105].

Client server architecture refers to a distributed application structure that allows clients or service requesters which often have less computing power to access services provided by one or more powerful service providers referred to as servers. In our case, the system is designed

such that the resource intensive diagnosis functionality is hosted on one server and end users can access the functionality using any internet capable device.

In addition to enabling end users access the solution regardless of computational power of their devices, this model also has the advantage that it ensures all users get consistent results from the latest deployed server code. i.e. For the same image, a user accessing the system through a high-end smartphone or laptop and a user accessing it using a low-end feature phone all get the same results.

4.2.3. Approach to ensuring usability

In Software Engineering, usability refers to a degree to which an application can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use. It is a large part of an active research field referred to as Human Computer Interaction (HCI) [106]. Usability is not our primary focus in this study hence, we do not evaluate it. However, given our application's target demographic and the fact that the system is designed to be accessed using different devices with varying screen sizes whose users have different interaction habits, we had to take into consideration some usability guidelines.

To ensure a seamless, high quality user experience for all our end users regardless of devices they use to interact with the system, we designed the system's web client following Responsive Web Design(RWD) guidelines. Believed to have been coined by Ethan Marcotte [107], RWD is an approach to user interface design that allows web pages to be viewed in response to the web browser screen one is viewing with. A site designed following RWD guidelines adapts its layout to the viewing environment by using fluid, proportion-based grids, flexible images and Cascading Style Sheets 3 (CSS3) media queries.

To ensure minimal learning curve, when designing all our user interfaces, we adopted the popular, tried and tested user interface design guidelines referred to as flat design guidelines [108]. Flat design guidelines are recommended by most operating system vendors, including Microsoft [109], Google [110] and Apple [111]. Following these guidelines ensures familiarity which in turn promotes learnability.

4.3. Architectural overview

Figure 4-1 illustrates the core modules of the system and how they communicate with each other. From a very high level perspective, our system is made up of responsive web client, the application server and the system administrator portal. The responsive web client is the user facing end of the system through which users interact with the whole system. Through it, they upload images they want diagnosed and view the diagnosis results. The application server module is the core of the system where all major resource intensive functionality resides. Its responsible for all the image processing and classification processes. The system admin portal is mainly for configuring and testing the diagnosis engine. Through it, the system administrator can add and test new models for diagnosing more crops diseases. More details are on individual models are provided in the subsequent implementation chapter.

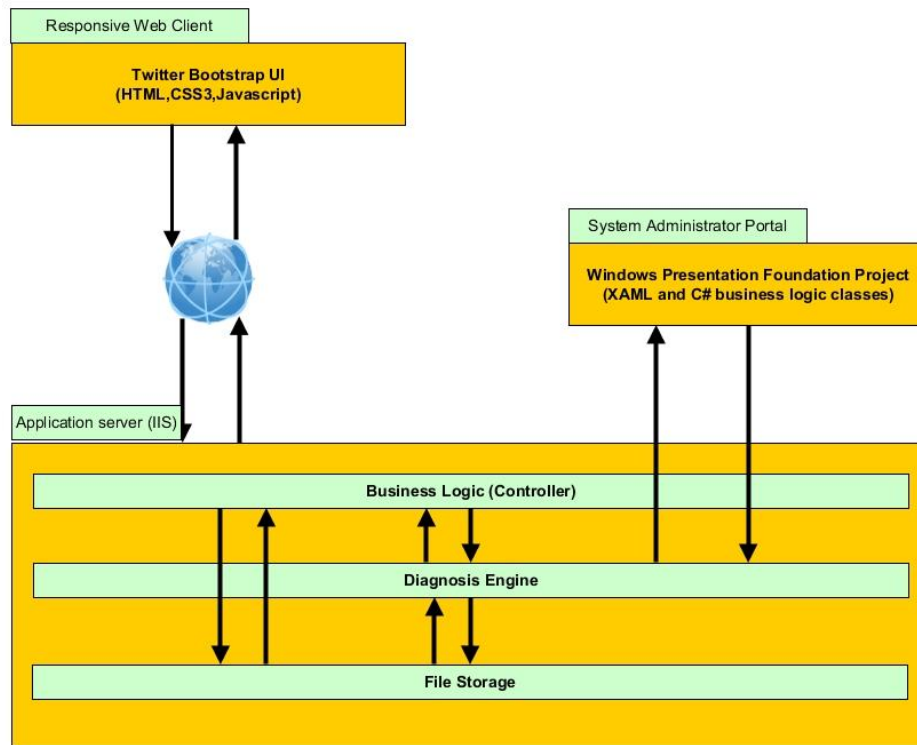


Figure 4-1 A high level architectural overview of the system showing main modules. Pointed arrows indicate communication between different modules

4.3.1. Use cases

Figure 4-2 illustrates the use cases the system caters for.

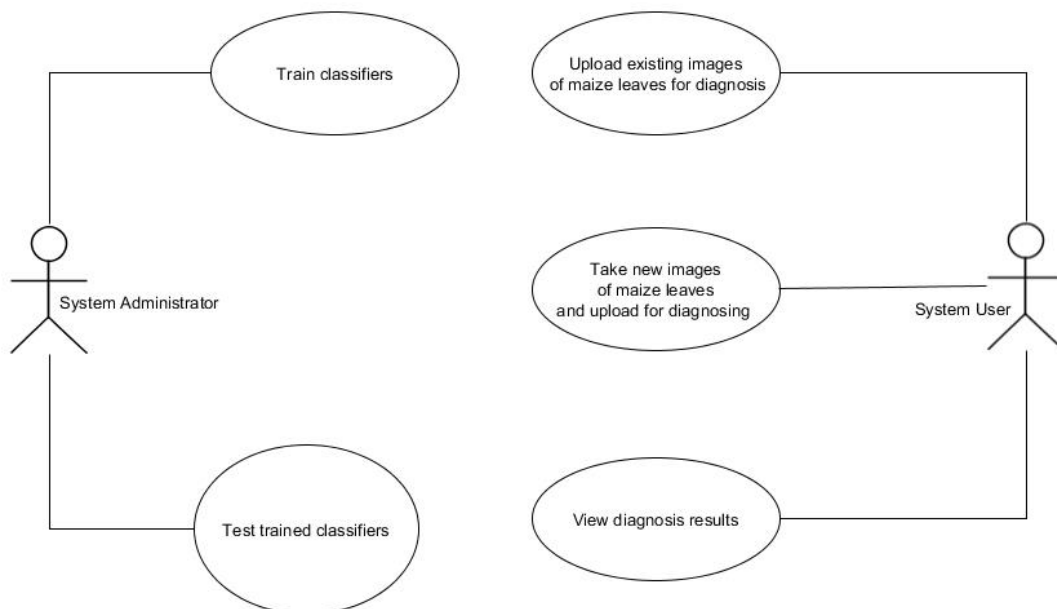


Figure 4-2 System use cases from both the system administrator and system user perspectives.

4.3.2. Activity flow charts

Figure 4-3 outlines activity flow we envisioned a system administrator would follow when setting up a new model for diagnosing crop diseases and figure 4-4 outlines an activity flow we envisioned an end user would follow when using our system.

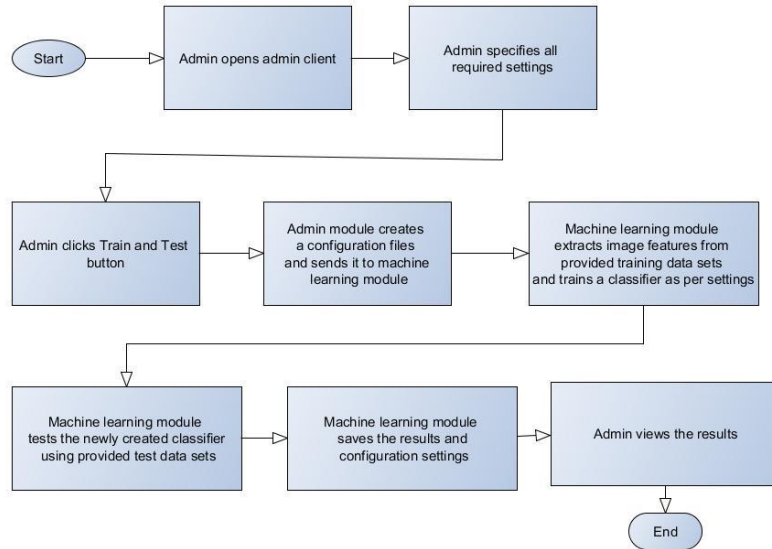


Figure 4-3 Flow diagram of system administrator using the administrator panel to set up a classifier

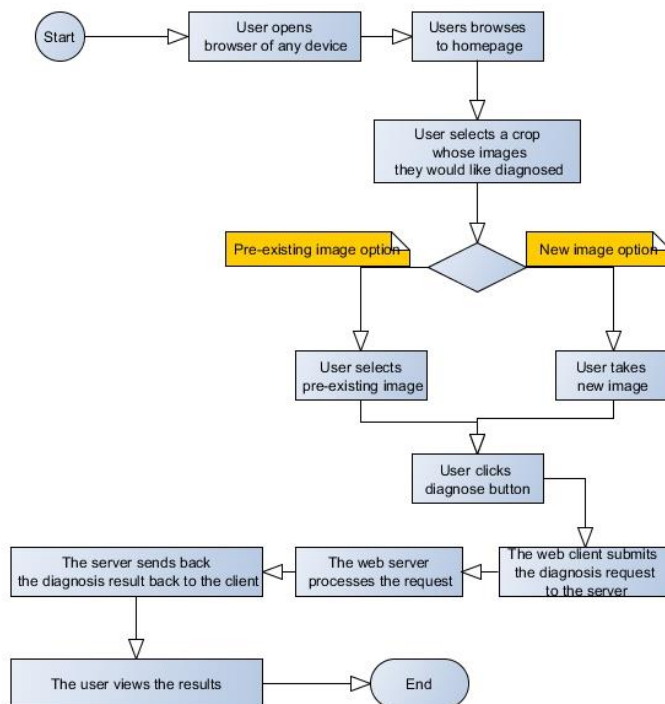


Figure 4-4 Flow diagram of a system using when diagnosing images using the responsive web client.

4.1 Conclusion

This chapter has outline the thought process that guided us in architecting our solution. We have discussed factors we took into consideration and our approach to solving them. In addition, we have provided a modular overview of the system, introduced use cases it caters for and activity flows we envisioned when designing the system.

In the next chapter, we continue our discussion of the system by providing more detail on how individual components were constructed.

5 Implementation

In this chapter, we provide more details on each individual component of the system. We discuss technologies utilized, how they were implemented and in addition, discuss the roles of each individual component.

It is worth mentioning that we do not go as far as providing class diagrams because all our components are constructed using open source third party libraries whose architectural compositions are readily available online.

5.1 Responsive web client

This is the user facing end of the system through which users interact with the whole system. It is constructed using Twitter Bootstrap [112], a popular free open-source front-end web framework that contains HTML, CSS and JavaScript based design templates for typography, forms, buttons, navigations and other interface components.

It has two main functions, allowing the users to submit diagnosis requests and displaying the diagnosis results. In its entirety, it consists of two web pages namely, the home page shown by figure 5-1 and the results page shown by figure 5-2.

