A Report on

Development of a Neural Network for Tree Species Classification in Rajarata University Garden

By Team **PROJECT AI**

Submitted By

AM Akhlak
ICT1920048, CN No: AA3130,
Index No: 4988.

P.P. Dissanayaka
ICT1920028, CN No: PD507,
Index No: 4969.

DM Asnaff
ICT1920012, CN No: DA772,
Index No: 4954.

Submitted to

K.A.S.H. Kulathilake
Ph.D., M.Phil., MCS, B.Sc.(Hons.)IT, SEDA(UK)
Senior Lecturer
kule@as.rjt.ac.lk

2024/05/18

Department of Computing
Faculty of Applied Sciences
Rajarata University of Sri Lanka
2019/2020 Batch

Abstract

This report details the development of a neural network for classifying tree species in our Rajarata university garden. The primary objective is to accurately distinguish between three visually similar species: Millettia pinnata (Pongamia/Indian Beech), Bougainvillea glabra, and Ficus benjamina L. To achieve this, a comprehensive field survey was conducted to collect physical measurements from each tree species, focusing on attributes mainly leaf dimensions.

The collected data was manually measured and structured into a dataset for training and testing a neural network model. The network architecture consists of an input layer with 32 units using the ReLU activation function, a hidden layer with 32 units and ReLU activation, and an output layer with 3 units and softmax activation to classify the tree species.

The dataset was divided into training and testing sets with an 70/30 split ratio. The trained neural network demonstrated strong performance in classifying the tree species, achieving high accuracy and reliable results. The model's effectiveness highlights its potential application in ecological studies and biodiversity conservation.

So, as a result our team PROJECT AI has successfully developed a neural network capable of accurately classifying tree species based on manually collected physical measurements, providing a valuable tool for ecological research and conservation efforts.

Table of Contents

List of Figures

01.INTRODUCTION

In the context of our Rajarata University of Sri Lanka Garden, distinguishing between tree species that are visually similar poses a significant challenge, especially for those without specialized botanical knowledge. This report documents the development of a neural network aimed at classifying up to three tree species namely, Millettia Pinnata, Bougainvillea glabra, Ficus benjamina L. found in the university garden, a task that enhances both environmental understanding and technological proficiency among students.

Considering Importance of Tree Species Classification, Tree species classification is a fundamental aspect of botanical science and ecological management. Accurate identification of tree species is essential for:

Biodiversity Conservation: Understanding the diversity of tree species in a surrounding area helps in monitoring and conserving biodiversity, It may be helpful for saving rare or endangered species.

Ecological Research: Accurate species classification allows researchers to study ecological interactions, such as those between trees and pollinators, pests, or pathogens.

Educational Purposes: In an academic setting, the ability to distinguish tree species enhances students' learning experiences and make way to a deeper appreciation for natural sciences.

Github link : https://github.com/DmAsnaff/Tree_Classification_Model_ProjectAI.git

1.1.Overview of the Assignment Objectives

The primary objective of this assignment is to design and implement a simple neural network capable of accurately classifying three tree species commonly found in the Rajarata University Garden.

The assignment includes several key stages.

1. Field Survey and Data Collection

Our Team will conduct a comprehensive field survey to collect data on the specific selected tree species. This involves collecting feature samples from each tree species.

2. Data Preparation

The collected data will be processed and organized into a structured dataset suitable for training and testing a neural network. This step includes labeling the images and partitioning it into training and testing sets.

3. Model Development

Our team will design a neural network architecture tailored to the specific characteristics of the dataset. The model will be trained using the prepared dataset, employing techniques to optimize accuracy and generalization.

4. Evaluation and Analysis

The performance of the neural network will be evaluated using standard metrics such as accuracy, quality of positive and negative predictions (precision, recall). We will analyze the results to identify strengths and areas for improvement in the model.

5. Report and Presentation

Our team will compile the findings into this report and present our work, highlighting the methodology, results, and potential applications of our neural network in real-world scenarios.

