Addis Ababa University

College of Natural Sciences

Locust Identification System Using Deep Neural Network

Mebrahtu Chaklu Gebretensiae

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

March 2021

Addis Ababa University

College of Natural Sciences

Mebrahtu Chaklu Gebretensiae

Advisor: Yaregal Assabie (PhD)

This is to certify that the thesis prepared by Mebrahtu Chaklu Gebretensiae, titled: *Locust Identification System using Deep Neural Network* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Advisor: __**Yaregal Assabie (PhD)**_____ Signature _____Date_____

Examiner: _____ Signature _____Date_____

Examiner: _____ Signature _____Date_____

Abstract

Locusts are the most dangerous pests that cause the devastation of crops and plants around the world. The consequences of locust attacks are disastrous for both food security and livelihoods of the rural populations. Currently, effective identification of these insects requires large numbers of trained experts and may cost millions of dollars. On top of that, the large amount of chemical insecticides used can have serious side effects on the environment. This means precise automated locust identification systems could minimize total amount of chemical usage to combat the locusts.

To identify the desired locust this research mainly focusses on analyzing and extracting unique features of the locust from the acquired training dataset and then train the proposed system accordingly. However, to include all locust types the collected 412 total training image datasets have distinctive ranges of sizes, shapes, backgrounds and views. Having such distinctions makes them complex for identification. Thus, to solve this kind of complexity and to increase identification accuracy, we use deep neural network model such as ResNet50.

Therefore, the design of the proposed system model includes various stages starting from acquiring locust images up to identifying locust. Finally, the performance of the proposed system is assessed by its ability to identify locusts from a given test dataset. According to the experiment results, this research work has shown that the performance of the proposed system is 95.2% when it comes to classification of locusts from non-locusts using ResNet50.

Key Words: - Locust Swarms, Locust Detection, Locust Identification, Deep Neural Network.

Acknowledgements

First, I wish to express my sincere appreciation to my advisor Dr. Yaregal Assabie for his motive and initiative to convincingly comment, suggest, guide, appreciate and encourage me right from the moments of problem formulation to the completion of the research work. It is real with his continuous follow up, encouragement and advice this thesis came up to realization. Without his persistent help, the goal of this thesis would not have been realized.

I would also like to take this opportunity to express my special regards to acknowledge the support and great love of my family, my wife Belay Jejaw; my daughter, Kidist Mebrahtu; my mother, Birke Gebrekidan; my father, Chaklu Gebretensae and all my sister and brothers. They kept me going on and this work would not have been possible without their input.

I would also like to thank my best friend Mebrahtu Teshale and my colleague Liku Zelleke for sharing of valuable ideas, knowledge and encouragement. Without their support, I may not have the opportunity to finish my thesis. Finally, I would like to recognize the invaluable assistance that you all provided me during my study.

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

ANN: Artificial Neural Network

CV: Computer Vision

DCNN: Deep Convolutional Neural Network

GPU: Graphics Processing Unit

MRA: Microsoft Research Asia

PDE: Partial Deferential Equation

RCNN: Region-Based Convolutional Neural Networks

RGB: Red Green Blue

SURF: Speed-Up Robust Features

SVM: Support Vector Machines

TTI : Total Train Images

Chapter One: Introduction

1.1 Background

Locusts are a collection of certain species of short-horned grasshoppers in the Acrididae family group that have a swarming phase [1]. These insects are usually solitary, but under certain circumstances they become more abundant and change their behavior and habits, becoming gregarious [1]. Locusts represent perhaps the most conspicuous of all insect pests. They can be found in sandy, dry grassland, and deserts in abundance. Locusts have had a devastating effect since the early days of agriculture and they are among the most dangerous agricultural pests. Locusts were first recognized as the causes of serious agricultural problem in the 1920s by British authorities in Iraq [1].

Locusts are extremely different from other pests. Their populations can quickly grow to catastrophic levels, and some species form very dense bands and swarms that can cause a great deal of damage in a very short time [2]. Locust can migrate hundreds of kilometers per day and invade areas covering millions of square kilometers, resulting in major economic, social, and environmental impacts on an international scale. This behavior of locusts makes them an international challenge.

The consequences of locust attacks can be disastrous for both food security and livelihoods of the rural populations in affected areas [2]. Thus, proper locust identification is critical to food security worldwide and often requires governmental or international involvement. The aim of identification is to treat as many locust hotspots as possible before they can damage crops by intervening early during outbreaks. However, outbreak areas are not always limited in size, and some locusts have widespread favored habitats that cover hundreds of thousands of square kilometers or more and some areas can be remote with limited access [2].

Locusts will damage most green plants and crops. As per Food and Agriculture Organization estimates, a single locust can consume roughly its own weight in fresh food per day, which is about two grams' worth [3]. The number of locusts in a swarm can vary from less than one square km to

several hundred square km and in a single small swarm may have around 80 million individual locusts. This makes it important to find an efficient way to fight these pests.

Today, the identification of locusts is done using various techniques like farmers or local officials' visual inspection and field survey reports by experts [4]. Expert field survey result can be reported every two weeks to the higher concerned department using the attached desert locust standard survey form that shown on Annex A. This means the accuracy of identifying a locust may highly rely on the expert's field survey and farmer's visual inspection reports results. However, locusts can move and change their location within a short time, and this leads the field survey report result to vary in location from the actual one from time to time. This means, the correct identification of the locusts is highly important for the next stage which is combating the pests using insecticides.

Since these traditional locust identification methods are the main input for spraying insecticides to combat locusts, it makes the currently available insecticide spraying techniques traditional and not as effective as they should be. This traditional insecticide spraying techniques encounters many problems like the loss of large amounts of chemicals, increase of negative environmental impacts, destruction or killing of other insects due to the inability to properly identify the target locusts. This is due to the fact that spraying insecticides on the whole estimated areas as per the field survey locust identification report is a waste of chemical and highly pollutes the environment.

In addition, the identification campaigns usually cost millions of dollars and needs large number of experts to collect locust information through field surveys. Thus, development of new, optimal and effective locust identification strategies is mandatory and quite beneficial. In order to minimize the cost, time consumption and labor energy it is necessary to develop a system that identifies or detects locusts using latest technology. Due to the rapid development of digital technology, there is an opportunity for digital image processing technology to be used in the field of locust identification.

Image processing is a method of performing operations on an image, in order to get an enhanced image or to extract some useful information from it [5]. It is a type of signal processing in which input is an image and output may be image or characteristics or features associated with that image. Image processing is a term that indicates the processing of an image (or video frame) that is taken as an input and results in a set of related parameters of the image [6]. Image processing plays a

key role in object identification and tracking. Thus, in this research image processing may provide us with a realistic opportunity for the automation of locust identification using deep neural network model.

1.2 Motivation

Locusts continue to pose a threat to agriculture in Africa. This has been especially visible from 2019 up to 2021 in Ethiopia. Losses due to locusts are not limited to damage to pastures and crops. However, the rapid loss of vegetation cover may result in soil erosion and increase the risk of food insecurity. Ethiopian government reported that some parts of the country have been attacked by huge swarms of locust [7]. FAO disclosed that the rapid incursion of the locust across many regions of Ethiopia has already resulted in significant losses on crops and jeopardized the livelihoods of smallholder farmers that depend on the crops and livestock.

Farmers and local officials have been working with the FAO to combat the locusts. Ethiopian agriculture officials have also reported that they were taking steps to handle a major locust infestation that could threaten some of the country's staple crops [8]. As per the government reports, it had sent planes to the affected areas to try to deal with the problem from the air and farmers are using traditional methods such as dispersing the swarms with smoke and picking the locust off crops by hand [8]. However, as per the guideline the control measures in place have failed to stop the locust swarms. Therefore, it is this frequent and devastating locust infestation and the disaster they cause on the country as a whole that motivated us to find a solution that could, at least to identify the locust easily using digital technology.

1.3 Statement of the Problem

Up to now, field survey reports are still perceived as the only effective way of locust identification techniques. However, locusts are by nature highly mobile insects and can quickly move from region to region, country to country, and even from continent to continent and covering more than 100 km per day. This behavior of locusts makes them difficult to identify. Because the identified reports from the field surveys may not reflect the exact current location of the locusts. Thus, if there is a variation on the field survey report after a time or days it is difficult to spray insecticide over all the areas reported in the past.

Then there is the fact that currently we use locust identification techniques like using farmers and field survey experts through inspection. However, identifying locusts in abyssal or inaccessible remote areas by human inspection is difficult. Thus, these unidentified locusts found in the abyssal areas, may reproduce and return to devastate crops if they are not properly identified soon.

Due to this fact, we propose to modify or upgrade the current available traditional identification technique using digital technology. Thus, developing a system that uses a digital image will help to identify locusts easily and effectively. Therefore, in this research work, we plan to develop a locust identification system that identify locust which found within an image easily using deep neural network model.

Few researches have been conducted an automatic identification system for different agricultural pests which are related to locust identification such as Automatic Segmentation Cotton Insects and Pests [25], Cognitive Vision Method for Insect Pest Image Segmentation [26], Pest Control in Agricultural Plantations Using Image Processing [30], Early Pest Detection from Crop Using Image Processing and Computational Intelligence [27], etc. However, the technique applied for these related works can not directly apply for the identification of locusts due to the difference in morphological, shape and texture features. Thus, this research work aims at developing automatic locust identification system taking locust specific characteristics into account.

1.4 Objectives

General Objective

The general objective of this research is to develop a locust identification system using deep neural network model.

Specific Objectives

In order to achieve the general objective, the following specific objectives have been identified.

- Review previous works that are related to locust identification.
- Collect locust sample representing of the different feature of locust.
- Design an algorithm for extracting feature of locusts.
- Design architecture of the proposed system.
- Develop a prototype.

❖ Test and evaluate the system.

1.5 Methods

To achieve the general and specific objectives the research work will follow different methodologies. They are described as follows:

Literature Review

To understand the subject matter, literatures on currently available technologies that are related to the work we are going to develop will be reviewed, studied and analyzed.

Data Collection

Locust images dataset will be collected to be used for training and testing. The data will be collected from the web and Desert Locust Control Organization for East African field survey reports.

Tools

To assess the performance of the locust identification system, a prototype will be developed using MATLAB software. This tool has a great capability on providing a comprehensive set of reference-standard algorithms and workflow applications for image processing, analysis, visualization and algorithm development. This will provide an insight on the applicability of the system.

1.6 Scope and Limitations

The main focus of this research is to develop a system which automatically identifies locusts in an image. However, identification of the locust in a video will not be included. This implies that the locust movement direction and speed will not be identified.

1.7 Application of Results

Some of the advantages of automated identification of locust system are listed as follows:

- Easily identifying locusts: the development of this system increases the effectiveness of the identification of locusts, which found in remote areas that may not be accessible to humans

by taking the image using a drone. It also increases the correctness and reliability of the identification process by minimizing the report time gap and number of experts assigned in the field survey to identify the locusts by replacing through digital technology.

- Optimal use of resources: to identify locust, many resources are required including manpower, time, vehicles and airplanes. Thus, this research work helps with the optimization of these resources.
- Minimize negative environmental impact: the proposed solution opens an opportunity to spray chemicals using drones only on identified locusts. This decreases the pollution of natural resources like water and air, while protecting other insects like bees and reptiles that have a positive impact on the ecosystem. However, current available technology follows a full-cover spraying technique based on the last reported field survey results and by estimating next locust movement by considering wind direction.

1.8 Organization of the Rest of the Thesis

The remaining part of the thesis is organized as follows. Chapter Two is about literature review which discusses the general background of locust, digital image processing and the related subject areas. Understanding the general background of the locust enables us to grasp some basic ideas about the domain of this research area. Chapter Three presents different types of related works that are attempted to develop on different digital solutions related to locust identification systems. Chapter Four gives a detailed description of the architecture and design issues of the proposed system. The main components of the proposed system architecture and their functional operation are discussed in detail. Chapter Five presents the experimental results of the proposed system prototype. In Chapter Six conclusions, contributions of the thesis and future works will be pointed out.

Chapter Two: Literature Review

2.1Introduction

This Chapter focuses on detail theoretical background description of locust and digital image processing techniques. The first part presents the definition, anatomy and lifecycle of the locust. The second part describes in detail about the fundamental steps of digital image processing and deep neural network. In general, to identify target locust using deep neural network model, we start from acquisition. After the images are captured there are a number of processes that we follow to reach the desired goal of a machine vision system.

2.2 Definition of Locust

Locusts are members of the grasshopper family Acrididae, which includes most of the short-horned grasshoppers [9]. Locusts differ from grasshoppers because they have the ability to change their behavior and physiology, in particular their color and shape (morphology) in response to changes in density. Adult locusts can form swarms which may contain thousands of millions of individuals and which behave as a unit.

Locusts are typically in motion and can cover vast distances, some species may travel 81 miles or more a day [10]. Locusts devastate crops and cause major agricultural damage, which can lead to famine. Locusts are large herbivorous insects that can be serious pests of agriculture due to their ability to form dense and highly mobile swarms. Locusts occur in many parts of the world. However, nowadays locusts are most destructive in subsistence farming regions of Africa. In Europe the term locust denotes large acridids, whereas smaller species are called grasshoppers. In North America the names locust and grasshopper are used for any acridid [10].

A phase theory has been developed to account for the sporadic appearance and disappearance of locust swarms [10]. According to the theory, a plague species has two phases: one **solitary** and the other **gregarious**. The phases can be distinguished by differences in color, form, physiology, and behavior. For instance, a solitary-phase nymph, adjusts its color to match that of its surroundings, does not collect in groups, has low metabolic and oxygen-intake rates, and is sluggish. A

gregarious-phase nymph, on the other hand, has black and yellow or orange color in a fixed pattern, gathers in large groups, has high metabolic and oxygen intake rates, and is active and nervous.

Adult locusts differ more in form than in color. The solitary phase has shorter wings, longer legs, and a narrower pronotum, or dorsal sclerite (with higher crest and larger head), than the gregarious phase. The adult of the gregarious phase has a more saddle-shaped pronotum, broader shoulders, and longer wings.

When a nymph of a solitary-phase locust matures in the presence of many other locusts, it undergoes a physiological change and produces offspring of the gregarious type. If crowding is sufficiently dense and of long enough duration, the majority of a local population will shift to the gregarious migratory phase. The nymph of a gregarious-phase locust, on the other hand, will produce offspring that reverts to the solitary phase if it matures in isolation. The solitary phase is the normal state of the species, the gregarious phase being a physiological response to violent fluctuations in the environment.

A gregarious-phase locust is restless and irritable, and it flies spontaneously on warm dry days, when its body temperature is high [10]. The muscular activity of flight further raises its temperature. A swarm ceases flying only when environmental conditions change such as if there is rain fall, temperature decrease, or darkness occurs. The long-distance dispersal of these swarms is usually associated with either frontal winds of storm systems or high-level jet-stream winds. The acridids typically fly almost straight up into these fast-moving winds and then are carried with the winds until they slow to the point where gravity overcomes wind speed, causing them to drop from the sky.

2.2.1 Anatomy of Locust

The body of a locust can be divided into three main parts: head, thorax and abdomen. Figure 2.1 [11] shows the anatomy of locust.

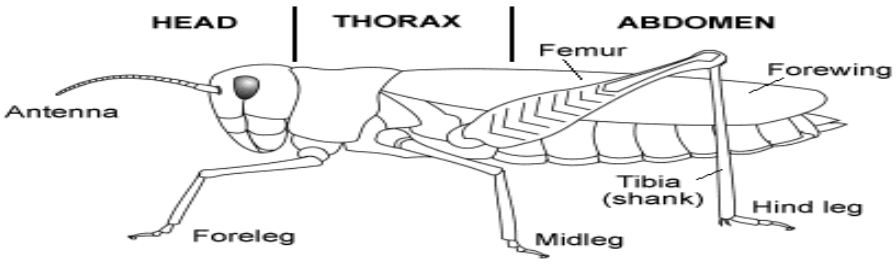

Figure 2.1: Anatomy of Locust

I. Head

On the head can be seen:

- ✓ A pair of jointed antennae or feelers which the locust uses to recognize things by touch or smell.
- ✓ A pair of large compound eyes which give the locust a wide field of vision and enable it to detect movement easily. It is not known how many colors locusts can recognize but it has been shown that they react to green.
- ✓ The mouth which has several parts that can easily be separated and identified. These are the upper lip, a pair of hard, black, serrated jaws which move sideways to cut through plant food, a pair of secondary jaws which help in holding the food, and a lower lip.
- ✓ Jointed appendages which are called palps. These are used for tasting food.

