MAIZE DISEASE CLASSIFICATION MODEL BASED MULTI TASK LEARNING - CONVOLUTIONAL NEURAL NETWORKS

DIANE NIYOMWUNGERE

MASTER OF SCIENCE

(Information Technology)

JOMO KENYATTA UNIVERSITY

OF

AGRICULTURE AND TECHNOLOGY

2024

Maize Disease Classification Model Based Multi Task Learning - Convolutional Neural Networks

Diane Niyomwungere

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Technology of the Jomo Kenyatta University of Agriculture and Technology

2024

DECLARATION

This thesis is my original work and has not been presented for a degree in any other university

Signature……………………………………………..Date……………………………

Diane Niyomwungere

This thesis has been submitted for examination with our approval as the University Supervisors

Signature……………………………………………..Date……………………………

Prof. Waweru Mwangi, PhD

JKUAT, Kenya

Signature……………………………………………..Date……………………………

Dr. Richard Rimiru, PhD

JKUAT, Kenya

DEDICATION

To my parents Ir. Fabien Bizoza and Victoire Hatungimana, my brothers and sisters, my Husband Dr. Landry Desire Masabo, my children Gino, Mara Rohi and Niko Masabo, I dedicate this work.

ACKNOWLEDGEMENT

I thank the almighty God for enabling me to write this Research Thesis. Many thanks to my supervisors Prof. Waweru Mwangi and Dr. Richard Rimiru for their guidance and support throughout the process. It has been a great and amazing learning opportunity.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF APPENDICES

ACRONYMS AND ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Networks
CNN	Convolutional Neural Networks
CNN-MTL	Convolutional Neural Networks- Multitask Learning
DL	Deep Learning
ML	Machine Learning
MLP	Multilayer Perceptron
MTL	Multitask Learning

ABSTRACT

One of the top machine learning algorithms for image classification is Convolutional Neural Networks, according to experts. Convolutional Neural Networks has been extensively used in the agriculture sector for a variety of solutions, including identifying plant diseases, forecasting crop production, and categorizing land cover, among others. Unfortunately, creating Convolutional Neural Networks models requires a significant amount of data, which is extremely difficult to come by in agriculture. In this study, we suggest combining the Convolutional Neural Networks and Multitask Learning techniques since it has been shown to be a good algorithm to use when there isn't enough data by utilizing its ability to share layers. Additionally, Multitask Learning enabled us to simultaneously identify pathogens and diseases that affect maize, which is not possible when using a single Convolutional Neural Networks model. Indeed, recognizing pathogen may help at preventing the disease to spread throughout the whole field. Multitask Learning helped in improving the performance of our model by reducing overfitting. In this research, we combined Multitask Learning with other regularization techniques for a better performance. Indeed, the test accuracy of the overfitting model increases from 60.08% for the single maize disease model to 74.48% when combining the maize disease identification model with the maize pathogen identification model in one model using Multitask Learning. The accuracy rises to 77.44% while combining Multitask Learning to the early stopping method. However, the test accuracy goes up to 85.22 % when Multitask Learning is combined with Early Stopping and Transfer Learning.

Keywords: Multitask learning, Convolutional Neural Networks, Overfiting, Image Classification, and Regularization Methods.

CHAPTER ONE

INTRODUCTION

1.1 Background

Computer science and its distinctive subjective knowledge such as Artificial Intelligence, Machine Learning, Deep Learning, Computer vision, Image processing, have previously undergone significant advancements in various aspect of real life.

Deep Learning (DL) methods are relatively emerging Machine Learning algorithms and are applied in different domains. Deep Learning models may be supervised, semi-supervised, or unsupervised as will be detailed in the next chapter. Deep Learning is based on human brain modeling. Compared to other Machine Learning algorithms, Deep Learning are the best because of their robustness, generalization and scalability, they can be applied to almost all fields of science indeed (Alom et al., 2019). The most popular Deep Learning methods are:

1. Convolutional Neural Network (CNN): are a specialized kind of ANNs that use convolution in place of general matrix multiplication in at least one of their layers. Convolutional Neural Networks involve many connections, and the architecture is typically comprised of different types of layers, including convolution, pooling and fully-connected layers, and realize form of regularization(Namatēvs, 2018).
2. Recurrent Neural Network (RNN) are NN which are capable of modelling sequential data for sequence recognition and prediction(Salehinejad et al., 2017). They include Long Short-Term Memory (LSTM) capable of learning long term sequences, and Gated Recurrent Units (GRU) is the Recurrent Neural Network (RNN) appropriate for regression modeling of deep learning tasks such as prediction(Kirori & Ireri, 2020). It is like a long short-term memory (LSTM) with a forget gate, but has fewer parameters than LSTM.

3. Auto-Encoder (AE): aims at learning an identity mapping on a given dataset with a bottleneck latent dimension. They are suitable at learning compressed, interpretable, and structured data representations.(Cosentino et al., 2021).
4. Deep Belief Network (DBN): method of solving the problems from neural network with deep layers, such as low velocity and the overfitting phenomenon in learning.(Yuming et al., 2015)
5. Generative Adversarial Network (GAN), and
6. Deep Reinforcement Learning (DRL). (Alom et al., 2019)&(H. Wang & Raj, 2017)

Convolutional Neural Networks algorithms, the subject of this study, are used to solve graph problems, image processing and computer vision, speech processing, and medical imaging, etc.

In their study, (Tyr Wiesner-Hanks et al., 2017) showed that Convolutional Neural Networks are a class of Machine Learning models that can be trained to detect objects accurately in images, making them the current standard for object recognition.(Stewart et al., 2019).

Specifically, in agriculture domain, many researchers have since recently proposed and adapted Convolutional Neural Networks to implement crop cultivation, weather forecasting, yield prediction, insect detection, distinguishing weeds from crops, detecting the absolute environment to grow plants perfectly, and so on.

(Andreas K. and Francesc X. P.-B, 2018). noticed that Deep Learning offers better performance in solving various agricultural problems related to computer vision and image analysis, including classification or prediction.(Kamilaris & Prenafeta-Boldú, 2018)

In their study, (Ji et al., 2018) built a three-dimensional (3D) Convolutional Neural Networks based approach that classifies crops automatically from spatio-temporal multi-spectral remote sensing images.

Deep Convolutional Neural Networks model has been proposed by (Nagasubramanian et al., 2018) for plant disease (the charcoal rot of soybeans) identification using hyperspectral data.

(Lottes, Behley, Milioto, & Stachniss, 2018) proposed a crop-weed classification system based on a fully convolutional network with an encoder-decoder structure and incorporating spatial information through the use of sequential data

For image processing, distinct techniques such as image segmentation(Sartin et al., 2014), image filtering(Sansao et al., 2012), image histogram analysis(Wu et al., 2014), image background extraction(Eerens et al., 2014) have been proposed by many researchers to face agricultural challenges.

1.2 Problem Statement

Maize, a critical global food source, is susceptible to various diseases caused by pathogens that significantly impact crop yield and food security. The identification of both the maize disease and its causative pathogen is crucial for implementing targeted and effective agricultural interventions. Current methods for disease and pathogen classification often involve separate analyses, leading to increased complexity and resource requirements.

Traditional approaches to disease and pathogen identification in maize are labor-intensive, time-consuming, and rely heavily on expert knowledge. Leveraging the power of Convolutional Neural Networks (CNNs) presents an opportunity to automate this process, offering a scalable and efficient solution.

However, building an integrated model for the simultaneous classification of maize diseases and their associated pathogens using multitask learning remains an underexplored area of research as illustrated below by the models proposed in the past.

(Khatib et al., 2022) built a maize leaf disease identification model based on Convolutional Neural Networks. They achieved an accuracy of 98.3% using AlexNet architecture.

(Bedi & Gole, 2021) proposed a hybrid model based on Convolutional Autoencoder (CAE) network and Convolutional Neural Network (CNN) applied to detect Bacterial Spot disease present in peach plants using leaf images. The testing accuracy achieved 98.38%. The model can be used for any plant disease detection.

This research aimed to address this gap by proposing an innovative Maize Disease and Pathogen Classification Model based on Multitask Learning-Convolutional Neural Networks.

Classically, a single Convolutional Neural Networks model is dedicated to a single task, whereas our goal is to classify both the disease and its pathogen. Fortunately, Multitask Learning allows for the learning of multiple tasks in a single model at the same time (Huq, Gani, Sherif, & Abid, 2021).

The Multitask Learning method was then combined with convolutional Neural Networks to classify the disease and its pathogen at the same time images. Furthermore, in the agricultural field, it is nearly impossible to find large labelled datasets required for the best Convolutional Neural Networks model performance. Multitask Learning is also a good technique to use when there is a lack of data. Indeed, Multitask Learning makes use of useful information from related tasks to address the data sparsity issue (Y. Zhang & Yang, 2018). Finally, Multitask Learning is one of the regularization techniques used to avoid overfitting while training a Convolutional Neural Networks model; it has been combined with other regularization techniques to improve the proposed model's performance.

1.3 Objectives

1.3.1 Global Objective

The broad objective of this research thesis is to build a Multitask Learning-Convolutional Neural Networks model for maize disease identification.

1.3.2 Specific Objectives

i. Analyze how multilayer perceptron is used in deep learning to establish complex relationships among entities.
ii. Evaluate different regularization methods used in deep learning to minimize the loss function.
iii. Develop an appropriate multitask Learning-Convolutional Neural Networks based model for identification of maize disease.
iv. Test and validate the proposed model.

1.4 Research Question

The goal of this study is to answer the following questions:

i. How the multilayer perceptron is used in deep learning to establish complex relationships among entities?
ii. Which current regularization methods are used in deep learning to minimize the loss function?
iii. How shall an appropriate Multitask Learning-Convolutional Neural Networks based model be developed for maize diseases classification?
iv. What are the test results of the proposed model?

1.5 Justification

Maize is an important cereal for humans and animals all over the world. It can be transformed into a variety of food and industrial products, such as scratch sweeteners, oil, beverages, glue, industrial alcohol, and fuel ethanol (Ranum, Peña-Rosas, & Garcia-Casal, 2014).

Agriculture employs a large portion of Kenya's population, as it does in many African countries. Kenya also has a diverse plant diversity due to a variety of habitats. In fact, Kenya has an estimated 7,500 plant species (Kainyu, 2014), and maize can grow in 90% of the Kenya region (Wambuga, P.W. and Muthamia, 2009) as shown in Figure 1. It is commonly consumed in the form of ugali, uji, mahindi, choma, and githeri. Kenya was a major exporter of maize until the 1990s (Barmao & Tarus, 2019), and the amount imported has increased exponentially since then.

This shift from exporter to importer is the result of numerous challenges, including the impact of climate change and persistent biotic and abiotic stresses, a lack of good maize production policies, and a lack of power technology and strategy (Ray, 2013).

One of the main challenges faces the maize crop is the presence of diseases, both in the field and in the storage. Foliar diseases range from fungal, bacterial and viral and include maize rust, maize smut, northern leaf blight (NLB), ear rots, gray leaf spots (GLS), maize streak disease and the Maize lethal necrosis disease (Kainyu, 2014) are the most frequently encountered diseases in Kenya. Yield losses due to GLS range from 10 to 70% though during epidemics, 90 to 100% losses have been reported (Charles, Muiru, Miano, & Kimenju, 2019).

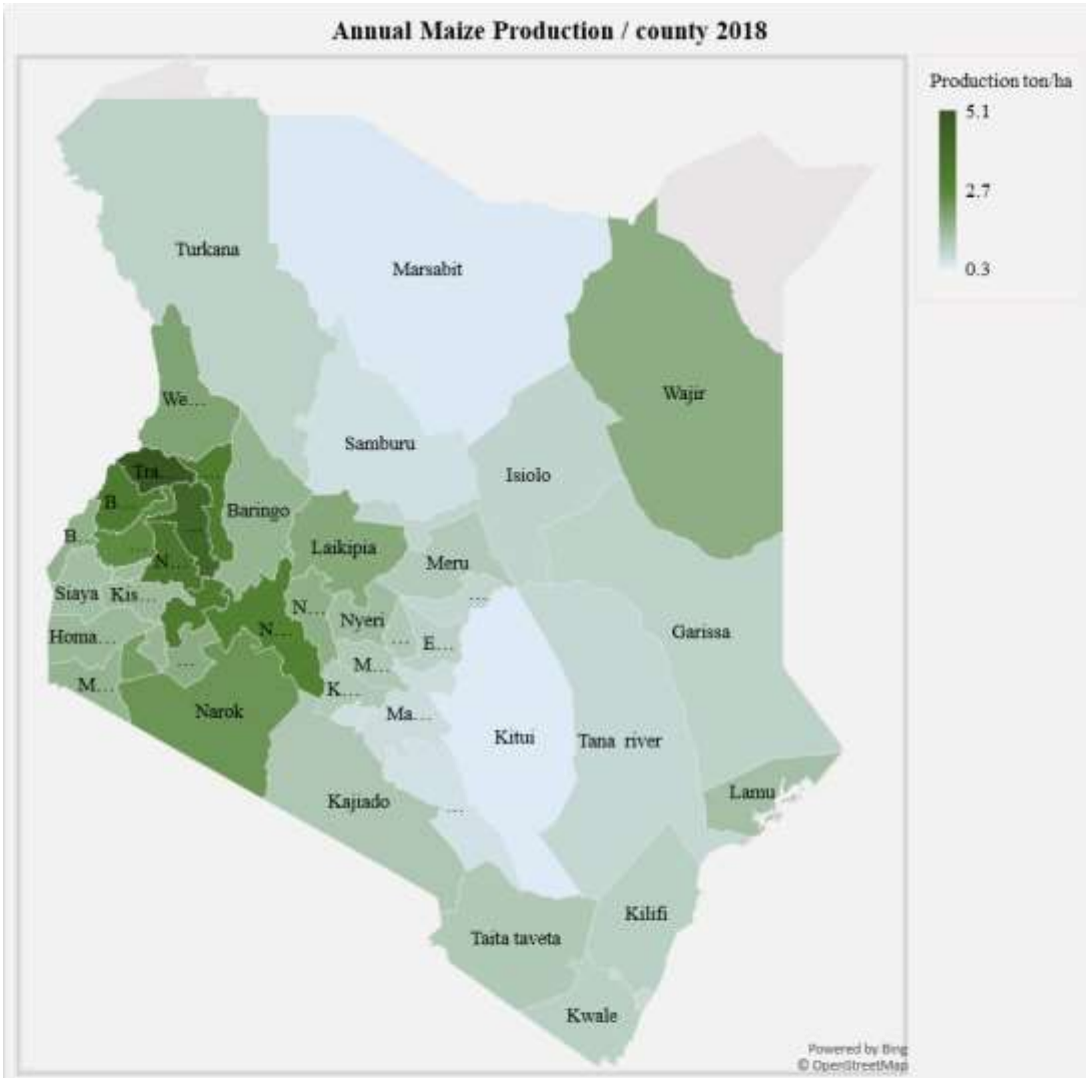


Figure 1.1: Maize Production in Kenya 2018

Source: (*(Dienya, 2020)*

Note that data for Mandera County is missed (see Appendix 1 for the data used to draw this map))

Some of these diseases are spreading so quickly that they must be identified sooner in order to be managed. For example, GLS, one of the most common maize diseases in Kenya, is spreading so quickly that an entire field can be considered diseased in just one month (Subedi, 2015). (Charles et al., 2019) found in their research that most farmers were unable to identify the various diseases without the assistance of experts.

This study proposed a classification model for maize disease, which could be a solution to the decrease in maize production by assisting farmers in identifying the various diseases without the assistance of expensive experts who are not always available. We proposed a model for identifying maize diseases using Convolutional Neural Networks.

Recognizing the plant pathogen may allow for earlier intervention because some pathogens may not require chemical treatment. In the case of a virus, for example, the best solution is to remove the infected plant(Lamp'l, 2013) and identifying the disease may aid with the selection of resistant seeds the following season. However, a traditional CNN cannot classify the disease and its pathogen within a single model. As a result, we propose a Multitask Neural Networks Convolutional Learning model that allows for the training of a single model for multiple tasks.

Furthermore, as previously stated, Multitask Learning is a good technique to use when the dataset is not large, which is still a real issue in the agricultural field. Multitask Learning is also a regularization technique used to avoid overfitting, which happens when the model fails to generalize to previously unseen data. In this research, we combined Multitask Learning with other regularization methods to enhance the performance of the proposed model.