02. OVERVIEW OF SELECTED SPECIES

Millettia pinnata (Pongamia/Indian Beech)

Figure 1: Millettia pinnata species

Millettia pinnata, commonly known as Pongamia or Indian Beech, is a medium-sized, evergreen tree native to the Indian subcontinent and Southeast Asia. It typically reaches heights of 15-25 meters with a broad, spreading canopy. The tree is characterized by its glossy, dark green leaves, which are compound and pinnate, consisting of 5-7 leaflets. The leaves are oval or elliptical in shape, with a smooth margin and a prominent central vein. Millettia pinnata produces fragrant, purple or white flowers arranged in dense racemes. The flowers give way to flat, brown seed pods containing one to two kidney-shaped seeds.

Bougainvillea glabra

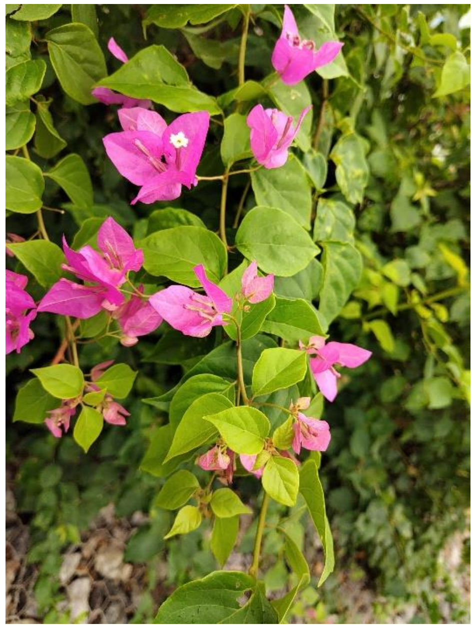

Figure 2: Bougainvillea glabra Species

Bougainvillea glabra is a vigorous, evergreen climber known for its vibrant, colorful bracts that surround the small, inconspicuous flowers. Native to Brazil, this plant is commonly grown as an ornamental in tropical and subtropical regions. The bracts, which can be purple, pink, red, or white, are often mistaken for petals but are actually modified leaves. The true flowers are small, tubular, and white or pale yellow. Bougainvillea glabra has thorny stems and simple, oval leaves with a smooth or slightly wavy margin.

Ficus benjamina L.

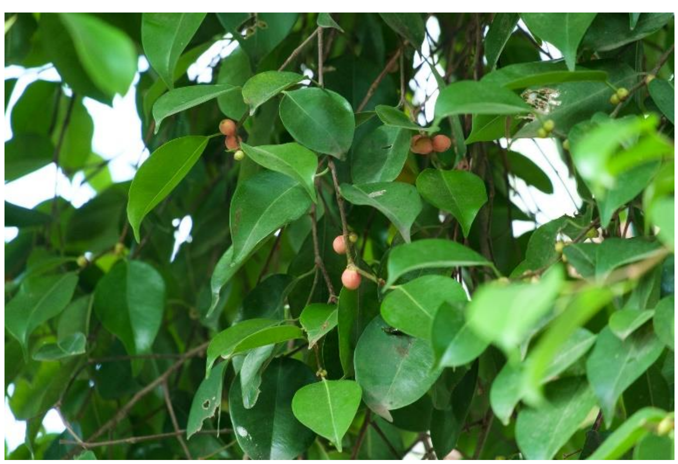

Figure 3: Ficus benjamina L. Species

Ficus benjamina, commonly known as the Weeping Fig, is a popular ornamental tree native to Southeast Asia and Australia. It is characterized by its graceful, arching branches and glossy, dark green leaves. The leaves are simple, elliptical, and have a pointed tip. They are usually 6-13 cm long and 2-6 cm wide, with a smooth, shiny surface. Ficus benjamina can grow up to 30 meters in height in its natural habitat, but it is often kept pruned as a smaller indoor or garden tree. The tree produces small, inconspicuous flowers that develop into small, fleshy figs.