II. Thorax

This is the part of the body which contains the muscles for walking, jumping and flying, and to which the wings and legs are attached [11]. On the thorax can be seen:

- ✓ A sheath covering the top and sides of the front part of the thorax. It is called the pronotum.
- ✓ Three pairs of legs, the hind legs being large and used for jumping. Each leg has three main parts: - The femur, tibia and the tarsus or foot.
- ✓ Two pairs of wings. At rest the harder front wings cover and protect the softer hind wings which are folded fan-wise.

III. Abdomen

On the abdomen can be seen:

- ✓ The ear, on the first section of the abdomen just behind the first joint of the large back legs. This is where the locust receives sounds. Locusts can hear one another up to about 2m apart.
- ✓ The ovipositor valves in the female. These are two pairs of short, curved, black hooks which form the tool with which the female locust digs a hole in the soil when the eggs are laid. This is how the female can be distinguished from the male, as the male does not have these hooks.

2.2.2 Lifecycle of Locust

Locusts undergo incomplete or direct metamorphosis. Unlike in insects such as butterflies or moths, there is no pupal stage and juveniles are similar in appearance to adults [11]. There are three main stages of development: egg, nymph and adult. The nymph or hopper stage can be further divided into growth stages called instars, with a moult between each. Figure 2.2 [11] shows the life cycle of the Australian plague locust which has five instar stages. The times given for development are approximate for optimum conditions during summer.

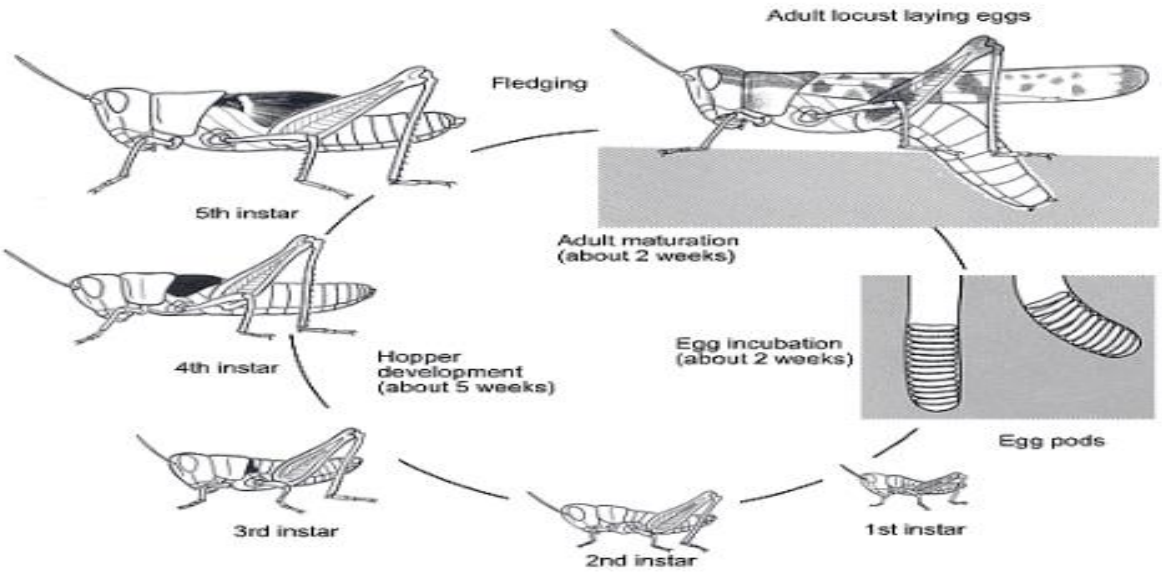

Figure 2.2: Lifecycle of Locust

Egg Stage

Locust eggs are usually laid in areas of bare sandy soil [11]. The female drills a hole into the ground using the ovipositor at the tip of the abdomen and lays a 'pod' of eggs which is sealed with froth. The froth helps to protect the eggs from desiccation, disease and predation. The eggs look like rice grains and are arranged like a miniature hand of bananas. The female will not lay unless the soil is moist at about 5-10 cm below the surface. In soft sandy soils, females have been known to lay when moisture is only found at depths below 12 cm. Before laying, the female will often probe the soil, inserting the tip of her abdomen to determine if there is enough moisture.

Nymphs stage

The locust eggs hatching takes about two weeks after they were laid. These baby locusts are referred to as "hoppers" or "nymphs." Over the next month to two months after hatching, the nymph locusts go through five molting stages called "instars." After the fifth instar, the locust's wings are fully developed.

Adult Stage

The final moult into the adult stage is known as fledging, when the locust develops fully formed flying wings. It takes a few weeks before the young adults of most species lay eggs. How long it takes for a locust to reach maturity depends on the species, habitat conditions and temperature [11]. Nymph and adult locusts are able to regulate their body temperature by basking in the sun or moving to shade.

2.3 Fundamental Stages of Digital Image Processing

An image is a 2-dimensional matrix of intensity (gray or color) values. It is defined as a two-dimensional function, f(x, y), where x and y are spatial (plane) coordinates, and the amplitude of *f* at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point [12]. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image. Therefore, digital image is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels. The field of digital image processing refers

to processing digital images by means of a computer. Digital image processing focuses on developing a computer system that is able to perform processing on an image.

Many image processing applications may follow different stages. However, there are some common fundamental steps that all most many image processing applications pass through. These fundamental steps of image processing are shown in Figure 2.3 [12] .

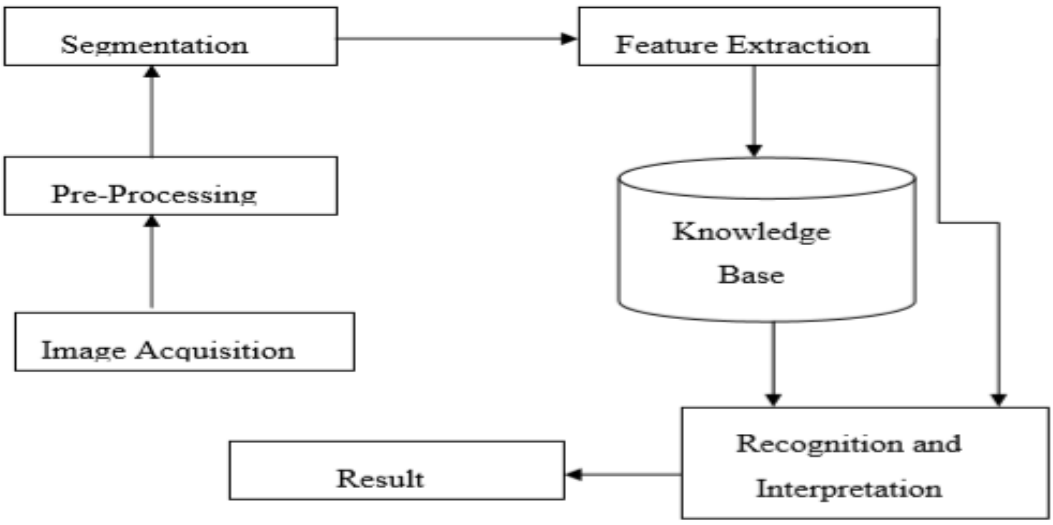

Figure 2.3: Fundamental Steps of Digital Image Processing

2.3.1 Image Acquisition

Image Acquisition is purely hardware dependent process, in which reflected light energy from the object being imaged is converted into electrons and spread over the internal sensor chip which is like a 2-D array of cells and the cell is called photo-site and contain amount of charges which is further converted to digital form using analog to digital converter [12]. Image acquisition is the first point of data entry into a picture archiving and communication system. Image acquisition in image processing can be broadly defined as the action of retrieving an image from some source, usually a hardware-based source, so it can be passed through whatever process needed to occur afterward. In order to capture an image a camera requires some sort of measurable energy. The energy of interest in this context is light or more generally electromagnetic wave. One of the ultimate goals of image acquisition is to have a source of input that operates within such controlled

and measured guidelines that the same image can, if necessary, be nearly perfectly reproduced under the same conditions so anomalous factors are easier to locate and eliminate [13].

2.3.2 Image pre-processing

Pre-processing is a common name for operations with images at the lowest level of abstraction - both input and output are intensity images [14]. These iconic images are of the same kind as the original data captured by the sensor, with an intensity image usually represented by a matrix of image function values. The aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing, although geometric transformations of images. Some of image pre-processing tasks are image enhancement, restoration, noise removal, resize, etc.

2.4.3 Image Segmentation

Image segmentation is the technique of dividing or partitioning an image into parts, called segments [15]. Image segmentation is a fundamental process in many image, video, and computer vision applications. Image segmentation technique is used to partition an image into meaningful parts having similar features and properties. The main aim of segmentation is simplification such as representing an image into meaningful and easily analyzable way and divide an image into several parts/segments having similar features or attributes. Image segmentation is necessary first step in image analysis. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

The popular techniques of image segmentations are threshold method, edge detection-based techniques, region-based techniques, clustering based techniques, watershed based techniques, partial differential equation based and artificial neural network based techniques. Figure 2.4 [15] shows techniques of image segmentation.

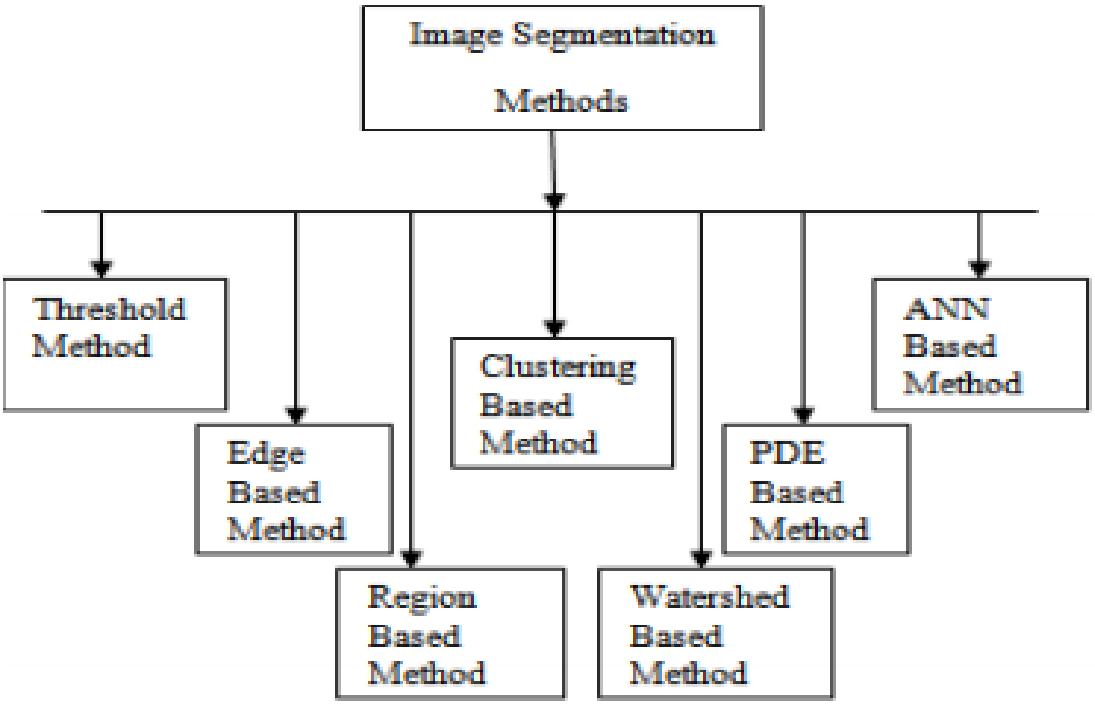

Figure 2.4: Image Segmentation Techniques

The threshold methods are the simplest methods for image segmentation and it divides the image pixels with respect to their intensity level [16]. However, the edge-based segmentation methods are based on the rapid change of intensity value in an image because a single intensity value does not provide good information about edges. In edge-based segmentation methods, first of all the edges are detected and then connected together to form the object boundaries to segment the required regions [17]. Farther to that, the region-based segmentation methods are segments the image into various regions having similar characteristics. In the other way the clustering-based method are the techniques, which segment the image into clusters having pixels with similar characteristics [18]. In the proposed system we will use the threshold segmentation method because this is simple and effective for locust shape or anatomy based feature extraction. Table 2.1 shows a comparison between various segmentation techniques by specifying its advantage and disadvantages.

Table 2.1: Comparison of Various Segmentation Techniques

Segmentation technique	**Description**	**Advantages**	**Disadvantages**
Threshold Method	based on the histogram peaks of the image to find particular threshold values	no need of previous information, simplest method	highly dependent on peaks, spatial details are not considered
Edge-Based Method	based on discontinuity detection	good for images having better contrast between objects	not suitable for wrong detected or too many edges
Region Based Method	based on partitioning image into homogeneous regions	more immune to noise, useful when it is easy to define similarity criteria	expensive method in terms of time and memory
Clustering Based Method	based on division into homogeneous clusters	fuzzy uses partial membership therefore more useful for real problems	determining membership function is not easy
Watershed Based Method	based on topological interpretation	results are more stable, detected boundaries are continuous	complex calculation of gradients
PDE Based Method	based on the working of differential equations	fastest method, best for time critical applications	more computational complexity
ANN Based Method	based on the simulation of learning process for decision making	no need to write complex programs	more wastage of time in training

2.3.4 Feature Extraction

Feature is defined as a function of one or more measurements, each of which specifies some quantifiable property of an object and is computed such that it quantifies some significant characteristics of the object [19]. Feature extraction is the process to represent raw image in a

reduced form to facilitate decision making such as pattern detection, classification or recognition. Finding and extracting reliable and discriminative features is always a crucial step to complete the task of image recognition and computer vision. Furthermore, as the number of application demands increase, an extended study and investigation in the feature extraction field becomes very important. The goal of feature extraction is to generate features that exhibit high information-packing properties such as extract the most relevant information from the raw data for discrimination between the classes and discard redundant information.

Features can be coarsely classified into low-level features and high-level features. Low-level features can be extracted directed from the original images, whereas high-level feature extraction must be based on low-level features [20]. Low-level features are basic features that can be extracted automatically from an image without any shape information (information about spatial relationships). Whereas, high-level feature extraction concerns finding shapes in computer images. For example, to be able to recognize faces automatically, one approach is to extract the component features. This requires extraction of the eyes, the ears and the nose, which are the major face features.

The low-level features can be categorized as follows: General features, Global features and Domain-specific features.

- ✓ **General features**: Application independent features such as color, texture, and shape. These features, according to the abstraction level, they can be further divided into:
    - Pixel-level features: Features calculated at each pixel, e.g., color, location.
    - Local features: Features calculated over the results of subdivision of the image band on image segmentation or edge detection.
- ✓ **Global features**: Features calculated over the entire image or just regular sub-area of an image.
- ✓ **Domain-specific features**: Application dependent features such as human faces, fingerprints, character recognition and conceptual features.

Feature Extraction Types

Many feature extraction types have been used to extract features from images. Some of the commonly used feature extraction types are as follows:

i. *Color features*: The color feature is one of the most widely used visual features in image processing. Typically, the color of an image is represented through some color models. There exist various color model to describe color information. A color model is specified in terms of 3-D coordinate system and a subspace within that system where each color is represented by a single point. The more commonly used color models are RGB (red, green, blue), HSV (hue, saturation, value) and Y,Cb,Cr (luminance and chrominance).

ii. *Shape features*: Visual features of objects are called the shape characteristics or visual features [21]. For example, circular object or triangular objects or other shapes, perimeter boundary of the object, the diameter of the border and so on. The visual features showed intuitively all belong to shape features. Moment, perimeter, area and orientation are some of the characteristics used for shape feature extraction technique. The accuracy of the shape feature extraction result depends on the pre-segmentation effect.

iii. *Texture features*: texture is a repeated pattern of information or arrangement of the structure with regular intervals [21]. In a general sense, texture refers to surface characteristics and appearance of an object given by the size, shape, density, arrangement, or proportion of its elementary parts. A basic stage to collect such features through texture analysis process is called as texture feature extraction. Due to the significance of texture information, texture feature extraction is a key function in various image processing applications like remote sensing, medical imaging and content-based image retrieval.

iv. *Spatial Features*: Spatial features of an object are characterized by its gray level, amplitude and spatial distribution. Amplitude is one of the simplest and most important features of an object. In X-ray images, the amplitude represents the absorption characteristics of the body masses and enables discrimination of bones from tissues.

2.4.5 Recognition and Interpretation

Once after completing segmentation, these segmented pixels are to be represented and described in any form for further processing. The pixels are to be described, based on representation. Representation can be either internal or external. External representation usually focuses on shape characteristics, whereas internal representation focuses on color and texture based regional properties. Recognition is carried over by any of the soft computing techniques.