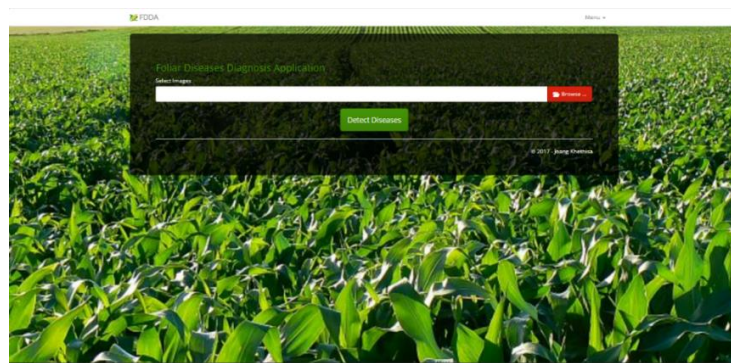


Figure 5-1 An image of the application's home is rendered by a windows 10 desktop chrome browser.

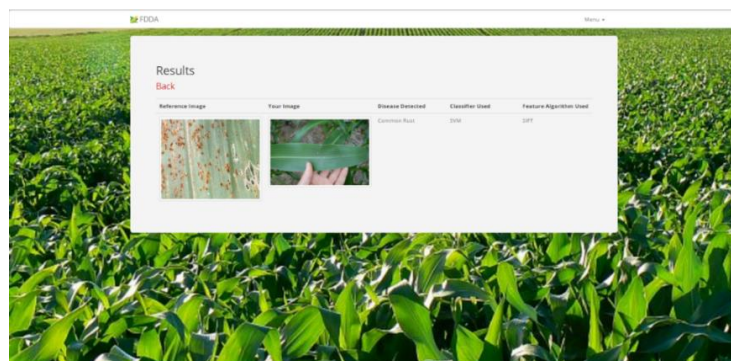


Figure 5-2 The application's results page as rendered by desktop version of Chrome browser. The image on the right is an example of a diagnosed disease and the image in the "Your Image Column" is the image the user uploaded. The other columns show the diagnosed di

The home page allows users to upload images pre-existing images or depending on a device the user is using, capture an image and upload it straight away. It is basically a html form consisting of a file input and a submit button. The file input is set to accept only image files and it has a multiple attribute set to true, allowing the user to upload multiple images for diagnosis in one go. It also has capture attribute set to true so that users accessing the site using browsers that support the html5 file API to be able to take images and upload them straight away. It also has a preview section so that the user is aware what images they are uploading. Figure 5-3 shows the home page with some images selected for upload. The submit button is for submitting the form. Upon submitting, the images are uploaded to the server where the diagnosis takes place and the user is redirected to the results page.

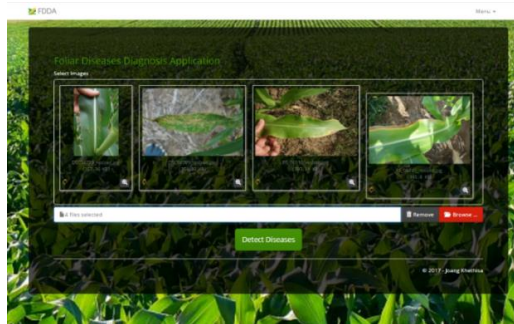


Figure 5-3 Application's home page showing preview of images the user is about to submit for diagnosing.

The results page is for displaying the diagnosis results. It consists of a html table for displaying the results. The table has five columns, Submitted Image, Sample Image, Disease Detected, Classifier Used, and Features Used columns. Submitted image column is for displaying the image the user uploaded for diagnosis. The Sample Image column is for displaying an image of the diagnosed diseases for user's reference. The Submitted image is for displaying a thumbnail of the image the user submitted for diagnosis. The Disease Detected column is for displaying the diagnosed disease. Finally, the classifier used and features used columns display the used machine learning algorithm and the used features extraction algorithm respectively. They are for the more technical users. Figure 5-4 shows the results page with diagnosis results of images uploaded as per figure 5-3.

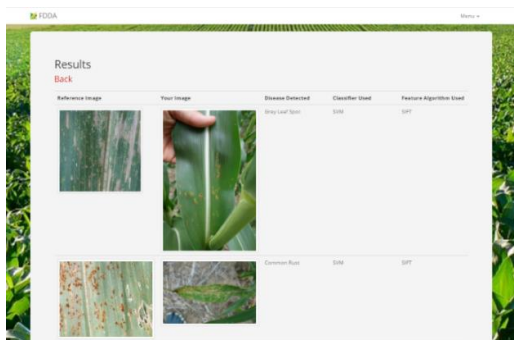


Figure 5-4 Results page showing diagnosis results of images shown in figure 5-3. Only the top two results are shown as other results were beyond screen print's reach.

Both pages are designed following Responsive Web Development guidelines. They scale based on the browser size the user is using to interact with the system. Figure 5-1 showed the home

page as rendered by the desktop chrome browser, figure 5-5 shows the same page on opera mobile browser on a Nokia C200 feature phone emulator and figure 5-6 shows the same page on a Google Nexus S5 smartphone emulator browser



Figure 5-5 Home page rendered on a Nokia C200 feature phone emulator's opera mobile browser

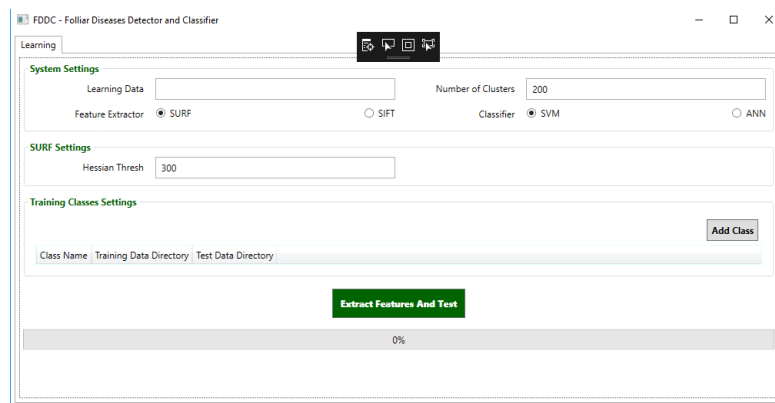


Figure 5-6 Home page rendered on a Google Nexus 5 smartphone emulator's mobile chrome browser.

5.2 System Administrator Portal

This part of the system is used by the administrator to train new classification models. It is constructed using Windows Presentation Foundation (WPF), a graphical subset of the .Net framework used for rendering user interfaces in windows based applications. Its presentation layer is constructed using XAML while its business logic layer is constructed using C# classes.

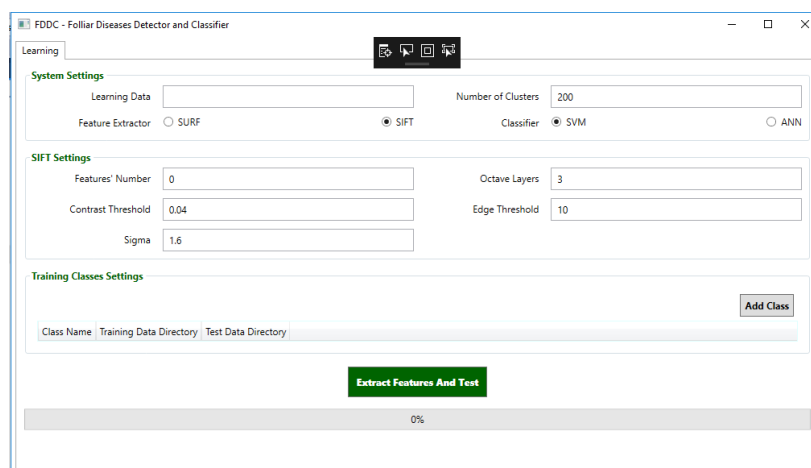
It consists of one main window and one dialog window. The main window is shown in figure 5-7. It consists of three sections. The first section is labelled System Settings and within it the admin can specify the path of the folder containing all training data, number of clusters to be used when creating the bag of features, the feature extraction algorithm to use and the classifier to use. The user has an option to select either SIFT or SURF for feature extraction.



The screenshot shows the 'FDDC - Foliar Diseases Detector and Classifier' application window. The 'Learning' tab is active. The 'System Settings' section includes a 'Learning Data' text box, a 'Number of Clusters' text box with the value '200', a 'Feature Extractor' section with radio buttons for 'SURF' (selected) and 'SIFT', and a 'Classifier' section with radio buttons for 'SVM' (selected) and 'ANN'. The 'SURF Settings' section has a 'Hessian Thresh' text box with the value '300'. The 'Training Classes Settings' section has a text box for 'Class Name' containing 'Training Data Directory | Test Data Directory' and an 'Add Class' button. At the bottom, there is a green 'Extract Features And Test' button and a progress bar showing '0%'.

Figure 5-7 Admin Panel Interface showing required fields when SURF is selected as a feature algorithm.

Depending on the feature algorithm selected, the second section contains either the SURF configuration form or SIFT configuration form. Both sections are pre-populated with recommended defaults. Figure 5-7 shows SURF parameters while figure 5-8 shows SIFT parameters. The third section is for configuring classes. Figure 5-9 shows the screen used for configuring data classes.



The screenshot shows the 'FDDC - Foliar Diseases Detector and Classifier' application window. The 'Learning' tab is active. The 'System Settings' section is the same as in Figure 5-7, but the 'Feature Extractor' radio button for 'SIFT' is selected. The 'SIFT Settings' section includes four text boxes: 'Features' Number' (0), 'Contrast Threshold' (0.04), 'Sigma' (1.6), 'Octave Layers' (3), and 'Edge Threshold' (10). The 'Training Classes Settings' section is the same as in Figure 5-7. At the bottom, there is a green 'Extract Features And Test' button and a progress bar showing '0%'.

Figure 5-8 Admin Panel Interface showing required fields when SURF is selected as a feature algorithm.

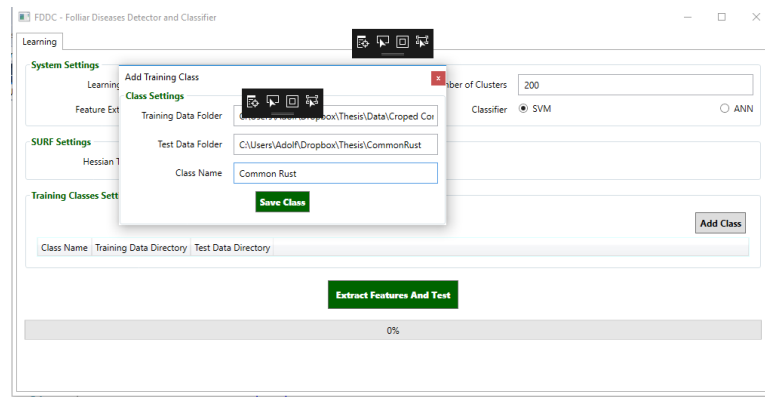


Figure 5-9 A pop up used for configuring data classes for the model. The system admin has to specify the full path to training data folder, full path to cross validation data and the class name.

When configuring a class, the system administrator must specify the full path to the folder containing images to use for training, a full path to the folder containing validation data for that class and the name of the class. The user can set up unlimited classes. However, in all our experiments, we only set up three classes, Common Rust, Grey Leaf Spot and Healthy.