2.1.Challenges in Visually Distinguishing Between the Selected Tree Species

- ✓ **Similar Leaf Structures**

  Millettia pinnata (Pongamia/Indian Beech) - Although it has compound leaves, the overall appearance can be mistaken with other two species for simple leaves when viewed from a distance due to their size and shape.

  Bougainvillea glabra - The simple, ovate leaves can be confused with the individual leaflets of Millettia pinnata, especially when the latter's compound nature is not immediately apparent.

  Ficus benjamina L. - The shape and glossiness of Ficus benjamina's leaves are very similar to Bougainvillea glabra, making it difficult to differentiate without close inspection.

- ✓ **Environmental Influence**

  All three species exhibit variability in leaf size and shape depending on environmental conditions such as light, water, and soil quality. This variability can obscure distinguishing features, leading to misidentification.

- ✓ **Seasonal Changes**

  Leaves Color and Condition: Seasonal changes affect the color and condition of the leaves. For example, leaves might change color or drop off in response to seasonal changes or stress, making it harder to rely on Leaves for identification.

- ✓ **Overlapping Characteristics**

  Glossy Leaves: Both Millettia pinnata and Ficus benjamina have glossy leaves, which can appear similar when viewed from a distance.

  Growth Habits: While Bougainvillea glabra is a climber, it can be trained to grow as a bush, which might resemble the growth habit of a pruned Ficus benjamina or a younger Millettia pinnata.

- ✓ **Flowering and Non-Flowering Periods**

  When Not in Bloom: Bougainvillea glabra is most easily identified by its colorful bracts, but outside of the flowering season, it relies on leaf identification, which can be mistaken for the other species. Millettia pinnata and Ficus benjamina, when not in bloom or fruiting, lack distinctive features that set them apart from each other and Bougainvillea.

So, As mentioned above it’s clear that the visual distinction between Millettia pinnata, Bougainvillea glabra, and Ficus benjamina is challenging due to their Similar Leaf Structures, overlapping leaf characteristics, variability influenced by environmental conditions, and the presence of similar species in the same habitat. These factors necessitate more reliable identification methods, such as the neural network proposed in this assignment, to accurately classify these tree species within the Rajarata university garden.