Soft Computing techniques normally deal with imprecision, uncertainty and partial truth [22]. It forms the basis for various machine learning algorithms. Soft computing techniques normally employ hard tasks to those problems, in which there is no algorithm to find better solution. It is also different from hard computing, since it is tolerant to imprecision, uncertainty, etc. Soft computing solutions are usually unpredictable and uncertain. Their values are always between 0 and 1. Soft computing techniques resemble a biological neuron of human mind. Various components of soft computing techniques include Support Vector Machines (SVM), Artificial Neural Networks (ANN) and Deep Convectional Neural network (DCNN).

Support vector machines are the supervised learning models that analyze and recognize patterns [22]. It is a classifier method performing classification by constructing a hyperplane in a multidimensional space and separating the class labels. Given a set of training samples each sample belonging to one of two classes, SVM builds a training model that assigns the samples into one of the categories, making it a binary classifier. It is the decision plane that separates between objects having different class memberships. As an example, let us consider two objects having colors dark and light in Figure 2.5 [22]. The separating line here is the boundary, in which objects to the right of the boundary are dark and to the left are light. Any new object that falls on the right side is labeled as dark and those on the left is labeled as light.

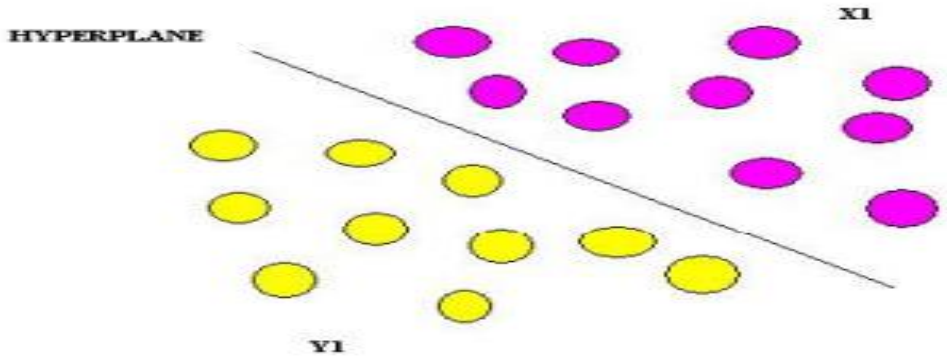

Figure 2.5: Linear Classifier

In the case of nonlinear functions, the original objects are mapped using a set of mathematical functions known as kernels. The process of rearranging the objects is known as mapping. The only thing here is to find an optimal line that separate dark and light objects.

Artificial Neural Networks are used to learn patterns and relationships between patterns [22]. The main motive of neural network is to mimic the human ability by adapting themselves to the changing environment. They also have the capability to learn from past events and react accordingly. The networks have the capability to learn complex nonlinear relationships and can adapt themselves according to the data provided and produce better results. ANN (Artificial Neural Network) consists of a number of nodes that are analogous to the neurons in the human brain. Each node has a function with a set of parameters determining the output, for the given input. By altering the parameters, the node function gets modified and the process continues till correct response is obtained. The signal that exists between the nodes is transmitted through the connection links between them. The connection links possess weight value that is multiplied with the given input signal to obtain the net input value. The output is obtained by applying the activation to the net input. The process continues until the desired output is obtained.

Architecture of Artificial Neural Network

Figure 2.6 [23] shows the architecture of artificial neural network. This artificial neural network is formed in three layers, called the input layer, hidden layer, and output layer. Each layer consists of one or more nodes or neurons. As we can see in Figure 2.6, neurons are represented by the small circles. Each neuron has an activation function that defines the output of the neuron. The activation function is used to introduce non-linearity in the modeling capabilities of the network. The lines between the nodes indicate the flow of information from one node to the next.

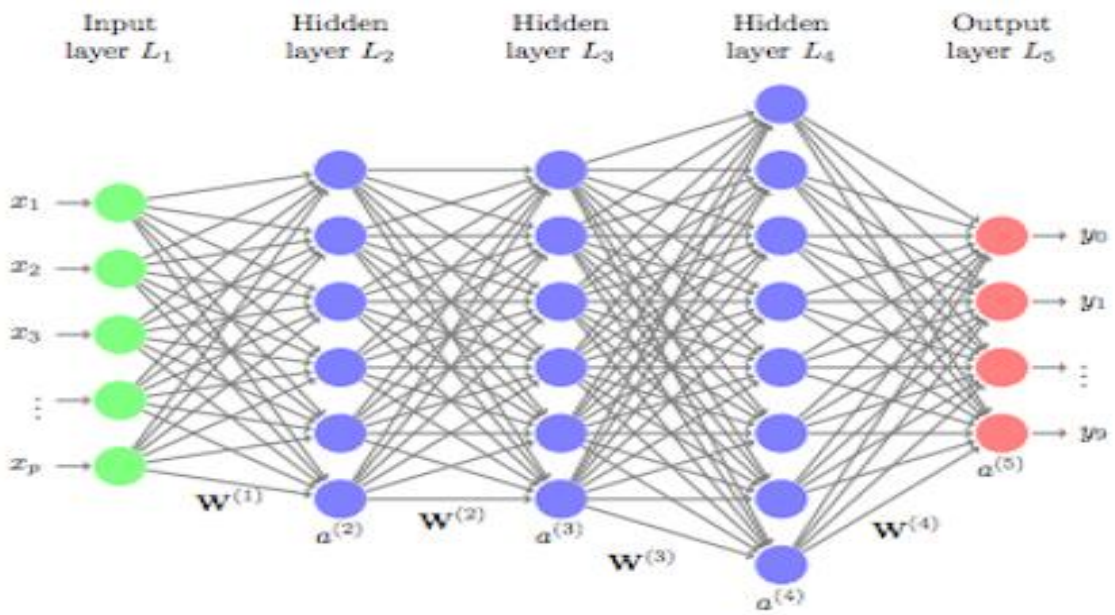

Figure 2.6: Architecture of Artificial Neural Network

Input layer of an artificial neural network is composed of artificial input neurons and brings the initial data into the system for further processing by subsequent layers of artificial neurons [23]. These inputs can be loaded from an external source such as a dataset or a csv file. Always there must be one input layer in a neural network. The input layer takes in the inputs, performs the calculations via its neurons and then the output is transmitted onto the subsequent layers. The number of neurons in an input layer is dependent on the shape of the training data. Equation 1 [23] shows how to get the total numbers of neurons,

Number of neurons = Number of training Data Features + 1 *(1)*

where *one* additional node is to capture the bias term.

Hidden layer is located between the input and output of the algorithm, in which the function applies weights to the inputs and directs them through an activation function as the output [23]. The hidden layers perform nonlinear transformations of the inputs entered into the network. There could be zero or more hidden layers in a neural network. Usually, each hidden layer contains the same number of neurons. Many experiments have shown us the optimum number of neurons in a hidden layer can be determined as shown in Equation 2 [23].

$$Number\ of\ neurons = \frac{Training\ Data\ Samples}{Factor\ *(Input\ Neurons + Output\ Neurons)} \quad (2)$$

where the factor is used to prevent over-fitting and it is a number between 1 and 10.

Output layer is the last layer of neurons that produces given outputs for the program [23]. The output layer is responsible for producing the final result. Always there must be one output layer in a neural network. The number of neurons on output layer is depends on the problem we need to solve such as whether we are attempting to work on a classification or regression problem. Thus, if our neural network is a classifier, then it has a single node. However, if we use a probabilistic activation function such as *softmax* then the output layer has one node per class label in our model.

Weights (W) are the co-efficients of the equation which we are trying to resolve. Negative weights reduce the value of an output. When a neural network is trained on the training set, it is initialized with a set of weights. These weights are then optimized during the training period and the optimum weights are produced.

Activation function (a) is nothing but a mathematical function that takes in an input and produces an output. The function is activated when the computed result reaches the specified threshold. The input in this instance is the weighted sum plus bias as show in Equation 3 [23]:

$$Output = activation\ function\ (X1W1 + X2W2 + ...XnWn + Bias) \quad (3)$$

where X1, X2,....Xn are the inputs and W1,W2..Wn are the weights. Bias is simply a constant value (or a constant vector) that is added to the product of inputs and weights. Bias is utilized to offset the result. The bias is used to shift the result of activation function towards the positive or negative side.

Limitation of Artificial Neural Network is that the output may sometimes become unstable when applied to larger problem. The mathematical theories that are used to guarantee the performance of a neural network need improvement. The better solution for it may be to train and test the intelligent systems by varying the number of hidden layers. It is also proved by some researchers that, increasing the number of hidden layers improves the recognition accuracy [23].

Deep Convolutional Neural Networks are neural networks used primarily to classify images (i.e., name what they see), cluster images by similarity (photo search), and perform object recognition within scenes [23]. The efficiency of convolutional nets in image recognition is one of the main reasons why the world has woken up to the efficiency of deep learning. In a sense, Convolutional neural networks (CNN) are the reason why deep learning is famous. Deep learning is the name we use for "stacked neural networks" that is, networks composed of several layers [23]. In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. CNNs are powering major advances in computer vision (CV), which has obvious applications for self-driving cars, robotics, drones, security, medical diagnoses, and treatments for the visually impaired.

Convolutional neural networks can also perform more banal (and more profitable), business-oriented tasks such as optical character recognition (OCR) to digitize text and make natural language processing possible on analog and hand-written documents, where the images are symbols to be transcribed. CNNs are not limited to image recognition, they have been applied directly to text analytics also and they can be applied to sound when it is represented visually as a spectrogram, and graph data with graph convolutional networks [23]. Deep convolutional Neural networks help us to *cluster* and *classify* any data store.

How Deep Convolutional Neural Networks Work?

The first thing to know about convolutional networks is that they don't perceive images like humans do [23]. Therefore, we are going to have to think in a different way about what an image means as it is fed to and processed by a convolutional network. Convolutional networks perceive images as volumes, i.e., three-dimensional objects, rather than flat canvases to be measured only by width and height. That's because digital color images have a red-blue-green (RGB) encoding, mixing those three colors to produce the color spectrum humans perceive. A convolutional network ingests such images as three separate layers of color stacked one on top of the other. So, a convolutional network receives a normal color image as a rectangular box whose width and height are measured by the number of pixels along those dimensions, and whose depth is three layers deep, one for each letter in RGB.

We will need to pay close attention to the precise measures of each dimension of the image volume, because they are the foundation of the linear algebra operations used to process images [23]. Now, for each pixel of an image, the intensity of R, G and B will be expressed by a number, and that number will be an element in one of the three, stacked two-dimensional matrices, which together form the image volume. Those numbers are the initial raw sensory features being fed into the convolutional network, and the convolutional network's purpose is to find which of those numbers are significant signals that actually help it classify images more accurately. Rather than focus on one pixel at a time, a convolutional network takes in square patches of pixels and passes them through a filter. That filter is also a square matrix smaller than the image itself, and equal in size to the patch. It is also called a kernel, which will ring a bell for those familiar with support-vector machines, and the job of the filter is to find patterns in the pixels. Figure 2.7 [24] shows Architecture of DCNN.

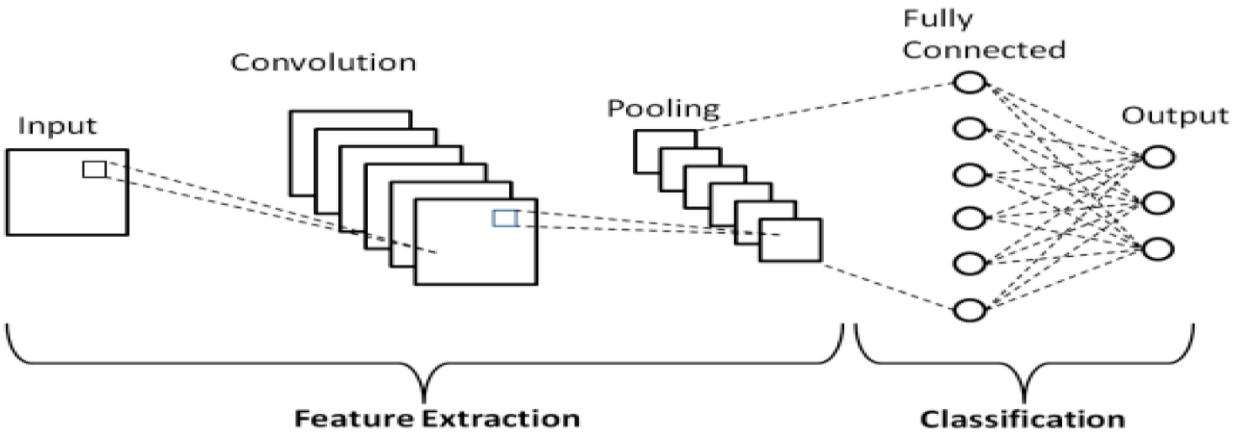

Figure 2.7: Architecture of Deep Convolution Neural Network

Nowadays, convolutional neural networks have led to a series of breakthroughs for image classification [24]. Many other visual recognition tasks have also greatly benefited from very deep models. Thus, over the years there is a trend to go *deeper*, to solve more complex tasks and also to increase or improve the classification or recognition accuracy. However, as we go deeper, the training of convolutional neural network becomes more difficult and also the accuracy starts saturating and then degrades [24]. Because, if we keep on stacking convolution layers in a linear fashion, we will find that not only the time and memory requirements of the network tend to increase but also the networks perform worse, because the problem of finding the optimal weights

becomes increasingly difficult as the depth of the network increases. Thus, to solve this kind of problems advanced new model is introduced into DCNNs which is called **ResNet**. ResNet is a short name for Residual Network. As the name of the network indicates, the new terminology that this network introduces is residual learning.

DCNN naturally integrate low/mid/high level features and classifiers in an end-to-end multilayer fashion, and the "levels" of features can be enriched by the number of stacked layers [24]. This case makes deep convolutional neural network training more difficult, time consuming, high cost and high resource utilized (i.e., RAM, GPU, storage, etc.). However, in residual learning, instead of trying to learn some features, we try to learn some residual [24]. Residual can be simply understood as subtraction of feature learned from input of that layer. ResNet does this using shortcut connections (directly connecting input of n^{th} layer to some $(n+x)^{th}$ layer). It has proved that training of ResNet is easier than training linear fashion of deep convolutional neural networks and also the problem of degrading accuracy is resolved. It also offers scalability of the network (number of parameters) by adjusting a widen factor (i.e., the channel of feature maps) and depth.

The Residual networks (ResNet) are deep neural networks that follow the basic idea of skipping blocks using the shortcut connections. The blocks follow two simple design structures:

- For the same output feature map size, the number of filters remains same.
- If the size of the feature map is halved, the number of filters is doubled.

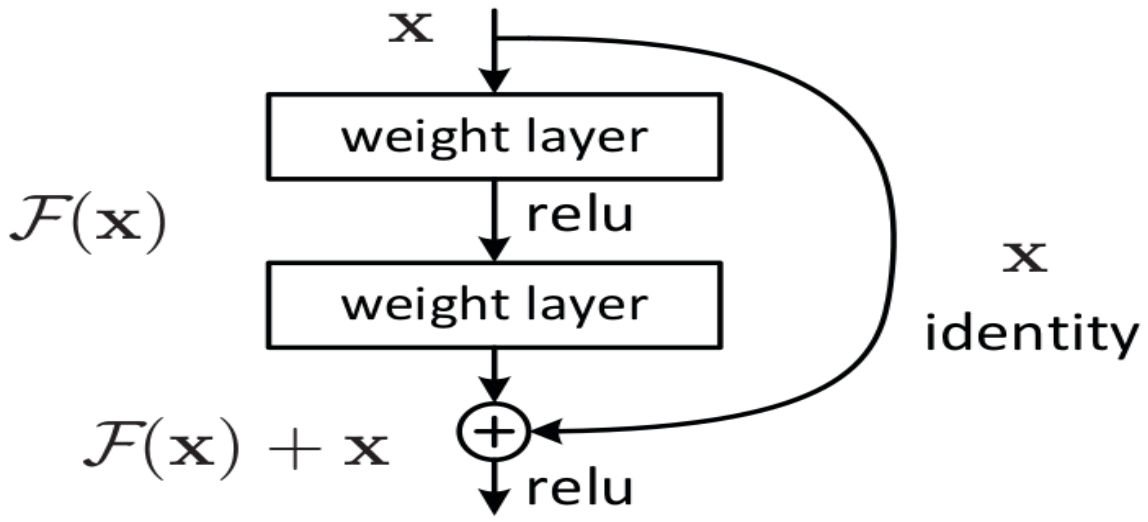

Figure 2.8: Architecture of ResNet

They explicitly let the layers fit a residual mapping and denoted that as H(x) and they let the nonlinear layers fit another mapping F(x)=H(x)−x so the original mapping becomes H(x)=F(x)+x as can be seen in Figure 2.8 [33] and the benefit of these shortcut identity mapping was that there were no additional parameters added to the model and also the computational time was kept in check. ResNet50 models can be used to achieve desired outcomes in areas with a deficient data via a process called transfer learning [33]. ResNet50 is a convolutional neural network which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer totally 50 layers deep.