1.6 Scope of the Research

Several Machine Learning algorithms are used in agricultural issues, but this study focused on the Multitask Learning-Convolutional Neural Networks algorithm for identifying maize disease. Our principal interest was on model regularization. Different deep learning regularization methods were compared and combined to improve the model's performance. Only image data from maize diseases commonly reported in Kenya were considered.

1.7 Thesis Organization

This thesis is divided into five chapters. Chapter 1 is an introductory chapter which started with a brief history of deep learning and computer vision. This chapter describes the problem statement, research objectives, justification, scope of the research and finally the thesis organization.

The second chapter begins with an introduction to Machine Learning and associated algorithms, followed by a discussion of Machine Learning applications in agriculture. The description of the Multilayer Perceptron in Deep Learning and the Convolutional Neural Networks structure with regularization techniques follows. The chapter next discusses loss functions and performance metrics in Convolutional Neural Networks before concluding by examining similar studies and identifying research gaps.

Chapter 3 presents the methodology used in this study. After the introduction of the chapter, then present the data to be used. The proposed model design and concept is then defined, followed by an explanation of the implementation the proposed model.

Experiments and results are covered in Chapter 4. It begins with outlining data processing, then proceeds to explain the four experiments carried out in this study, and concludes with a description of the results.

Chapter 5 summarizes this study and provides a general conclusion, followed by a recommendation for further work.

CHAPTER TWO

LITTERATURE REVIEW

2.1 Introduction

Machine learning (ML) is a subset of Artificial Intelligence (AI) that can automatically acquire, integrate, and develop knowledge from large-scale data, and then autonomously expand the acquired knowledge by discovering new information, without really being explicitly programmed to do so. ML and AI are usually used interchangeably, but the reality is that not all AI is ML, as illustrated in Figure 2, along with DL, which is a subset of ML, and Convolutional Neural Networks, which is a subset of DL.

In this section, we review research on ML algorithms and studies on ML in agriculture, as well as the Multilayer Perceptron (MLP) structure in DL, Convolutional Neural Networks and its regularization methods, loss functions, and performance metrics for Convolutional Neural Networks model construction, before explaining our research gaps after reviewing related works.

2.2 Machine Learning Algorithms

Machine Learning is one of the most rapidly growing areas of computer science. Currently, numerous researchers have proposed various Machine Learning algorithms to address a wide range of real-world challenges.

Machine Learning aims to enable machines to make predictions, cluster data, extract patterns, and make decisions. The ML techniques are mainly grouped in four categories as following (Malinowski et al., 2019; Sarker, 2021; Series, 2021):

- The supervised learning: the model learns from labeled data; for example, regression

- Unsupervised leaning: the model learns from unlabeled data; for example, clustering
- Semi-supervised learning: the model learns from the mixture of labeled and unlabeled data; for example, classification
- Reinforced learning: the model learns based on reward or penalty and then with no data.

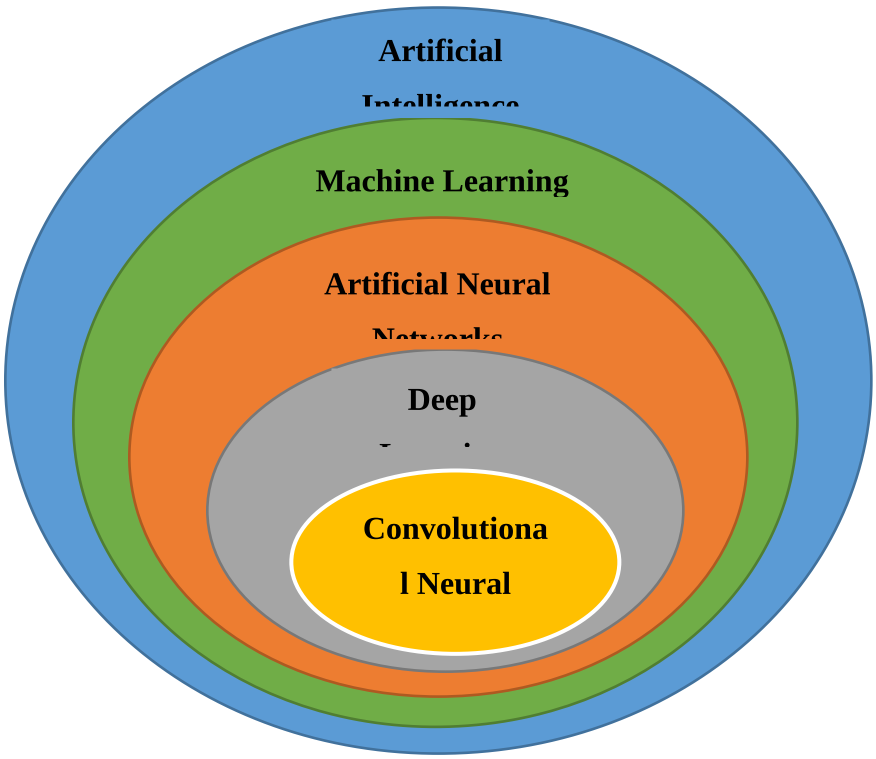


Figure 2.1: AI vs ML vs ANN vs DL vs CNN

Source: (Zhang *et al.*, 2022)

2.3 Machine Learning in Agriculture

Various Machine Learning algorithms have been used in agri-technology for a variety of solutions ranging from pre-harvesting, harvesting, and post-harvesting tasks, predicting harvest production, detecting disease in advance, recommending fertilizer for specific crops, and so forth.

(Thi et al., 2020) created and compared three different models for time series forecasting using the Keras toolkit: artificial neural network (ANN), recurrent neural network (RNN), and long short-term memory (LSTM). They discovered that the LSTM model outperformed the other tested models for long time scales in summer and spring.

A study on crop yield prediction was conducted by (Cedric et al., 2022). They built three models: a decision tree, a multivariate logistic regression model, and a k-nearest neighbor model, all of which were trained on rice, maize, cassava, seed cotton, yams, and banana crops from West African countries using climatic, weather, agricultural yield, and chemical data variables. The decision tree algorithm outperformed the other two algorithms.

(Fabio et al., 2022) developed a hybrid model based on M5P and Support vector regression for precipitation forecasting. The four metrics they used to evaluate the models' performance were coefficient of determination (R2), mean absolute error (MAE), root square mean error (RSME), and relative absolute error (RAE). The hybrid model outperformed the two models trained and tested separately.

Deep learning is one of the most widely used ML algorithms for a variety of agricultural problems. (Lottes et al., 2018) proposed a crop-weed classification system based on a fully convolutional network with an encoder-decoder structure and spatial information incorporated via sequential data. They trained their model on data from various growth ranges to distinguish crops at different stages, giving 96.1 recall and 96.6 precision. The sequential model also generalizes well to previously unknown fields. As a result, the experiment revealed that changing the visual appearance of the image data reduced the model's performance.

Recurrent Neural Networks were used to classify land cover using multi-temporal spatial data derived from a time series of satellite images (Ienco et al., 2017). They used two different datasets and applied one LSTM layer to both pixel-based and object-based classification. They discovered that their method worked well for underrepresented and difficult classes, but the accuracy was only 75.15 percent.

According to (Liakos, Busato, Moshou, Pearson, & Bochtis, 2018), farm management systems are evolving into real-time artificial intelligence enabled programs that provide recommendations and insights for farmer decision support by applying machine learning to sensor data.

In 2019, (Jha, Doshi, Patel, & Shah, 2019) examined the Machine Learning algorithms used in a variety of precision agriculture applications. They discovered that by applying machine learning to sensor data, farm management systems are evolving into true AI systems, providing optimal insights for decision-making and action. The authors developed a plant and weed distinguishing technique using machine learning and image processing techniques to successfully manage weed separation from crops. It is an important study because weed management is critical in the crop harvesting field. They used the Local Binary Pattern (LBP) algorithm to extract textural features from crop leaves and the Support Vector Machine (SVM) algorithm to classify the multiclass plant.

(Khaki & Wang, 2019) proposed a Deep Neural Networks model for crop yield prediction that outperformed other popular methods such as Lasso, shallow neural networks (SNN), and regression tree (RT). They trained the model using crop genotype, yield performance, and corn hybrid environment data.

A deep convolutional neural networks model were proposed by (Liu et al., 2018) for detecting four common apple leaf diseases: mosaic, rust, brown spot, and Alternaria leaf spot. They used an AlexNet-based architecture and a dataset of 13,689 images of diseased apple leaves. They achieved an accuracy of 97.62% by implementing the model in the Caffe framework on the GPU platform.

Bayesian algorithms(Amatya et al., 2016), decision trees(Veenadhari et al., 2011), ensemble learning(Kung et al., 2016), and support vector machines(Han et al., 2020) are examples of other Machine Learning algorithms used in agriculture

2.4 Multilayer Perceptron in Deep Learning

The architecture of Artificial Neural Networks was inspired by the architecture of the human brain, which is made up of a massive number of neurons and synapses. In the field of Neural Networks, a perceptron represents a single neuron and neurons are connected together to form a layer. A multilayer perceptron (MLP) (Aleshin-Guendel & Alvarez, 2017) is an ANN architecture made up of one input, one or more hidden, and one output layers whereas Deep Learning is a type of ANN that has more than one hidden layer. Figure illustrates an MLP that has two hidden layers. The nodes are linked together by links, and each link has a numerical weight assigned to it (Russell & Norvig, 2010).

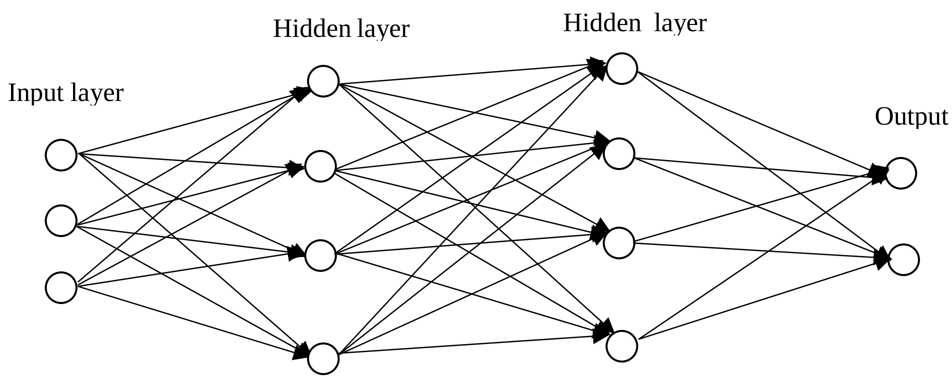


Figure 2.2: Two Hidden Layer MLP Architecture

The weights are real numbers that represent the importance of each input to the output (Nielsen, 2015). The perceptron learns by adding the bias (b) to the sum of inputs (x) multiplied by their weights (w). It is shown mathematically in equation 1.

$$y = \sum_i x_i w_i + b \qquad (1)$$

The feedforward network is formed when each layer feeds into the layer above until the output is produced. There are no loops; each node receives input from upstream nodes and sends output to downstream nodes. Through the activation function, the MLP converts the hidden layer output (Czum et al., 2020). Rectified Linear Unit function, softmax function, tanh function, and Sigmoid function are examples of common activation functions.

2.3.1 Rectified Linear Unit Function

Rectified Linear Unit function (ReLU) allows the network to converge rapidly. It is a derivative function, which permits backpropagation. It computes the following function:

$$f(x) = \max(0, x) \tag{2}$$

This function returns x if x is positive and 0 otherwise.

The disadvantage of using the ReLU function is that the network will die if the inputs approach zero or are negative. (Agarap, 2018).

2.3.2 Softmax Function

The softmax function is used to calculate probability distribution from a vector of real numbers. It is usually used for classification tasks typically at the output layer.(Chigozie N. et al., 2018). Softmax function is able to handle multiple-class models where it returns probabilities of each class. It is defined by equation 3 below.

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{3}$$

2.3.3 Sigmoid Function

The output of a sigmoid function is bounded (Y. Wang et al., 2020) between 0 and 1 what is called squashing function. It is useful in probability prediction. The equation 4 describes the sigmoid function. Its computation is expensive.

$$f(x) = \frac{1}{1+e^{-x}} \tag{4}$$

2.3.4 Tanh Function

The tanh function is a sigmoid function, which is bounded between 1 and -1(Feng & Lu, 2019). It is symmetric and centred on zero. The tanh function is given by equation 5. For a very high or very low value of x, the vanish gradient problem may occur.

$$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{5}$$

2.3.5 CNN Structure

Convolutional Neural Networks (ConvNets/CNN) are Deep Learning (DL) techniques used to solve computer vision problems. As previously stated, Neural Networks take a single vector as input and transform it through a series of hidden layers. Each hidden layer consists of a set of neurons, each of which is fully connected to all neurons in the previous layer, and neurons in a single layer function are completely independent and do not share any connections. The "output layer" is the final fully-connected layer. In addition to the input layer, Convolutional Neural Networks architecture mainly consists of three layers: convolutional, pooling, and fully connected. Figure illustrates a convolutional Neural Networks with two convolution layers, two max pooling layers and two fully connected layers.

2.3.6 Convolutional Layer

This is the first layer used to extract the different features from the input images. This layer performs the mathematical operation of convolution between the input image and a filter of size MxM. The dot product between the filter and the parts of the input image with respect to the size of the filter (MxM) is calculated by sliding the filter over the input image. The output is known as the Feature map, and it contains information about the image such as its corners and edges. This feature map is then fed to other layers, which learn several other features from the input image.

The convolutional filter is in charge of underlying the local image patch (Q. Zhang et al., 2019). For controlling the number of parameters, it has two shared parameters: kernel and scalar bias. Each neuron is only connected to a small portion of the input volume. The hyper-parameters that control the size of the output volume are depth, stride, and zero-padding.

2.3.7 Pooling Layer

The pooling layer is located between two convolutional layers. Its role is to combine the outputs of the neuron clusters and gradually reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and thus to control overfitting (Singh et al., 2017). The Pooling Layer operates independently on each depth slice of the input and spatially resizes it. The following are two common functions used in the pooling operation:

Max Pooling selects the maximum element from each of the feature map's windows. As a result, the output of the max-pooling layer would be a feature map containing the most dominant features of the previous feature map.

Average Pooling Average Pooling computes the average of the elements present in the filter's feature map region. It simply takes the features from the feature map and averages them.

2.3.8 Fully Connected Layer

The fully connected layers act as classifiers(Kamilaris & Prenafeta-Boldú, 2018). Their role is to understand the patterns generated by the previous layers (Ferreira & Giraldi, 2017). Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.

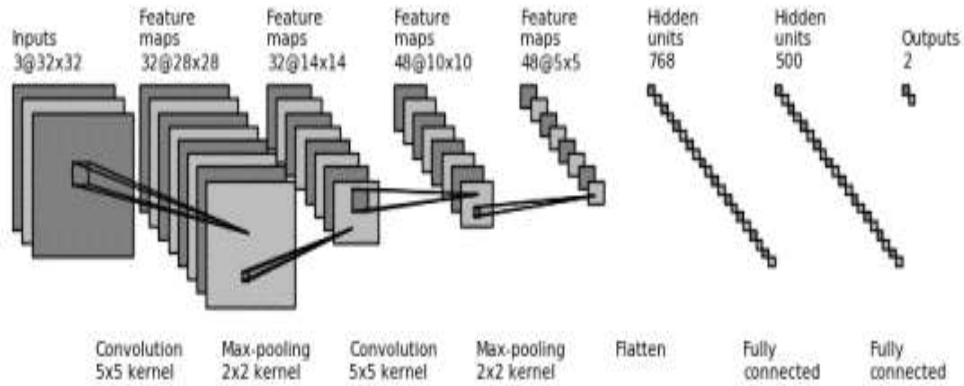


Figure 2.3: Example of Convolutional Neural Network structure

Source: (Murphy, 2016))

2.3.9 Popular Convolutional Neural Networks Architectures

There are no set rules for how many layers are required to build a Convolutional Neural Networks model or how the layers are connected to one another, but there are popular architectures that have proven to be successful over time:

- LeNet: developed by LeCun et al. in 1998, LeNet is made up of seven layers, including two convolution layers, two sub-sampling layers, two fully connected layers, and an output layer.
- AlexNet: The architecture was very similar to LeNet, but it was deeper, with more filters per layer, and with stacked convolutional layers. It included convolutions, max pooling, dropout, data augmentation, ReLU activations, and SGD with momentum.

