# **CHAPTER THREE**

# **METHODOLOGY**

# **INTRODUCTION**

# I applied deep learning techniques to classify plant diseases in maize and cassava images using Convolutional Neural Networks (CNNs) and transfer learning. The aim was to build a robust deep learning model to distinguish between healthy and diseased maize and cassava plants to aid in early disease detection, which is vital for crop management and food security.

* 1. **DATASET The "Dataset for Crop Pest and Disease Detection" (Mensah Kwabena et al., 2023)**

Raw Data.zip, a 1.22 GB zip file, contains the raw photographs of 24,881 raw images and 102,976 Augmented images. The four (4) folders are in the Cashew, Cassava, Maize, Tomato (CCMT) Dataset subdirectory of the Raw Data folder after it has been unzipped. CCMT Dataset.zip, a 6.81 GB zip file, contains the expanded dataset. Following unzipping, the four (4) folders are in the main folder which has 22 classes in total.

Focusing on maize and cassava, **Cassava has (5) 7508 images**: Bacterial Blight, Mosaic, healthy, Brown Spot, and Green Mite while. **Maize has (7) 5389 images**: Fall armyworm, Grasshopper, healthy, Leaf beetle, Leaf blight, Leaf spot, and Streak. All images were captured, separated, and saved in their respective folders according to the plant type (Mensah Kwabena et al., 2023). The images were gathered from October 2022 till December 2022 and was published in 2023. The photographs were taken in a variety of settings and with a range of backgrounds, including genuine, dark, white, and lighted (Mensah Kwabena et al., 2023).

# **DATA PRE-PROCESSING**

## **Data source**

The Mendeley site provided the dataset in this project. To guarantee that the model is successfully trained and verified on unseen data to evaluate generalization performance, the entire dataset is divided into training (80%), validation (10%), and test (10%) sets after data collection.

* **Location:** University of Energy and Natural Resources

P.O. Box 214, Sunyani – Ghana

Website: [http://https//www.uenr.edu.gh](http://https/www.uenr.edu.gh)

African Technology Policy Society Network (ATPSNET)