Chapter Three: Related Work

3.1 Introduction

In this Chapter, different digital solutions related to locust identification and detection systems will be reviewed. For simplicity we summarized and grouped these related works based on their functionality and method used to develop the system into three groups as insect and pest segmentation, insect and pest detection and pest control in agriculture.

3.2 Insect and Pest Segmentation

Patil and Hegadi [25] demonstrated the segmentation of the cotton crop insects from an image. Before developing this new method of cotton insect's pest segmentation, the authors described that identifying pests and insects on cotton crops was a time consuming and laborious method. Thus, on the research work they proposed automatic segmentation based on edge detection. The segmentation of the cotton crop insects from the image was performed by using the sobel edge detection method. However, they did not show the results of the experiment in numbers or percentage to assess the accuracy of the proposed system. They simply list detected pests from the given input of cotton images as an experimental result of the research work and not figure out in number to show the accuracy.

Wang *et al*. [26] conducted cognitive vision method to improve the accuracy and stability of pest image segmentation. The main idea of this research work was to differentiate the pests from the crops' leaves using pest-intelligent diagnosis. In their pipelined procedures of intelligent diagnosis methods, image segmentation was applied to separate insects from the crops' leaves, which is the precondition for feature extraction and disease diagnosis. The authors method works in the following way: first, a pest image was divided into blocks via an image block processing method. Second, an adaptive learning algorithm was used to accurately select the initial cluster center. Third preliminary segmentation result were achieved using k-means clustering. Finally, three digital morphological features of ellipse were adopted to remove leaf veins. They conducted the experiment in an image of whiteflies on three kinds of leaves (i.e., corn, tomato, and pepper). To test the accuracy of the segmentation algorithm they calculated the misclassified rate by comparing a manually segmented image against the same automatically segmented image. The automatic

segmentation methods used for comparison included a fixed threshold method, the Otsu method, the traditional k-means cluster method. Based on the comparison the cognitive segmentation method used by the authors had a lower average misclassification rate for the different crops. The mean error rate was 0.0364 which was 20.6%,15.88% and 16.8% lower than that of fixed threshold method, Otsu method, the traditional K-means cluster method respectively.

3.3 Insect and Pest Detection

Gondal and Kha [27] developed an automatic approach for early pest detection in agricultural crops using image processing and computational intelligence. This research work uses erosion algorithm to enhance images and remove noise. Then, threshold technique was conducted to detect pests by separating background images of leaves from the pests. In addition, Support Vector Machine (SVM) was used for classification of images with and without pests based on the image features. To evaluate the pest detection accuracy, they used a total of 100 images (i.e., 50 images of leaves with whiteflies and 50 without whiteflies). Finally, the developed pest detection system's results showed an overall accuracy of 97%.

Bhadane *et al*. [28] also developed early pest identification in agricultural crops using image processing techniques. The authors developed a software prototype system for pest detection on the infected images of different leaves. Images of the infected leaves are captured by digital camera. Image pre-processing and image segmentation techniques were used to detect infected parts of the particular plants. Then the detected part was processed for further feature extraction which gives general idea about pests. Objects feature extraction was performed using background subtraction and threshold technique. Then image segmentation was used to partition the image into multiple regions. For these particular type of edge detection sobel operator has been used. The drawback of this system is that if the color of pest and leaf is almost similar then the background subtraction method cannot identify the pest. If so, the color change on the leaf is also identified as pest. In addition, the accuracy of pest identification depended on the selection of threshold values. However, they did not describe or figure out the accuracy of the proposed solution in number or percentage whereas they said it was three times faster and it covers three times more leaf surface when compared with the classical manual approaches.

Miranda *et al.* [29], developed an automatic pest detection and extraction system using image processing. The authors used image processing techniques to detect and extract insect pests by establishing an automated detection and extraction system for estimating pest densities in paddy fields. The authors' top priority was to find effective methods to reduce the level of infestation in the paddy fields. For this, the authors presented an automatic pest detection and extraction system and they used different image processing techniques to detect and extract the pests in the captured image. They used 'background modeling', to detect the presence of insect pests in the captured image, and a 'median filter' was used to remove the noise produced by different lighting conditions. To assess the performance of the proposed solution the authors tested it practically at Pampanga State Agricultural University farms using 4 wireless cameras in the paddy field for five consecutive days and the accuracy was 89%.

3.4 Pest Control in Agriculture

Krishnan and Jabert [30] developed automatic pest control in agricultural plantations using image processing. In this paper they developed an algorithm for easily identifying the pest infected areas of crops. The algorithm can be further modified to find the diseased areas in the crops using sophisticated software and better image acquisition devices. The authors' main objective was to detect diseases that affect crops like coffee berries on plantations and other bioagressors (e.g., aphids) or other plant diseases. In the proposed method to detect the plant diseases automatically, pre-processing was added before the segmentation, volume estimation was added after the segmentation and an image subtraction step was added after the algorithm. Therefore, the proposed method reduces the problems due to noises and other irregularities in acquired images and also it overcomes the disadvantages like intensity in homogeneity and artefacts. To detect biological objects on a complex background, they combined scanner image acquisition, sampling optimization and advanced cognitive vision. Segmented insect pest screenshot images were shown on the research study as experimental result.

Babu and Rao [31] developed leaves pest and disease recognition using Neural Network model. This software model is developed to suggest remedial measures for pest or disease management in agricultural crops. Using this software, the user can scan an infected leaf to identify the pest or disease that affected it and can obtain solutions to control it. The authors used prewitt edge

detection algorithm to develop the system. Prewitt edge detection produces an image where higher grey level values indicate the presence of an edge between two objects. Back propagation Neural Network is used for classification. The training set contains minimum five species for each type of leaves in each data file. Using more number of species in training set and number of output nodes can enhance the recognition ability. Recognized pests of the leaf screenshot image was shown on the research study as an experimental result.

Both these research studies did not describe or mention the accuracy or the performance result of their system. Plus to that they are also dependent to background color of crop leaves and affects all kinds of insects. Therefore, they are not applicable for locust identification.

3.5 Summary

To summarize the techniques applied in these related works cannot be directly applied for identification and detection of locusts because all these related works are dependent on the background color of crop leaves, threshold value and affect any kind of insects. The main objective of these related works was to segment or detect any object or insect that exists on given crop leaves. Due to this fact we cannot directly apply these systems for locust identification and detection. Table 3.1 shows the summary of related works. Thus, in this research work we will add some additional basic features of locusts on the existing related works by identifying locusts from another non-locust objects and detect the exact location of locust in a given image.

Table 3.1: Summary of Related Works

No	Research Title	Algorithm	Authors	Limitation
1	Automatic Segmentation Cotton Insects and Pests	Sobel edge detection method	Roopa.Patil, Ravindra S. Hegadi[25]	Dependent on cotton crop background color.
2	Pest Control in Agricultural Plantations Using Image Processing	Differential clustering	Zhibin Wang, Kaiyi Wang, Zhongqiang Liu, Xiaofeng Wang, and Shouhui Pan [26]	Only adds pre-processing before segmentation
3	Cognitive Vision Method for Insect Pest Image Segmentation.	adaptive learning, k-means clustering and three digital morphological features of ellipse	Muhammad Danish Gondal and Yasir Niaz Kha [27]	Background color dependent
4	Early Pest Detection from Crop Using Image Processing and Computational Intelligence	Threshold technique and Support Vector Machine (SVM)	Ganesh Bhadane, Sapana Sharma, and Vijay B. Nerkar, [28]	Background color dependent and Threshold value dependent
5	Early Pest Identification in Agricultural Crops using Image Processing Techniques	Threshold technique	Johnny L. Miranda, Bobby D. Gerardo, and Bartolome T. Tanguilig [29]	Background color and Threshold value dependent
6	Pest detection and extraction system using image processing	background modeling and median filter	Murali Krishnan and Jabert. G [30],	Background Color Dependent
7	Leaves Recognition Using Neural Network	Prewitt edge detection and Back propagation Neural Network	Prasad Babu and B.Srinivasa Rao[31]	Back Propagation Wight value dependent

Chapter Four: The Proposed Locust Identification System

4.1 Introduction

In this Chapter, we are presenting the design and architecture of the proposed system. The different components of the proposed system architecture are also described along with relevant techniques and algorithms. In general, the proposed system passes through many processes such as image acquisition, preprocessing, segmentation, training, identification, and classification of locusts.

To identify locusts from another insect we have to extract some unique feature from the shape or anatomy of the locust using different digital image processing techniques. After we have identified the locusts from non-locust objects, the system draws rectangular bounding box on the surrounding of identified locust objects in the given test image. To do this, in our research work we used the deep neural network model.

Thus, in this chapter, we will explain the design and system architecture of locust identification using deep neural network model.

4.2 The Proposed System Architecture

The overall concept and framework of any vision related algorithm for image identification, classification and recognition is almost the same. It can be started from acquiring digital images from the environment using a digital camera or another method. Then image pre-processing techniques are applied to enhance and improve the quality of the acquired images. Next, the useful features can be extracted for further analysis. Finally, several analytical techniques are applied to identify, classify, understand and recognize the images according to the specific problem.

Therefore, the image acquisition process is the initial input to the proposed system architecture and it is responsible for acquiring different locust images. After the locust image has been obtained, various methods of processing can be applied to the image dataset to develop the proposed system. Accordingly, it is necessary to capture and collect different locust input images from different sources, i.e., from the web. Furthermore, we divide the acquired locust image dataset into the following three parts:

a) Training Dataset

A locust training dataset is a sample of data used to implement and build up a locust identifier model. It is a set of locust examples used to fit the parameters (e.g., weights of connections between neurons in the deep neural networks) of the model. The model is trained on the training dataset using a supervised learning method. The use of representative training datasets is of significant importance for the performance of all identification and classification methods. In this proposed system we are collecting a total of 412 image dataset for training and testing purposes. Among the collected image datasets 80% of the images (i.e., 330 images) are used for training while the remaining 20% (i.e., 82 images) are used for validation and test datasets; both sets of images are chosen randomly.

b) Validation dataset

A validation dataset is a sample of data held back during the training of our proposed model that is used to give an estimate of model accuracy while tuning the model's hyper-parameters. The validation dataset is different from the test dataset that is also held back from the training of the model. Actually, it can be regarded as a part of training set, because it is used to build the proposed system's deep neural network model.

c) Test dataset

A test dataset is a dataset that is independent of the training dataset, but that follows the same probability distribution as the training dataset. Test dataset is used for performance evaluation of the proposed solution. Therefore, in our case it is a collection of some examples or samples of locust and non-locust images used only to assess the performance of a proposed system model.

As a result of the division of the acquired locust image, the proposed system architecture also has three phases. The first phase is the training phase. In this phase, sample training locust images are already collected from different sources. Then, these training locust images dataset are pre-processed and segmented using an image processing technique. After pre-processing and segmenting the dataset, useful image features are extracted using different feature extraction

techniques (i.e., SURF feature extraction) and then a deep neural network model is used to train the proposed solution using the extracted features to create the knowledge base.

The second phase is the validation phase. In this phase, sample random locust validation dataset is chosen from the acquired locust image dataset and the same procedures with the training phase are applied to validate the proposed system during training.

The third phase of the proposed system architecture is test phase. To test the developed system, the test datasets can be chosen randomly from the overall locust dataset or from any web sources. Then pre-processing is done on the collected test dataset using different image processing techniques. In the last step, useful features are extracted from the test data using different extraction models. They are then compared, or matched, with the previously acquired knowledge based on the training features.

Thus, in our proposed system to identify locust objects from non-locust objects and annotate the bounding box on the identified locust, we train, validate, and test the locust image dataset using the proposed system architecture as shown in Figure 4.1. In the proposed system architecture, the three phases i.e., train, validation and test phases are represented in three different colors (rose, blue and green respectively). There are some components which are represented in yellow – these are our contributions in the proposed system architectures.

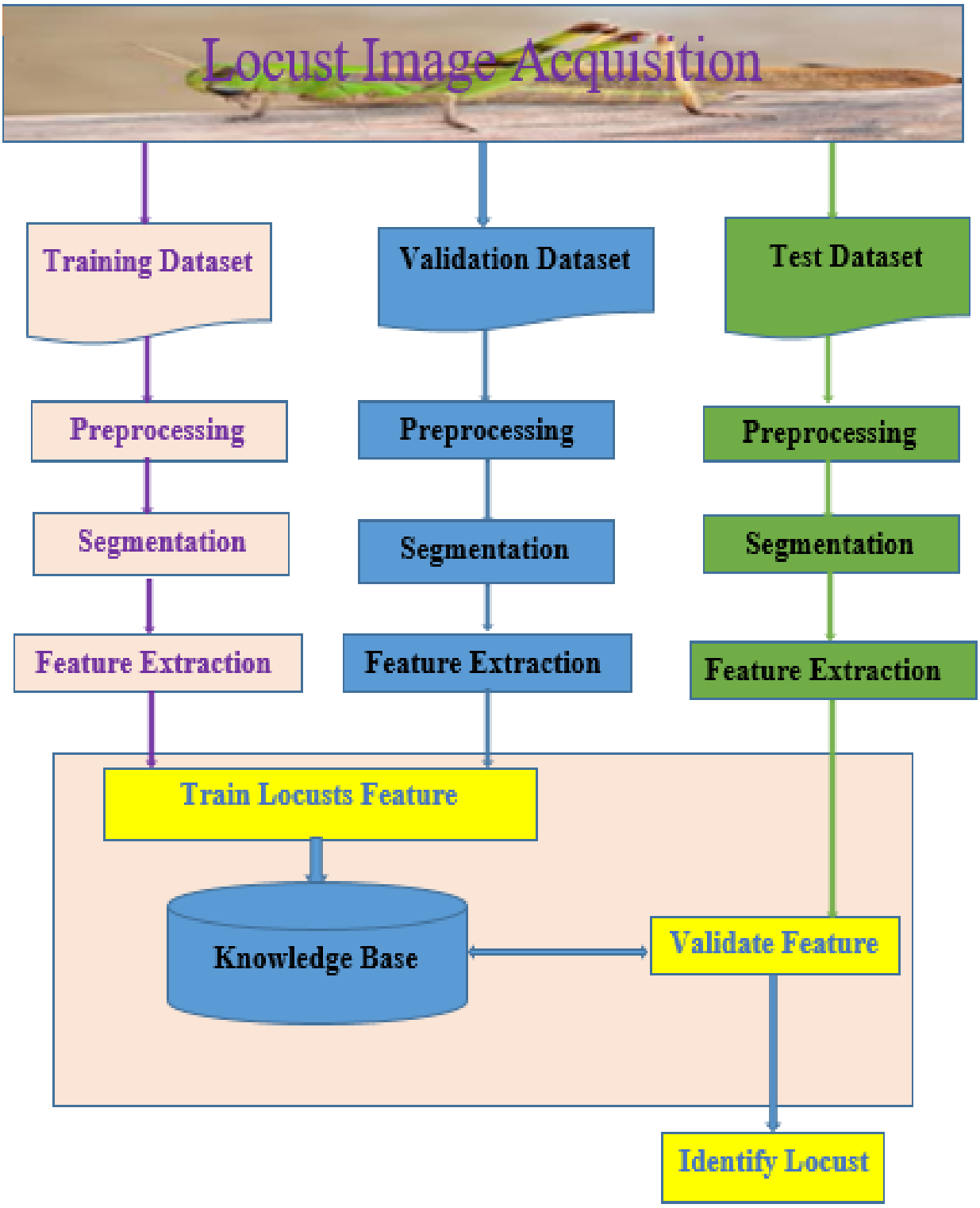

Figure 4.1: The Proposed System Architecture

4.3 Components of the Proposed Architecture

4.3.1 Pre-processing

Once an image has been acquired, the next step is pre-processing. This step is required to improve the locust image dataset by suppressing unnecessary distortions or enhancing major image features that could be important for further recognition processes. We can pre-process both locust training datasets as well as test datasets to improve, enhance and adjust the appearance of our dataset. The main goal of preprocessing is to identify the locust in an image and discard all other information by considering the shape and texture of locusts. As part of image processing, we resized the original locust image sizes, converted RGB image to gray scale and removed noise from the images.

Image Resize: Training large size locust image datasets is computationally expensive and takes a long time. In addition to that, the training image datasets may contain different image sizes, appearance, quality and formats which makes it difficult to train the proposed model. Therefore, it is mandatory to pre-process the input image based on our training model's requirements. Based on this, locust images with different sizes can all be resized to meet the expected size requirements. To avoid one by one re-saving of all images to this format, we use the *augmentedImageDatastore* MATLAB function. This function transforms batches of training, validation, and test data, with optional preprocessing such as resizing, rotation, and reflection. To do this we give the path of the image dataset with respect to the required parameter to the augmented function then the function compute and give an output based on the received parameters.