- GoogleNet: The network was inspired by LeNet but added a new element known as an inception module. Batch normalization, image distortions, and RMSprop were all used. This module relies on a series of very small convolutions to drastically reduce the number of parameters. Their architecture included a 22-layer deep Convolutional Neural Networks, but the number of parameters was reduced from 60 million (AlexNet) to 4 million.
- VGGNet: Visual Geometry Group is a network of 16 convolutional layers that is appealing due to its very uniform architecture. Similar to AlexNet, only 3x3 convolutions are used, but there are many filters.
- ResNet: so-called Residual Neural Network, ResNet has been implemented by Kaiming He et al, 2016. They proposed a novel architecture with "skip connections" and heavy batch normalization. These skip connections, also known as gated units, are similar to recent successful RNN elements. They were able to train a NN with 152 layers while remaining less complex than VGGNet using this technique.
- DenseNet: refers to Densely Connected Convolutional Networks, was proposed by (Gao et al., 2018). It is a type of convolutional neural network that uses Dense Blocks to connect all layers directly with each other. To maintain the feed-forward nature, each layer receives additional inputs from all preceding layers and sends its own feature-maps to all subsequent layers.
- FractalNet: It is a type of convolutional neural network proposed by (Larsson et al., 2017) that foregoes residual connections in favor of a "fractal" design. They entail repeatedly applying a simple expansion rule to generate deep networks with precisely truncated fractal structural layouts. These networks have interacting subpaths of varying lengths but no pass-through or residual connections.
- CapsuleNet (G. Hinton et al., 2018): are networks that can retrieve spatial information and other important features in order to overcome the information loss seen in pooling operations. It focuses on replicating biological neural networks in order to improve recognition and segmentation.

2.4 Regularization Techniques in Deep Learning

Regularization is a machine learning technique that allows an algorithm to generalize well on previously unseen data (Courville, 2016). It refers to any changes made to the learning algorithm in order to reduce its generalization error (Goodfellow et al, 2016). Deep learning uses several regularization methods, the most common of which are as follows:

2.4.1 L2 Parameter Regularization

Commonly known as weight decay, L2 regularization is a method which aims to reduce the loss by adding a regularization term R(w) to the objective function L(w).

$$\hat{L}(w) = L(w) + R(w) \tag{6}$$

$$R(w) = \frac{\lambda}{2} \|w\|^2{}_2 \tag{7}$$

Where λ is the weighting term controlling the regularization over the consistency(Kukačka et al, 2017).

2.4.2 Data Augmentation

In practice, the amount of data available is limited, whereas deep learning algorithms generalize better when trained on larger amounts of data. Thus, data augmentation is a method of increasing the size of the training set. To augment data, common techniques include Translation, rotation, flipping, brightness adjustment, zoom in/zoom out, cropping, noise injection, and so on ((Shorten & Khoshgoftaar, 2019),(Mikołajczyk & Grochowski, 2018)). This method, however, is not applicable to all tasks. For example, it is effective for image recognition but not for density estimation. (Ian et al., 2016). There exist two types of data augmentation:

- **Online Data Augmentation**: it is the technique where images from training data are randomly selected and data augmentation techniques are used. The model is then trained using the original data, which includes randomly augmented images. The augmented images are never saved anywhere in this case, and it is impossible to tell which image is augmented.
- **Offline Data Augmentation:** this method allows to generate new data that are saved on the disk. After applying data augmentation technique to each and every training image, augmented images are obtained. This expands the dataset and strengthens the model. Such a method that can be used to increase the number of images in a dataset.

2.4.3 Multi-Task Learning

This is a method for regularizing a model by leveraging important information contained in related tasks. A Multitask Learning model is split into two parts: shared layers and task-specific layers. There are two common Multitask Learning methods used in Deep Learning: one is hard parameter sharing, which involves sharing hidden layers across all tasks while maintaining task-specific output layers. The alternative method is soft parameter sharing, in which each task has its own model with its own set of parameters (Thung & Wee, 2018). Given T number of tasks and an objective function L with a regularization term **Ω**, the regularized function is shown in equation (8).

$$\hat{L}(w1, \dots, wT) = \sum_{t=1}^{T} L(wt) + \Omega(w1, \dots, wT) \qquad (8)$$

2.4.4 Transfer Learning

Transfer learning(Shrivastava et al., 2019) is a machine learning method in which a model developed for one task is reused as the foundation for another model developed for a different task. It is the process of improving one's learning in a new task by transferring knowledge from a previously learned related task. Transfer learning can be applied in two ways, as explained below:

- **Feature Extraction**: Taking advantage of features learned by a model previously trained on a larger dataset in the same domain is a common practice. As the name implies, it is all about using the representations learned by a previous network to extract meaningful features from new samples by training a new classifier on top of the pre-trained model so that you can repurpose the feature maps learned previously for the dataset. This technique is recommended when working with a small dataset.
- **Fine-tuning**: This technique entails fine-tuning the top-level layers of the pre-trained models to the new dataset in order to improve performance even further. In this case, the weights were adjusted so that the model learned high-level dataset features. When the training dataset is large and very similar to the original dataset on which the pre-trained model was trained, this technique is usually recommended.

2.4.5 Early Stopping

Early stopping is one of the most common regularization methods used in deep learning to prevent overfitting. This happens at the Optimal Stopping Point, and that is the point at which the validation loss begins to rise while the training loss falls. The stopping criterion is a predicate that indicates when to discontinue training (Karlsruhe, 2015).

2.4.6 Dropout

Dropout(G. E. Hinton et al, 2012) is a method that prevents overfitting and allows for the efficient combination of increasingly diverse networks (Hinton et al , 2014). At each iteration, this technique involves randomly removing some nodes from the networks. Dropout is a data-dependent regularizer as well. (Z. Li et al, 2016).

2.5 Loss Functions in Deep Learning

The loss function is a method of assessing the performance of a machine learning model by calculating the difference between the predicted and actual output. Deep Learning has several loss functions that are used depending on the type of problem to solve. Some popular loss functions are as follows:

2.5.1 Cross-Entropy Losses

Cross entropy is measured by the distance between the output probabilities and the actual values as illustrated by the following equation 9.

$$L_{CE} = -\sum_{i=0}^{n} x_i \log(p_i) \qquad (9)$$

Source : (Kiprono E., 2020)

Where n the number of classes, x_i the true label and p_i the softmax probability for the i^{th} class. Categorical Cross Entropy Loss is used for multi-class classification problems and binary Cross Entropy Loss for binary classification problems.

2.5.2 Hinge Loss

The hinge loss is a loss function used usually for binary classification problems. For x inputs and y outputs, the Hinge Loss (HL) is given by equation 10:

$$HL = \max(0,1 - y * f(x)) \qquad (10)$$

Source: (Varma, 2018)

2.5.3 Mean Squared Error (MSE) Loss

The mean squared Error is a regression loss function used as the average of the difference of the predictions and the real value squared across the whole dataset. It is given by the equation 10:

$$MSE = \frac{1}{n}\sum_{i}^{n}(y_i - \hat{y}_i)^2 \qquad (11)$$

Where n is the number of samples, y the actual value and ŷ the predicted value for the i^{th} class (Dufourq & Bassett, 2017).

2.3.4 Kullback Leibler Divergence Loss

Kullback Leiber Divergence (or KL Divergence) also called relative entropy is used especially in multiclass classification problem by comparing two probability distributions as shown in equation 11. KL Divergence calculates how much a given distribution is away from the true distribution

$$D_{KL} = \sum_{x \epsilon X} p(x) log \frac{p(x)}{q(x)} \quad (12)$$

source: (Nelken & Shieber, 2006)

2.5.5 Performance Metrics

Metrics are used to monitor and measure a model's performance. Evaluation Following training, metrics are used to assess overall performance.

2.5.6 Accuracy

The accuracy is calculated by multiplying the number of correctly classified data items by the total number of observations by 100. It is also the most widely used metric for evaluating model performance in classification problems.

2.5.7 Precision

Precision is defined as the fraction of true positives to total positives predicted (equation 13). It simply displays "how many of the selected data items are relevant." In other words, how many of the observations predicted by an algorithm to be positive are actually positive.

$$TP = \frac{TP}{TP+FP} \quad (13)$$

Where TP is the true positives and FP, the false positives.

2.5.8 Recall

The recall is the ratio of true positives to all positives in the ground truth. It displays "the number of relevant data items selected." In other words, how many of the observations that are actually positive have been predicted by the algorithm. It equals to true positives (TP) divided by the sum of true positives and false negatives (FN) as shown by equation 14.

$$TP = \frac{TP}{TP+FN} \tag{14}$$

2.5.9 F1-Score

F1-score utilizes a combination of precision and recall. The F1 score is really the harmonic mean of the two. The basic formula for the two is as follows:

$$F1 = \frac{2}{\frac{1}{P}+\frac{1}{R}} \tag{15}$$

With P being the precision and R the recall.

A high F1 score now indicates high precision as well as high recall. It has a strong blend of precision and recall and performs well on unbalanced classification issues.

2.5.10 AU-ROC

AUC stands for "Area under the ROC Curve" (Yang et al., 2019). AUC, in other words, assesses the full two-dimensional area beneath the complete ROC curve: A receiver operating characteristic curve (ROC curve) is a graph that depicts the performance of a classification model over all classification levels. This graph depicts two parameters:

- True Positive Rate

- False Positive Rate

2.6 Related Works and Gaps

Convolutional Neural Network (CNN) is a deep learning architecture inspired by living creatures' natural visual perception mechanisms. Image classification, object detection, object tracking, pose estimation, text detection, visual saliency detection, action recognition, scene labeling, speech and natural language processing are some of the main applications of CNN.

Several researchers have worked on the maize disease classification model. In their research, (Da Rocha et al., 2021) proposed convolutional neural network architectures to classify maize leaf diseases and enhance the models' performance using Bayesian hyper-parameter optimization, data augmentation, and fine-tuning strategies. (DeChant et al., 2017) built a Convolutional Neural Networks model to identify North leaf disease from maize crop leaf images, and data augmentation was used as a regularization method.

Many of these experiments identified maize disease using a single Convolutional Neural Networks algorithm. However, knowing which pathogen is causing the disease may assist prevent it from spreading earlier, whereas classic Convolutional Neural Networks (CNNs/ConvNets) were designed to solve a single task classification problem and are currently inadequate for multi-output classification.

Recently, several works combining Convolutional Neural Networks to Multitask Learning algorithms have been proposed to permit learning multiple tasks at a time in a single model. Thus, Multitask Learning is a technique used to improve the performance of multi related learning tasks(Y. Zhang & Yang, 2018). Using Multitask Learning may reduce storage space and training time. Also, learning tasks simultaneously may increase the accuracy of each task and then reduce the risk of overfitting.

In their study, (Zeng & Ji, 2016) proposed a multi-instance multi-task convolutional neural networks (MIMT-CNN) model. The shared sub-Convolutional Neural Networks are connected to the input images and their outputs became the inputs of the additional convolutional and fully conned layers. They trained their model to mouse's brain gene images. The regularization methods included L2 norm and AUC are used as performance measure. The loss function was given by the summation of the loss functions of the individual tasks. Their model outperformed the VGG pretrained model to image Net used as the baseline.

A multi task learning for large scale image classification has been developed by (Kuang et al., 2017)/ they used soft parameter sharing method where discriminative tree classifier over concept ontology replaced the N-way flat softmax classifier. Three ontology networks based on AlexNet was built to be trained to fashion60 and ILSVRC2012 datasets clustered into grained groups. They found that tree classifier-based model has advantage over storage space. In this paper, they did not mention how they measured performance of their model.

Another multi-task convolutional neural network has been proposed by (C. Zhang & Zhang, 2014) to improve multi-view face detection. Two face and non-face examples datasets were trained using stochastic gradient descent, momentum, and weight decay. Cross entropy loss and mean squared error were minimized for the model regularization.

(Su et al., 2019) proposed another type of multi-task learning where parameters of the neural network were regularized by low rank. The model was iteratively optimized by gradient descent. An L1 norm regularization technique was also used, and their model compared to other multi-task learning methods from the literature.

The preceding studies did not indicate how Multitask Learning performed as a regularization technique, which is one of our research goals. The proposed model focused on which regularization approaches should be employed to get the best generalization. We relied on a hard parameter sharing architecture, which is made up of two major parts: sharing parameter layers and specific-task layers. Different regularization techniques were combined to improve model performance and then compared. The purpose of this study is then to develop a regularized Multitask learning-convolutional neural networks model for concurrently recognizing maize diseases and pathogens, which will assist farmers in automatically identifying different diseases without the assistance of an expert.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

To carry out various research activities, each successful project must have appropriate and approved methodology. This chapter discusses the proposed research methodology for the study. It starts by introducing the dataset that was used. The proposed model design and concept is then defined, followed by an explanation of how to implement the proposed model using the various tools available.

3.2 Data Sets

This proposed research applied use of images from several sources were collected from two websites, Kaggle and Plantwise, to form a single dataset The dataset consists of 2636 images of leaves, including five classes of maize diseases divided into two pathogens as listed below:

- **Northern Leaf Blight** (NLB) is caused by the fungus Exserohilum turcicum Long, narrow, tan lesions that form parallel to leaf margins are the first signs of NLB. As these lesions grow, the typical NLB signs such as long, oblong, or "cigar-shaped" tan or grey lesions will be visible. When humidity levels are high, the lesions generate olive-green or black fungus spores that can give them a dirty or dark appearance. If one looks at lesions with a hand lens, one can see the spores. Depending on hybrid susceptibility, the lesions can be anywhere between 1 and 7 inches long. A leaf may develop several lesions, and lesions may combine to generate significant, asymmetrical patches of dead tissue(Wise, 2011).

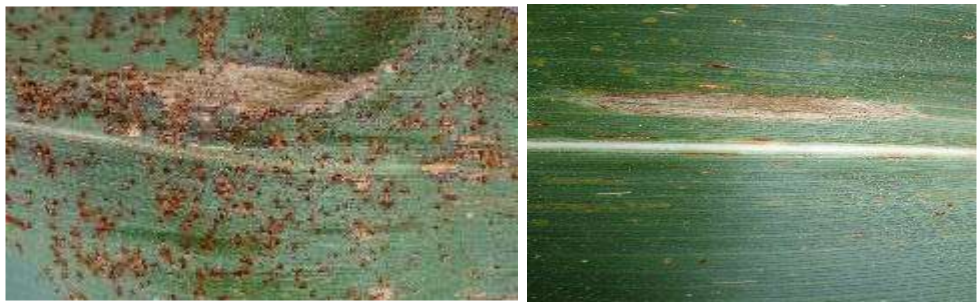


Figure 3.1: Maize Leaves with NLB Disease

- **Common Rust**: is fungal disease caused by Puccinia sorghi pathogen. Symptoms include little brown dots, Spots develop into jagged, elongated pustules that range in color from brick red to cinnamon brown. both the upper and lower leaf surfaces contain this (unlike southern rust) Late in the season, pustules shift from dark brown to black. occurs only on leaves; it does not affect husk leaves, sheaths, stalks, or ear shanks.Small, round specks that are pale and in clusters characterize early lesions on leaves. Small, powdery pustules that appear on the underside of the leaves may help identify lesions as they progress. Early signs of infection on leaves include brownish-red, rectangular pustules; later, the fungus bursts through the leaf surface (epidermis), exposing masses of powdery urediniospores. When walking through an infected field, spores produced within the pustules are easily dislodged and can be seen on a white shirt. Severe infections that occurred while the leaf tissue was in the whorl may result in the formation of a rust band closer to the base of the leaf. The entire leaf blade perishes as a result of band formation(Yan, 2015) .

Figure 3.2: Maize Leaves with Common Rust Disease

- **Gray Leaf Spot** is caused by the fungus Cercospora zeae-maydis.

  On leaves, the early signs of gray leaf spot appear as tiny, pinpoint lesions encircled by yellow haloes. The condition can be difficult to correctly diagnose at this point, but when lesions progress, they become long, narrow, brown to gray dots. Lesions can grow 1.5 to 2 inches long and spread parallel to the leaf veins. Lesions could also develop on the husks and sheaths of vulnerable hybrid plants. The major leaf veins prevent lateral leaf lesions from expanding, giving them a blocky appearance. Lesions can combine to generate big, erratic regions of dead tissue on the leaves in the right circumstances(Elliott & Harmon, 2011).