8th Floor – The Chancery – Valley Road - Nairobi

P.O. Box 10081-00100, Nairobi, Kenya

Website: http://www.atpsnet.org

Data accessibility Repository name: Dataset for Crop Pest and Disease Detection

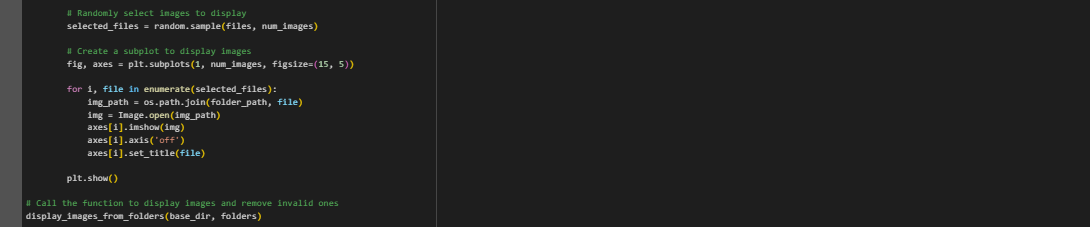
DOI: 10.17632/bwh3zbpkpv.1

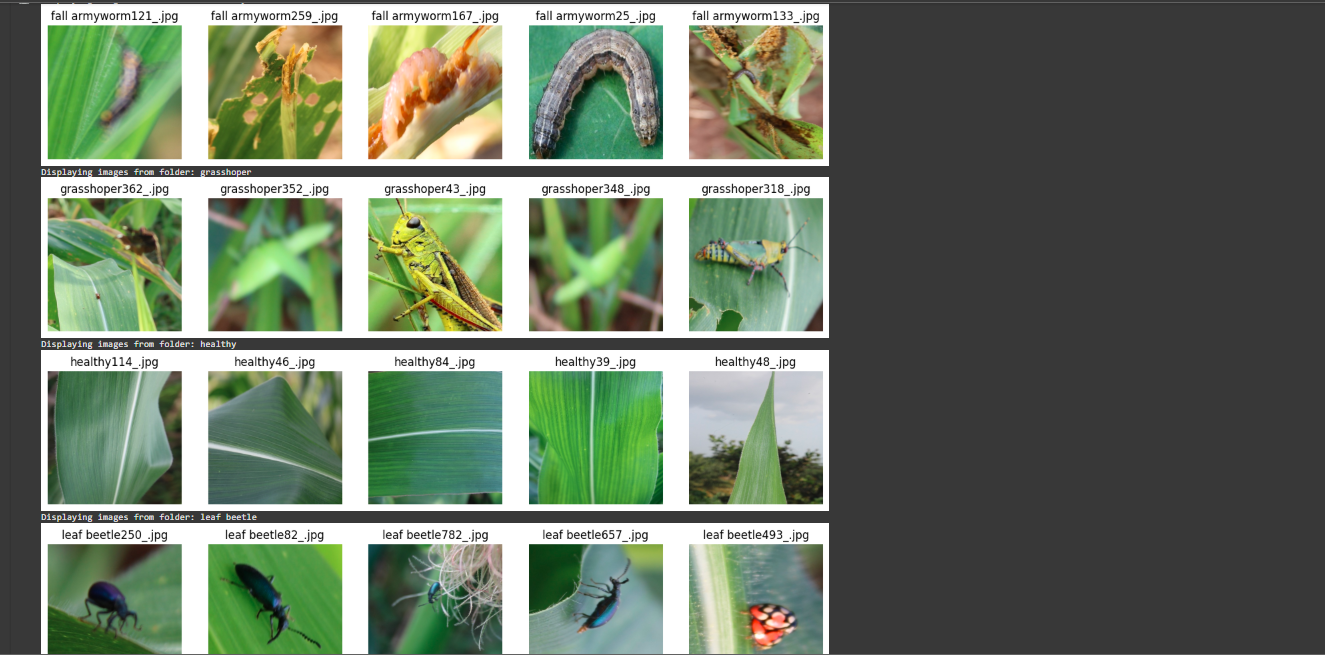
Direct URL to data: <https://data.mendeley.com/datasets/bwh3zbpkpv>

Source: (Mensah Kwabena et al., 2023)

A screenshot of a computer

AI-generated content may be incorrect.





A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 3.1: loading of maize folder dataset.

A screen shot of a computer

AI-generated content may be incorrect.

A collage of green leaves

AI-generated content may be incorrect.

A collage of green leaves

AI-generated content may be incorrect.

Figure 3.2: loading cassava folder dataset

After I loaded the dataset separately, I firstly merge them together then, I divided the entire dataset into three subsets: training (80%), validation (10%), and test (10%) sets. This division ensured that I could train the model on one subset and evaluate its performance on unseen data to assess generalization.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A close-up of a leaf

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 3.3: merging both the maize and cassava dataset together.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A black rectangular object with white text

AI-generated content may be incorrect.

Figure 3.4: splitting (maize and cassava) dataset into 80%, 10%, 10%

## **Data cleaning**

I performed a cleaning process to ensure the quality of the images used for model training. The cleaning steps included removing irrelevant or corrupted images and ensuring that the dataset only included high-quality, relevant data. This resulted in the following cleaned dataset sizes:

|  |  |  |
| --- | --- | --- |
|  | Maize | Cassava |
| original | 5,389 | 7,508 |
| After cleaning | 5,361 | 7,950 |

## **Data pre-processing**

I applied several preprocessing steps to raw image data before training:

* **Cleaning:** I removed any noisy or irrelevant parts of the images and eliminated low-quality images to focus on the regions of the plants where diseases were visible.
* **Resizing**: I resized all images to a consistent size of 150x150 pixels to ensure they could be processed by the deep learning model.
* **Normalization:** I normalized the pixel values of the images to the range [0, 1] to aid in faster model convergence during training.
* **Data Augmentation**: To improve the generalization ability of the model and prevent overfitting, I used several data augmentation techniques, such as random rotations, zooms, flips, and brightness adjustments. This artificially expanded the dataset and increased its diversity, especially for the minority classes.