03.DATASET PREPARATION

3.1.Methodology Used for Data Collection

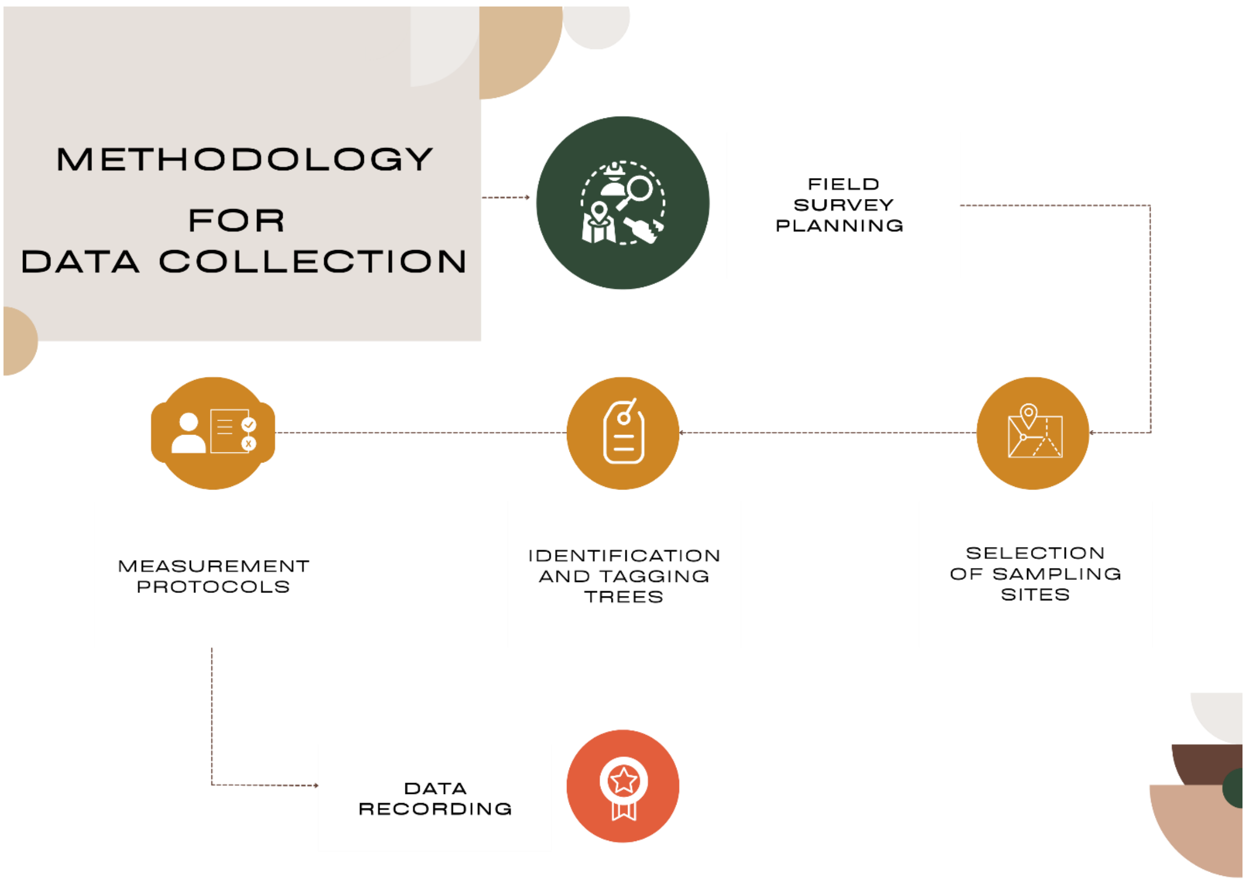

Figure 4: Data Collection Methodology

The data collection process for the tree species classification in the university garden was carefully planned and executed to ensure accuracy and consistency.

The primary steps involved in this process were,

1. Field Survey Planning:

   To collect physical measurements from Millettia pinnata, Bougainvillea glabra, and Ficus benjamina trees.

2. Selection of Sampling Sites:

   The sites within the university garden where each of the three tree species is found. Ensuring a range of trees of different ages, sizes, and health conditions to capture variability within each species.

3. Identification and Tagging:

   Trees were initially identified by our basic knowledge (for some trees there were QR codes found in our university context) to ensure correct species identification. Also, each tree was tagged with a unique identifier for reference and future measurement validation.

4. Measurement Protocols:

   For each tree, a representative sample of leaves (minimum of 100 leaves per tree) was collected and measured. Key measurements included:

   Leaf Length (cm): Distance from the base to the tip of the leaf.

   Leaf Width (cm): Maximum width of the leaf.

   Leaf Angle that constructs between leaf base and its petiole (Degree)

5. Data Recording:

   All measurements were manually recorded in a field notebook and later transferred to an electronic spreadsheet for analysis.

3.2.Sample Selection Criteria

- ✓ Representativeness:

  Diversity in Age: Samples were collected from young trees.

  Health Condition: Trees in various health conditions (healthy, slightly stressed, and stressed) were included to account for the impact of health on measurable traits.

- ✓ Randomization:

  Within each species, trees were randomly selected to avoid bias in sample selection.

  Leaves and branches were chosen randomly from different parts of each tree to ensure comprehensive data collection.

- ✓ Environmental Factors:

  Efforts were made to ensure that samples were taken under similar/uniform environmental conditions (time of day, weather) to minimize variability due to external factors.