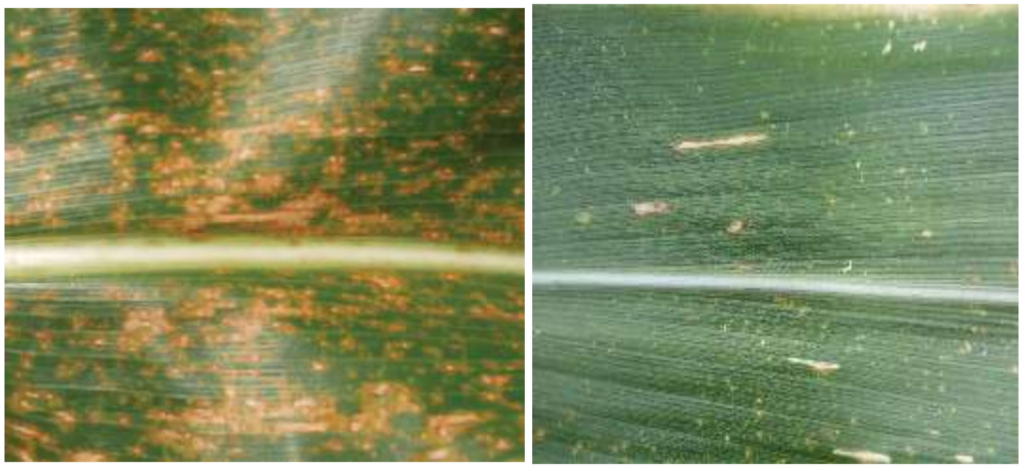


Figure 3.3: Maize Leaves with Gray Leaf Spot Disease

- **Maize Lethal Necrosis** (Osunga, M., Mutua, F.N., & Mugo, 2017) is a devastating viral disease of maize caused by double infection with Maize chlorotic mottle virus (MCMV) and any one of the Potyviridae family members. The symptoms include: Dying leaves; leading to premature plant death; Failure to tassel and sterility in male plants; Malformed or no ears; Rotting cob(Tonui et al., 2020).

Figure 3.4: Maize Leaves with MLN Disease

- **Leaf Streak** is a virus is transmitted mainly by Cicadulina mbila (maize leaf hopper) but other leafhopper species such as C. storeyi, C.arachidis and C. dabrowski have also been found to transmit the virus .
  The sucking mouth parts of the leafhopper allow them to enter plant cells with the help of digestive and salivary enzymes as well as physical force. Plant

stunting and the cessation of ear formation, development, and grain filling in diseased plants are the main causes of yield loss. Plants may succumb to serious infections and die too soon.

Very small, spherical, dispersed disease symptoms appear in the youngest leaves one week after infection and are the first sign of the disease. With plant growth, the number of spots grows, and they widen parallel to the leaf veins. In contrast to the dark green of typical foliage, fully extended leaves develop a chlorosis with broken yellow streaks along the veins. Maize that has been extensively diseased is shown stagger. The only insects known to spread the maize streak virus from one maize plant to another are Cicaduna species(Charles et al., 2019).

Figure 3.5: Maize Leaves with MLS Disease

The dataset was split into 80% and 20% for training and test sets, respectively. The training set was utilized to fit the classifier's parameters. To tune the parameters for the trained model optimization, 20% of the training set was used as a validation set. The test set was used to evaluate the model's performance.

3.3 Model Design

The goal of this study is to build a regularized Multitask Learning-based Convolutional Neural Network model. Multitask Learning was created to perform a model when there are multiple classes and/or a small dataset. Multitask Learning (Caruana et al., 1997) is a machine learning technique that improves learning for one task by sharing representations between related tasks. The proposed Multitask Learning model (see **Error! Reference source not found.**) is a hard parameter sharing model with shared nd task-specific layers. The model used an image of disease maize as input to classify it as various maize diseases and pathogens. Feature extraction occurs at the shared layer level, whereas classification occurs at the task-specific layer level

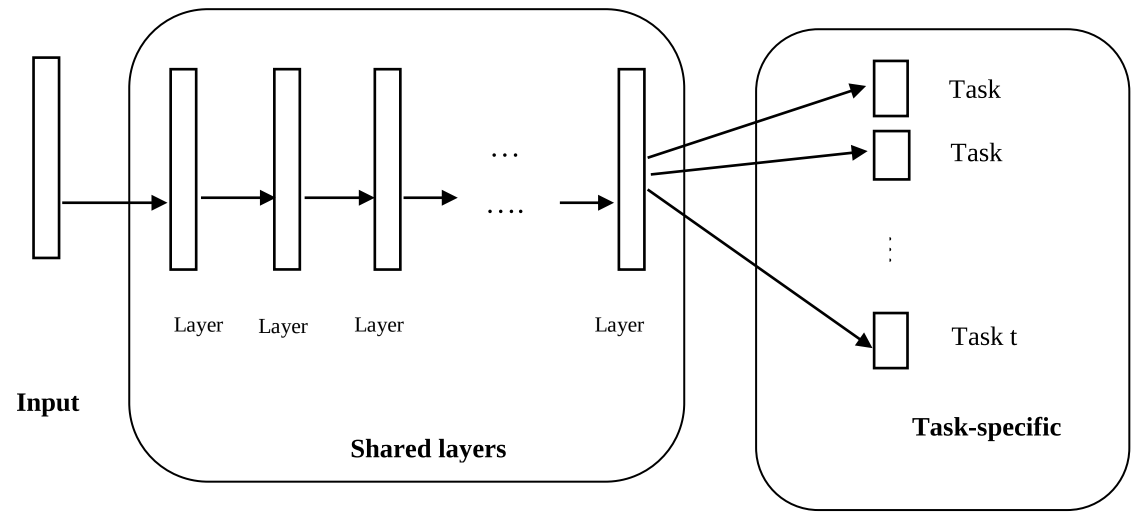


Figure 3.6: Hard Parameter Sharing Model

3.4 Conceptual Model

The Machine Learning workflow was followed by to develop the proposed model. The collected data was prepared, trained, and validated using the built Multitask Learning-Convolutional Neural Networks model, which was then evaluated as shown in

Images of diseased maize leaves were gathered from different online sources and combined to make one dataset. The collected data were then split into training and test set. The next step was the data was augmented using offline data augmentation. Then the data was normalized and rescaled.

This research aims at solving two tasks through a single model that is classification of maize disease and pathogen. Then, building the model consists of choosing the algorithm such as Convolutional Neural Networks and Multitask Learning, create layers and apply loss functions and create the optimizer.

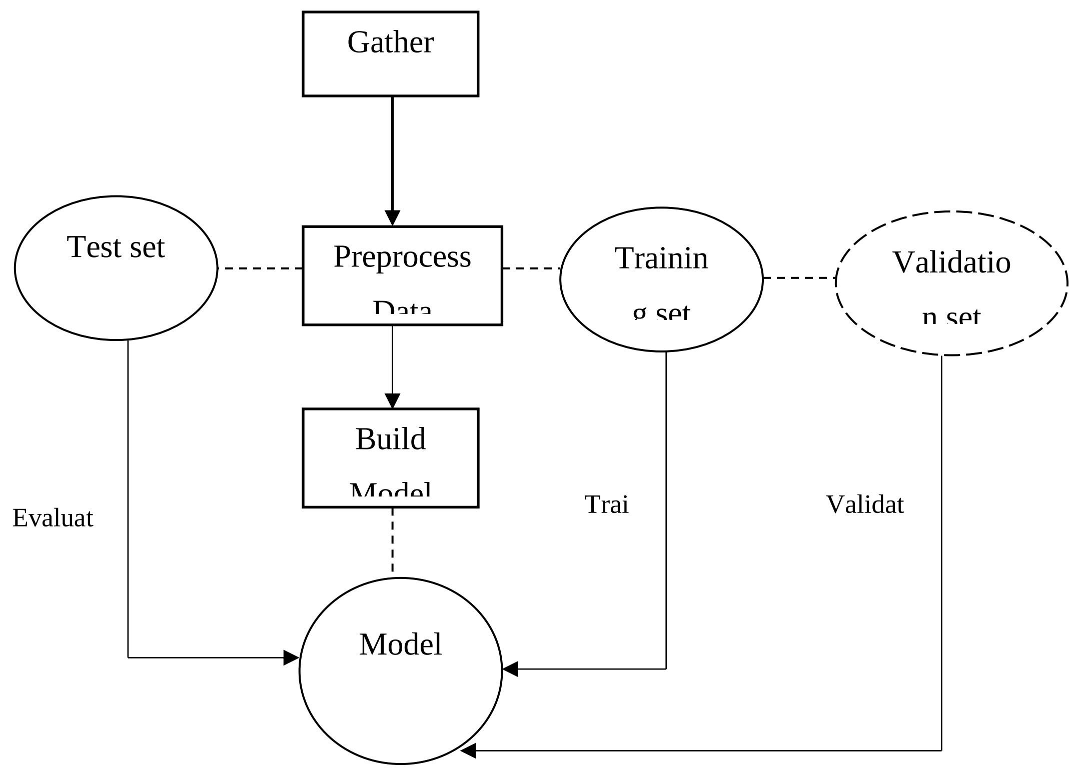


Figure 3.7: ML Model Workflow

The model was trained using eighty percent of the dataset and validated with twenty percent of the training set. Finally, the model was evaluated with previously unseen data (test set), twenty percent of the dataset.

There exist two Multitask Learning architectures (hard parameter sharing and soft parameter sharing), whereas we aimed to take advantage of sharing parameters, parameters are not shared between task models in soft parameter sharing (Crawshaw, 2020). We then proposed a Hard parameter sharing method divided into shared and task-specific layers. Feature extraction is done at the shared layers' level, while classification is done at task-specific layer. Given an input image, the model classifies it into two related tasks: the maize disease and its type. Figure shows the proposed model. The Input layer of an image is about diseased maize in RGB color with 224X224 of size, that is (224, 224, 3). In CNN's models, the feature extraction is done automatically (Navamani, 2019) at the shared layers level. We have six convolutional layers, five pooling layers, and three dense layers. Parameters are learned at convolutional and dense layers only. To calculate the number of learnable parameters, we consider the input of the convolutional layer from the previous layer, the kernel size, the number of filters, the stride, and the bias. The number of nodes from the previous layer and the bias are considered for the dense layer. The maize disease name corresponds to multiclass classification with softmax and categorical cross-entropy for activation and loss functions at task-specific layers. For the disease type, we have a binary classification task with sigmoid and binary cross-entropy for activation and loss functions, respectively. Adam optimizer is used for all the experiments with a learning rate of 0.0001 with the Accuracy Metrics.

The other objective of this Thesis is also to improve the model performance using different combined regularization methods. The first regularization used is Multitask Learning itself, as it is also the best algorithm to use in the case of training multiple related tasks. Because the dataset is not big enough, we use transfer learning as a regularization method by taking advantage of reusing a pre-trained model for the feature extraction. Data augmentation is also a recommended method in the case of small datasets; however, in this work, it has been used to solve an imbalanced data problem and cannot be reused on the same dataset. We also utilize Early stopping, which aids in the termination of model training before overfitting occurs.

Building a model in machine learning, specifically convolutional neural networks, follows well-defined steps, such as data collection, pre-processing the data, building the multi-task neural networks-convolutional learning model, training model, then evaluating the model and testing the model on new data. Because mobile smartphones are increasingly popular among Kenyans (X. Li et al., 2020), the model could be implemented through a mobile application.

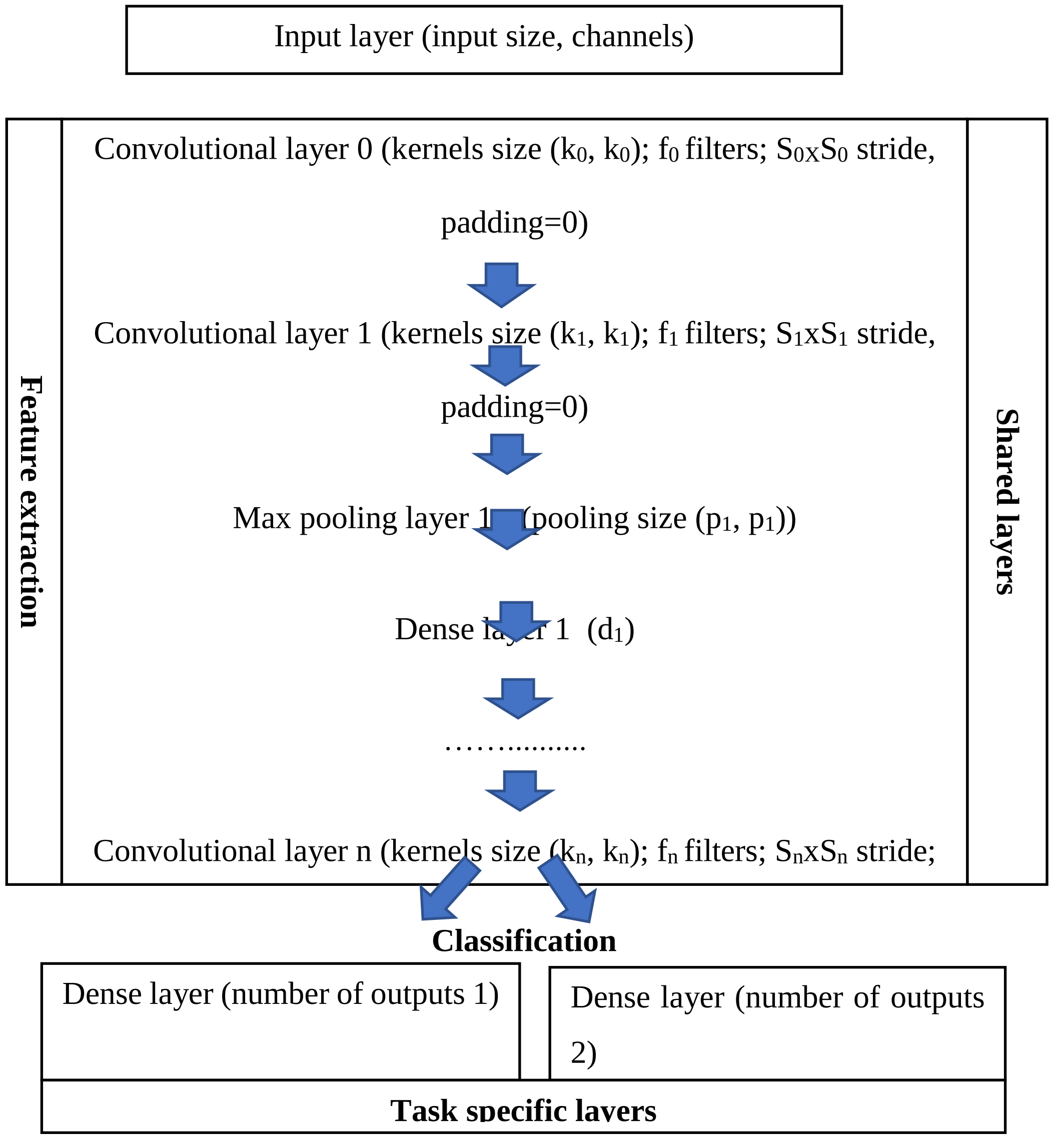


Figure 3.8: The Proposed CNN-MTL Model

3.5 Implementation

All experiments were carried out in Google Colaboratory (Colab), a Jupiter notebook environment developed by Google Brain that runs in the cloud. We used the Keras Functional API, a Tensorflow API that allows us to create complex models with flexibility, such as sharing layers, which is critical for multitask learning algorithms, specifically for the hard parameter sharing proposed in this study. Indeed, this technique allows you to create multiple models in one model at the same time, with sharing layers and task-specific layers.

Google Colab also includes a free Graphics Processing Unit (GPU) to speed up processing. Colab also provides free Random Access Memory (RAM) of 12GB, which is required to store input data, weight parameters, and activations as an input propagates through the network. Activations from a forward pass must be saved in training until they can be used to calculate error gradients in the backwards pass. Google Drive was used to save all of the files.

Python is the programming language used in this study to implement the proposed model. Python is a scripting language that is high-level, interpreted, interactive, and object-oriented. Python is intended to be highly readable. Python is widely used in a variety of fields, including machine learning, web development, and software development. It is also used for data analytics.

CHAPTER FOUR

RESEARCH RESULTS AND DISCUSSIONS

4.1 Introduction

To achieve our main goal of developing a maize disease identification model based multitask Learning-Convolutional Neural networks, we conducted various experiments ranging from data processing to model building, training, and testing, as detailed in this chapter. In addition, the results were thoroughly discussed.

4.2 Experimental Setup

All the experiments were executed in Google Colaboratory, which provides a GPU accelerator and the Tensorflow platform with Keras (version 2.6) Application Programming Interface (API). We used Keras functional API, which can handle complex models with non-linear topology, shared layers, and even multiple inputs or/and outputs. Python (version 3.7) was the Programming language used for all the experiments.