Convert RGB Image to Gray Scale: grayscale images are entirely sufficient for many tasks. This means there is no need to use color images which are complicated and hard to process. Algorithm 4.1 shows how to convert color image into gray scale image. Thus, based on the algorithm we used *rgb2gray* MATLAB function to convert RGB locust images to gray scale images after conversion we easily extract texture features from the gray scale image. Equation 4 [32] shows how the rgb2gray function is computed.

grayscale = 0.299*R + 0.587*G + 0.114*B (4)

where R is red, G is green, and B is blue the pixel value of the gray image can be represented in 8 bit and the range values could be from 0 up to 255.

1. *Input: RGB locust image*
2. *Output: Grayscale locust image*

I = RGB image;

Im = imread(I);

Gm = rgb2gray(Im);

Algorithm 4.1: Convert RGB Image to Gray Scale Image Algorithm

Noise Removal: We also remove noise from the images to improve the quality of the image during pre-processing stage and we applied the classical spatial filters based on pixel operation algorithm to remove the noise of the gray image [32]. The key of the effect of noise removal lies in the mathematical model of the applied spatial filter. Let gn(i,j) be the gray noisy image and g(i,j) be the gray image of noise removed. Sij be a rectangular image window with the size of $m \times n$ and the center pixel of (i,j). Bothe Equation 5 and 6 [32] shows how the maximum filter value and minimum filter value can be computed respectively. According to the principle of algorithm 4.2 [32] we found both noise and de-noised locust images as shown on Figure 4.2.

$$g(i,j) = \max_{(s,t)\in S_{ij}} \{g_n(s,t)\} \quad (5)$$

$$g(i,j) = \min_{(s,t)\in S_{ij}} \{g_n(s,t)\} \quad (6)$$

Algorithm g = gray_NOISE_REMOVAL_PIXEL (gn)

Input: gray noisy image gn(i,j) (i=1,2,…, N, j=1,2,…, M)

Output: gray image after removing the noise g(i,j) (i=1,2,…, N, j=1,2,…, M)

Step 1: Input gray noisy image.

Step 2: Go to Step 8, if the pixels are all fit to neighboring pixels.

Step 3: Read the gray level gn(i,j) of the pixel (i,j) (i=1,2,…, N, j=1,2,…, M) in a set order.

Step 4: Go to Step 7 if gn(i,j) isn't the maximum or minimum filter value.

Step 5: Count the number c of the maximum or minimum value in the neighborhood of (i,j).

Step 6: Replace the pixel value gn (i,j) with output g(i,j) of the spatial filter if c≥TH, then go to Step 2.

Step 7: Keep gn(i,j) unchanged, then go to Step 2.

Step 8: Output the result image.

Algorithm 4.2: Gray Image Noise Removal Algorithm Based on Pixel Operation

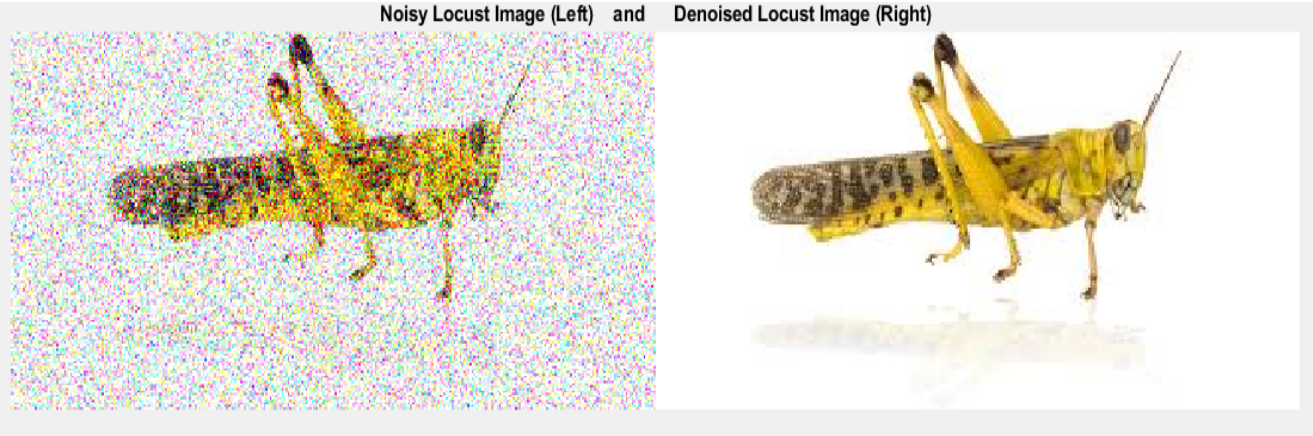

Figure 4.2: Noise Removal of Locust Image

4.3.2 Segmentation

In this section locust images are partitioned into segments or regions for easy analysis and extraction of useful information. Accordingly, we divide the locust image into different important segments or regions to process and easily extract meaningful information from the locust images.

Basically, we segment the locust dataset for the purpose of:

- Easily finding the shapes of a locusts based on the anatomy of locusts
- Easily finding all body parts of a locust by considering the lifecycle of locusts

Therefore, in the proposed system we use different methods to segment locust image data, i.e., we use Threshold method to segment locusts image. This could be performed by converting the locust image into a binary image by replacing all pixels in the input image with luminance greater than the threshold value with a value 1 (white) and replacing all other pixels with a value 0 (black). The binary image should contain all of the essential information about the position and shape of the objects of interest. The selection of the optimum threshold value can be performed manually based on prior knowledge or automatically based the information of image features. Figure 4.3 shows locust image segmentation using the threshold method. Algorithm 4.3 shows how threshold segmentation computed.

```
Input: Preprocessed Locust Image
Output: Segmented Locust Image
Step 1: Load Locust Input Image L(i,j)
Step 2: Obtain Image Histograms H(i,j)  (Distribution of pixels)
Step3: Get threshold value T manually
    For each pixel location H in L
      If Saturation > T
         L(i,j) = white (0);
          Else
        L(i,j) = Black (1);
       End
    End
Return L(i,j);
```

Algorithm 4.3: Threshold Algorithm to Segment Locust Image

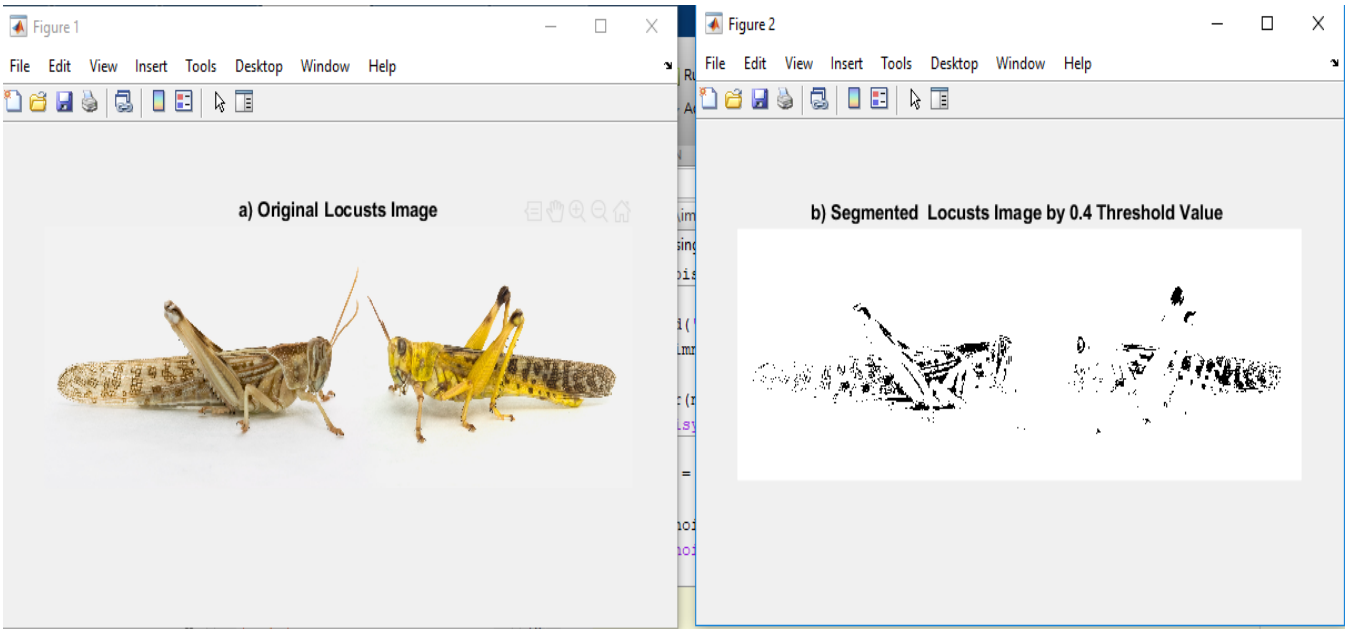

Figure 4.3: Locust Image Segmentation using Threshold Methods

The segmented locust image is highly dependent on the threshold values. Therefore, when we change the threshold values it is reflected in the resulting segmented image. For instance, Figure 4.4 shows the incremented threshold values from 0.4 to 0.7.

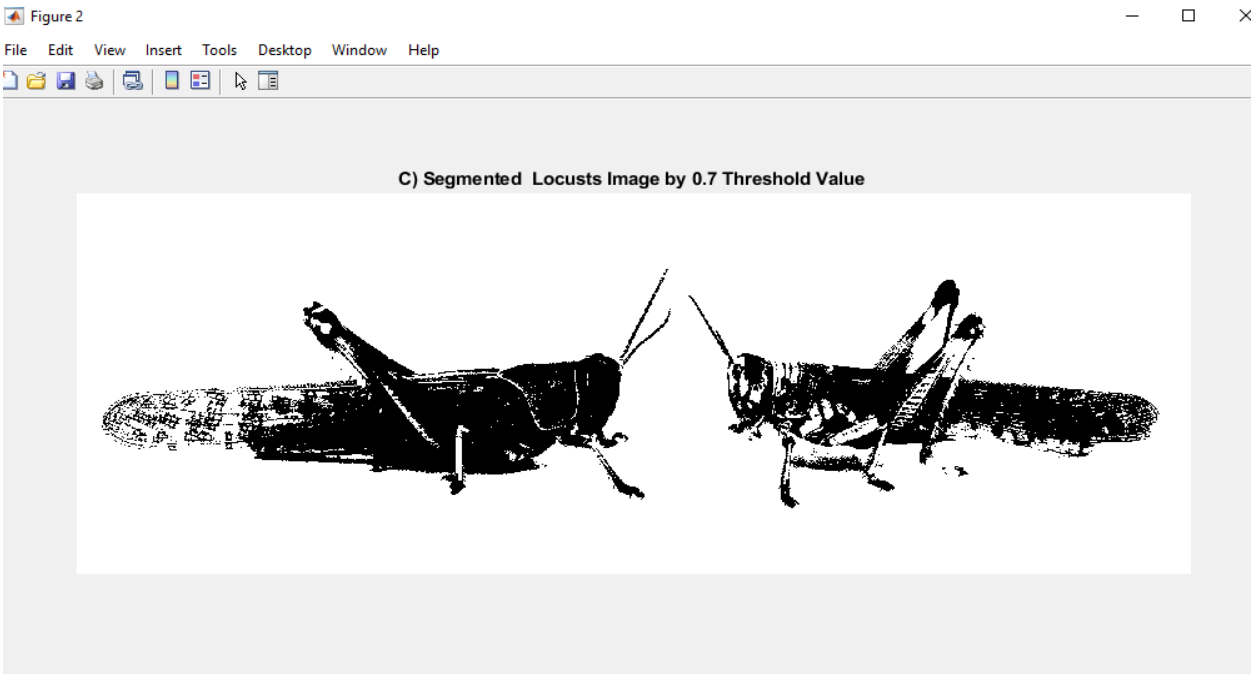

Figure 4.4: Segmented Locust Image by 0.7 Threshold Value

4.3.3 Feature Extraction

Feature extraction is a main part of locust identification and detection system development. It is a process of extracting meaningful information from the locust image. To identify and detect locusts, we must use different techniques to extract the basic unique features of a locust. Thus, the most important feature extraction techniques we used to identify locusts are shape and texture feature which found in the training locust images. Some of the basic component of a locust anatomies which make it unique are: -

- ✓ having special abdomen,
- ✓ upper lip,
- ✓ a pair of jointed antennae or feelers,
- ✓ a pair of large compound eyes,
- ✓ a pair of hard black serrated jaws,
- ✓ a sheath covering the top and sides of the front part of the thorax,
- ✓ three pairs of legs (and each leg has three main parts such as the femur, tibia and the tarsus or foot),
- ✓ and two pairs of wings (harder front wings cover and protect the softer hind wings which are folded fan-wise).

However, the training locust images have distinctive ranges of image background views, shapes, and sizes. Therefore, it is difficult to extract common feature to all types of locusts in the given locust dataset using linear feature extraction function to identify all kind of locust object from non-locust objects. Because extracting common features using such linear or simple feature extraction functions is too complex due to varying backgrounds, sizes, patterns and shape of locust image in different areas and development stages. Therefore, to deal with such complexities in identification needs deeper analysis and a larger amount of different training data [33]. That's why in our proposed system we used deep neural network model.

Basically, Figure 4.5 shows the extracted basic features of locust image using Speed-up Robust Features(SURF) extraction function. Because SURF is a scale and in-plane rotation invariant feature. It contains interest point detector and descriptor. The detector locates the interest points in the image, and the descriptor describes the features of the interest points and constructs the feature

vectors of the interest points [34]. SURF relies on the determinant of the Hessian matrix to compute the interest points. Algorithm 4.4 shows how to extract unique features of a locust.

Input: Segmented locust image dataset

Output: Locust image with strong selected features

Step 1: Load segmented locust images

Step 2: get image pixel I(x,y)

Step 3: Compute the Hessian Matrix of pixel I(x,y) = H(I(x,y))

Step 4: Calculate the determinant of Hessian Matrix H(I(x,y))

Step 5: Select top determinant values

Step 6: Choose the interest point from the determinant values.

Step 7: Save the interest point and end

Algorithm 4.4: Algorithm of Locust Feature Extraction.

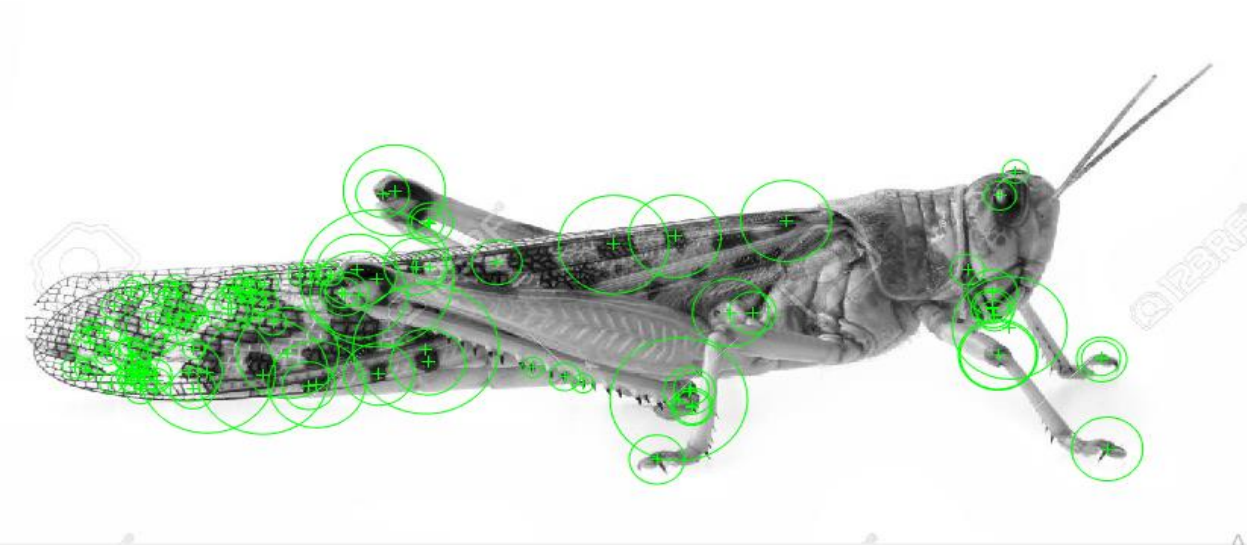

Figure 4.5: Top Strongest Extracted Locust Features Using SURF Function

4.3.4 Train Locust Feature

After we have extracted the unique feature of the locust using the feature extraction component part then we must train our proposed system to build knowledge base. Thus, Train sample locust feature component is responsible to build knowledge base for many different locust types by training the design model using the collected training dataset and validation dataset as a sample of the locust.