## **Class Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
| Category | Maize | Category | Cassava |
| Leaf spot | 1,249 | Bacterial blight | 2,772 |
| Streak virus | 1,002 | Brown spot | 1,506 |
| healthy | 206 | Healthy | 1,283 |
| Leaf blight | 998 | Mosaic | 1,235 |
| Leaf beetle | 938 | Green mite | 1,154 |
| Grasshopper | 638 |  |  |
| Fall armyworm | 285 |  |  |

# **DISEASE CLASSIFICATION MODEL**

## **Model Selection**

In this research project I used 4 different models which include the following architectures.

1. **Custom CNN Architecture:**

A simple, straightforward CNN is flexible and allows customization based on the dataset.

**Layout:**

* **Convolutional Layers**: These layers are responsible for feature extraction. The filters (32, 64, 128) progressively capture more complex patterns such as edges, textures, and shapes.
* **Max-Pooling Layers**: These layers reduce the spatial size of the image and help the model focus on the most important features, thus improving the model's ability to generalize.
* **Flatten Layer:** Converts the 2D feature maps into a 1D vector that can be fed into fully connected layers.
* **Fully Connected Layers:** These layers perform the classification task by learning high-level features extracted by the convolutional layers.
* **Dropout Layer**: Reduces overfitting by randomly disabling a fraction of neurons during training, preventing the model from memorizing the training data.
* **SoftMax Output Layer**: Outputs the probability of each class for classification, with the class having the highest probability as the predicted label.

1. **VGG16 (Transfer Learning):**

known for its deep architecture, allows to capture complex hierarchical features. Great for tasks that require a deeper level of understanding of the images. Also has a strong track record in image classification tasks, especially when pre-trained on ImageNet, which made it suitable for fine-tuning in plant disease classification.

**Layout:**

* **Deep Convolutional Layers**: VGG16 has 16 layers (13 convolutional and 3 fully connected) and is known for its simplicity and depth. It uses small 3x3 filters and max-pooling layers to reduce dimensionality.
* **Global Average Pooling**: After the convolutional layers, Global Average Pooling is applied to average the features across all spatial locations, resulting in a fixed-size output irrespective of input size.
* **Fully Connected Layers**: These are used to make predictions based on the extracted features.
* **SoftMax Output Layer:** Final classification layer for multi-class classification.

1. **MobileNetV2 (Transfer Learning):**

Optimized for mobile and edge devices, making it an excellent choice for real-time applications with limited computational resources.Despite being lightweight, it performs well on image classification tasks due to its use of depth wise separable convolutions.Pre-trained on ImageNet as a good transfer learning approach, provides good feature extraction backbone, enabling faster training on specialized tasks (e.g., plant disease classification).

**Layout:**

* **Depth wise Separable Convolutions**: MobileNetV2 uses more efficient form of convolution, each input channel is processed separately in two steps: depth wise convolution followed by pointwise convolution (1x1). This reduces the number of parameters and computations, making it lightweight.
* **Pre-trained on ImageNet**: The base model was pre-trained on ImageNet, so it already knows how to extract features from a wide variety of images, including plants. This allows fine-tuning on the plant disease classification task.
* **No Fully Connected Layer**: It’s used as a feature extractor by removing the top classification layer and adding custom layers after it.

1. **Hybrid Model (Ensemble Approach):**

Each model (ResNet50, EfficientNetB3, and MobileNetV2) has different strengths. By combining them, I can create a model that captures diverse patterns and features, improving robustness and accuracy. Ensemble models are often more powerful than single models because they combine the strengths of different approaches, which can lead to better performance, especially on complex datasets. Each of the base models might capture different aspects of the image data (e.g., texture, edges, fine details), and merging their features gives the hybrid model an advantage in handling different types of plant diseases.