4.3 Datasets

The original dataset is imbalanced as shown by, which means that the classes are not well distributed. Indeed, imbalanced data is defined as a situation in which the number of samples assigned to each class differ widely. Imbalanced data has both majority and minority classes. As shown in Figure, the majority class is North leaf blight- fungus with 988 samples and the minority class is leaf streak- virus with only 77 samples.

The dataset is divided into 80% and 20% training and test sets, respectively. Figure shows that the original dataset is imbalanced, indicating that the classes are not evenly distributed. In fact, imbalanced data is defined as a situation in which the number of samples assigned to each class varies significantly. Data that is imbalanced includes both majority and minority classes. The extreme majority class in our case is North leaf blight-fungus, which has 988 samples, and the extreme minority class is leaf streak-virus, which has only 77 samples.

Most machine learning techniques assume that data is evenly distributed. When there is a class imbalance, the machine learning classifier tends to be more biased towards the majority class, resulting in incorrect classification of the minority class.

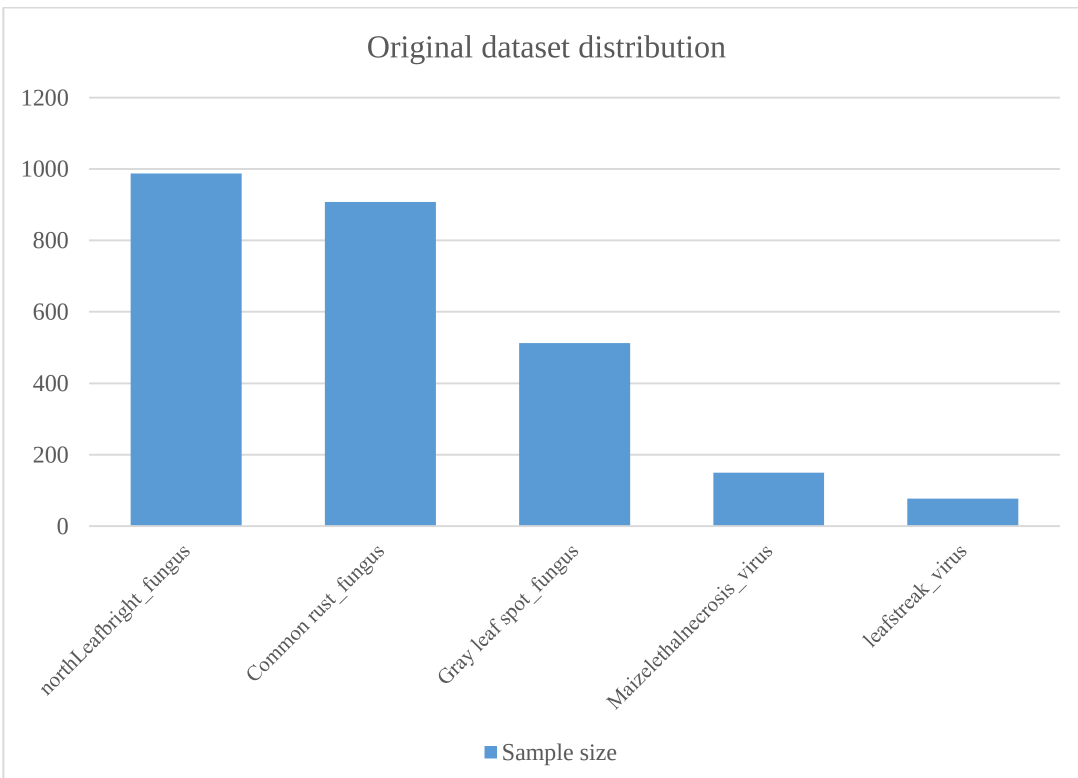


Figure 4.1: Imbalanced Dataset

To solve the problem of imbalanced data, we used Oversampling, a technique for adjusting the class distribution of a dataset. We used the Offline data augmentation technique on the minor classes "Gray leaf spot-fungus," "maize fatal necrosis-virus," and "leaf streak-virus" after splitting the data into training and test sets. Techniques for artificially increasing the amount of data by generating new modified copies of current data are referred to as data augmentation.

In this research, Rotation, Width shift, Height range, Shear, Zooming, and Horizontal flip of the images were used (See Appendix 2 for code snippet used for offline data augmentation). Figure shows the balanced data after oversampling, consisting of 5077 images. Next, the images were resized to (224, 224, 3) and the labels encoded to one hot-encoding after the feature being scaled.

After data processing, building, training and testing the models are the next steps.

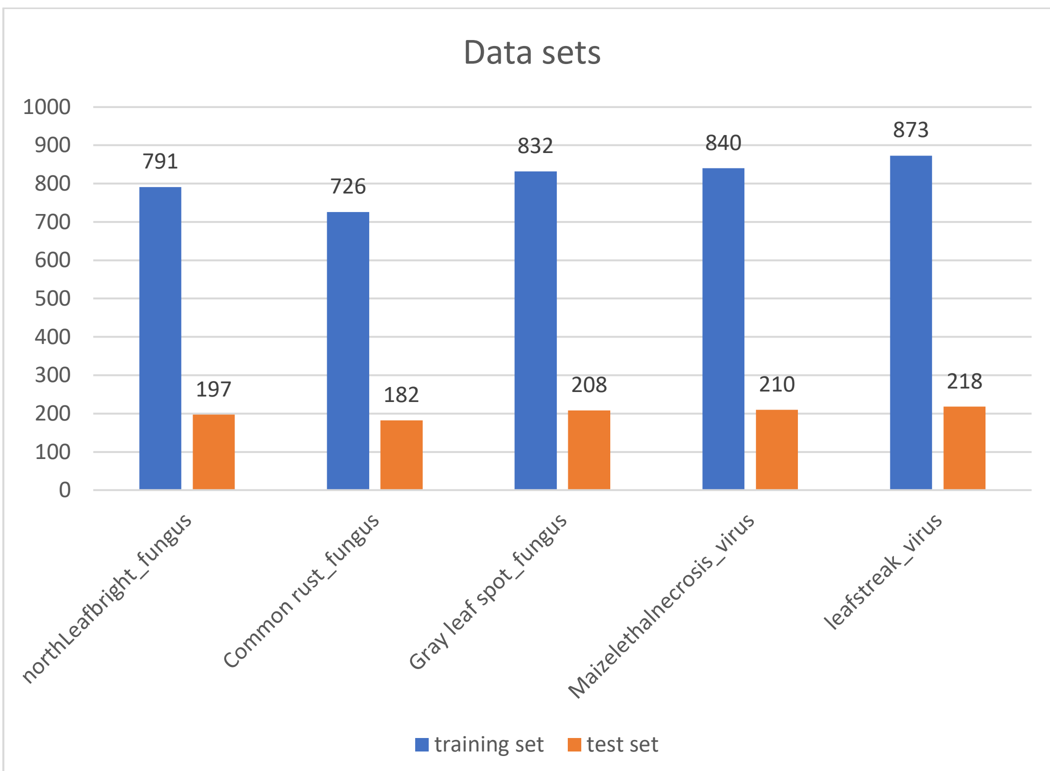


Figure 4.2: Data Sets after Oversampling

4.4 Training Using No Regularization Technique

Our baseline consists of building disease name and disease type (pathogens) classification models without using any regularization techniques. The disease name classification model's architecture consisted of an input layer with an input shape of (224,224,3), six convolutional layers, five pooling layers, four dense layers, and a flatten layer. Except for the output layer, where the softmax activation function was utilized, we used Rectified Linear Unit (ReLU) as the activation function.

The architecture of the pathogen classification model included an input layer with the input shape (224,224,3), six convolutional layers, five pooling layers, four dense layers, and a flatten layer. Except for the output layer, where the sigmoid activation function was utilized, we used the Rectified Linear Unit (ReLU) as the activation function.

The pooling size for all experiments was (2,2), and the Adam optimizer with a learning rate of 0.0001 was utilized. Accuracy was the performance metric (See Appendix 4 for the corresponding snippet). The validation set was made up of 20% of the training set. We utilized a batch size of 32 and 200 epochs. We trained the two models separately. Figure and Figure demonstrate the two model architectures.

The disease name classification model's training accuracy reached 98%, however its validation accuracy did not improve in anyway. Meanwhile, as seen in Figure, the "disease name" classification model was overfitting. The disease type classification model's training and validation accuracy curves followed the same proportional pattern. Furthermore, the training and validation losses exhibit similar trends (see Figure).

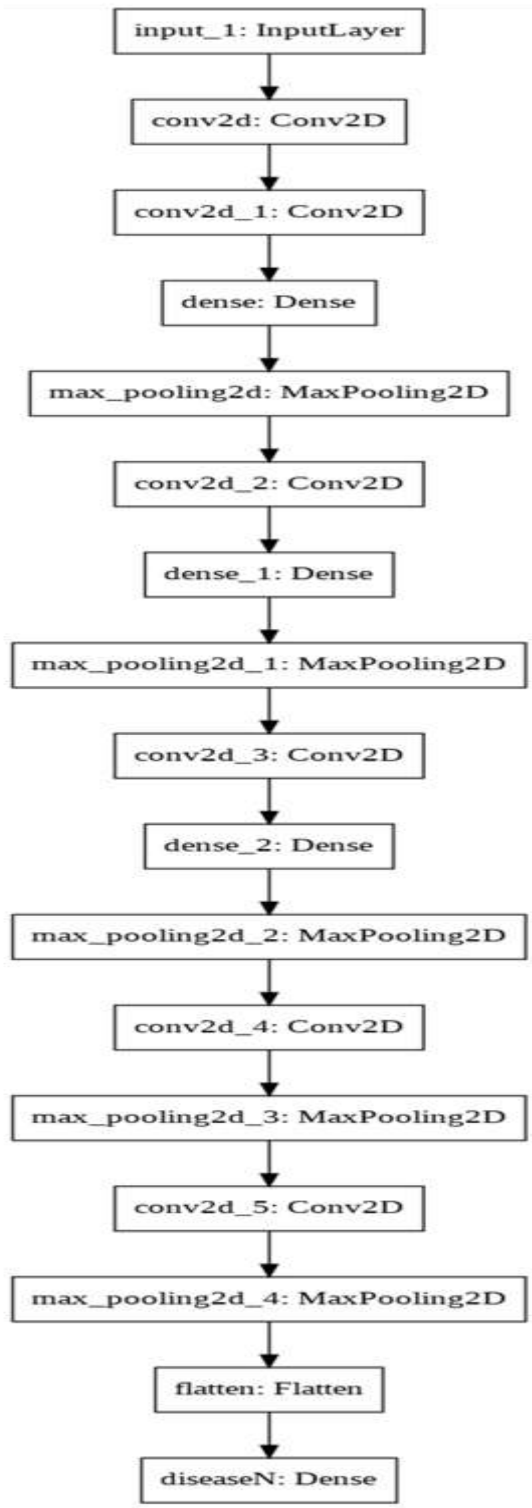


Figure 4.3: Disease Name Classification Model's Architecture

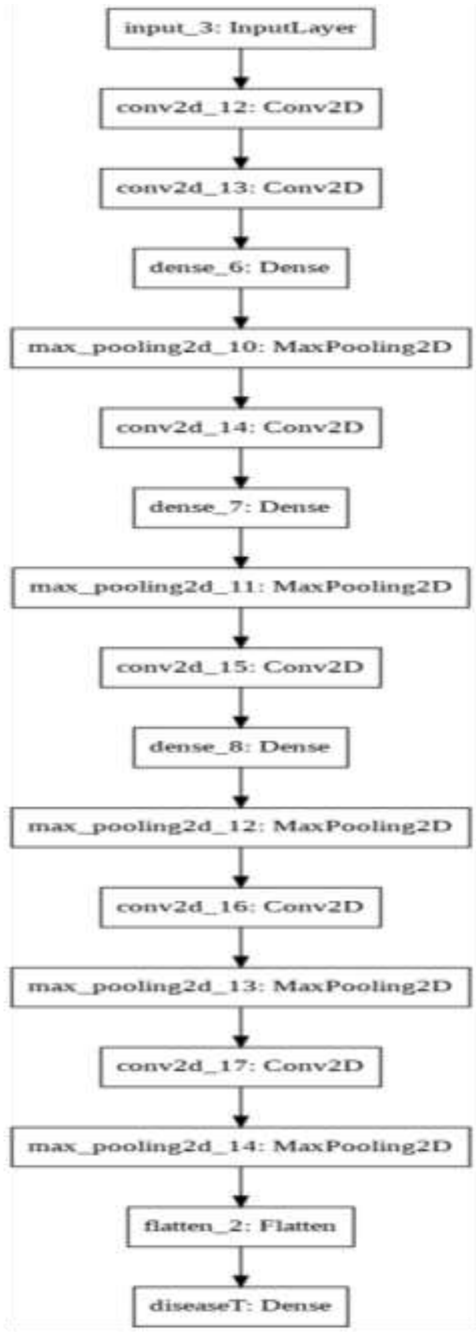
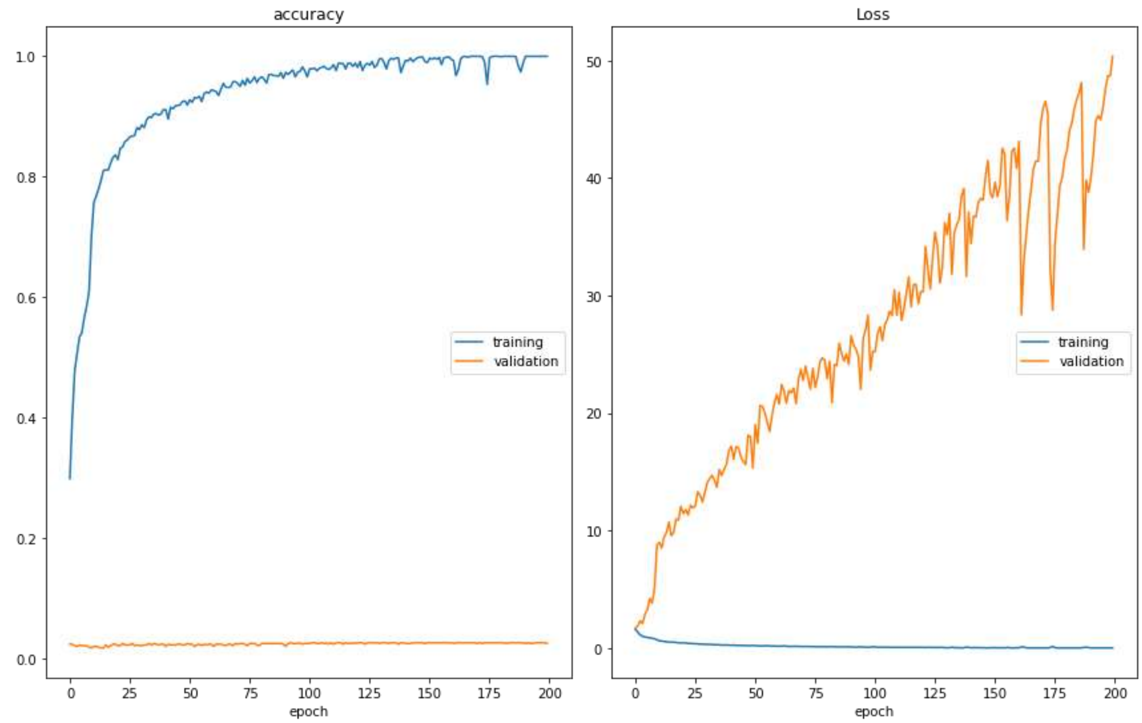


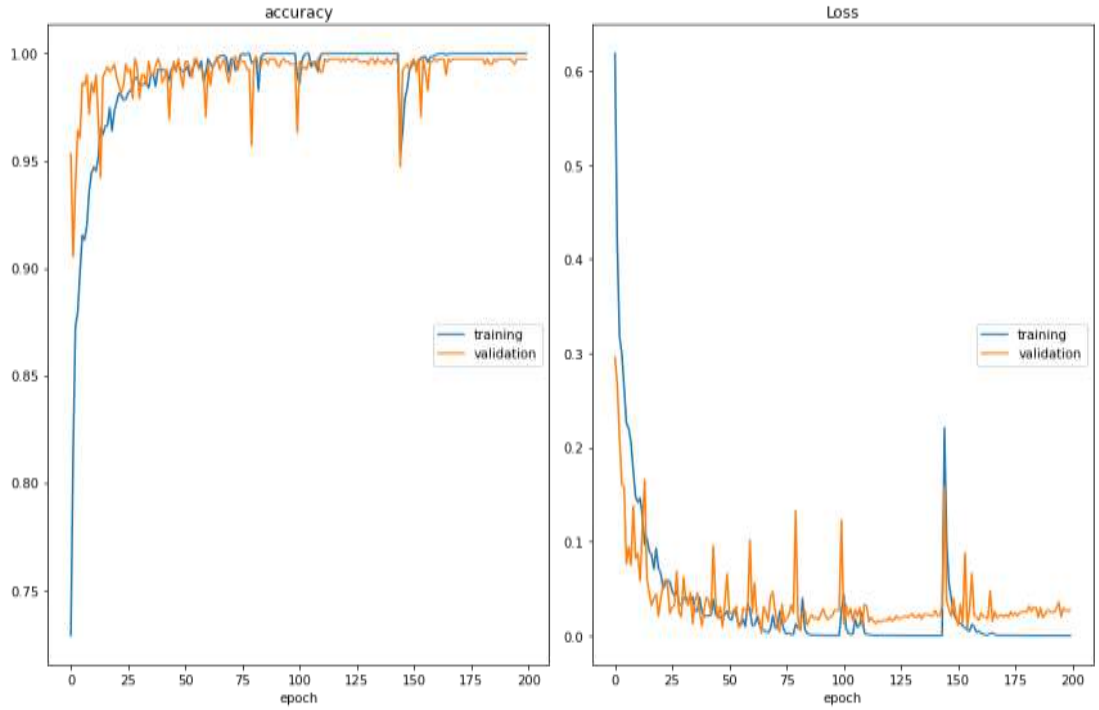
Figure 4.4: Disease Name (Pathogen) Classification Model Architecture

The test loss is 7.5268 and Test accuracy is 61.08% for the disease name classification whereas for the disease type classification, the test loss was 0.0789 and the test accuracy was 98.62%.