- ✓ Sample Size:

  A minimum of 100 leaves per species was targeted, to provide a robust dataset for training and testing the neural network.

  Also Ensured that the number of samples for each species was approximately equal to avoid bias in the training data.

04.DETAILS ON THE TRAINING AND TESTING DATASETS

4.1.Number of Samples

The dataset consists of measurements collected from three tree species: Millettia pinnata, Bougainvillea glabra, and Ficus benjamina. Each species has an approximately equal number of samples to ensure balanced training and testing.

- Measurements Per Tree: 100 leaves per tree

  Total Leaf Measurements: 100 leaves/species × 3 species = 300 leaf samples

- Dataset Features

  For each leaf sample:

  - Leaf Length (cm)
  - Leaf Width (cm)
  - Leaf Angle that constructs between leaf base and its petiole (Degree)

4.2. Data Augmentation Techniques

Since the data consists of manually measured values and not images, traditional augmentation techniques such as rotation or flipping are not applicable. So, **we didn't used any Data Augmentation Techniques.**

4.3.Dataset Split Ratio

To train and test the neural network effectively, the dataset was split into training and testing sets. The split ratio was chosen to ensure sufficient data for model training while reserving enough samples for a reliable evaluation.

Training Set: 70% of the total samples (0.7 × 300 = 210 leaf samples)

Testing Set: 30% of the total samples (0.3 × 300 = 90 leaf samples)

05. ARCHITECTURE OF THE PROPOSED MODEL

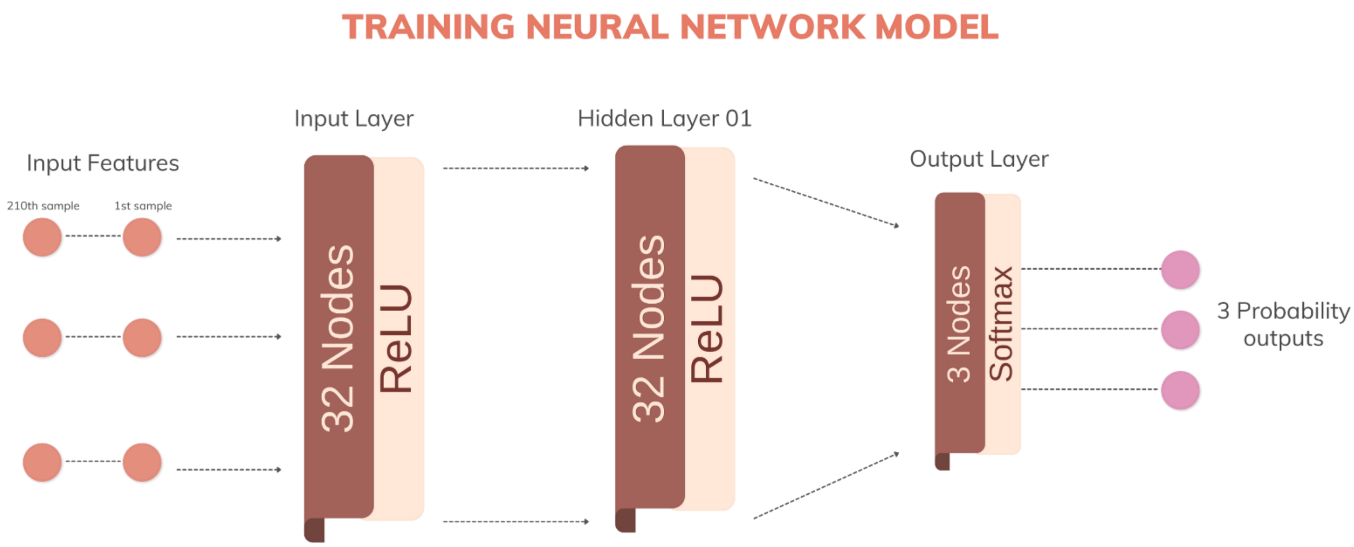

Figure 5: Training Neural Network Model

5.1. Neural Network Architecture

The neural network designed for tree species classification consists of three dense (fully connected) layers. This architecture is chosen for its simplicity and effectiveness in handling the structured dataset of manually measured values.