**Layout:**

* **Base Models**: I used the pre-trained models ResNet50, EfficientNetB3, and MobileNetV2 as feature extractors. These models were frozen to preserve their pre-trained weights.
* **Global Average Pooling**: After extracting features from each model, the output is pooled globally, reducing the dimensionality and retaining only the important features.
* **Concatenatio**n: I pooled features from all three models and are concatenated into a single vector, combining the strengths of each model’s feature extraction.
* **Fully Connected Layers**: These layers help me in learning the final decision based on the combined features.
* **SoftMax Output Layer**: This layer handles multi-class classification.
  + 1. **LAYOUT SUMMARY**
* **CNN**: Basic, customizable architecture for simpler tasks and learning from scratch.
* **MobileNetV2**: Chosen for its lightweight design and efficiency, making it ideal for resource-constrained devices.
* **VGG16**: Used for deeper feature extraction and better performance on more complex tasks.
* **Hybrid Model**: Combines multiple models' strengths, using ensemble learning to improve robustness and accuracy by learning from diverse feature extraction perspectives

# **TRAINING THE MODEL**

## **Model Training Process**

The training process involved feeding labeled data into the model and allowing it to learn the distinguishing patterns between healthy and diseased plants. For training:

1. **Learning Rate**: I used a moderate learning rate of 0.001, balancing between fast convergence and the risk of overshooting the optimal solution.
2. **Batch Size**: I set the batch size to 32 for efficient training, ensuring that the model was exposed to enough images before updating the weights.
3. **Epochs:** The model was trained for a total of 10, 20 and 30 epochs, allowing enough time for the model to learn and adjust its weights appropriately.

## **Loss Function and Optimization**

For multi-class classification, I used categorical cross-entropy as the loss function, which is standard for tasks involving multiple classes, it was used for all models. The Adam optimizer was employed to improve convergence speed and model performance due to its adaptive learning rate, was used in all models except hybrid. The Hybrid used AdamW because applies weight decay in a way that works as intended which leads to better generalization and more stable training which works in line with ReduceLROnPlateau.

## **Model Evaluation Metrics**

Accuracy, Precision, Recall, F1-Score. The metrics explanation will be detailed in chapter four. Metrics were calculated on both the validation and test datasets

* + 1. **Regularization Techniques**
* Dropout: A dropout rate of 0.5 was applied to randomly disable neurons during training, which helps improve generalization. Was used by all models.
* Early Stopping: Training was stopped if the validation loss started to degrade to ensure model didn’t overfit.
  + 1. **Activation Function**
* ReLU (Rectified Linear Unit): Prevents the vanishing gradient problem and accelerates convergence (Nair & Hinton, 2010).
* SoftMax: Converts logits into probability distributions for multi-class classification.
* Sigmoid and Tanh: Used for binary classification but can suffer from saturation issues (Goodfellow et al., 2016). The output of a sigmoid function is between 0 and 1, called the squashing function
  + 1. **Callbacks**

This allows the model to monitor and modify the training pattern automatically which helps saves the best model and training efficiency.

* ModelCheckpoint: Saves the best model only when validation accuracy improves which prevents losing the most performing model. It was used by CNN, VGG16 and MobileNet V2.
* ReduceLROnPlateau: Automatically reduces the learning rate anytime the model stops improving. Was used by only Hybrid.
* EarlyStopping: It stops training early if the metrics stops improving which saves time. Was used by VGG16 and Hybrid Model.

# **ARCHITECTURAL DESIGN**

Overall structure from the initial dataset collection to the selection of the model.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.5 Architectural diagram

# **3.6 FLOW CHART**

illustrates the sequence of steps **I** followed during the research project.

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 3.6 Flow chart for the research project