(a) (b)

Figure 4.5: (a) Disease Name Classification Model's Training Accuracy; (b) Disease Name Classification Model's Training Loss



(a) (b)

Figure 4.6: (a) Disease Type Classification Model's Training Accuracy; (b) Disease Type Classification Model's Training Loss

4.5 Training Using Multi Task Learning Technique

The next experiment was to create a Multitask Learning model by combining the two models, as shown in

Figure. The training and validation accuracies for disease classification were increasing concurrently, which is a promising sign of overfitting reduction. Figure shows the similar scenario for their losses, which were reducing at the same period. The disease name's test accuracy was enhanced from 60.89% to 74.48%. Yet, there was no noticeable change in the disease type's training, validation, or testing data from the previous experiment (see Figure 4.9)

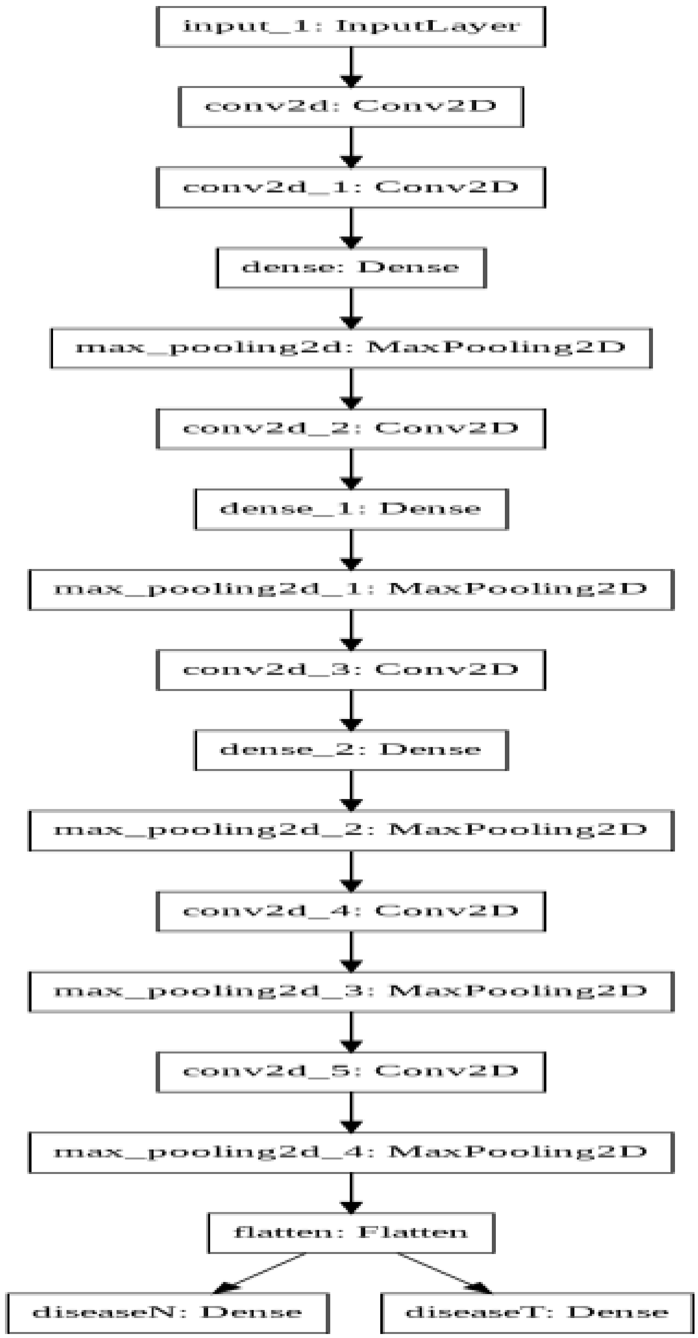


Figure 4.7: The MTL Model's Architecture for Maize Disease Classification (See Appendix 3 for the Corresponding Code Snippet)

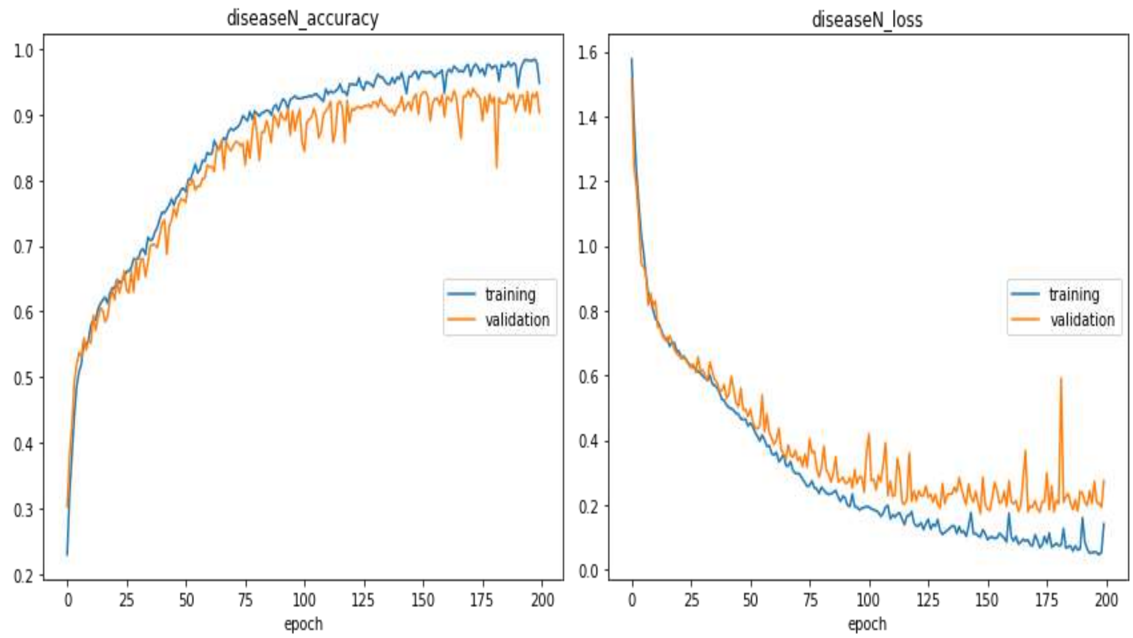


Figure 4.8: Training and Validation Accuracies and Losses for Disease Name (DiseaseN) Classification Model Using MTL

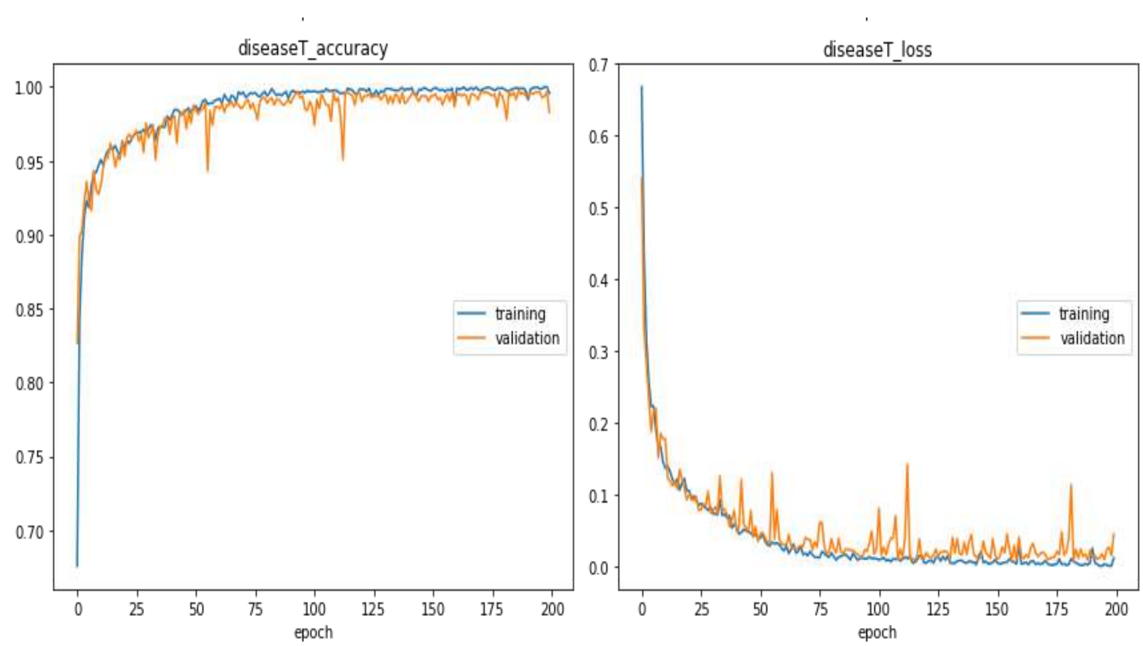


Figure 4.9: Training and Validation Accuracies and Losses for Disease Type (DiseaseT) Classification Model Using MTL

4.6 Training using Multi Task Learning and Early Stopping Combined

The third experiment involved adding an early stopping regularization technique to the MTL model with a patience of 10. Early stopping helps at ending the training process at the optimal point of the model performance which helps at reducing the overfitting and the training time (See Appendix 5 for the corresponding snippet). In our case, the optimal point was reached after 138 epochs, whereas the total number of epochs was 200. And as illustrated by Figure, the training and validation curves were increasing almost in parallel, indication that overfitting was reduced. The test accuracy for the disease name classification model rose to 77.44%.

There was no discernible difference in the disease type's training, validation, or testing accuracy and loss from the previous experiment as shown by Figure.

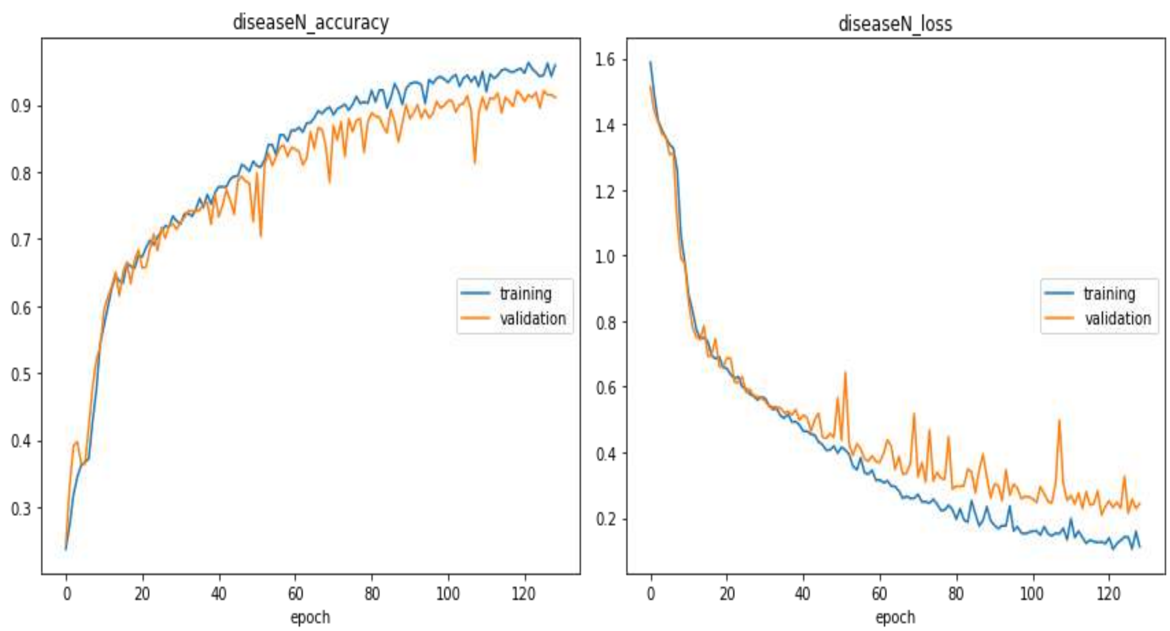


Figure 4.10: Training and Validation Accuracies and Losses for Disease Name (DiseaseN) Classification Models Using MTL and Early Stopping

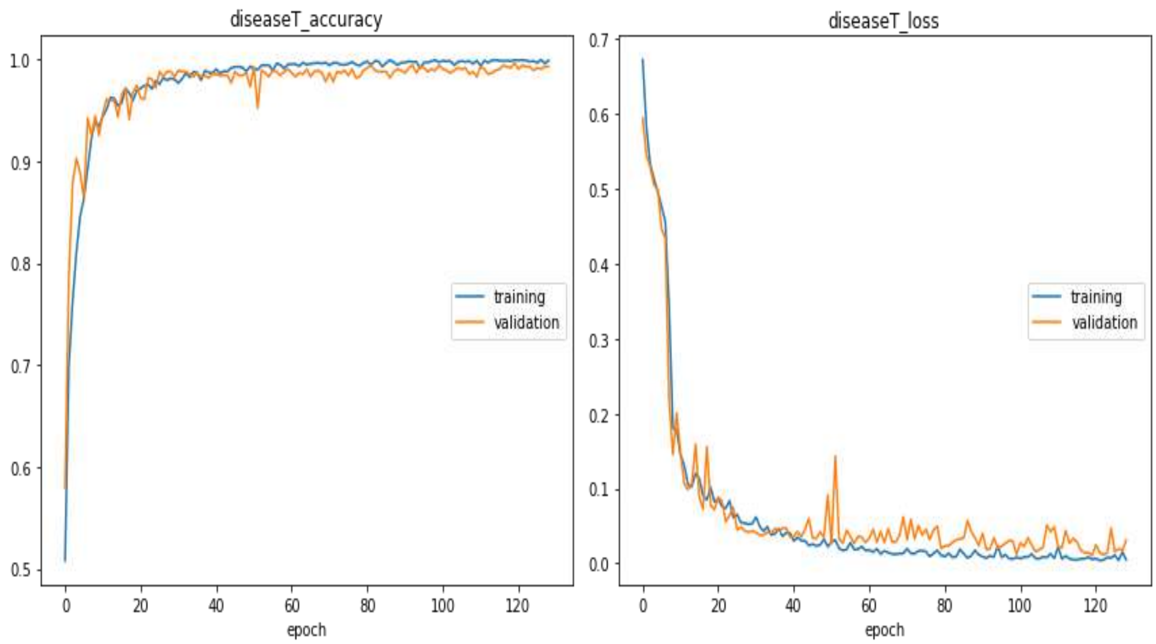


Figure 4.11: Training and Validation Accuracies and Losses for Disease Type (DiseaseT) Classification Model Using MTL and Early Stopping

4.7 Training Using Multi Task Learning, Early Stopping and Transfer Learning Combined

The fourth and final experiment was to use a pre-trained Resnet50 model on Imagenet for the feature extraction as our dataset was not big enough. The pre-trained model is connected to our model by a Global pooling layer for shared layers. Then the output of the global pooling layer is connected to the specific task layers, which are about two output layers, one is a softmax for the five diseases and another is sigmoid for the two types of diseases to be classified as in the above experiments (See Appendix 6 for the corresponding code snippet). The optimal point was at only 30 epochs, as illustrated by Figure. The test accuracy jumped to 85.22% for the disease name classification model and 97.93% for the disease type classification model (see Figure).