4.3.5 Validate Feature

Already we have built the knowledge base using the training sample locust features component. This implies that, after this the developed proposed model already has now how about what is a locust. Accordingly, to test the proposed system the validate feature component is responsible to cross check or validate the test image feature with knowledge base feature to decide whether the test image feature having similarity or not. Figure 4.6 shows matched extracted feature of two different locust images. Algorithm 4.5 shows how to cross checks or matches between features of test image across with the feature of the train image.

Input: Test Image
Output: Putatively Matched Feature of Train and Test image
Load test data
Identify training interest points
Extract training interest points descriptors
Initialize match objects
Read test image
Identify test interest points
Extract test interest point descriptors
Match Query points with training points
If (Match points > threshold)
Compute homograph transform line
end if

Algorithm 4.5 Validate Test Feature with Training Feature

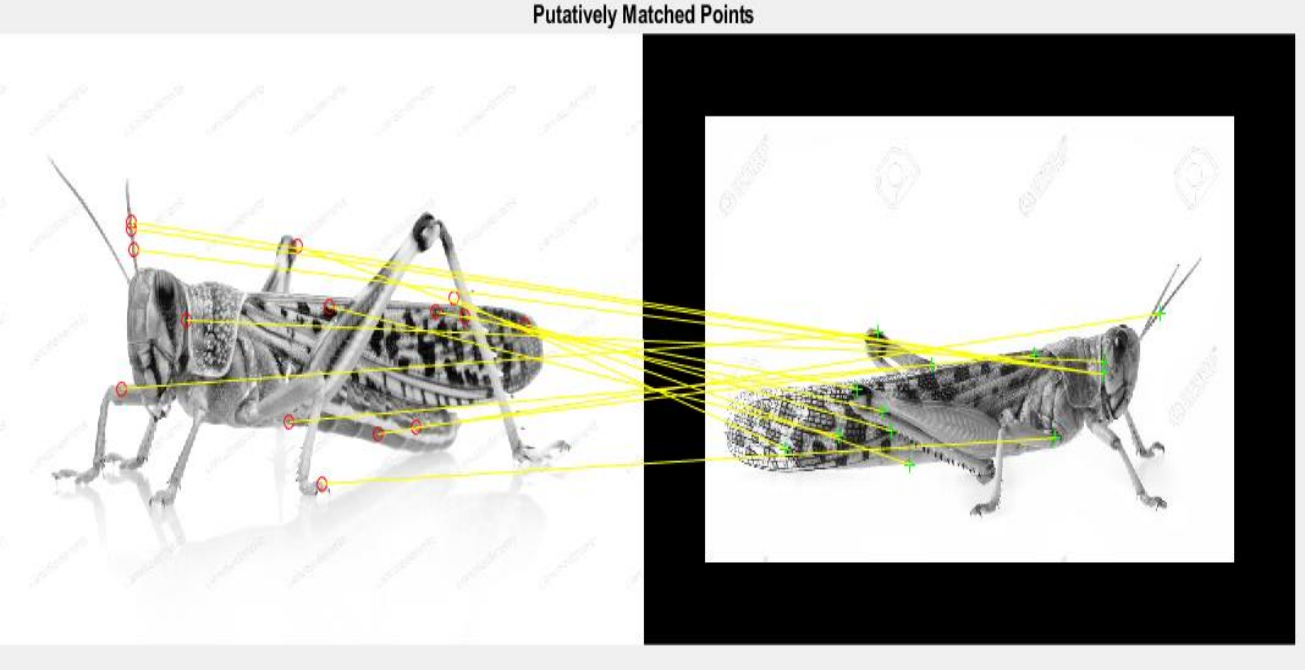

Figure 4.6: Matched Test and Train Locust Image Features

4.3.6 Identify Locust

Locust identification is the basic task and the last result of the proposed system. We first did the locust identification part which is responsible for the identification or classification by analyzing locust objects from another non-locust objects based on the acquired knowledge. Next, the proposed system draws rectangular bounding box on the identified locust to show the locust object in the input image. Algorithm 4.6 shows how the proposed system identifies a locust and draws a bounding box in the given image.

Input: Test and Train Feature

Output: Identified locust images.

Step 1: Compute convolutional neural network layer activations

Step 2: Fit multiclass models for support vector machines or other classifiers

Step 3: Predicts the output of an identified locust

Step 4: Identify locust within an image based on prediction value

Step 5: Select maximum score of the identified value

Step 6: A*nnotate with rectangular shape and label at the identified locust and end.*

Algorithm 4.6: Locust Identification Algorithm

4.4 Summary

In this Chapter we developed specific architectural design which helps to automatically identify locust objects from another non-locust object using a digital technology. Then, we described each component of the proposed architecture with the respective algorithms in detail. The proposed system can accept locust image dataset as input and divide these acquired locust image datasets into three phases. Lastly, these three phases pass through different image analytical processes (i.e., preprocessing, segmenting, feature extracting, and feature validating) to meet the desired goal of the proposed system.

Chapter Five: Experimentation and Evaluation

5.1 Development Tools

We used some tools to develop a prototype for a locust identification system. The basic goal and functionality of the developed prototype of the proposed system is identifying locusts from non-locust objects. After locust objects have been identified, the system draws labeled rectangular bounding box on the identified locust in the given test image. For this purpose, we used the MATLAB software for the proposed system prototype development. MATLAB stands for "matrix laboratory". It is a multi-paradigm numerical computing environment and proprietary programming language developed by Math Works. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages. It also provides a comprehensive set of reference-standard algorithms and workflow applications for image processing, analysis, visualization, and algorithm development. In this development of a prototype for the proposed system we used MATLAB version R2018b software**.**

Basically, to identify or classify locusts from non-locust objects we used the *ResNet50* model which is a deeper model of the deep conventional neural network. Moreover, the specification of the computer on which the system is implemented is Intel Core i5, Windows 10 laptop computer with 4GB RAM and 2.60 GHz processor.

5.2 Dataset Collection

For training, evaluating, and testing of the developed proposed prototype system, a total of 412 different types of locusts and non-locust images were collected from the web. During the data collection we tried to include all kinds of locusts by considering classifications of locusts, life cycle of locusts, and background or views of the locusts in a given image. Thus, to train and test the proposed system we collected images that were different in shape, background, number, appearance, and size. As it is described in Section 4.3, we classified the collected datasets into three parts: training dataset, validation dataset and test dataset. From the total image dataset 80% of the image dataset were used for training (i.e., 330 images in total) and 20% of the image dataset were used for validation and testing (i.e., 82 images).

5.3 System Prototype

Based on the designed model, the prototype system is developed to show the performance of the proposed solution model. Thus, we developed a prototype as shown in Figure 5.1 to demonstrate and test the proposed locust identification system model and to see if it is capable of performing the identification process based on the designed and trained model. The sample code, and its implementation for the proposed system, has been presented in Annex B.

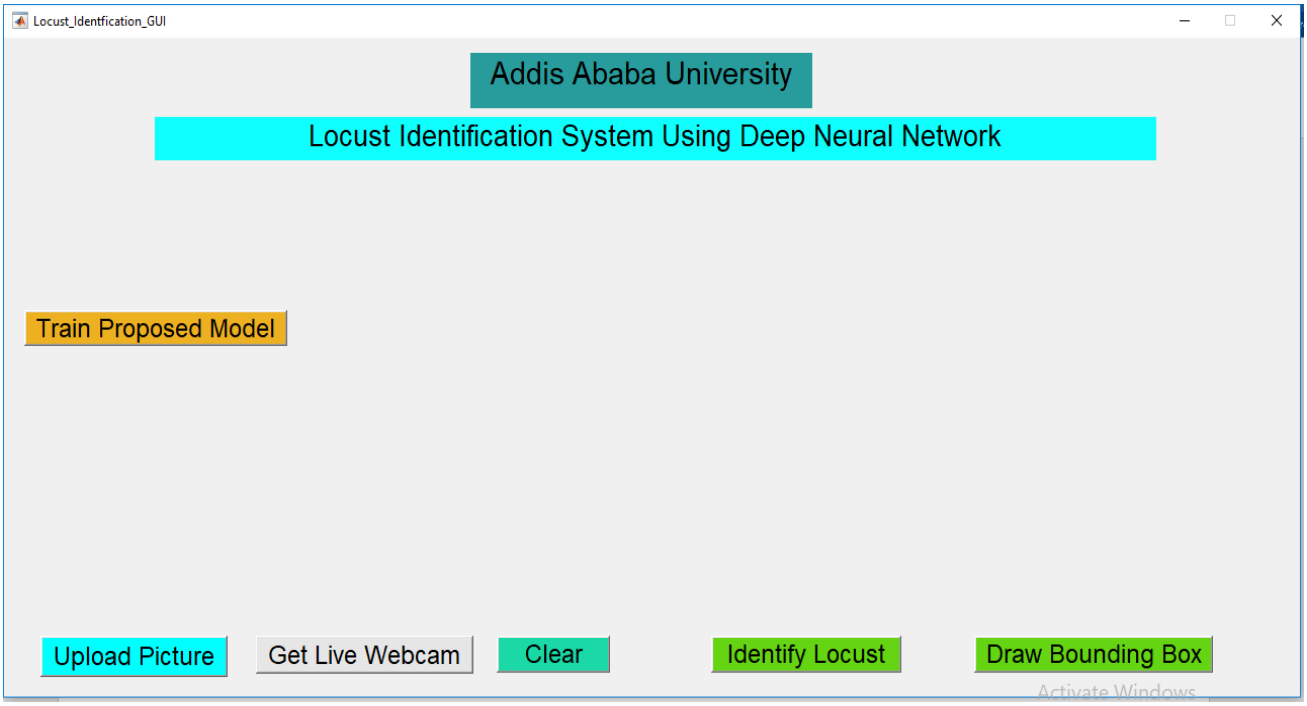

Figure 5.1: User Interface of the Proposed System Prototype

5.4 Experimental Results

We evaluated the performance of the proposed model based on our prepared dataset. We ran our experiments on collected locust and non-locust image datasets. We used deep neural network model, i.e., ResNet50 model to identify and classify locusts from non-locust objects. When performing the locust image identification or classification, we first pre-processed, segmented, and extracted locust features from the training dataset and trained the proposed model using the "Train Proposed Model" button.

Once the training is done, first click on the “Upload Picture” or “Get Live Webcam” buttons to load the test data. Next, click on the “Identify Locust” button to find out if the system has found a locust in the loaded test data. Then we get “Input Image Contains Locust Object” labeled image if the loaded test data contains locust object or “No locust on the input image” labeled image if the loaded test data is non-locust object based on the prediction knowledge of the proposed system. Figure 5.2 shows the properly identified and labeled locust image after clicking the “Identify Locust” button and Figures 5.3 and 5.5 shows us the result of non-locust test data. On the other hand, locust recognition is not only telling us what is in the image but also showing us where in the input image the locust object is found via drawing bounding box *(x, y)*-coordinates. Figure 5.4 shows the properly detected locust images after clicking the “Draw Bounding Box” button.

In general, performance of the proposed system was observed that, ResNet50 model has tremendous performance on identifying and classifying locust from non-locust object. In fact, as our experiment results show the developed proposed system’s performance was 95.2% accurate as shown on Figure 5.7.