After filling all the necessary configuration settings, the system administrator clicks on “Extract Features and Test” button. Upon clicking the button, the administrator portal prepares the training requests according to the specified settings and submits the request to the application server where all training and validation occurs. The administrator is kept updated on the progress by the progress bar which is asynchronously updated by the server side code.

5.3 Application server

The application server is the core of the system where all major resource intensive functionality resides. It is responsible for handling all the diagnosis requests from the web client and creating the diagnosis models as per administrator portal’s instructions.

It consists of three layers, namely, the business logic layer, diagnosis engine and the file storage layer. The business logic is composed mainly of the web controller and associated data objects. It accepts requests from the web client, formats them, forwards them to the diagnosis engine for further processing and diagnosing, and returns the results back to the client. The diagnosis engine is responsible for diagnosing images using existing models and for creating new models. The file storage layer is for saving classifiers, bag of features dictionary, uploaded images and diagnosis results.

The business logic layer is a C# asp.net MVC controller while the file storage layer consists of XML files which store different system configurations and diagnosis results, YML files that store bag of features dictionaries and trained SVM models, and images files uploaded by users. At the core of the diagnosis engine is OpenCV, a C++ open source computer vision and machine learning software library built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products [113]. To enable simple integration of OpenCV with the rest of our system which is constructed mainly

using C#.net, we utilized EmguCV, a cross platform .Net wrapper to OpenCV library that allows OpenCV functions to be called from .NET compatible languages such as C# [114].

The application server is configured to process three kinds of requests, the diagnosis request and results request from the web client, and the training request from the system administrator portal. A training request is basically a data transfer object(DTO) whose properties are the configurations set in the administrator portal. These properties include: number of clusters to use, path of parent folder containing subfolders of different classes' data, feature extraction algorithm, feature algorithm settings, classifier name and individual class settings. This request is processed directly by the diagnosis engine. Upon receiving the request, the diagnosis engine extracts image features from all images in the parent folder and subfolders, processes them to create a bag of features as per specified settings and serializes the resulting dictionary into a YML file as specified in Appendix A. Afterwards, it loops through individual classes, extracting image features from all images in the training folder, matching them against the stored bag of features and storing them in individual class matrices. Afterwards, the individual class matrices are fed to an SVM for training. When training is complete, the resulting model is serialized to a XML file like Appendix B. The next step is then to validate the resulting model. The diagnosis engine achieves this by looping through images contained in individual classes' test data folders and using the trained model to predict individual images' classes. The overall results are serialized to a XML file with the schema specified in Appendix C.

The diagnosis request from the web client is simply a form consisting of a jagged array of images. This request is received by the MVC controller. Upon receiving the request, the controller loops through previously saved training results and selects one with the highest average classification score. Using the settings from the found file, it modifies the diagnosis request to include the settings from the file and forwards the request to the diagnosis engine. The diagnosis engine loops through the images, extracting individual image features as per request settings and predicting their classes using the model specified in the request. The prediction results are serialized to an XML file with the schema specified in Appendix D and returned to the controller which returns a redirect header to the web client.

The results request is a simple get request specifying the name of the results xml. It is also processed by the MVC controller. Upon receiving the results request, the controller locates the results XML file, translates it to a HTML file and returns the results to the web client where a user can view their diagnosis results.

5.4 Conclusion

In this chapter, we have discussed in more detail individual component of the system. We have discussed the technologies used to build the individual components, responsibilities of the component and how they all tie together.

In the next chapter, we outline steps we followed to evaluate how well our system meets our study's primary objectives and tests we carried out to answer research questions we posed at the beginning of our research.

6 Evaluation and results

The objective of this research was to determine the feasibility of developing a crop disease diagnosis application utilizing computer vision and machine learning techniques, that in addition to being accessible through personal computers and smartphones, can also be accessed through feature phones.

This chapter outlines the steps we undertook to evaluate this primary objective and experiments we carried out in attempt to answer the research questions we posed.

6.1 Data collection

The bulk of the data we used throughout our experiments was acquired from www.plantvillage.org [115], an open access database of over 50,000 images of healthy and diseased crops. All in all, we collected 160 images of maize leaves infected by Common Rust, 160 images of maize leaves infected by Grey Leaf Spot and 160 images of healthy maize leaves. We ensured that images in each group are representative of different stages of the respective infection and that each group's images have varying orientations. In addition, we ensured that images in each set have varying levels of background noise.

6.1.1 Data preparation

The next step before any features extraction or model training, was to divide our data into sets. We split each collection into three subsets, the training subset used for training the models, validation subset used for estimating the prediction error of trained models during model selection and the testing subset used for assessment of the final models. When dividing the data into the respective sets, we ensured that the images in the training and validation sets of Common Rust and Grey Leaf Spot were representative of different stages of the infections. i.e. There were enough early stage infection images, mid stage infection images and advanced stage infection images in each set. At the end of this exercise our data was as presented in table 6-1 below.

Table 6-1 Data collected from www.plantvillage.org before any pre-processing.

Collection	Training Set	Validation Set	Testing Set
Healthy	50	50	60
Common Rust	50	50	60
Grey leaf spot	50	50	60

The next step we took was to crop out backgrounds from the training sets' images to avoid the risk of a classifier considering the background features as part of the significant descriptors. The images were cropped manually using Windows Paint software. Afterwards, we used GIMP software to split individual images in the training sets into smaller images so that we could

have more data to test different scenarios with different sets of images. Eventually, our data was presented in table 6-2

Table 6-2 Data after cropping training set images to create more training images.

Collection	Training Set	Validation Set	Testing Set
Healthy	100	50	60
Common Rust	100	50	60
Grey leaf spot	100	50	60

It is worth mentioning that we ensured that the images in our testing sets resembled typical images that a regular user would upload to the system for diagnosing. We did not remove any background objects from them nor pre-process them in any way. Figure 6-1 presents examples of typical images in our testing data sets.



Figure 6-1 Typical images in our testing data sets. No background noise is removed so that images resemble images our system users are likely to upload for diagnosis.

6.2 Hardware and software used

All experiments were carried out on a HP laptop with Intel core i5-3230m, 2.60GHz CPU and 4GB RAM running a 64-bit Windows 10 Professional edition. All programming and benchmarking was done on Microsoft Visual Studio 2015 Professional Edition with Internet Information Services (IIS) version 10.0 Express edition installed. Google Chrome browser developer tools, Opera browser developer tools and visual studio emulators were used to emulate different mobile phone conditions. Opera mobile browser emulator, Chrome browser for Android, Internet Explorer 9, Edge browser and Safari browser were used for testing browser compatibility.

6.3 Evaluation of the crop disease diagnosis engine

The first component of the system we evaluated was the diagnosis engine as it is the core of the system. Our admin panel allows for the system administrator to configure the number of clusters to use when creating the bag of features and what feature extraction algorithm to use. In addition, the administrator is also responsible for deciding the amount of data to use for training the SVM model. Moreover, both SIFT and SURF have configurable parameters the administrator also must decide on. Our first task was then to establish which feature extraction algorithm and parameters would produce the best results, as well as the influence the amount of training data has on the overall accuracy of the SMV models.

For every feature extraction algorithm and SVM combination, we carried out three suites of experiments. The first was geared towards establishing the number of clusters that produces the best results. To establish this, we used the default feature extraction algorithm's parameters, kept the training data quantities constant and only varied the number of clusters. Having established the best number of clusters, our second suite of experiments were designed to find which parameters produced the best results. We kept the number of clusters and the training data constant and varied individual algorithm parameters over several iterations. The third suite of tests was then designed to establish the influence the quantity of training data had on the classifier. We used the best parameters established from the first two sets of tests and varied the amounts of training data.

Having established the ideal parameters and training data quantity to use for each combination of feature extraction algorithm and SVM, our final task was then to evaluate each ideal model using the testing data sets.

Section 6.3.1. presents various test results of varying different parameters when training an SVM classifier using SURF features while section 6.3.2. presents various results of varying different parameters when training an SVM classifier using SIFT features. Finally, section 6.3.3. presents evaluation of the ideal SURF-SVM classifier and SIFT-SVM classifier as established in sections 6.3.1. and 6.3.2. Each test was run four times and the results presented in this dissertation are averages.

6.3.1 SURF-SVM tests' results

The first test we performed for SURF-SVM model was to determine the impact varying the number of clusters used when creating the bag of features would have on the overall classifier. Roughly speaking, this variable controls the trade-off between being distinctive and robust. If too low, it can result in visual words not representative of all features and if too large it can result overfitting. The task of this test is then to determine what the best number to use given our classification problem and data. We varied the number of clusters starting at 50 clusters and ending at 400 clusters, while maintaining the training data at 100 images per class and the Hessian Threshold to OpenCV's default value of 200. The best value to use is determined by highest average accuracy. The results are presented in figure 6-2.

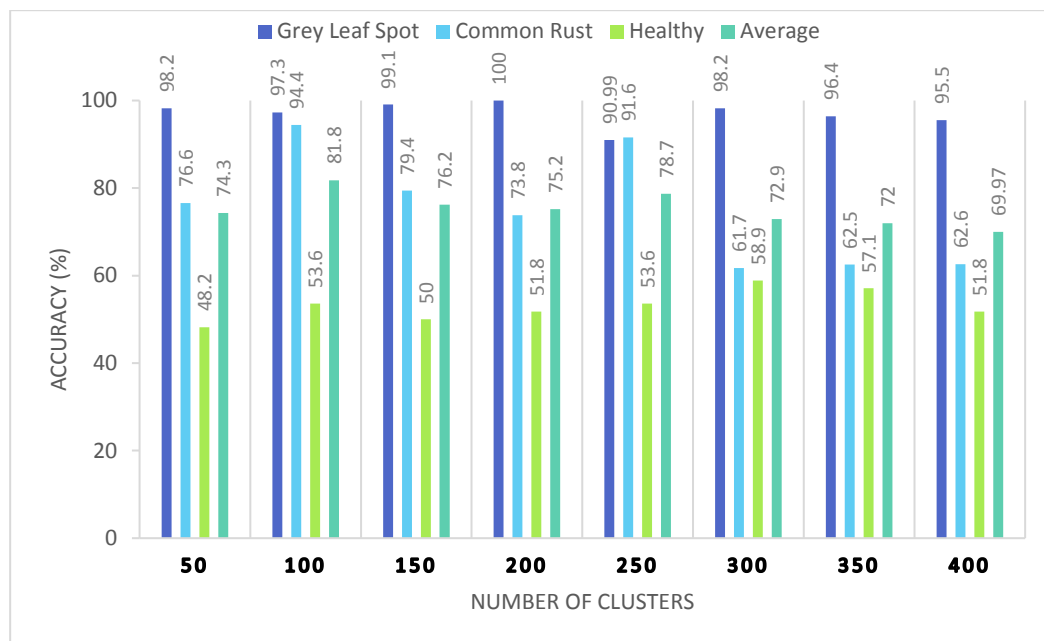


Figure 6-2 Effects of varying number of clusters while keeping Hessian Threshold fixed at 200 and training data at 100 images per class when training SURF-SVM classifier.

From the results presented in figure 6-2, we can see that the average accuracy of the classifier is low at 50 clusters, peaks at 100 clusters and afterwards gradually decreases. This pattern is in line with patterns Yang et established in [116].