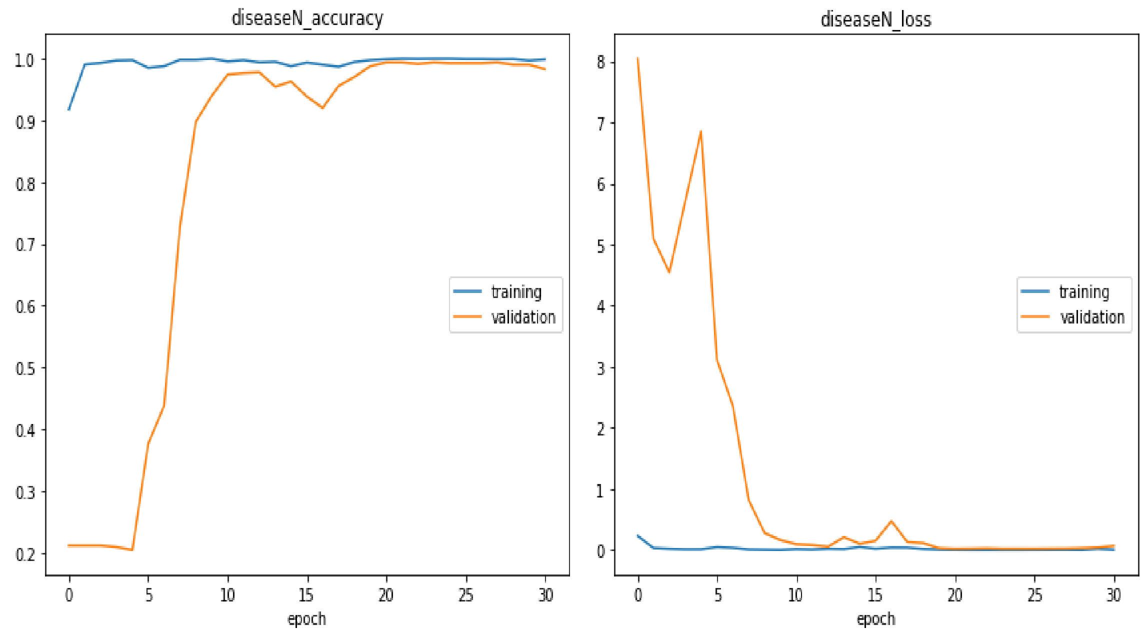


Figure 4.12: Training And Validation Accuracies and Losses for Disease Name (DiseaseN) Classification Models Using MTL, Early Stopping and Transfer Learning Techniques Combined

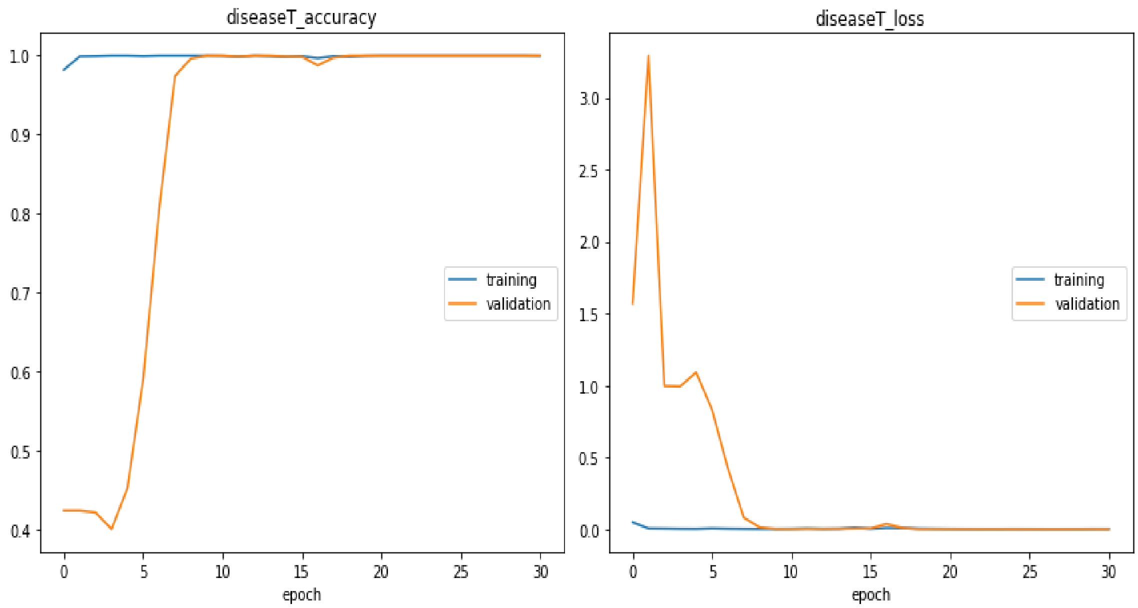


Figure 4.13: Training and Validation Accuracies and Losses for Disease Type or Pathogen (DiseaseT) Classification Models Using MTL, Early Stopping and Transfer Learning Techniques Combined

4.8 Summary of the Test Accuracies Results

Figure summarizes the test accuracies of all the experiments. The overfitting disease name classification model demonstrates constant variation. With no regularization methods, the model's test accuracy was 60.89%. When the MTL method was used, it increased to 74.48%. The test accuracy rose to 85.22% by combining MTL with early stopping and transfer learning in the third experiment. The test accuracy for the disease type classification model remained practically constant.

Indeed, one of the indications that overfitting had been minimized was an increase in test accuracy. MTL improved overall generalization by leveraging information included in training images of related tasks. The information learnt from related tasks improved the model's capacity to learn a meaningful representation of the data when using MTL, which decreased overfitting and improved generalization.

While overfitting occurs during the training process, the early stopping technique ends the process at the optimal point. allows you to provide an arbitrary large number of training epochs and to stop training whenever the model's performance on the validation dataset stops improving. The training error falls exponentially until the influence of increasing epochs on the error is no longer significant. The validation error, on the other hand, decreases initially with rising epochs before increasing at a certain point. This is the moment at which a model should be stopped because it will begin to overfit after this point.

Finally, using a large amount of data may aid in preventing overfitting. However, in practice, gathering enough data is challenging. In this study, using a pre-trained model for feature extraction allowed us to take advantage of features learned by a model trained on a larger dataset, ImageNet. This is done by instantiating the pre-trained model and adding a fully-connected classifier on top. The pre-trained model is frozen and only the classifier weights were modified during training. In this approach, the convolutional base collected all of the features associated with each image, and it is the trained classifier that decides the image class given the extracted features, which minimize overfitting. This technique is called transfer learning and in this experiment, ResNet50 was used on ImageNet dataset.

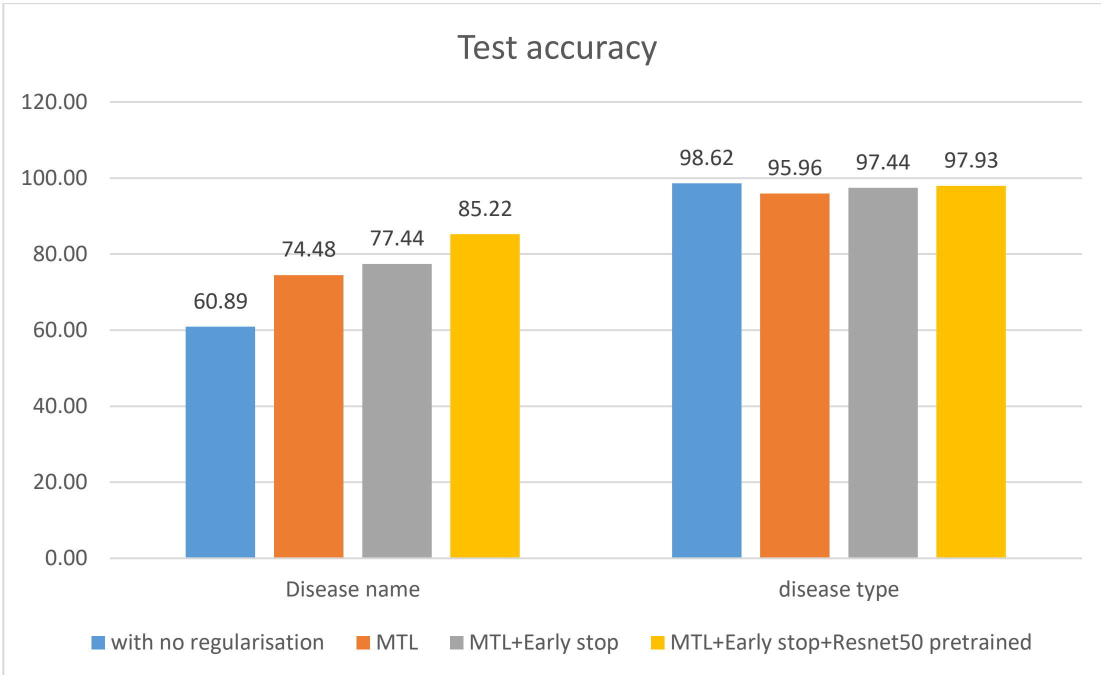


Figure 4.14: Summary of the Experimental Results of Test Accuracies

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORKS

5.1 Conclusion

This work proposes a regularized Multitask Neural Networks Convolutional Learning model by combining MTL methods with other regularization methods.

First, we build two models, one of which is overfitting as our baseline. We then construct an MTL model based on the two models, which increases the test accuracy of the overfitting model. In the subsequent experiments, we combine MTL with Early stopping and Transfer Learning, which increases the accuracy. MTL helped us build a multiclass model and a binary classification model in one model, simultaneously identifying maize disease and its pathogen. In hard parameter sharing, feature extraction is done at shared layers, which can help reduce overfitting. We realized that a good technique of improving a model performance is to combine different proper regularization methods. In our case, combining MTL, early stopping, and Transfer learning gives us better results. The lack of sufficient data may explain it.

As mentioned above, maize is an important cereal for Kenyan people, and its production continues to decrease because of diseases. Furthermore, some diseases spread rapidly and must be fixed quickly as possible. Unfortunately, finding experts in the domain is still not easy as they are few and not always available.

According to Ezinne et al., mobile smartphones are becoming increasingly popular among Kenyans; this model used through a mobile application would help maize farmers identify the disease and pathogen themselves earlier, which will help them at reducing costs and save time. Also, fighting the maize disease would increase maize production.

5.2 Recommendation for Future Work

While a significant amount of data is required for the best performance of a Deep Neural Networks, finding large labelled datasets remains a challenge in the Agriculture field. Following that, future research would focus on training a combination of different regularization methods on more extensive and more diverse datasets with more tasks. Furthermore, while maize is consumed in many countries, our application was limited to Kenya's most common maize diseases. As a result, future work would consider including more diseases for more users.

Finally, as one maize may suffer from more than one disease and pathogen, future research would work on maize images with multiple diseases and pathogens.

REFERENCE

Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. *1*, 2–8. http://arxiv.org/abs/1803.08375

Aleshin-Guendel, S., & Alvarez, S. (2017). *Examining the Structure of Convolutional Neural Networks*. Boston College.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, *8*(3), 1–67. https://doi.org/10.3390/electronics8030292

Amatya, S., Karkee, M., Gongal, A., Zhang, Q., & Whiting, M. D. (2016). Detection of cherry tree branches with full foliage in planar architecture for automated sweet-cherry harvesting. *Biosystems Engineering*, *146*, 3–15. https://doi.org/10.1016/j.biosystemseng.2015.10.003

Bedi, P., & Gole, P. (2021). Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network. *Artificial Intelligence in Agriculture*, *5*, 90–101. https://doi.org/10.1016/j.aiia.2021.05.002

Caruana, R., Mitchell, T., Pomerleau, D., Dietterich, T., & Simon, H. (1997). *Multitask Learning* (Issue September). Carnegie Mellon University.

Cedric, L. S., Adoni, W. Y. H., Aworka, R., Zoueu, J. T., Mutombo, F. K., Krichen, M., & Kimpolo, C. L. M. (2022). Crops yield prediction based on machine learning models: Case of West African countries. *Smart Agricultural Technology*, *2*(March), 100049. https://doi.org/10.1016/j.atech.2022.100049

Charles, A. K., Muiru, W. M., Miano, D. W., & Kimenju, J. W. (2019). Distribution of Common Maize Diseases and Molecular Characterization of Maize Streak Virus in

Kenya. *Journal of Agricultural Science*, *11*(4), 47. https://doi.org/10.5539/jas.v11n4p47

Cosentino, R., Balestriero, R., Baraniuk, R., & Aazhang, B. (2021). Deep Autoencoders: From Understanding to Generalization Guarantees. *Proceedings OfMachine Learning Research*, *107*, 1–26. http://arxiv.org/abs/2009.09525

Courville, A. Goodfellow, I., & Bengio, Y., (2016). Deep Learning.[online] Deeplearningbook. org.

Crawshaw, M. (2020). Multi-Task Learning with Deep Neural Networks: A Survey. *ArXiv*. http://arxiv.org/abs/2009.09796

Koech, K. E. (2020). Cross-Entropy Loss Function. A loss function used in most… | by Kiprono Elijah Koech | Towards Data Science. Towards Data Science. https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec864

Czum, J. M. (2020). Dive Into Deep Learning. *Journal of the American College of Radiology*, *17*(5), 637–638. https://doi.org/10.1016/j.jacr.2020.02.005

Da Rocha, E. L., Rodrigues, L., & Mari, J. F. (2021). Maize leaf disease classification using convolutional neural networks and hyperparameter optimization. *WVC 2020*, *October*, 104–110. https://doi.org/10.5753/wvc.2020.13489

DeChant, C., Wiesner-Hanks, T., Chen, S., Stewart, E. L., Yosinski, J., Gore, M. A., Nelson, R. J., & Lipson, H. (2017). Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning. *Phytopathology*, *107*(11), 1426–1432. https://doi.org/10.1094/PHYTO-11-16-0417-R

Dienya, T. (2020). *Kenya Maize Production By Counties - Dataset - Kilimo Open Data*. http://kilimodata.developlocal.org/dataset/kenya-maize-production-by-counties

Dufourq, E., & Bassett, B. A. (2017). Automated problem identification: Regression vs classification via evolutionary deep networks. *ArXiv*.

Eerens, H., Haesen, D., Rembold, F., Urbano, F., Tote, C., & Bydekerke, L. (2014). Image time series processing for agriculture monitoring. *Environmental Modelling and Software*, *53*, 154–162. https://doi.org/10.1016/j.envsoft.2013.10.021

Elliott, M. L., & Harmon, P. F. (2011). Gray Leaf Spot. *Edis*, *2011*(2), 1–4. https://doi.org/10.32473/edis-lh047-2011

Fabio, D. N., Francesco, G., Quoc, B. P., & Giovanni, de M. (2022). precipitation prediction.pdf. *Sustainability (Switzerland)*. https://doi.org/10.3390/su4052663

Feng, J., & Lu, S. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, *1237*(2), 111–122. https://doi.org/10.1088/1742-6596/1237/2/022030

Ferreira, A., & Giraldi, G. (2017). Convolutional Neural Network approaches to granite tiles classification. *Expert Systems with Applications*, *84*(September), 1–11. https://doi.org/10.1016/j.eswa.2017.04.053

Gao, H. (Cornell U., Zhuang, L. (Tsinghua U., & Laurens van der, M. (Facebook A. R. (2018). Densely Connected Convolutional Networks. *ArXiv*.

Han, J., Zhang, Z., Cao, J., Luo, Y., Zhang, L., Li, Z., & Zhang, J. (2020). Prediction of winter wheat yield based on multi-source data and machine learning in China. *Remote Sensing*, *12*(2). https://doi.org/10.3390/rs12020236

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv*, 1–18. http://arxiv.org/abs/1207.0580

Hinton, G., Nitish, S., Alex, K., Ilya, S., & Ruslan, S. (2014). Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning*

Research, *15*, 1929–1958.

Hinton, G., Sabour, S., & Frosst, N. (2018). Matrix capsules with EM routing. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 1–15.