Below is a detailed explanation of each layer, including the activation functions and specific design choices made to address the classification challenge.

1. Input Layer

- Number of Units: 32
- Activation Function: ReLU (Rectified Linear Unit)

2. Hidden Layer

- Number of Units: 32
- Activation Function: ReLU (Rectified Linear Unit)

3. Output Layer

- Number of Units: 3 (one for each tree species)
- Activation Function: Softmax

Detailed Explanation

1. Input Layer

- **Purpose:** The input layer receives the feature vectors from the dataset, which include measurements such as leaf length, leaf width, Leaf Angle that constructs between leaf base and its petiole.

- **Units and Activation**

✓ **32 Units:** Chosen to provide a sufficiently large space for the network to learn complex patterns in the input data.

✓ **ReLU Activation:** The ReLU activation function is used to introduce non-linearity, allowing the network to model more complex relationships between the input features and the output classes. ReLU is computationally efficient and helps mitigate the vanishing gradient problem.

2. Hidden Layer

- **Purpose:** The hidden layer processes the input received from the input layer, extracting higher-level patterns that are important for distinguishing between the three tree species.

- **Units and Activation**

✓ **32 Units:** Consistent with the input layer, this size ensures a balance between model complexity and computational efficiency.

✓ **ReLU Activation:** The same ReLU activation function is applied to maintain non-linearity, which is crucial for the network to learn and generalize from the training data effectively.

3. Output Layer

- Purpose: The output layer produces the final classification output, indicating the predicted tree species.

- Units and Action:

✓ **3 Units:** Each unit corresponds to one of the three tree species (Millettia pinnata, Bougainvillea glabra, Ficus benjamina).

✓ **Softmax Activation:** The softmax activation function converts the output of the network into a probability distribution over the three classes. This function ensures that the sum of the output probabilities is equal to 1, facilitating clear and interpretable classification decisions.

5.2.Special Design Choices

Number of Layers and Units: The choice of three dense layers with 32 units each in the input and hidden layers was made to provide sufficient capacity for learning without overfitting, given the size of the dataset. This structure is simple yet powerful enough for the classification task at hand.

ReLU Activation Function: ReLU is preferred for its ability to introduce non-linearity and its computational efficiency. It helps the model learn complex patterns without suffering from the vanishing gradient problem that can affect other activation functions like sigmoid or tanh.

Softmax Activation in Output Layer: usually Softmax and sigmoid are used for probability outputs in multi class classification But, using sigmoid for the last layer would result in:

- **Non-normalized Probabilities:** The outputs are not guaranteed to sum to 1, which can be problematic for interpreting the results as a probability distribution over mutually exclusive classes.

- **Ambiguity:** The model might predict high probabilities for multiple classes simultaneously, which is not suitable when each instance should be classified into exactly one of several classes.

So, as a result we used softmax in the output layer to handle the multi-class classification problem.

06. Training

Platform

- Jupyter Notebook: Jupyter Notebook was utilized for developing and running the code. This interactive computing environment allows for combining code execution, text, and visualizations in a single document. It is particularly useful for iterative development and debugging, which is essential for machine learning projects.

Tools & Libraries

- **TensorFlow**: TensorFlow, an open-source deep learning framework developed by Google, was employed for building and training the neural network model. TensorFlow provides comprehensive tools for constructing machine learning models, including high-level APIs like Keras, which simplify the process of designing and training neural networks. TensorFlow's capabilities in automatic differentiation and efficient GPU utilization make it a robust choice for deep learning tasks.