The next tests we performed were aimed at evaluating the impact of varying the Hessian Threshold on the classifier's prediction accuracy. The Hessian Threshold affects the number of features extracted from images and their repeatability, the tendency of re-detection of the same feature in another image of the same scene but different camera angle. Setting a low Hessian Threshold results in detection of a lot of feature points which however have less repeatability and setting it high results in fewer features with high repeatability. An ideal Hessian Threshold is one that produces enough repeatable features for a classifier to establish distinctive features of individual classes. To establish an ideal Hessian Threshold for our training data, we used the ideal number of clusters, 100, established in the last suite of tests, 100 training images per class



and varied the Hessian Threshold from 0 to 350. The results of the tests are presented in figure 6-3 on the next page.

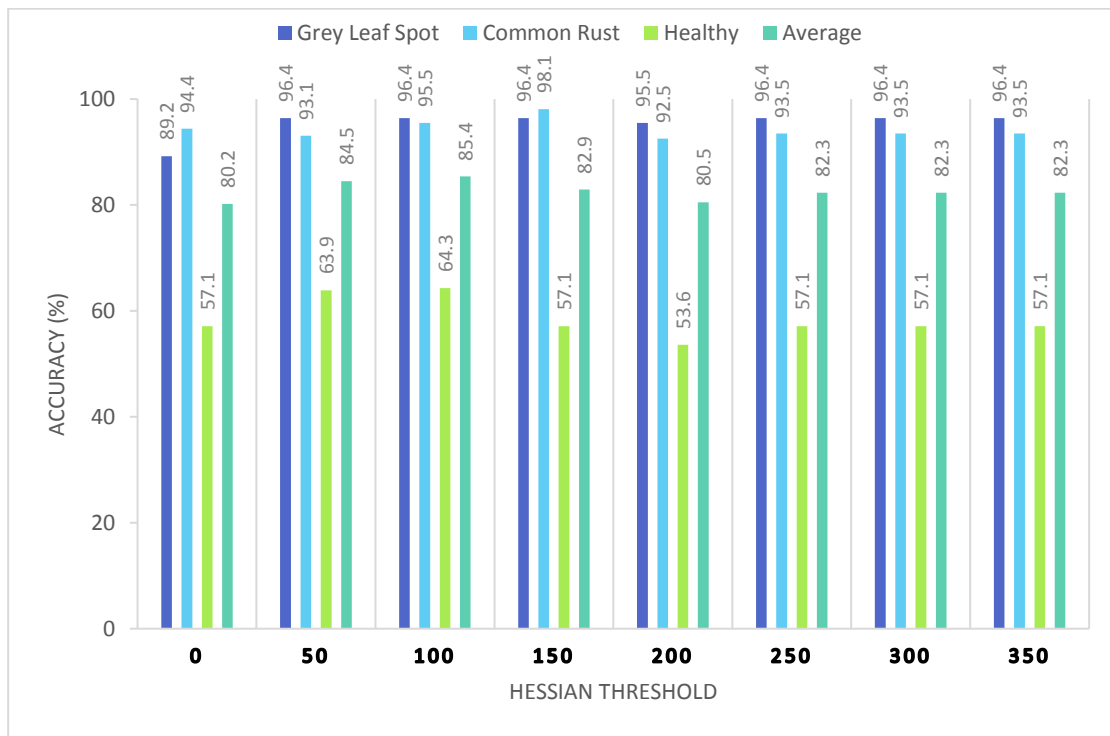


Figure 6-3 Effects of varying the hessian threshold clusters while keeping number of clusters fixed at 100 and training data at 100 images per class when training a SURF-SVM classifier.

From the results presented in figure 6-3, it is evident that the best Hessian Threshold to use for our training data is 100.

The last tests we performed on SURF-SVM model were aimed at establishing how the quantity of training images affect the accuracy of the classifier. To establish this, we used the ideal parameters as established in the last two suites of tests and varied the number of images per training set starting at 20 images per class to 100 images per class. The results of the tests are presented in figure 6-3.

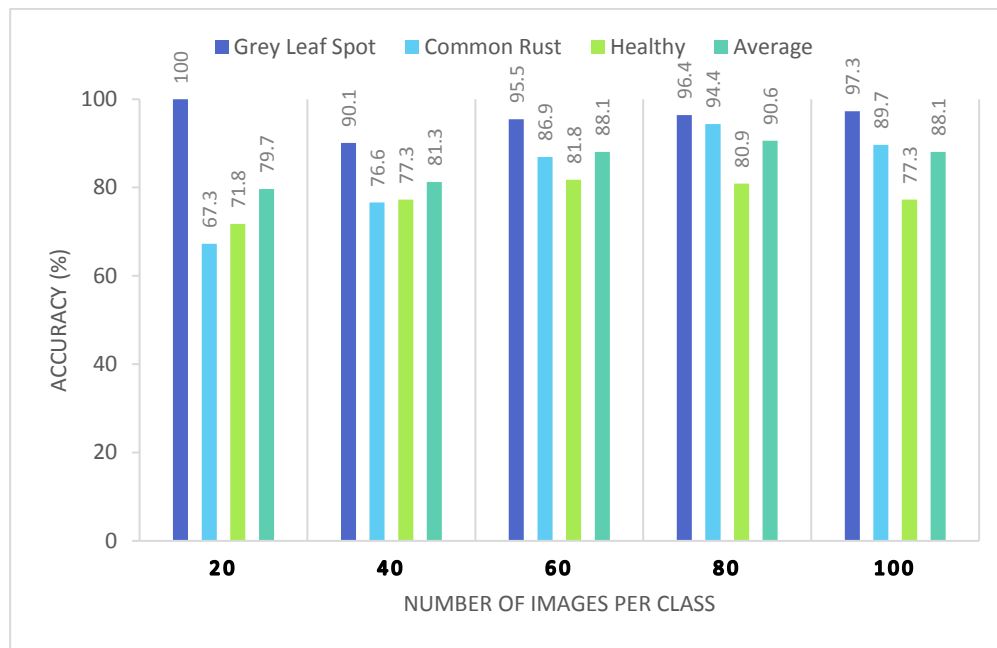


Figure 6-4 Effects of varying training data quantity while keeping number of clusters fixed at 100 and Hessian Threshold at 100 when training a SURF-SVM classifier.

A lot of studies have previously investigated the impact of data on machine learning algorithms' accuracy. Though there are varying views on exactly what constitutes good learning data and how much of it is enough to effectively train a classifier, there is a consensus that the more data one trains with, the more accurate the resulting model becomes [117] [118] [119]. As indicated in figure 6-4, our model produced a different pattern, using 80 images per class produced a more accurate model than the one produced using 100 images per class. This could be because our training data sets sizes are too small and the pattern probably would have changed to align with the expectations had we kept increasing the training data quantity. However, this is a phenomenon worth investigating in more detail on its own dedicated thesis. We therefore decided not to investigate it any further in this study, but rather accept the results as they are.

From the above experiments' results, we inferred that given our data, to produce best candidate classifier using SURF and SVM we must set a number of clusters to 100, Hessian Threshold to 100 and only use 40 images per class. As highlighted in our discussion of the results presented in figure 6-4, it is worth doing a different study to validate these results, especially those produced by varying quantity of training data

6.3.2 SIFT-SVM tests' results

As in the case of SURF-SVM model, the first test we performed was to determine the impact of varying the number of clusters. In this test, we also varied the number of clusters starting at 50 and ending at 400 while keeping the training data constant at 100 images per class and using the default OpenCV's values for octave layers, contrast threshold, edge threshold and sigma. The results are present in figure 6-5.

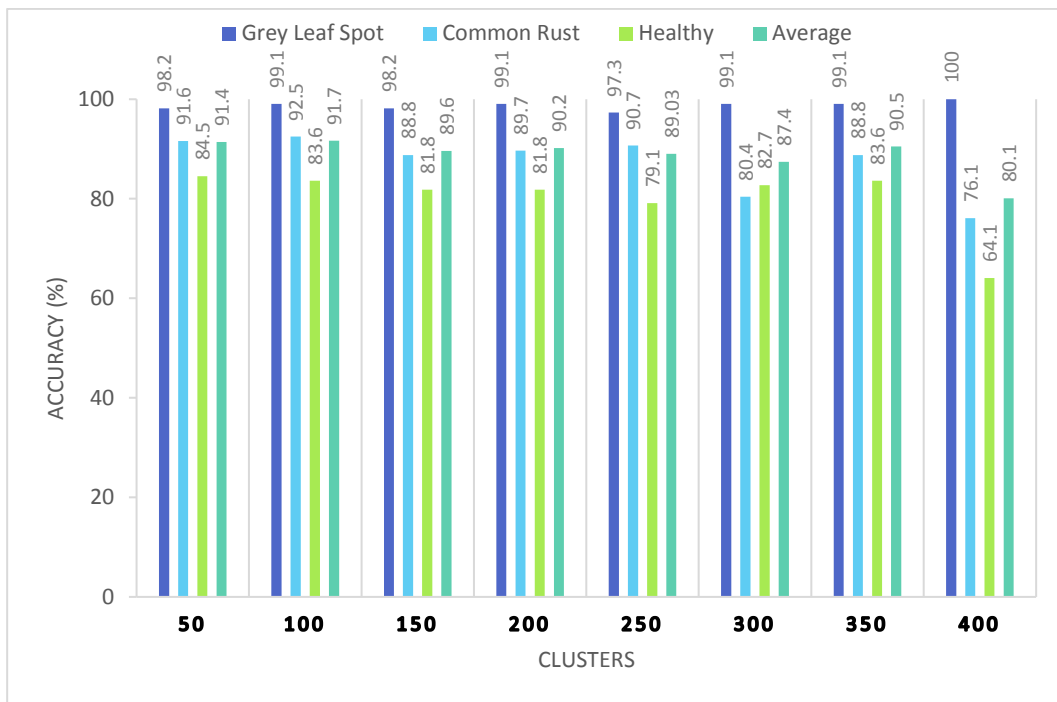


Figure 6-5 Effects of varying number of clusters when training a SIFT-SVM classifier

Like SURF-SVM model’s results, 100 clusters produced the highest average accuracy. However, unlike in the previous SURF-SVM test, the average accuracy did not continuously decrease after the peak. As indicated in figure 6-5, 200 clusters produced a higher average accuracy than 150 clusters. Similarly, 350 clusters produced higher average accuracy than 300 clusters. Although the results’ pattern varies from the SURF-SVM results’ pattern, we are not the first to observe this pattern. Liu et al also observed a similar pattern in [120]. Establishing the underlying reasons behind the differences is beyond the scope of this study, we therefore accepted the results as they are.

The next test we performed on the SIFT-SVM model was to establish the impact of varying the number of octave layers on the accuracy of the produced model. Octaves in SIFT emulate various representations of a given image from camera distances different from the original. In principle, increasing the number of octave layers gives the ability to detect features in more different sizes ranging from the smallest to the largest. However, it also increased computation time and makes the algorithm susceptible to noise. More details on Octave layers can be found in the original paper by Lowe [121]. In our tests, we varied the octave layers from the 3 which is the recommended value in Lowe’s paper [121] to 11. Our results are represented in figure 6-6.