Ian, G., Yoshua, B., & Aaron, C. (2016). *Deep Learning*. MIT Press. https://www.deeplearningbook.org/

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*, 70–90. https://doi.org/10.1016/j.compag.2018.02.016

Karlsruhe, U. (2015). *1 Early stopping but when? March 2000*. https://doi.org/10.1007/3-540-49430-8

Khaki, S., & Wang, L. (2019). Crop yield prediction using deep neural networks. *Frontiers in Plant Science*, *10*(May), 1–10. https://doi.org/10.3389/fpls.2019.00621

Khatib, J., Dalam, S., Satria, B., Sidauruk, A., Wardhana, R., Akbar, A. Al, Ihsan, A., Gama, A. M., Yogyakarta, U. A., Bengkulu, U. D., Selatan, P. A., & Kunci, K. (2022). *Indonesian Journal of Computer Science*. *11*(1), 566–576.

Kirori, Z., & Ireri, E. (2020). http://www.ijssit.com. *Ijssit*, *V*(VIII), 157–166.

Kuang, Z., Li, Z., Zhao, T., & Fan, J. (2017). Deep Multi-task Learning for Large-Scale Image Classification. *Proceedings - 2017 IEEE 3rd International Conference on Multimedia Big Data, BigMM 2017*, 310–317. https://doi.org/10.1109/BigMM.2017.72

Kukačka, J., Golkov, V., & Cremers, D. (2018). Regularization for Deep Learning: A Taxonomy. *ICLR*, 1–24. http://arxiv.org/abs/1710.10686

Kung, H. Y., Kuo, T. H., Chen, C. H., & Tsai, P. Y. (2016). Accuracy analysis mechanism for agriculture data using the ensemble neural network method. *Sustainability (Switzerland)*, *8*(8), 1–11. https://doi.org/10.3390/su8080735

Lamp'l, J. (2013). *Bacteria, Fungus, and Viruses, an Overview - Growing A Greener World®*. https://www.growingagreenerworld.com/bacteria-fungus-and-viruses-an-overview/

Larsson, G., Maire, M., & Shakhnarovich, G. (2017). *Fractalnet: Ultra-deep Neural Netwprks witout residuals*. 1–11.

Li, X., Grandvalet, Y., & Davoine, F. (2020). A baseline regularization scheme for transfer learning with convolutional neural networks. *Pattern Recognition*, *98*. https://doi.org/10.1016/j.patcog.2019.107049

Li, Z., Gong, B., & Yang, T. (2016). Improved dropout for shallow and deep learning. *Advances in Neural Information Processing Systems*, *Nips*, 2531–2539.

Liu, B., Zhang, Y., He, D. J., & Li, Y. (2018). Identification of apple leaf diseases based on deep convolutional neural networks. *Symmetry*, *10*(1). https://doi.org/10.3390/sym10010011

Lottes, P., Behley, J., Milioto, A., & Stachniss, C. (2018). Fully convolutional networks with sequential information for robust crop and weed detection in precision farming. *IEEE Robotics and Automation Letters*, *3*(4), 2870–2877. https://doi.org/10.1109/LRA.2018.2846289

Malinowski, E., Zimányi, E., Joseph, S. K., Warehouse, D., Inmon, B., Analytical, O., Olap, P., Gatziu, S., Vavouras, A., Nilsson, A. A., & Merkle, D. (2019). About the Tutorial Copyright & Disclaimer. *Data Vault 2.0, January 1999*, 1–15. https://doi.org/10.1007/978-3-322-94873-1

Mikołajczyk, A., & Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem. *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*, *May*, 117–122. https://doi.org/10.1109/IIPHDW.2018.8388338

Murphy, J. (2016). *An Overview of Convolutional Neural Network Architectures for Deep Learning*. 1–22.

Namatēvs, I. (2018). Deep Convolutional Neural Networks: Structure, Feature Extraction and Training. *Information Technology and Management Science*, *20*(1), 40–47. https://doi.org/10.1515/itms-2017-0007

Navamani, T. M. (2019). Efficient Deep Learning Approaches for Health Informatics. *Deep Learning and Parallel Computing Environment for Bioengineering Systems*, 123–137. https://doi.org/10.1016/B978-0-12-816718-2.00014-2

Nelken, R., & Shieber, S. M. (2006). Computing The Kullback-Leibler Divergence Between Probabilistic Automata Using Rational Kernels. *Applied Sciences*, 15.

Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *ArXiv*, 1–20. http://arxiv.org/abs/1811.03378

Osunga, M., Mutua, F.N., & Mugo, R. (2017). *Spatial Modelling of Maize Lethal Necrosis Disease in Bomet County, Kenya*. https://api.semanticscholar.org/CorpusID:134784404

Picking Loss Functions - A comparison between MSE, Cross Entropy, and Hinge Loss – Rohan Varma – machine learning, math, and other random thoughts. (n.d.). Retrieved January 9, 2021, from https://rohanvarma.me/Loss-Functions/

Russell, S., & Norvig, P. (2010). Artificial Intelligence A Modern Approach Third Edition. In *Pearson*. https://doi.org/10.1017/S0269888900007724

Salehinejad, H., Sankar, S., Barfett, J., Colak, E., & Valaee, S. (2017). Recent Advances in Recurrent Neural Networks. *Researchgate*, *March*. http://arxiv.org/abs/1801.01078

Sansao, J. P. H., Silva, M. C., Mozelli, L. A., Pinto, F. A. C., & Queiroz, D. M. (2012). Weed Mapping Using Digital Images. *International Conference of Agricultural Engineering CIGR-AgEng2012*, *i*.

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3). https://doi.org/10.1007/s42979-021-00592-x

Sartin, M. A., Da Silva, A. C. R., & Kappes, C. (2014). Image segmentation with artificial neural network for nutrient deficiency in cotton crop. *Journal of Computer Science*, *10*(6), 1084–1093. https://doi.org/10.3844/jcssp.2014.1084.1093

Series, I. (2021). Machine Learning Algorithms and Applications. In *Machine Learning Algorithms and Applications* (Vol. 7). https://doi.org/10.1002/9781119769262

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0197-0

Shrivastava, V. K., Pradhan, M. K., Minz, S., & Thakur, M. P. (2019). Rice plant disease classification using transfer learning of deep convolution neural network. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *42*(3/W6), 631–635. https://doi.org/10.5194/isprs-archives-XLII-3-W6-631-2019

Singh, M. K., Baluja, P. G. S., & Sahu, D. P. (2017). Understanding the Convolutional Neural Network & it ’ s Research Aspects in Deep Learning. *International Journal for Research in Applied Science and Engineering Technology*, *5*(Vi), 867–871.

Stewart, E. L., Wiesner-Hanks, T., Kaczmar, N., DeChant, C., Wu, H., Lipson, H., Nelson, R. J., & Gore, M. A. (2019). Quantitative Phenotyping of Northern Leaf Blight in UAV Images Using Deep Learning. *Remote Sensing*, *11*(19), 1–10. https://doi.org/10.3390/rs11192209

Su, F., Shang, H. Y., & Wang, J. Y. (2019). Low-rank deep convolutional neural network for multi-task learning. *ArXiv*, *2019*.

Thi, T., Tran, K., Lee, T., Shin, J., & Kim, J. (2020). Deep Learning-Based Maximum Temperature Forecasting Assisted with Meta-Learning for. *Atmosphere*, *11*, 1–21.

Tonui, R., Masanga, J., Kasili, R., Runo, S., & Alakonya, A. (2020). Identification of maize lethal necrosis disease causal viruses in maize and suspected alternative hosts through small RNA profiling. *Journal of Phytopathology*, *168*(7–8), 439–450. https://doi.org/10.1111/jph.12908

Veenadhari, S., Bharat Mishra, D., & Singh, D. C. (2011). Soybean Productivity Modelling using Decision Tree Algorithms. *International Journal of Computer Applications*, *27*(7), 11–15. https://doi.org/10.5120/3314-4549

Wang, H., & Raj, B. (2017). *On the Origin of Deep Learning*. 1–72. http://arxiv.org/abs/1702.07800

Wang, Y., Li, Y., Song, Y., & Rong, X. (2020). The influence of the activation function in a convolution neural network model of facial expression recognition. *Applied Sciences (Switzerland)*, *10*(5). https://doi.org/10.3390/app10051897

Wise, K. (2011). Diseases of Corn: Northern Corn Leaf Blight. *Purdue Extension*, *6*, 1–

3.

Wu, Y., Li, D., Li, Z., & Yang, W. (2014). Fast processing of foreign fiber images by image blocking. *Information Processing in Agriculture*, *1*(1), 2–13. https://doi.org/10.1016/j.inpa.2013.05.001

Yan. (2015). *Common Rust of Corn*.

Yang, C.-K., Yeh, J. C.-Y., Yu, W.-H., Chien, L.-I., Lin, K.-H., Huang, W.-S., & Hsu, P.-K. (2019). Deep Convolutional Neural Network-Based Positron Emission Tomography Analysis Predicts Esophageal Cancer Outcome. *Journal of Clinical Medicine*, *8*(6), 844. https://doi.org/10.3390/jcm8060844

Yuming, H., Junhai, G., & Hua, Z. (2015). "Deep Belief Networks and deep learning,." *International Conference on Intelligent Computing and Internet of Things Proceedings*, 1–4. https://doi.org/10.1109/ICAIOT.2015.7111524.

Zeng, T., & Ji, S. (2016). Deep convolutional neural networks for multi-instance multi-task learning. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2016-Janua*(October), 579–588. https://doi.org/10.1109/ICDM.2015.92

Zhang, C., & Zhang, Z. (2014). Improving multiview face detection with multi-task deep convolutional neural networks. *2014 IEEE Winter Conference on Applications of Computer Vision, WACV 2014*, 1036–1041. https://doi.org/10.1109/WACV.2014.6835990

Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y., & Yu, B. (2019). Recent advances in convolutional neural network acceleration. *Neurocomputing*, *323*, 37–51. https://doi.org/10.1016/j.neucom.2018.09.038

Zhang, Y. D., Jiang, X., & Wang, S. H. (2022). Fingerspelling Recognition by 12-Layer CNN with Stochastic Pooling. *Mobile Networks and Applications*, *February*.

https://doi.org/10.1007/s11036-021-01900-8

Zhang, Y., & Yang, Q. (2018). An overview of multi-task learning. *National Science Review*, *5*(1), 30–43. https://doi.org/10.1093/nsr/nwx105

APPENDICES

Appendix I: Maize Production in Kenya by County in 2018

COUNTY	Annual Area (Ha)	Annual Quantity (Ton)	Annual Production (Ton/Ha)
Baringo	37,658	58,476	1.6
Bomet	33,291	58,337	1.8
Bungoma	93,484	295,482	3.2
Busia	33,122	53,629	1.6
Elgeyo Marakwet	30,631	92,602	3.0
Embu	35,812	32,114	0.9
Garissa	138	119	0.9
Homa bay	69,055	100,742	1.5
Isiolo	358	288	0.8
Kajiado	16,663	18,698	1.1
Kakamega	95,387	238,291	2.5
Kericho	33,461	105,403	3.2
Kiambu	29,434	32,219	1.1
Kilifi	60,617	54,676	0.9
Kirinyaga	30,877	33,348	1.1
Kisii	74,162	154,182	2.1
Kisumu	50,470	70,914	1.4
Kitui	80,244	22,967	0.3
Kwale	68,886	62,103	0.9
Laikipia	26,313	48,008	1.8
Lamu	22,704	29,025	1.3
Machakos	130,298	81,374	0.6
Makueni	123,311	62,759	0.5
Mandera			
Marsabit	1,060	347	0.3
Meru	72,012	74,726	1.0
Migori	82,153	128,126	1.6
Mombasa	439	309	0.7
Murang'a	65,701	71,793	1.1
Nairobi	804	460	0.6

Nakuru	86,102	238,003	2.8
Nandi	67,451	229,736	3.4
Narok	91,602	208,307	2.3
Nyamira	60,101	91,041	1.5
Nyandarua	21,295	35,705	1.7
Nyeri	28,631	39,050	1.4
Samburu	8,010	4,364	0.5
Siaya	66,766	87,638	1.3
Taita taveta	18,977	20,364	1.1
Tana river	4,819	3,536	0.7
Tharaka nithi	24,436	19,128	0.8
Trans nzoia	107,681	548,197	5.1
Turkana	4,246	3,742	0.9
Uasin gishu	95,209	405,461	4.3
Vihiga	25,090	35,129	1.4
Wajir	610	1,034	1.7
West pokot	32,172	61,840	1.9

Appendix II: Code Snippet for Offline Data Augmentation

```
directory='/content/drive/MyDrive/Corn_unzipped/train//*.jpg'
directory_saved='/content/drive/MyDrive/Corn_unzipped/train/LeafStreak_virus/'
i = 0

datagen = ImageDataGenerator(
        rotation_range=40,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')

for f in glob.iglob(directory):
  img = load_img(f)
  x = img_to_array(img)  # this is a Numpy array
  x = x.reshape((1,) + x.shape)
  i = 0

    # generate 6 new augmented images
  for batch in datagen.flow(x, batch_size = 1,
                        save_to_dir = directory_saved,
                        save_prefix ='augmentedLS', save_format ='jpg'):
      i += 1
      if i > 6:
          break
```

Appendix III: Code Snippet for the MTL Model

```
from keras.models import Model
from keras.layers import Input, Dense, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D

inp = keras.Input(shape=(224, 224,3))
conv_0 = Conv2D(64, (5 , 5), padding='valid', activation='relu')(inp)
conv_1 = Conv2D(64, (5 , 5), padding='same', activation='relu')(conv_0)
pool_1 = MaxPooling2D(pool_size=(2, 2))(conv_1)
dense_1 = Dense(64, activation='relu')(pool_1)

conv_2 = Conv2D(32, (5 , 5), padding='valid', activation='relu')(dense_1)
pool_2 = MaxPooling2D(pool_size=(2, 2))(conv_2)
dense_2 = Dense(32, activation='relu')(pool_2)

conv_3 = Conv2D(16, (5 , 5), padding='valid', activation='relu')(dense_2)
pool_3 = MaxPooling2D(pool_size=(2, 2))(conv_3)
dense_3 = Dense(16, activation='relu')(pool_3)

conv_4 = Conv2D(8, (3 , 3), padding='valid', activation='relu')(dense_3)
pool_4 = MaxPooling2D(pool_size=(2, 2))(conv_4)

conv_5 = Conv2D(4, (3 , 3), padding='valid', activation='relu')(pool_4)
pool_5 = MaxPooling2D(pool_size=(2, 2))(conv_5)
flat = Flatten()(pool_5)

out1 = Dense(5, activation= 'softmax', name = "diseaseN")(flat)
out2 = Dense(1, activation= 'sigmoid', name = "diseaseT")(flat)

mtl_model = Model(inputs=inp, outputs=[out1,out2])
mtl_model.save_weights('mtl.h5')
mtl_model.summary()
```

Appendix IV: Code Snippet for Compiling the Model

```
from keras.optimizers import Adam as adam_v2
adam = adam_v2(learning_rate=0.0001)
# res_gen_model.compile(
mtl_model.compile(
                    optimizer= adam,
                    loss={'diseaseN': 'categorical_crossentropy', 'diseaseT': 'binary_crossentropy'},
                    metrics=['accuracy'])
```

Appendix V: Code Snippet for Training MTL and Early Stopping Combined

```
# training the model
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard
from livelossplot import PlotLossesKeras

tf.compat.v1.experimental.output_all_intermediates(True)

checkpointer = ModelCheckpoint(filepath='mtl.h5',
                               verbose=1,
                               save_best_only=True)

# # Early stopping to prevent overtraining and to ensure decreasing validation loss
early_stop = EarlyStopping(monitor='val_loss',
                          patience=10,
                           restore_best_weights=True,
                           mode='min')
history =     mtl_model.fit(data,[named, typed],
              batch_size = 32,
              epochs = 200,
              validation_split=0.2,
              callbacks=[early_stop, checkpointer, PlotLossesKeras()],
              verbose = 1)
```

Appendix VI: Code Snippet for the Transfer Learning Combined to MTL Technique

```
# using resnet50
from keras.models import Model
from keras.layers import Input, Dense, Activation, Flatten
from keras.layers import Conv2D,  GlobalAveragePooling2D,Dropout
from tensorflow.keras.applications.resnet50 import ResNet50

res_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
x = res_model.output
x = GlobalAveragePooling2D()(x)
x= Flatten()(x)

out1 = Dense(5, activation= 'softmax', name = "diseaseN")(x)
out2 = Dense(1, activation= 'sigmoid', name = "diseaseT")(x)

res_gen_model = Model(inputs=res_model.input, outputs=[out1,out2])
res_gen_model.save_weights('resnetmodel.h5')
res_gen_model.summary()
```

Appendix VII: Publication

Below is the link to follow for our publication:

https://ieeexplore.ieee.org/document/9845568