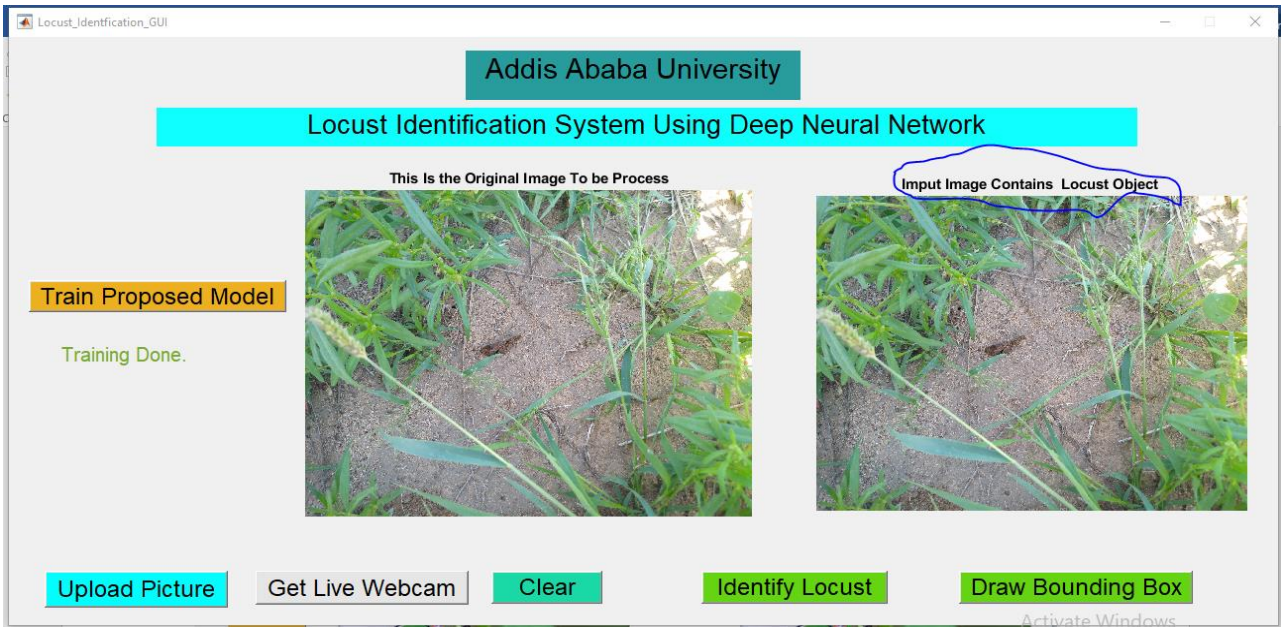

Figure 5.2: Check to Identify Positive Locust Test Image

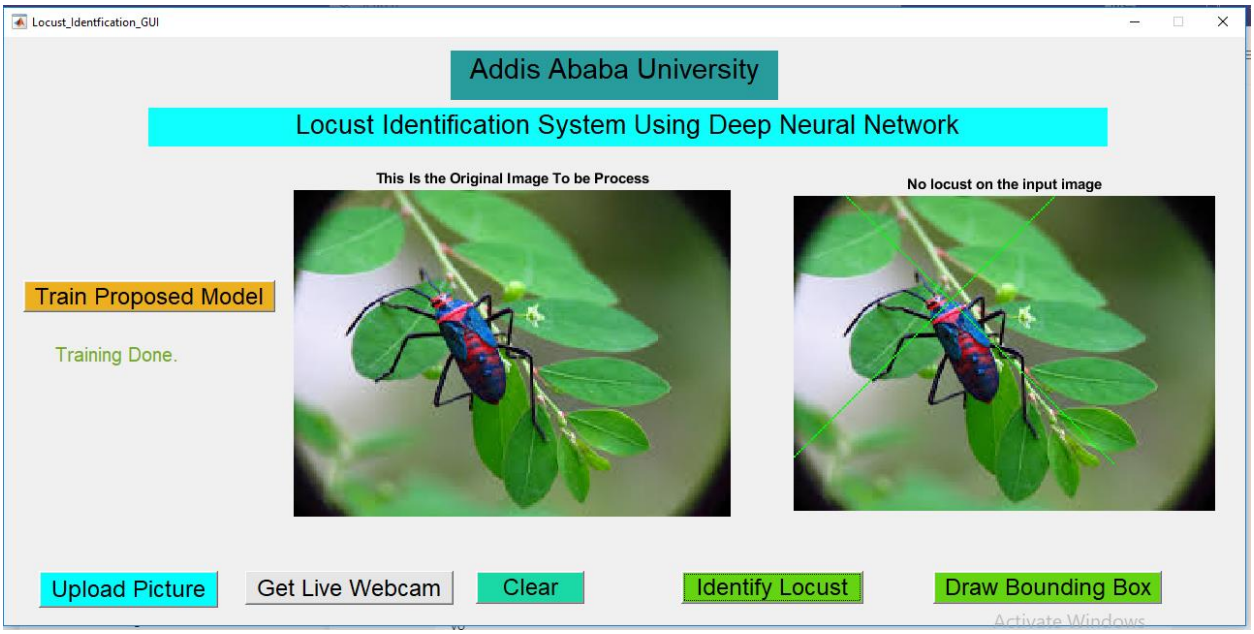

Figure 5.3: Check Other Non-Locust Image

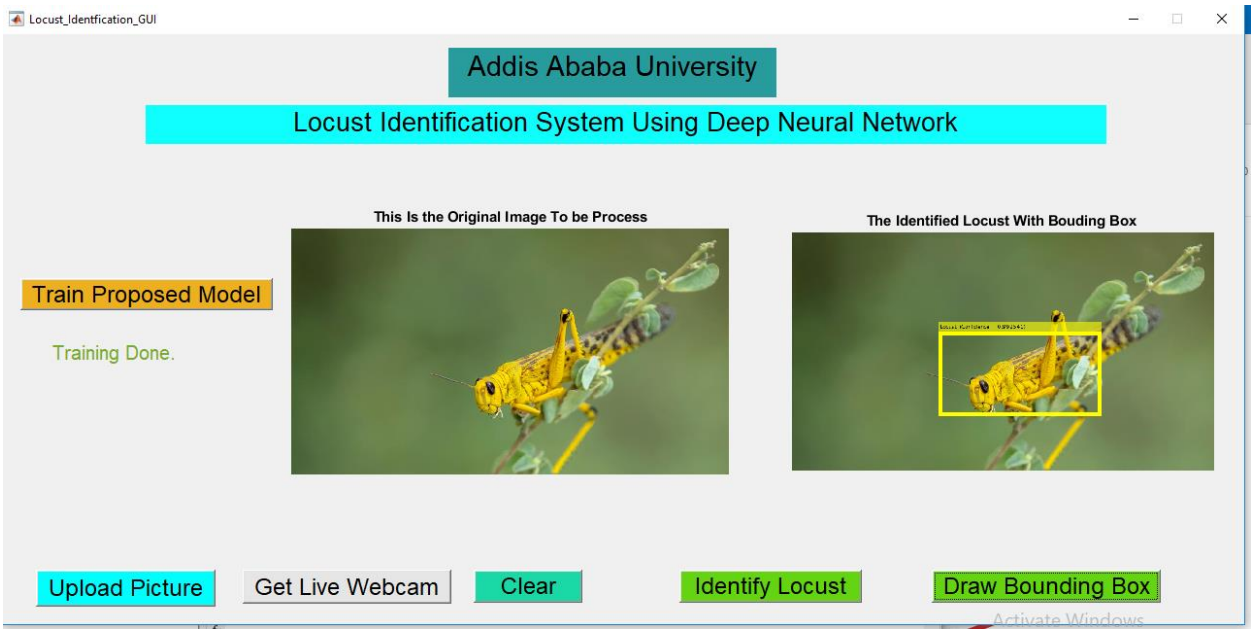

Figure 5.4: Identified Locust Object with Bounding Box

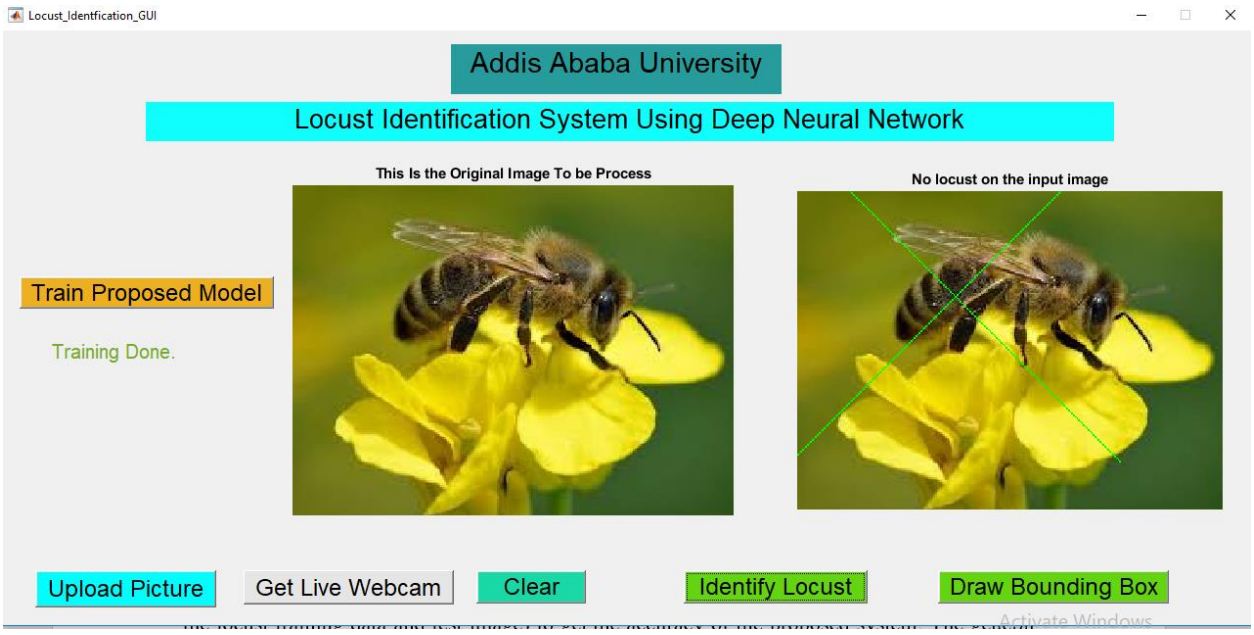

Figure 5.5: Sample Image Which Contains Bee

5.5 Accuracy Assessment of the Proposed System

After the training of the proposed system was accomplished, we were able to measure the proposed system network model's accuracy by comparing the target class and testing results.

Accordingly, we used the "confusionmat" MATLAB function that has two input arguments (i.e., the locust training data and test image) to get the accuracy of the proposed system. The general result of two input argument of a confusion matrices is represented as shown in Table 5.1. The recall, precision, and accuracy of a given two input argument confusion matrices also computed as shown on Figure 5.6 [35].

A Confusion Matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known and it allows the visualization of the performance of an algorithm. Even though the accuracy and performance of the proposed model depends on the number and the quality of the training data, we were still able to achieve an accuracy of 95.2% for our proposed system using our training dataset as shown on Figure 5.7.

Table 5.1: General Discerption of Proposed Solution Confusion Matrix

Total Training Image(TTI)	Predicted Yes	Predicted No
Actual Yes	True Positive (TP)	False Negative (FN)
Actual No	False Positive (FP)	True Negative (TN)

- There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a locust, for example, "yes" would mean they have the locust, and "no" would mean they don't have the locust.
- Out of 330 total locust and non-locust images, the proposed solution predicted "yes" 153 times, and "no" 161 times.
- In reality, 165 images in the sample have the locusts, and 165 images do not have locusts.

Let's now define the most basic terms in detail:

- **True Positives (TP):** These are images in which we predicted yes (they have the locust), and they do have the locust.
- **True Negatives (TN):** We predicted no, and they don't have the locust.
- **False Positives (FP):** We predicted yes, but they don't actually have the locust.
- **False Negatives (FN):** We predicted no, but they actually do have the locust.

We have added the row and column totals terms to the confusion matrix.

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Figure 5.6: Confusion Matrix with Advanced Classification Metrics

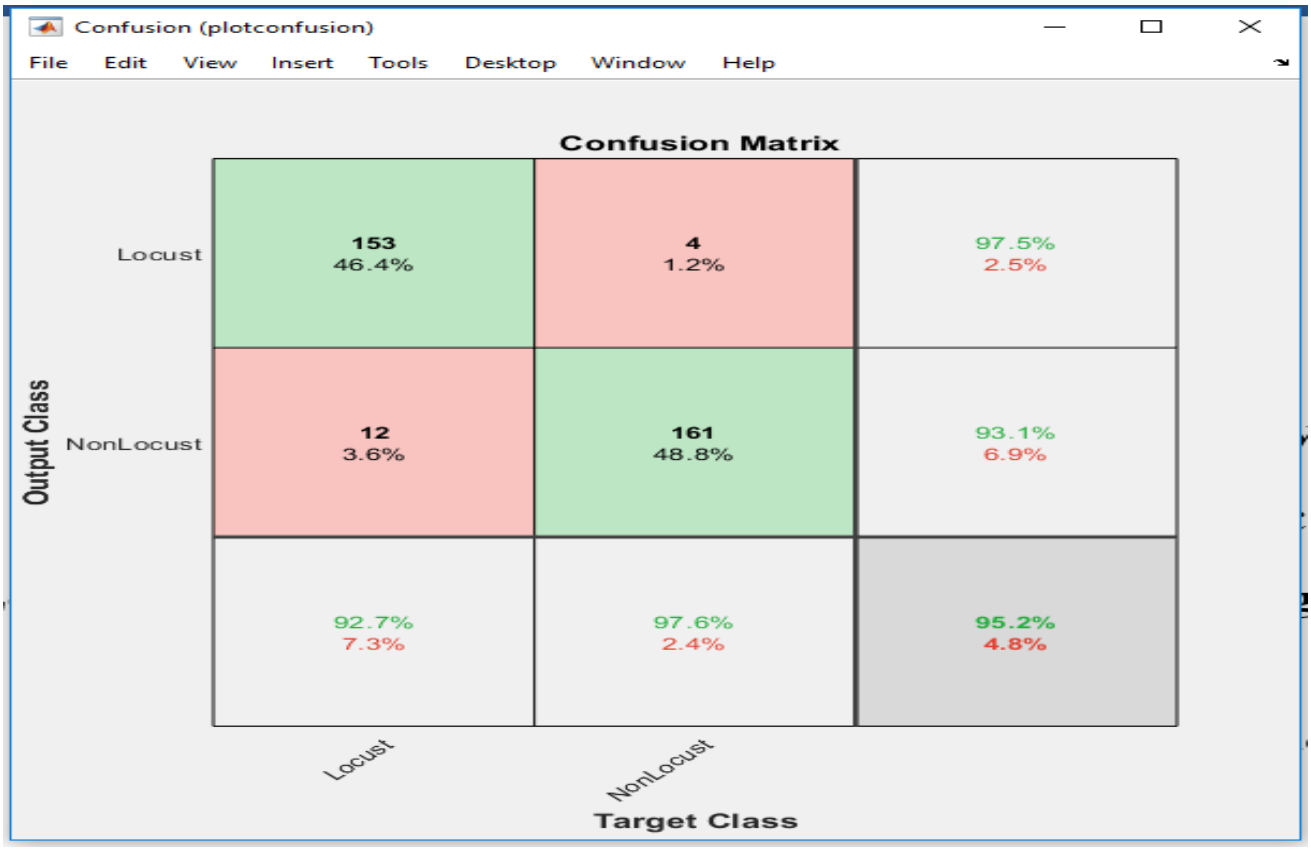

Figure 5.7: Confusion Matrix of the Proposed System

The proposed system result values shown on Figure 5.7 were computed as follow:

- **Accuracy:** Overall, how often is the classifier correct?
    - (TP+TN)/total = (153+161)/330 = 0.952 = 95.2 %
- **True Positive Rate:** When it is actually yes, how often does it predict yes?
    - TP/actual yes = 153/157 = 0.975 = 97.5 %
    - also known as "Sensitivity" or "Recall"
- **False Positive Rate:** When it is actually no, how often does it predict yes?
    - FP/actual no = 12/173 = 0.069 = 6.9 %
- **True Negative Rate:** When it is actually no, how often does it predict no?
    - TN/actual no = 161/173= 0.930 = 93.1 %
    - equivalent to 1 minus False Positive Rate
    - also known as "Specificity"
- **Precision:** When it predicts yes, how often is it correct?
    - TP/predicted yes = 153/165 = 0.927 = 92.7 %

5.6 Summary

In this Chapter we developed, tested, and evaluated the prototype of the proposed system using the collected locust image datasets. According to our evaluation, the performance of the proposed system prototype was assessed using precision, recall and accuracy result. Based on consecutive assessments we achieved a promising result of 92.7 %, 97.5 % and 95.2 % in precision, recall and accuracy respectively.

Chapter Six: Conclusion and Future Works

6.1 Conclusion

Locusts are overwhelmingly threatening large areas of pastures and crops in the countries of the Horn of Africa. This affects the lives of millions of people. In this thesis work, a literature review is done to understand the shape of locusts. Previous works related to locust identification are also reviewed. To alleviate the problems, using a locust identification system which supports the existing system on locust identification is proposed. This work contains an algorithm for extracting features of locusts and a design of the proposed system. We have collected sample locust image data, which is used for training and testing the proposed model.

Thus, in this research work an attempt has been made to develop a model for the identification of locusts. We have also presented a comparative evaluation of advanced deep neural network model (i.e., ResNet50 locust identification models) to support and facilitate the current available locust identification methods.

Identification of locust is performed mainly based on the extracted shape and texture feature of the training image. Thus, the performance of the proposed system shows that the overall success for the identification of locusts is 95.2 %. Moreover, the majority of the identification errors are attributed to the challenges faced by the quality of acquired image. Accordingly, as we can see from the proposed system prototype result on Figure 5.7, this study achieves promising results towards locust identification using deep neural network.

6.2 Contribution of This Work

The main contributions of this research work are given below.

- ✓ We developed a locust feature extraction algorithm as shown on Algorithm 4.4.
- ✓ We designed the proposed system architecture.
- ✓ We proposed the deep neural network model to identify locusts.
- ✓ We developed, tested, and evaluated proposed system prototype.

6.3 Future Works

Even though we have achieved encouraging results on the developed prototype, we believe that more additional functionality and performance improvement could be incorporated to come up with an even more effective and efficient locust identification method.

Hence, for the future we propose below points as a future works and advanced research direction:

- ✓ Improve the proposed system to use satellite image, GIS and Metrology data to forecast and predict the possibility of locust swarms before they occur and take proper pre-preventive and monitoring technique to manage the swarms at early stage.
- ✓ Improve the accuracy and performance of the proposed system by increasing the size and quality of the training data.
- ✓ The current study focuses only on locust identification within an image. Therefore, in the future studies can extended this work to include video processing to identify the speed and locust movement.
- ✓ Advance the locust identification system to include identification of locust egg from another non-locust egg because not all experts may have the same experience and knowledge about the egg of locust. Because, for non-expert person, there are other insect eggs that can be confused with locust eggs.

References

[1] C. J. Lomer, R. P. Bateman, D. L. Johnson, J. Langewald, and M. Thomas " Biological Control of Locusts and Grasshoppers,"Annual Review of Entomology, Vol. 46, 2001, pp .667–702.

[2] Long Zhang, Michel Lecoq, Alexandre Latchininsky, and David Hunter "Locust and Grasshopper Management ," Annual Review of Entomology, Vol.64, 2019, pp.15-34.

[3] Desert Locust Information Service, "Food Consumption of Locust," retrieved from http://www.fao.org/ag/locusts/oldsite/LOCFAQ.htm#q9, Last accessed on December 23, 2019.

[4] Anil Sharma, "Locust Control Management: Moving from Traditional to New Technologies," Empirical Analysis, Vol.4, 2014.

[5] University of Tartu," Digital Image processing," retrieved from https://sisu.ut.ee/imageprocessing/book/1, Last accessed on December 23, 2019.

[6] Srishtee Jain and Surendra Chadokar, "Object Detection in Image Processing," International Journal of Electrical Electronics and Computer Engineering,Vol.2, 2015, pp. 26-29.

[7] FAO, "Desert Locusts threaten agricultural production in Ethiopia," retrieved from: https://reliefweb.int/report/ethiopia/desert-locusts-threaten-agricultural-production-ethiopia , Last accessed on November 30, 2019.

[8] UN warning for Ethiopia, Kenya, Eritrea and Sudan, "Locust invasion," retrieved from: https://www.bbc.com/news/world-africa-50345204, Last accessed on November 30, 2019.

[9] P.M. Symmons and Keith Cressman," Biology and Behavior of Locust," FAO Agriculture Department, 2001.

[10] Encyclopaedia Britannica," Definition of Locust," retrieved from https://www.britannica.com/animal/locust-insect, Last accessed on May 29, 2020.

[11] Austrian Government Department of Agriculture “About locusts”, retrieved from https://www.agriculture.gov.au/pests-diseases-weeds/locusts/about/about_locusts, Last accessed on May 29,2020.

[12] Vikas Kumar Mishra, Shobhit Kumar, and Neeraj Shukla “Acquisition and Techniques to Perform Image Acquisition,” A Journal of Physical Sciences, Engineering & Technology, Vol. 09, 2017, pp.21-24.

[13] Arivazhagan, S., R. Newlin Shebiah, S. Ananthi, and S. Vishnu Varthini, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features." Agricultural Engineering International, Vol. 01, 2013, pp.211-217.

[14] V.Padmanabha Reddy and S.China Venkateswarlu, “Digital Image Processing,” retrieved from, https://www.iare.ac.in/sites/default/files/lecture_notes/DIP-LECTURE_NOTES.pdf, Last accessed on June 03, 2020.

[15] Dilpreet Kaur and Yadwinder Kaur, “Various Image Segmentation Techniques,” International Journal of Computer Science and Mobile Computing,Vol.3 Issue.5, May- 2014, pp. 809-814.

[16] Y. J. Zhang, “An Overview of Image and Video Segmentation in the last 40 years,” Proceedings of the 6th International Symposium on Signal Processing and Its Applications, pp. 144-151, 2001.