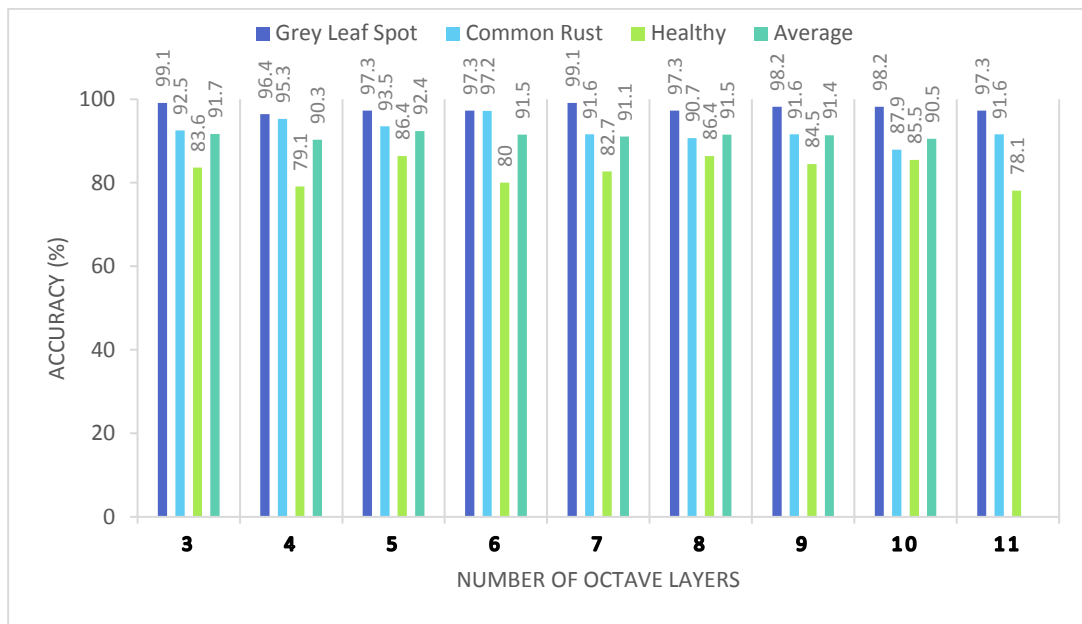


Figure 6-6 Effect of varying number of octave layer when training a SIFT-SVM classifier

From figure 6-6, it can be established that the best number of octave layers to use for our training data is 5.

The next tests we performed were to establish the appropriate contrast threshold value to use for our training data. In SIFT, contrast threshold is used to filter out weak features in semi-uniform (low-contrast) regions. The larger the threshold, the more refined but fewer features are produced by the algorithm. In our tests, we varied the value from 0 to 0.04 and our results are presented in figure 6-7.

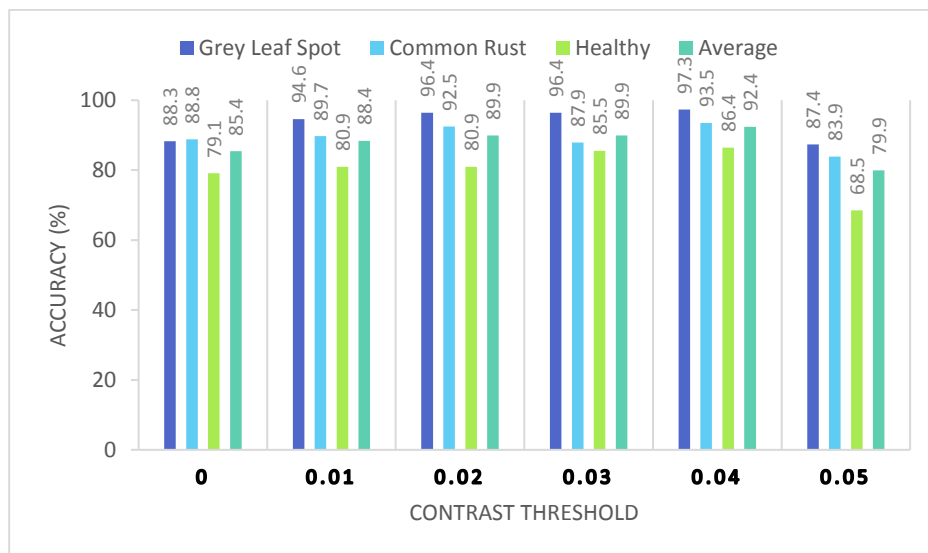


Figure 6-7 Effects of varying contrast threshold when training a SIFT-SVM classifier



As visible in figure 6-7, the accuracy of the model increases from 85.4 at 0 contrast threshold to the maximum of 92.4 at the threshold of 0.04. It therefore concludes that for our training data, the best value to set the threshold at is 0.04.

The next tests we performed were aimed at establishing the appropriate edge threshold to use. In SIFT, the edge threshold is used to filter out edge-like features. However, unlike in the case of contract threshold, the larger the threshold the less features are filtered out resulting in the algorithm producing more features. In our tests, we varied the edge threshold values from 0 to 14 and our results are present in figure 6-8.

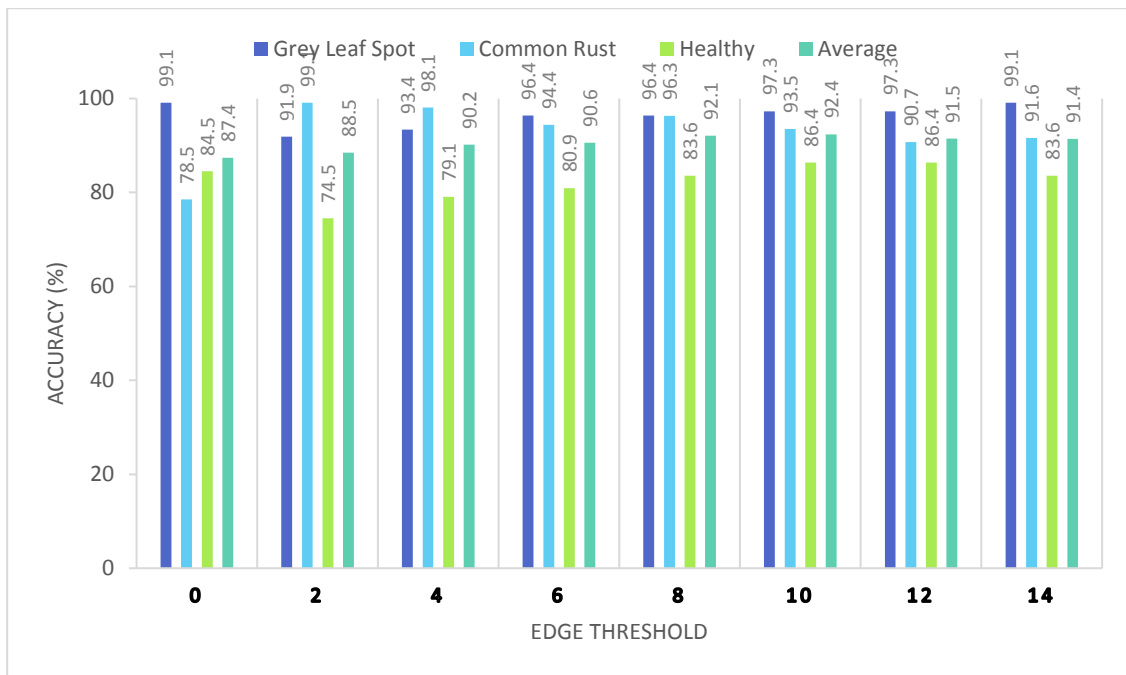


Figure 6-8 Effects of varying edge threshold when training a SIFT-SVM classifier.

From figure 6-8, it is evident that the most appropriate value to set the edge threshold when training a model with our data is 12.

Our next tests were aimed at finding the appropriate Sigma value to use when extracting features to train our classifier with. The detailed explanation of sigma is better left to this paper by Rey-Otero [122]. In simplified terms, low sigma value increases the visibility of edges and other detail present in a digital image which is beneficial if training images were captured using a weak camera with soft lenses. To find the appropriate value to use for our training data, we varied the value between 1.2 and 1.8 and our results are presented in figure 6-9.

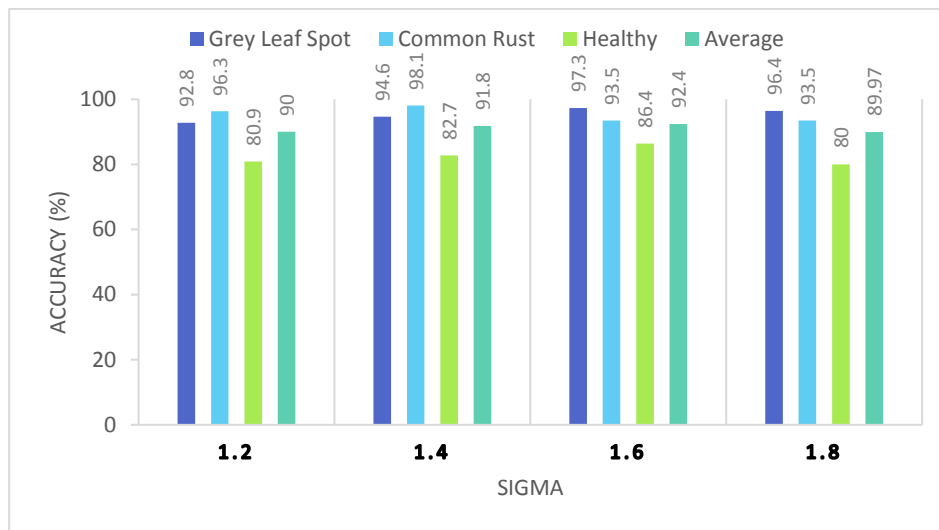


Figure 6-9 Effects of varying Sigma when training a SIFT-SVM classifier

From figure 6-9, it is evident that the most appropriate Sigma value for training a model with our data is 1.6.

As we did when investigating the best parameters for a SURF-SVM model, the last suite of tests we performed were aimed at establishing how the quantity of training images affects the accuracy of the classifier. To establish this, we used the ideal parameters as established in the previous SIFT-SVM tests and varied the number of images per training set starting at 20 images per class to 100 images per class. Our results of the tests are presented in figure 6-10.



Figure 6-10 Effects of varying quantity of training data when training a SIFT-SVM classifier



From figure 6-10, it is visible that varying the quantity of training when training a SIFT-SVM model results in a different pattern from that formed by the same test on a SURF-SVM model. In this case, the average accuracy increases as the training data is increased. This is more in line with surveyed past papers [117] [118] [119]. It is however worth investigating how the same data produced such different patterns. Unfortunately, this is beyond the scope of this dissertation.

6.3.3 Evaluation of established ideal classifiers

Having established the ideal parameters and training data quantity to use for each combination of a feature extraction algorithm and SVM in sections 6.3.1 and 6.3.2, our final task was then to train models using the established ideal values and evaluate the resulting models using our test data.

Our evaluation results are presented using confusion matrices. A detailed explanation of an identity matrix is provided in section 1.3, of this thesis. The confusion matrix of SURF-SVM model is presented in tables 6-3 and one for SIFT-SVM model is presented in table 6-4

Table 6-3 Confusion Matrix of the best SURF-SVM classifier

	Common Rust	Grey Leaf Spot	Healthy	Total Classification	Precision
Common Rust	37	23	0	60	61.667%
Grey Leaf Spot	14	45	1	60	75%
Healthy	3	21	36	60	60%
Overall Truth	54	89	37	180	
Recall	68.519%	50.562%	97.297%		
<i>Overall Accuracy</i>	$\frac{61.667 + 75 + 60}{3} = 65.556\%$				

NOTE – Number of clusters =100, Hessian Threshold =100, training data per class = 80



Table 6-4 Confusion matrix of the SIFT-SVM classifier

	Common Rust	Grey Leaf Spot	Healthy	Total Classification	Precision
Common Rust	41	19	0	60	68.333%
Grey Leaf Spot	13	47	0	60	78.333%
Healthy	0	16	44	60	73.333%
Overall Truth	54	82	44	180	
Recall	75.926%	57.317%	100%		
Overall Accuracy	$\frac{68.333 + 78.333 + 73.333}{3} = 73.333\%$				

Note – Number of clusters =100, octave layers = 5, contrast threshold = 0.04, edge threshold = 10, sigma = 1.6, training data per class =100.