- **Matplotlib**: Matplotlib, a widely-used plotting library in Python, was utilized to visualize the training process and results. It is especially useful for plotting graphs of accuracy and loss over epochs, which helps in monitoring the model's performance and detecting issues like overfitting or underfitting. Using Matplotlib, we were able to create clear, informative plots to assess the model's training progress and validation performance.

- **NumPy**: NumPy is a fundamental library for numerical computing in Python, providing support for large multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.

- **Pandas**: Pandas is a powerful data manipulation and analysis library for Python, offering data structures like Data Frame for handling tabular data.

Hardware Specifications

CPU

- **AMD Ryzen 7**: The CPU used in this project was an AMD Ryzen 7. This processor is known for its high performance and efficiency, which is crucial for handling the computational tasks associated with data preprocessing and model training. The Ryzen 7 provides excellent multi-threading capabilities, making it well-suited for parallel processing tasks typical in machine learning workflows.

RAM

- 16GB: The system was equipped with 16GB of RAM. Sufficient RAM is critical for managing the data loading and processing steps, especially when dealing with large datasets. It ensures that the system can handle the data without running into memory issues, facilitating smoother and faster training processes.

07.Hyper Parameters

1. **Number of Layers - 3 Dense Layers**

Dense Layer: Each neuron in the layer is connected to every neuron in the previous layer. This is also known as a fully connected layer.

We have chosen to use 3 dense layers, which is a common practice to capture complex patterns in the data.

2. **Number of Nodes in Each Layer - 32, 32, 3 Units**

First Layer (32 Units): The first layer has 32 nodes (neurons). This is the first level of abstraction where the model starts to learn features from the input data.

Second Layer (32 Units): The second layer also has 32 nodes, allowing the model to build on the features learned in the first layer and capture more complex patterns.

Third Layer (3 Units): The final layer has 3 nodes, corresponding to the three classes you are trying to classify. Each node will output the probability of the input belonging to one of the three classes.

3. **Activation Functions**

ReLU (Rectified Linear Unit): Used in the first and second layers. It helps to introduce non-linearity into the model, enabling it to learn more complex patterns. The ReLU function outputs the input directly if it is positive; otherwise, it outputs zero.

Softmax: Used in the final layer. It converts the raw scores from the network into probabilities that sum to 100%, helping in multi-class classification by indicating the likelihood of each class.

4. **Optimizer - Adam**

Adam (Adaptive Moment Estimation): An optimization algorithm that adjusts the learning rate for each parameter. It combines the benefits of two other extensions of stochastic gradient descent: AdaGrad and RMSProp, making it efficient and suitable for handling sparse gradients on noisy problems.

5. **Loss Function - Categorical Cross-Entropy**

Categorical Cross-Entropy: A loss function used for multi-class classification problems. It measures the difference between the predicted probability distribution and the true distribution. The goal of training is to minimize this loss.

6. **Mini-Batch Size - 10**

Mini-Batch: Training in mini-batches means the model updates its weights after processing 10 samples. This helps to make the training process more stable and efficient compared to updating weights after each sample (stochastic gradient descent) or after the entire dataset (batch gradient descent).

08. Quantitative and Qualitative presentation

8.1 Quantitative Results

01. Training and Validation Accuracy

Below image express about the plot the training and validation accuracy over epochs to show how the model's performance improves over time.