[17] S. Saleh, N. V. Kalyankar and S. Khamitkar, “Image segmentation by using edge detection,” (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 03, 2010, pp. 804-807.

[18] M. Yambal and H. Gupta, “Image Segmentation using Fuzzy C Means Clustering: A survey,” International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 7, July 2013.

[19] Ryszard S. Choras,” Image Feature Extraction Techniques and Their Applications for CBIR and Biometrics Systems,” Vol. 1, Issue 1, 2007,pp.209-2014.

[20] E. Saber and A.M. Tekalp," Integration of color, edge and texture features for automatic region-based image annotation and retrieval," Electronic Imaging, Vol.7, 1998, pp. 684–700.

[21] Jianhua Liu and Yanling Shi," Image Feature Extraction Method Based on Shape Characteristics and Its Application in Medical Image Analysis," 2011, pp. 172–178.

[22] "Pattern recognition and image processing techniques," retrieved from, https://shodhganga.inflibnet.ac.in/bitstream/10603/152244/8/08_chapter%201.pdf, Last accessed June 09, 2020.

[23] A Beginner's Guide to Neural Networks and Deep Learning, "Neural Network Definition," retrieved from https://pathmind.com/wiki/neural-network, Last accessed July 06, 2020.

[24] Kaiming He, Xiangyu Zhan, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA, 27–30 June 2016, pp. 770–778.

[25] Roopa Patil and Ravindra S. Hegadi, "Segmentation of Cotton Insects and Pests Using Image Processing," Vinayak Mission and Karnataka University Conference on Cotton Insects and Pest, India, February 2009.

[26] Zhibin Wang, Kaiyi Wang, Zhongqiang Liu, Xiaofeng Wang, and Shouhui Pan, "A Cognitive Vision Method for Insect Pest Image Segmentation," International Federation of Aoutmatic Control Conference Paper, 2018, pp. 85–89.

[27] Muhammad Danish Gondal and Yasir Niaz Kha, "Early Pest Detection from Crop using Image Processing and Computational Intelligence," Fast-NU Research Journal, Vol.1, Issue 1, 2015, pp. 59-66.

[28] Ganesh Bhadane, Sapana Sharma, and Vijay B. Nerkar, "Early Pest Identification in Agricultural Crops using Image Processing Techniques," International Journal of Electrical, Electronics and Computer Engineering , Vol.2, 2013, pp. 77-82.

[29] Johnny L. Miranda, Bobby D. Gerardo, and Bartolome T. Tanguilig , “Pest Detection and Extraction Using Image Processing Techniques,” International Journal of computer and communication Engineering, Vol.3, 2014, pp.189-199.

[30] Murali Krishnan and Jabert.G, “Pest Control in Agricultural Plantations Using Image Processing,” Vol.6, Issue 4, 2013, pp.68-74

[31] M. S. Prasad Babu and B. Srinivasa Rao, “Leaves Recognition Using Back propagation neural network - advice for pest and disease control on crops,” Andhra University, India, January 2007.

32 Wang Jianwei,” A Noise Removal Algorithm of Color Image,”, TELKOMNIKA Indonesian Journal of Electrical Engineering, Vol.12, January 2014, pp. 565-574.

[33] Aliyu Abubakar, Mohammed Ajuji, and Ibrahim Usman Yahya, “Comparison of Deep Transfer Learning Techniques in Human Skin Burns Discriminatio”, University of Bradford, UK, April 2020.

[34] Raúl Cid Carro, Juan-Manuel Ahuactzin Larios, Edmundo Bonilla Huerta, Roberto Morales Caporal, and Federico Ramírez Cruz, “Face Recognition Using SURF,” Proceedings of the 2015 Switzerland Conference on Face Recognition, vol.5, August 2015, pp 316-326.

[35] “Simple guide to confusion matrix terminology,” retrieved from https://genera.org.es/docs/are-locust-borers-dangerous-to-humans-42c3ce, last Accessed, August 25, 2020.

Annex A: The Standard Desert Locust Survey Form

Desert Locust Standard Survey/Control Form

Country: Date:

1	Survey Stop	1	2	3
1.1	Location name			
1.2	Time			
1.3	Latitude (DD MM SSS)			
1.4	Longitude (DD MM SSS / W, E)			
1.5	Surveyed area (ha)			
1.6	Locust (Present or Absent)			
1.7	4 GPS corner points of area to be treated **1 • → • 2** **↓** **4 • ← • 3**	**1:**	**1:**	**1:**
		2:	**2:**	**2:**
		3:	**3:**	**3:**
		4:	**4:**	**4:**
1.8	Area to be treated (ha)			

2	Ecology			
	Habitat			
2.1	Topography (Wadi, Plain, Plateau, Hills, Dunes, Interdunes, Crops, Pasture, Oasis, Reg, Salt flat, Depression, Well, Beach, Town)			
2.2	Soil type (Sand, Silt, Clay, Stone, Gravel, Rocks)			
2.3	Soil moisture (Dry or Wet)			
2.4	Wet soil depth (cm) From - To			

	Vegetation			
2.5	State (Greening, Green, Drying, Dry)			
2.6	Density (Low, Medium, Dense)			
2.7	Annual species (list the 3 dominant species)			
2.8	State (Greening, Green, Drying, Dry)			
2.9	Cover (%)			
2.10	Drying (%)			
2.11	Development stage (1,2,3,4,5)			
2.12	Perennial species (list the 3 dominant species)			
2.13	State (Re-greening, Green, Drying, Dry)			
2.14	Cover (%)			

2.15	Drying (%)			
2.16	Greening (%)			

	Weather			
2.17	Date of last rain			
2.18	Approximative quantity (Light, Moderate, Heavy)			
2.19	Quantity (mm)			
2.20	Temperature (°C)			
2.21	Wind coming from (N, NW, NE, W, E, S, SW, SE)			
2.22	Wind speed (m/s)			

3	**Locust**			
	Hoppers			
3.1	Stage (E-1-2-3-4-5-6-F)			
3.2	Dominant stage (E-1-2-3-4-5-6-F)			
3.3	Appearance (Solitary, *Transiens*, T/*congregans*, T/*dissocians*, Gregarious)			
3.4	Behaviour (Isolated, Scattered, Groups)			
3.5	Colour (Green, Green/Yellow, Green/Black, Yellow/Black, Black)			
3.6	Density (Low, Medium, High)			
3.7	Density minimum, average, maximum (per tuft or m^2)			
3.8	Average distance between tufts (m)			
3.9	Activity (Hatching, Marching, Feeding, Roosting, Moulting)			

	Bands			
3.10	Stage (E-1-2-3-4-5-F)			
3.11	Dominant stage (E-1-2-3-4-5-F)			
3.12	Density (Low, Medium, High)			
3.13	Density minimum, average, maximum (per m^2)			
3.14	Size minimum, average, maximum (m^2 or ha)			
3.15	Number of bands			
3.16	Average distance between bands (m)			
3.17	Colour (Black, Yellow/Black, Green)			
3.18	Activity (Hatching, Marching, Feeding, Roosting, Moulting)			

	Adults			
3.19	Stage (Immature, Maturing, Mature)			
3.20	Dominant stage (Immature, Maturing, Mature)			
3.21	Colour (Gray, Brown, Yellow Wings, Pink, Yellow)			
3.22	Appearance (Solitary, *Transiens*, T/*congregans*, T/*dissocians*, Gregarious)			

3.23	Behaviour (Isolated, Scattered, Groups)			
3.24	Breeding (Copulating, Laying)			
3.25	Density (Low, Medium, High)			
3.26	Number (per transect)			
3.27	Length (m) and width (m) of transect			

	Swarms			
3.28	Stage (Immature, Maturing, Mature)			
3.29	Dominant stage (Immature, Maturing, Mature)			
3.30	Colour (Pink, Yellow)			
3.31	Breeding (Copulating, Laying)			
3.32	Activity (Settled, Takeoff, Milling, Flying)			
3.33	Density minimum and maximum (per m^2)			
3.34	Density (Low, Medium, High)			
3.35	Size (ha, km^2))			
3.36	Flying from (N, NW, NE, W, E, S, SW, SE)			
3.37	Flying to (N, NW, NE, W, E, S, SW, SE)			
3.38	Flying height (Low, Medium, High)			
3.39	Flying duration (h and min)			
3.40	Cohesion (Weak, Medium, Strong)			
3.41	Shape (Cumuliform, Stratiform)			

4	**Control**			
4.1	Application type (Full cover, Barrier)			
4.2	Area treated (ha) and area protected (ha)			
4.3	Pesticide name			
4.4	Formulation (EC, ULV)			
4.5	Concentration (g a.i./L or %)			
4.6	Application rate (L/ha or g/ha)			
4.7	Quantity used (L or G)			
4.8	Method (Handheld, Backpack, Vehicle, Air)			
4.9	Treatment duration (h and min)			
4.10	Mortality rate (%)			
4.11	Time after treatment (hours)			
4.12	Phytotoxicity (Present, Absent)			
4.13	Zootoxicity (Present, Absent)			

5	**Safety**			
5.1	Protective clothing used: Goggles, Mask, Overalls, Boots (L, M, C, B, G)			
5.2	Intoxication (Yes, No)			
5.3	Cholinesterase rate monitoring (Yes, No)			

Annex B: Sample System Code

```
function varargout = Locust_Identfication_GUI(varargin)

% LOCUST_IDENTFICATION_GUI MATLAB code for Locust_Identfication_GUI.fig
%      LOCUST_IDENTFICATION_GUI, by itself, creates a new
LOCUST_IDENTFICATION_GUI or raises the existing
%      singleton*.
%
%      H = LOCUST_IDENTFICATION_GUI returns the handle to a new
LOCUST_IDENTFICATION_GUI or the handle to
%      the existing singleton*.
%
%      LOCUST_IDENTFICATION_GUI('.CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in LOCUST_IDENTFICATION_GUI.M with the given
input arguments.
%
%      LOCUST_IDENTFICATION_GUI('Property','Value',...) creates a new
LOCUST_IDENTFICATION_GUI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Locust_Identfication_GUI_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Locust_Identfication_GUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Locust_Identfication_GUI
 gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Locust_Identfication_GUI_OpeningFcn, ...
                   'gui_OutputFcn',  @Locust_Identfication_GUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
```

```
end
% End initialization code - DO NOT EDIT


% --- Executes just before Locust_Identfication_GUI is made visible.
function Locust_Identfication_GUI_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
set(handles.axes1,'visible','off');
set(handles.axes2,'visible','off');
% Update handles structure
guidata(hObject, handles);
function varargout = Locust_Identfication_GUI_OutputFcn(hObject, eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in upload.
function upload_Callback(hObject, eventdata, handles)
% hObject    handle to upload (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.output=hObject;
global  newImage;
[fname, path]=uigetfile('*.jpg,*.bmp,*.png','Give imput image','*.jpg');
 fname=strcat(path,fname);
 newImage=imread(fname);
 imshow(newImage,'Parent',handles.axes1);
 title('This Is the Original Image To be Process','Parent',handles.axes1);
guidata(hObject,handles);
% --- Executes on button press in getwebcam.
function getwebcam_Callback(hObject, eventdata, handles)
% hObject    handle to getwebcam (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global  newImage;
web=webcam;
newImage=web.snapshot;
newImage=imfill(newImage,'holes');
axes(handles.axes1);
 imshow( newImage);
 title('This is Original Image Get From Webcam');
% --- Executes on button press in trainDCNN.
function trainDCNN_Callback(hObject, eventdata, handles)
handles.output = hObject;
global data;
global net;
global augtrainSet;
global featureLayer;
```

```
global trainFeature;
global trainLabel;
global clasifier;
global imageSize;
global detector;
global valSet;
global augvalSet;
global valFeature;
set(handles.train,'string','Training....');
outputFolder='F:\MSc\Thesis'
rootFolder = fullfile(outputFolder, 'image')
categories={'Locust','NonLocust'};
data=imageDatastore(fullfile(rootFolder,categories),...
'LabelSource','foldernames');
net=resnet50();
tbl=countEachLabel(data)
minSetcount=min(tbl{:,2})
data=splitEachLabel(data,minSetcount,'randomize');
[trainSet,valSet]=splitEachLabel(data,0.2,'randomize');
imageSize=net.Layers(1).InputSize;
augtrainSet=augmentedImageDatastore(imageSize,trainSet,'ColorPreprocessing','gray2rgb');
augvalSet=augmentedImageDatastore(imageSize,valSet,'ColorPreprocessing','gray2rgb');
featureLayer='fc1000';
trainFeature=activations(net,augtrainSet,featureLayer,'MiniBatchSize',32,'OutputAs','columns');
trainLabel=trainSet.Labels;
valFeature=activations(net,augvalSet,featureLayer,'MiniBatchSize',20,'OutputAs','columns');
clasifier=fitcecoc(trainFeature,trainLabel,'Learner','Linear','Coding','onevsall','ObservationsIn','c
olumns');
%% detection of locust
cas = load('locustimage.mat')
%cas = load('locust.mat')
  imagepath=cas.gTruth.DataSource.Source
  Locust=cas.gTruth.LabelData.Locust
  trainingData=table(imagepath,Locust)
  inputLayer = imageInputLayer([32 32 3])
filterSize = [3 3];
numFilters = 32
middleLayers = [
   convolution2dLayer(filterSize, numFilters, 'Padding', 1)
   reluLayer()
   convolution2dLayer(filterSize, numFilters, 'Padding', 1)
   reluLayer()
   maxPooling2dLayer(3, 'Stride',2)  ];
finalLayers = [
   fullyConnectedLayer(64)
   reluLayer()
```

```
    fullyConnectedLayer(2)
    softmaxLayer()
    classificationLayer()];
layers2 = [
    inputLayer
    middleLayers
    finalLayers ];
 options = trainingOptions('sgdm', 'MiniBatchSize', 32, 'InitialLearnRate', 1e-3, 'MaxEpochs', 5,
      'VerboseFrequency', 200, ...
      'CheckpointPath', tempdir);
detector = trainRCNNObjectDetector(trainingData, layers2, options,'NegativeOverlapRange', [0
0.3],
      'PositiveOverlapRange', [0.6 1])
set(handles.train,'string','Training Done.');
set(handles.train2,'string','Total Image');
set(handles.train3,'string',num2str(minSetcount));
guidata(hObject,handles);
% --- Executes on button press in ExtractFeature.
function ExtractFeature_Callback(~, eventdata, handles)
global newImage;
gImage=rgb2gray(newImage);
gPoints = detectSURFFeatures(gImage);
axes(handles.axes2);
imshow(gImage);
hold on;
gImage=plot(selectStrongest(gPoints, 100));
title('100 Strongest Feature ');
% --- Executes on button press in Identify.
function Identify_Callback(hObject, eventdata, handles)
global newImage;
global ds;
global newImageFeature;
global output;
global imageSize;
global net;
global featureLayer;
global clasifier;
 ds=augmentedImageDatastore(imageSize,newImage,'ColorPreprocessing','gray2rgb');
 newImageFeature=activations(net,ds,featureLayer,'MiniBatchSize',32,'OutputAs','columns');
 output=predict(clasifier,newImageFeature,'ObservationsIn','columns');
  if (output=='Locust')
global  identfid;
   identfid=newImage;
  axes(handles.axes2);
    imshow(identfid);
    title('Imput Image Contains  Locust Object ');
```

```
  else
     %newImage=im2bw(newImage);
    [x, y, z] = size(newImage);
    K=insertMarker(newImage,[x/2 y/4],'X','Size',100);
    axes(handles.axes2);
    imshow(K);
     title('No locust on the input image');
end
function in1_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function in1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in binary.
function binary_Callback(hObject, eventdata, handles)
global  newImage;
 bImage=im2bw(newImage,0.7);
 %newImage=imfill(newImage,'holes');
 axes(handles.axes2);
 imshow(bImage);
 title('This is Binary Image');
% --- Executes on button press in detectbt.
function detectbt_Callback(hObject, eventdata, handles)
global  newImage;
global detector;
global output;
if (output=='Locust')
%bbox=step(detector,newImage)
[bbox, score, label] = detect(detector, newImage,'MiniBatchSize', 32)
[score, idx] = max(score)
bbox = bbox(idx, :);
annotation = sprintf('%s: (Confidence = %f)', label(idx), score);
identfid=insertObjectAnnotation(newImage,'rectangle',bbox,annotation,'LineWidth',8);
%identfid=insertShape(newImage,'rectangle',bbox,'LineWidth',8);
axes(handles.axes2)
imshow(identfid);
    title('Identified Locusts');
else
    imshow(newImage)
    title('Unable to Identify b/c No locust on the input image');
end
```

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

Name: __Mebrahtu Chaklu_____

Signature: _____

Date: _____

Confirmed by advisor:

Name: __Yaregal Assabie(PhD)_____

Signature: _____

Date: _____