From tables 6-3 and 6-4 it is evident that SIFT-SVM model is significantly more accurate than SURF-SVM model. This is in line with all tests carried in sections 6.3.1. and 6.3.2 where SIFT-SVM always had higher accuracy on all similar tests. It is also worth noting that in both cases, the accuracy is significantly lower than the accuracies established in 6.3.1 and 6.3.2 when using cross validation data. This was expected. As discussed in section 6.1.1, the images in our testing data sets resemble typical images that a regular system user would upload for diagnosing. Thus, their backgrounds have not been cropped out.

Comparing Figure 6-11 and Figure 6-13 which are typical of test images the system misclassified with images from corresponding validation sets, Figure 6-12 and Figure 6-14 respectfully, two characteristics we believe influenced their misclassification stand-out. Firstly, unlike Figure 6-12 and Figure 6-14 which have homogenous backgrounds, both Figure 6-11 and Figure 6-13 have complex backgrounds. We believe local features extracted from the background objects made it harder for the classifiers to correctly classify the images. In future works, this impact can probably be minimized by adding a segmentation step to remove irrelevant backgrounds. Secondly, both Figure 6-11 and Figure 6-13 show early stages of Grey Leaf Spot and Common Rust respectively. As mentioned in section 1.4, the two diseases have identical symptoms in early stages of infection. This impact can probably be minimized by adding more images of early stage infections in the training data. However, this is worth investigating in more detail in a dedicated thesis.



Figure 6-11. Example of misclassified Grey Leaf Spot image.



Figure 6-12. Example of Common Rust validation set images.



Figure 6-13. Example of misclassified Common Rust image.

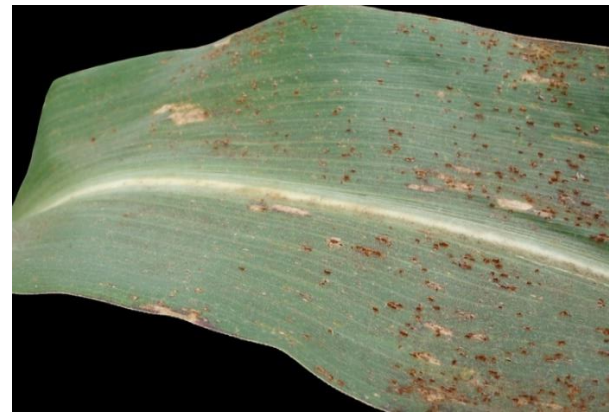


Figure 6-14. Example of Grey Leaf Spot validation set images.

What was surprising is the low recall rate of Grey Leaf Spot evident in both table 6-3 and table 6-4, given that in our validation tests, Grey Leaf Spot was predicted with the highest accuracy. This pattern is also worth investigating in more detail in its own research to establish what causes the trade-off.

6.4 Overall system evaluation

From the evaluation done in [section 6.3](#), we established that the best diagnosis engine to use was one that used SIFT feature extraction algorithm with number of clusters set to 100, octave layers to 5, contrast threshold to 0.04, edge threshold to 10, sigma set to 1.6 and 100 training images per class. We trained and deployed an SVM using these parameters.

Our next task was then to evaluate the overall system as the end user would use it. Our first test was to establish our system's performance in terms of page loading times against google.com, the world's most visited site [123]. The test was set up to evaluate the average time it takes for our system's home page to load and the average time it takes to upload an image and get back diagnosis results on different network speeds and compared the results against the times it takes for Google's homepage to load and the time it takes to return image search results on similar



network speeds. We used a 466kb, 800px by 533px image as a testing image and used Google Chrome developer tools to throttle the network and achieve different network speeds. For each network speed, the test was carried out five time and the average test results are presented in table 6-5 below.

Table 6-5 Average load times of system pages compared to Google Search Engine load times

Network	Home Page Load Time	Diagnosis and Results Load Time	Google.com home page load time	Google.com results page load time
GPRS (50 kb/s, 20 kb/s)	11.78 seconds	2.1 minutes	23.03 seconds	2.7 minutes
Regular 2G (250 kb/s,50 kb/s)	3.02 seconds	18.22 seconds	2.32 seconds	12.71 seconds
Good 2G (750 kb/s, 250 kb/s)	1.63 seconds	9.93 seconds	1.44 seconds	8.38 seconds
Good 3G (1.5 Mb/s, 750 kb/s)	1.07 seconds	4.03 seconds	1.01 seconds	6.72 seconds
Regular 4G (4.0 Mb/s, 3.0 Mb/s)	543 milliseconds	2.95 seconds	1 second	6.4 seconds
DSL (2.0 Mb/s, 3.0 Mb/s)	398 milliseconds	1.15 seconds	1 seconds	5.6 seconds
Wi-Fi (30 Mb/s, 15 Mb/s)	366 milliseconds	689 milliseconds	853 milliseconds	5.1 seconds

From table 6-5, it is evident that a user accessing our system from different networks would experience similar waiting periods they are used to when accessing the world’s most popular site. It is worth mentioning that during this tests, our system was hosted on a computer within



our local network while Google pages were served from a remote server, so results are likely to vary in life situations. However, the purpose of this experiment was not to benchmark our system’s performance against Google load times, but rather to establish if our system’s load times were within acceptable ranges.

Our second batch of tests was aimed to establish how well our system renders and functions in different web browsers. To limit the number of tests to perform whilst being thorough, we resolved to only test the system on the most popular browsers, Opera, Chrome, Internet Explorer (IE 10), Firefox, Edge, Opera Mobile and Safari. To test compatibility with feature phones, we resolved to test the system on opera mini 4 [124], a free Java Me platform browser that is compatible with most feature phones. We tested that users can view the home page, upload images and view results on each browser. The results are presented in table 6-6 on the next page.

Table 6-6 Compatibility test of system's web pages on popular browsers

Browser	Home Page loads	User can upload an image	User can view diagnosis results
Opera Mini 4	Yes	Yes	Yes
Opera	Yes	Yes	Yes
Opera Mobile	Yes	Yes	Yes
Chrome	Yes	Yes	Yes
IE 10	Yes	Yes	Yes
Firefox	Yes	Yes	Yes
Edge	Yes	Yes	Yes
Safari	Yes	Yes	Yes

From the results presented in table 6-6, it is evident that our system is accessible and functions properly on all popular browsers.

Given that our system users will use different devices to capture the images resulting in images of different resolutions, we then set out to establish how well our system would function given low resolution images as input. To establish this, we took 20 images from our test data that were previously correctly diagnosed and resized them to an average of 128 × 160 pixels, an average size of an image taken using Nokia C200 feature phone we had access to, and diagnosed the resulting images using the system. In all cases, the diagnosis results remained the same.



6.5. Conclusion

In this chapter, we outlined the experiments we performed to evaluate the system. We have discussed different experiments we performed to determine the best classifiers to deploy and how we evaluated them to determine their accuracy when using data that resembles typical data end users would upload for diagnosis. In addition, we have discussed experiments we performed on the overall system to determine how it would behave in life situations. We tested its usability when accessed from different networks, and its browser compatibility.

In the next chapter, we relate the evaluation results to the primary objective of this study and answer the research questions we posed at the beginning of the study.



7 Conclusion

A considerable amount of research on utilizing image-processing and computer-vision algorithms to detect and classify crop diseases has been carried out over the past decades. Unfortunately, the bulk of these previous studies have mainly focused on either developing large scale remote sensing applications more suited for large scale farming or on developing traditional desktop/laptop applications and a few others on developing high end smartphone applications. Thus, making them inaccessible to many subsistence farmers, especially those in sub-Saharan Africa where ownership of personal computers and smartphones is minimal.

The primary objective of this study was to establish the feasibility of utilizing similar techniques to develop a crop disease diagnosis application that in addition to being accessible through personal computers and high end smartphones can also be accessed through any internet enabled feature phone.

During our study, we reviewed several past studies that have successfully developed crop diagnosis applications utilizing computer vision and machine learning techniques and observed both their successes and shortfalls. Leveraging on their successes, we successfully developed a prototype crop disease diagnosis application capable of diagnosing two of the most popular maize foliar diseases, Common Rust and Grey Leaf Spot, that is not only accessible through personal computers and high end smartphones but is also accessible through any internet enable feature phone.

The developed solution is a web based application constructed using open source libraries. Its web client is constructed using Twitter Bootstrap front-end web framework and was designed following responsive web guidelines. It adapts its layout depending on the size of the device being used to ensure smooth interaction. The diagnosis engine of the application was constructed using OpenCV and EmguCV libraries. It is based on the bag of features classification model and can be trained using either SIFT and SVM or SURF and SVM.

Several tests were conducted, mainly on the diagnosis engine and on the overall integrated solution to evaluate how well it can diagnose diseases, if is usable when accessed on different networks and its cross-browser support. The detailed experiments and their results are presented in the preceding evaluation chapter. Based on these results, we conclude that it is possible to utilize computer vision and machine learning algorithms to develop a highly scalable crop diagnosis application that is not only accessible through personal computers and high end smartphones but also accessible through feature phones. In this regards, our study was successful.

7.1 Answers to research questions

At the beginning of this dissertation we posed several questions we considered relevant in evaluating the success of our study. The answers to the posed research questions are examined as follows:



- Can all algorithms necessary for diagnosing images successfully execute and return the results before connection time-out for clients?

Our tests were carried on a typical laptop with specifications indicated in section 6.2 which are fairly low specifications compared to standard web servers. Throughout all our tests, we never got a server connection timeout error. It is therefore safe to conclude that yes, all algorithms can execute before connection times out.

- Can the system still be accessible and usable to users connecting from slow networks such as GPRS?

Admittedly, the page load times are significantly longer, up-to 2.1 minutes to load results, when connecting from GPRS and 2G networks. However, as indicated in table 6-5, these are in line with load times users would experience when using Google search engine from the same network. Given google search engine is the most visited site globally, it is safe to conclude that our system remains accessible and usable even at slow speed networks.

- What is the highest classification score we can get?

During training when using the validation data which had less background noise, the highest average accuracy was 92.4 as indicated in figure 6-10. However, when evaluating the model using test data which resembles typical images end users would upload for diagnosis, the overall accuracy of our model dropped to 73.3% as indicated in table 6-4. We therefore conclude that the highest accuracy the system can produce in practical situations is 73.3%.

7.2 Future work

The primary objective of this dissertation has been achieved and all research question satisfactory answered. However, there is always room for improvement.

In future works, we would like to work on ways to improve the accuracy of the diagnosis engine. One way to achieve this would be by adding a pre-processor module whose responsibility would be to enhance users' input files by removing background objects and any other noise that might be present in the files.

In addition, there are some observations we made during this study that are worthy of further investigation. In figure 6-4 we observed that increasing the quantity of training data when training a SURF-SVM model did not necessarily increase the produced model's accuracy yet the same test using the same data on SIFT-SVM model produced different but more expected results as indicated in figure 6-10. In addition, we observed that varying a number of clusters produced varying patterns. In future works, it is worth investigating the underlying reasons behind these differences.

8 References

- [1] United Nations, “World Population Prospects: The 2015 revision, key findings and advance table,” 2015.
- [2] Alliance for a Green Revolution in Africa, “Africa agriculture status report: Climate change and smallholder agriculture in sub-Saharan Africa,” Nairobi, 2014.
- [3] FAO, IFAD and WFP, “The State of Food Insecurity in the World 2015. Meeting the 2015 international hunger targets: taking stock of uneven progress,” Rome, 2015.
- [4] World Bank, 2014. [Online]. Available: <http://wdi.worldbank.org/table/5.12>. [Accessed 3 April 2016].
- [5] Ericsson, “Sub-Saharan Africa: Ericsson Mobility Report,” Stockholm, 2015.
- [6] Balancing Act, “Feature phone user survey,” Balancing Act, 2014.
- [7] International Data Corporation (IDC), *Smartphone shipments stall in Africa as demand for basic devices surges*, IDC, 2016.
- [8] The Angiosperm Phylogeny Group, “An update of the Angiosperm Phylogeny Group classification for the orders and families of flowering plants: APG III,” *Botanical Journal of the Linnean Society*, vol. 161, no. 2, p. 105–121, 2009.
- [9] Gann and Duignan, *Colonialism in Africa 1870-1960*, illustrated ed., P. D. L. H. Gann, Ed., CUP Archive, 1975.
- [10] M’mboyi et al., *Maize Production and Improvement in sub-Saharan Africa*, Nairobi: African Biotechnology Stakeholders Forum, 2010.
- [11] Gianessi, “INTERNATIONAL PESTICIDE BENEFITS CASE STUDY NO. 104, AUGUST 2014,” CropLife Foundation, August 2014. [Online]. Available: <https://croplife.org/case-study/importance-of-pesticides-for-growing-maize-in-sub-saharan-africa/>. [Accessed 10 April 2016].
- [12] Rossi et al., “Grey leaf spot of maize: A disease on the move,” *Plant disease*, vol. 67, no. 8, pp. 842-847, August 1983.
- [13] Ward et al., “Gray leaf spot, a disease of global importance in maize production,” *Plant disease*, vol. 83, no. 10, pp. 884-895, 1999.
- [14] Mpeketula et al., “An investigation on the biological variability of *Cercospora zeae maydis*, the incitant of gray leaf spot in maize in Malawi,” *African Crop Science Conference Proceedings*, vol. 6, pp. 286-289, 2003.
- [15] Lyimo, “Improving farmers’ access to and management of disease resistant cultivars in the Southern Highland of Tanzania – Phase 2,” 2006.
- [16] Ward, J.M.J., and Nowell D.C., “Integrated Management Practices for the Control of Maize Grey Leaf Spot,” *Integrated Pest Management Reviews*, vol. 3, no. 3, pp. 177-188, September 1998.

- [17] Tweer, Kloppers and Stephanie, "Maize diseases: common rust," 2009.
- [18] Gold Country Seed, "Common rust of corn," 2013.
- [19] Erik, Solem and Jan, Programming Computer Vision with Python, Draft ed., Jan Erik Solem, 2012.
- [20] Klette, Concise computer vision: An introduction into theory and algorithms, London: Springer, 2014.
- [21] Boden, Mind as Machine: A history of Cognitive Science, 1 ed., Oxford: CLARENDON PRESS , 2006.
- [22] Lew, Robust Computer Vision: Theory and applications, 26 ed., Springer Netherlands, 2003.
- [23] Gonzalez et al., Digital image processing, 3rd ed., Pearson Education, 2008.
- [24] Pratt, Digital image processing, Fourth Edition ed., Los Altos: Wiley & Sons, Inc, 2007.
- [25] Bernal J, Vilarino F., and Sanchez J., Feature Detectors and Feature Descriptors: Where are we now, Barcelona: Computer Vision Center and Computer Science Department UAB, 2010.
- [26] Lowe, "Object recognition from local scale-invariant features," in *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, Kerkyra, 1999.
- [27] Funayama et al., "SURF: Speeded Up Robust Features".
- [28] Bauer et al, "Comparing Several Implementations of Two Recently Published Feature Detectors," in *Proceedings of International Conference on Intelligent and Autonomous Systems*, 2007.
- [29] Panchal et al, "A Comparison of SIFT and SURF," *International Journal of Innovative Research in Computer and Communication Engineering* , vol. 1, no. 2, pp. 324-327, April 2013.
- [30] Otsu N., "A Threshold Selection Method from Gray-Level Histograms," *A Threshold Selection Method from Gray-Level Histograms*, vol. 9, no. 1, pp. 62-66, January 1979.
- [31] MacQueen J., "Some methods for classification and analysis of multivariate observations," Los Angeles, 1967.
- [32] Hugo, "Sur la division des corps matériels en parties," *Bull. Acad. Polon. Sci. Cl. III*, vol. 4, pp. 801-804, 1957.
- [33] Pitts, McCulloch and Walter, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115-133, 1943.
- [34] Boser et al., "A training algorithm for optimal margin classifiers," *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144-152 , 1992.
- [35] Corinna Cortes, and Vladimir Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, September 1995.

- [36] JLi, "An Empirical Comparison between SVMs and ANNs for Speech Recognition," Piscataway, New Jersey.
- [37] Lomakinab, Roman M. Balabina and Ekaterina I., "Support vector machine regression (SVR/LS-SVM)—an alternative to neural networks (ANN) for analytical chemistry? Comparison of nonlinear methods on near infrared (NIR) spectroscopy data," *Analyst*, vol. 136, no. 8, pp. 1703-1712, 2011.
- [38] Crookston, "A top 10 list of developments and issues impacting crop management and ecology during the past 50 years," *Crop Science*, vol. 46, pp. 2253-2262, 2006.
- [39] Larson W.E., and Robert P.C., "Soil management for sustainability, Soil and Water Conservation," *Farming by soil*, p. 103–112, 1991.
- [40] Zhang et al, "Precision agriculture: a worldwide overview," *Computers and Electronics in Agriculture*, vol. 36, p. 113–132, 2002.
- [41] Mulla, "Mapping and managing spatial patterns in soil fertility and crop yield," in *Soil specific crop management*, Madison, ASA Publishing Co, 1993, pp. 15-26.
- [42] Mulla et al., "Modeling the effect of precision agriculture: pesticide losses to surface waters," in *Terrestrial field dissipation studies*, Washington, ACS, 2002, p. 304–317.
- [43] Mulla et al., "A site-specific farm-scale GIS approach for reducing groundwater contamination by pesticides," *Journal of Environmental Quality*, vol. 25, p. 419–425, 1996.
- [44] Tian L, "Development of a sensor-based precision herbicide application system," *Computers and Electronics in Agriculture*, vol. 36, p. 133–149, 2002.
- [45] Mulla, "Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps," *Biosystems Engineering: Sensing Technologies for Sustainable Agriculture*, vol. 114, no. 4, p. 358–371, 2013.
- [46] Larson W.E., and Robert P.C., "Soil management for sustainability, Soil and Water Conservation," in *Farming by soil*, Ankeny, 1991, p. 103–112.
- [47] Mulla, "Using geostatistics and GIS to manage spatial patterns in soil fertility," in *Automated agriculture for the 21st century*, Michigan, ASAE, 1991, p. 336–345.
- [48] Mulla et al., "An evaluation of indicator properties affecting spatial patterns in N and P requirements for winter wheat yield," in *Precision agriculture '97: Spatial variability in soil and crop*, Oxford, BIOS Sci. Publ, 1997, p. 145–154.
- [49] Mulla et al., "A comparison of winter wheat yield and quality under uniform versus spatially variable fertilizer management," *Agriculture, Ecosystems & Environment*, vol. 38, p. 301–311, 1992.
- [50] Marchant et al., "Computer Vision for potato inspection without singulation," *Computers and electronics in agriculture*, vol. 4, pp. 235-244, 1990.

- [51] Sadegaonkar et al., “Automatic Sorting Using Computer Vision & Image Processing For Improving Apple Quality,” *International journal of innovative research and development*, vol. 4, no. 1, pp. 11-14, January 2015.
- [52] Yang et al., “Recognition of weeds with image processing and their use with fuzzy logic for precision agriculture,” *Canadian Agricultural Engineering*, vol. 42, no. 4, pp. 195-200, December 2000.
- [53] Tellaache et al., “A computer vision approach for weeds identification through Support Vector Machines,” *Applied Soft Computing*, vol. 11, no. 1, p. 908–915, January 2011.
- [54] Aitkenhead et al., “Weed and crop discrimination using image analysis and artificial intelligence methods,” *Computers and Electronics in Agriculture*, vol. 39, no. 3, pp. 157-171, August 2003.
- [55] Garcia, Barbedo and Jayme, “Digital image processing techniques for detecting, quantifying and classifying plant diseases,” *SpringerPlus*, vol. 2, no. 1, 2013.
- [56] Oppelt N., and Mauser W., “Hyperspectral monitoring of physiological parameters of wheat during a vegetation period using AVIS data,” *International Journal of Remote Sensing*, vol. 25, no. 1, pp. 145-159, 2004.
- [57] Bock et al., “Plant Disease Severity Estimated Visually, by Digital Photography and Image Analysis, and by Hyperspectral Imaging,” *Critical Reviews in Plant Sciences*, vol. 29, no. 2, pp. 59-107, 2010.
- [58] Barbedo et al., “Digital image processing techniques for detecting, quantifying and classifying plant diseases,” *SpringerPlus*, vol. 2, no. 1, pp. 1-12, 2013.
- [59] Mahlein, “Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping,” *Plant Science*, vol. 100, no. 2, pp. 241-251, February 2016.
- [60] Kim et al., “Machine vision technology for agricultural applications,” *Computers and Electronics in Agriculture*, vol. 36, no. 2, pp. 173 - 191, 2002.
- [61] De Jong, S. M. and Van de Meer F.D., *Remote Sensing Image Analysis: Including the Spatial Domain*. Bookseries on Remote Sensing Digital Image Processing, Dordrecht: Kluwer Academic Publishers, 2006.
- [62] Neblette, “Aerial photography for the study of plant diseases,” *Photo-Era Magazine*, vol. 58, p. 346, 1927.
- [63] Lindenthal M., Steiner U., Dehne H.W., and Oerke E.C., “Effect of downy mildew development on transpiration of cucumber leaves visualized by digital infrared thermography,” *Phytopathology*, vol. 95, no. 3, pp. 233-240, 2005.
- [64] Campbell J, *Introduction to Remote Sensing*, New York: Guilford Press, 2007.
- [65] McAllister E.D., and Myers J, “The time course of photosynthesis and fluorescence observed simultaneously,” *Smithson Misc Collect*, vol. 99, p. 1–37, 1940.

- [66] Kautsky H., Apel W., and Amann H., “Chlorophyllfluoreszenz und Kohlensäureassimilation,” *Biochem Z*, vol. 322, p. 277–292, 1960.
- [67] Baker, “Chlorophyll fluorescence: a probe of photosynthesis in vivo.,” *Annual Review: Plant Biology*, vol. 59, pp. 89-113, 2008.
- [68] Congming, Zhang and Jianhua, “Effects of water stress on photosystem II photochemistry and its thermostability in wheat plants,” *Journal of Experimental Botany*, vol. 50, no. 336, p. 1199–1206, 1999.
- [69] Stoll M., Schultz H. R. and Loehnertz B. B., “Exploring the sensitivity of thermal imaging for *Plasmopara viticola* pathogen detection in grapevines under different water status,” *Functional Plant Biology*, vol. 35, no. 4, p. 281–288, 2008.
- [70] Oerke, Steiner, Dehne and Lindenthal, “Thermal imaging of cucumber leaves affected by downy mildew and environmental conditions.,” *Journal of Experimental Botany*, vol. 57, no. 9, p. 2121–2132, 2006.
- [71] Marcassa et al., “Detection of citrus canker in citrus plants using laser induced fluorescence spectroscopy,” *Precision Agriculture*, vol. 10, p. 319–330, 2009.
- [72] Zhang et al., “Detection of stress in tomatoes induced by late blight disease in California, USA, using hyperspectral remote sensing,” *International Journal of Applied Earth Observation and Geoinformation*, vol. 4, no. 4, pp. 295-310, 2003.
- [73] Mahlein et al., “Early detection and classification of plant diseases with Support Vector Machines based on hyperspectral reflectance,” *Computers and Electronics in Agriculture*, vol. 74, no. 1, pp. 91-99, 2010.
- [74] Lindenthal, Steiner, Dehne, and Oerke, “Effect of downey mildew development on transpiration of cucumber leaves visualized by digital thermography,” *Phytopathology*, vol. 95, p. 233–240., 2005.
- [75] Oerke et al., “Thermal imaging of cucumber leaves affected by downy mildew and environmental conditions,” *Journal of Experimental Botany*, vol. 57, p. 2121–2132, 2006.
- [76] Stoll et al., “Early pathogen detection under different water status and the assessment of spray application in vineyards through the use of thermal imagery,” *Precision Agriculture*, vol. 9, pp. 407-417, 2008.
- [77] Cséfalvay et al., “Pre-symptomatic detection of *Plasmopara viticola* infection in grapevine leaves using chlorophyll fluorescence imaging,” *European Journal of Plant Pathology*, vol. 125, no. 2, pp. 291-302, 2009.
- [78] Munns et al., “New phenotyping methods for screening wheat and barley for beneficial responses to water deficit.,” *Journal of Experimental Botany*, vol. 61, no. 13, pp. 3499-3507, 2010.
- [79] Scholes et al. “Chlorophyll fluorescence imaging as tool for understanding the impact of fungal diseases on plant performance: a phenomics perspective,” *Functional Plant Biology*, vol. 36, p. 880–892., 2009.

- [80] Chaerle et al., “Multi-sensor plant imaging: towards the development of a stress-catalogue,” *Biotechnology Journal*, vol. 4, p. 1152–1167, 2009.
- [81] Lu, “Detection of bruises on apples using near-infrared hyperspectral imaging,” *Transactions of the ASAE*, vol. 46, no. 2, p. 523–530, 2003.
- [82] Xing and Baerdemaeker, “Bruise detection on ‘Jonagold’ apples using hyperspectral imaging,” *Postharvest Biology and Technology*, vol. 37, no. 2, p. 152–162, 2005.
- [83] Nicolaï et al., “Non-destructive measurement of bitter pit in apple fruit using NIR hyperspectral imaging,” *Postharvest Biology and Technology*, vol. 40, no. 1, pp. 1-6, 2006.
- [84] ElMasry et al., “Early detection of apple bruises on different background colors using hyperspectral imaging,” *LWT Food Science and Technology*, vol. 41, no. 2, p. 337–345, 2008.
- [85] Mahlein, “Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping,” *Plant disease*, vol. 100, no. 2, pp. 241-251, 2016.
- [86] Abdullah et al., “Classification of rubber tree leaf diseases using multilayer perceptron neural network,” in *5th student conference on research and development*, Selangor, 2007.
- [87] Huang and Kuo-Yi, “Application of artificial neural network for detecting Phalaenopsis seedling diseases using color and texture features,” *Computers and Electronics in Agriculture*, vol. 57, no. 1, p. 3–11, 2007.
- [88] Barbedo, “Digital image processing techniques for detecting, quantifying and classifying plant diseases,” *J.G. SpringerPlus*, vol. 2, 2013.
- [89] Mintel, “Sales of digital cameras decline as consumers snap up smartphones,” 29 June 2012. [Online]. Available: <http://www.mintel.com/press-centre/technology-press-centre/sales-of-digital-cameras-decline-as-consumers-snap-up-smartphones>. [Accessed 26 November 2016].
- [90] Statista, “Digital camera ownership: Number of people living in households that own a digital camera in the United States (USA) from spring 2008 to spring 2015 (in millions),” 2015. [Online]. Available: <https://www.statista.com/statistics/228876/people-living-in-households-that-own-a-digital-camera-usa/>. [Accessed 26 November 2016].
- [91] Statista, “Number of smartphone users worldwide from 2014 to 2020 (in billions),” 2016. [Online]. Available: <https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>. [Accessed 27 November 2016].
- [92] Peat UG, “Plantix,” Peat UG, 2015. [Online]. Available: <http://plantix.net/>. [Accessed 28 November 2016].

- [93] Rural and Marketing, "Plantix APP: Detecting crop disease and recommending treatment," 12 October 2016. [Online]. Available: <https://www.pressreader.com/>. [Accessed 28 November 2016].
- [94] Pongnumkul S., Chaovalit P., and Surasvadi N., "Applications of Smartphone-Based Sensors in Agriculture: A Systematic Review of Research," *Journal of Sensors*, vol. Vol 2015, no. 195308, p. 18, 2015.
- [95] Pew Research Center, "Cell Phones in Africa: Communication Lifeline," 15 April 2015. [Online]. Available: <http://www.pewglobal.org/2015/04/15/cell-phones-in-africa-communication-lifeline/>. [Accessed 29 November 2016].
- [96] Baydogan et al., "A Bag-of-Features Framework to Classify Time Series," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2796 - 2802, 2013.
- [97] Rueda et al., "Bag of Features for Automatic Classification of Alzheimer's Disease in Magnetic Resonance Images," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Berlin, Springer Berlin Heidelberg, 2012, pp. 559-566.
- [98] Caicedo J.C., Cruz A. and Gonzalez F.A., "Histopathology Image Classification using Bag of features," Bioingenium Research Group National University of Colombia, Colombia.
- [99] Camps-Valls and Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 6, pp. 1351 - 1362, 2005.
- [100] Melgani and Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 8, pp. 1778 - 1790, 2004.
- [101] Dixon B. and Candade N., "Multispectral landuse classification using neural networks and support vector machines: one or the other, or both?," *International Journal of Remote Sensing*, vol. 29, no. 4, pp. 1185-1206, 2008.
- [102] Lunetta et al., "Comparison of support vector machine, neural network, and CART algorithms for the land-cover classification using limited training data points," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 70, p. 78-87, 2012.
- [103] Z. E.A., "Support Vector Machines (SVMs) versus Multilayer Perception (MLP) in data classification," *Egyptian Informatics Journal*, vol. 13, no. 3, p. 177-183, 2012.
- [104] Keir, Turner and Yocom, "Client-Server Architecture," in *The Definitive Guide to Linux Network Programming*, Berkeley, Apress, 2004, pp. 99 - 135.
- [105] Bevan, "International Standards for HCI and Usability," *International Journal of Human Computer Studies*, vol. 55, no. 4, pp. 533-552, 2001.
- [106] Ethan Marchotte, "Responsive Web Design," 25 May 2010. [Online]. Available: alistapart.com/article/responsive-web-design. [Accessed 26 December 2016].

- [107] Webs Agency, “Why Flat Web Design is Always a Good Choice,” Webs Agency, 19 September 2016. [Online]. Available: <https://www.webs.agency/blog/why-flat-web-design-is-always-a-good-choice/>. [Accessed 14 December 2016].
- [108] Microsoft, “Microsoft Design,” [Online]. Available: <https://www.microsoft.com/en-us/design>. [Accessed 5 January 2017].
- [109] Google, “Google Design,” Google, [Online]. Available: <https://design.google.com/>. [Accessed 2 January 2017].
- [110] Apple Inc, “iOS Human Interface Guidelines,” Apple Inc, [Online]. Available: <https://developer.apple.com/ios/human-interface-guidelines/overview/design-principles/>. [Accessed 2 January 2017].
- [111] Bootstrap, “Bootstrap,” Twitter Org, [Online]. Available: getbootstrap.com. [Accessed 15 January 2017].
- [112] Itseez, “About OpenCV,” OpenCV.org, 2017. [Online]. Available: <http://opencv.org/about.html>. [Accessed 17 February 2017].
- [113] EmguCV, “EmguCV Main Page,” EmguCV, 9 January 2017. [Online]. Available: http://www.emgu.com/wiki/index.php/Main_Page. [Accessed 16 February 2017].
- [114] Hughes and Salathe, “An open access repository of images on plant health to enable the development of mobile disease diagnostics,” 12 April 2016. [Online]. Available: <https://arxiv.org/abs/1511.08060>. [Accessed 08 December 2016].
- [115] Yang et al., “Evaluating bag-of-visual-words representations in scene classification,” in *Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval*, Augsburg, Bavaria, 2007.
- [116] Alon Halevy, Peter Norvig and Fernando Pereira, “The Unreasonable Effectiveness of Data,” *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8-12, 2009.
- [117] Hepp, Stoll and Martin, “Detection of E-Commerce Systems with Sparse,” in *10th International Conference on e-Business Engineering*, 2013.
- [118] Banko, Michele, and Eric Brill, “Scaling to very very large corpora for natural language disambiguation,” in *Proceedings of the 39th annual meeting on association for computational linguistics*, 2001.
- [119] Wei-Xue Liu, Jian Hou and Hamid Reza Karimi, “Research on Vocabulary Sizes and Codebook Universality,” *Abstract and Applied Analysis*, vol. 2014, p. 7, 2014.
- [120] Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.
- [121] Mauricio, “Anatomy of the SIFT Method,” *Image Processing On Line*, vol. 4, p. 370–396, 2014.
- [122] Alexa Internet Inc, “The top 500 sites on the web,” Alexa Internet, February 2017. [Online]. Available: <http://www.alexa.com/topsites>. [Accessed 26 February 2017].

[123] Opera, "Opera Mini," Opera, [Online]. Available: <http://www.opera.com/mobile/mini>. [Accessed 17 February 2017].

[124] FAO, IFAD, WFP, The State of Food Insecurity in the World 2015. Meeting the 2015 international hunger targets: taking stock of uneven progress, Rome: FAO, 2015.

9 Appendices

Appendix A: Bag of features file

<https://www.dropbox.com/s/mjcntp5u30a1s/ClusteredSiftFeatures.yml?dl=0>

Appendix B: SVM classifier model

<https://www.dropbox.com/s/w7exhm62v77olh0/1022017015797.xml?dl=0>

Appendix C: Training results

<https://www.dropbox.com/s/n6tzwjly70nkg0/TestResults.xsd?dl=0>

Appendix D: Diagnosis results

<https://www.dropbox.com/s/mly5edxjadd7phe/DiagnosisResults.xsd?dl=0>