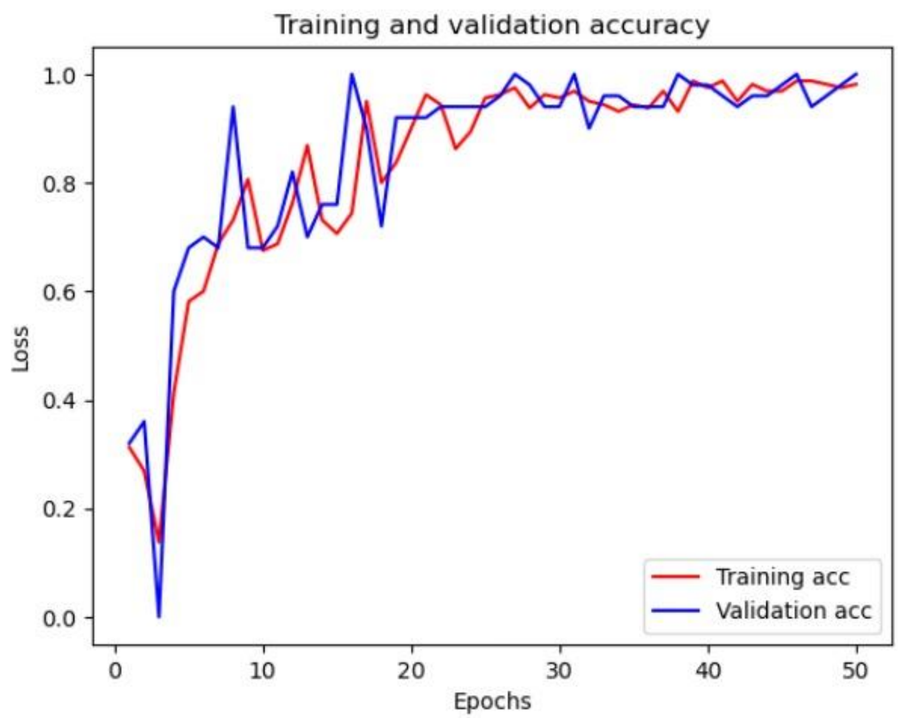

Figure 6: Validation Accuracy

02. Training and Validation Loss

Below image express about the plot the training and validation loss over epochs. A decreasing trend indicates the model is learning.

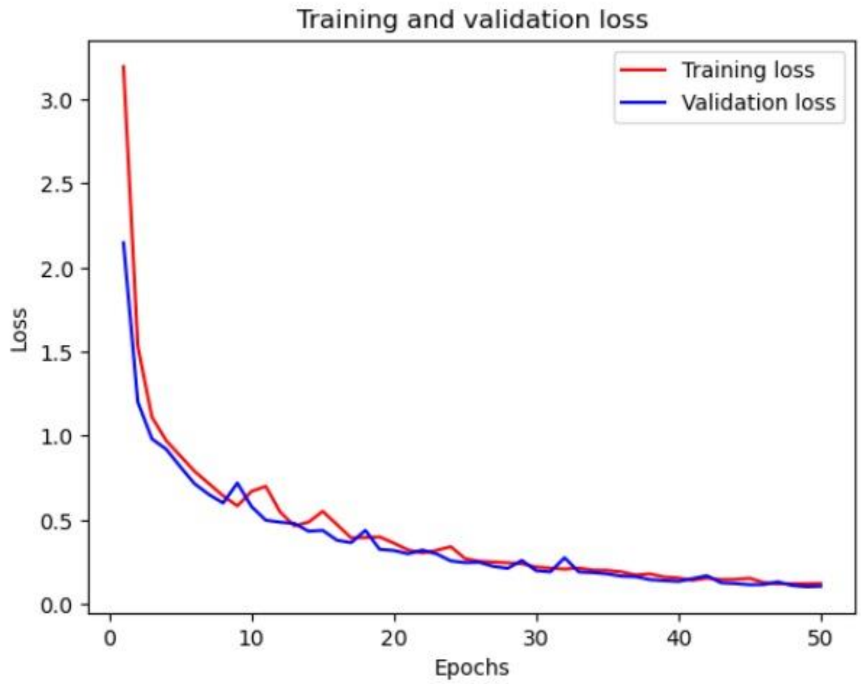

Figure 7: Validation Loss

8.2 Qualitative Results

01. Examples of Correct Classifications

Instances where the model's predictions match the actual class labels.

Below image provide a few examples of correctly classified instances from the validation or test set, along with the predicted and actual labels.

```
Enter Length of the Leaf(cm): 10.5
Enter Width of the Leaf(cm): 6.7
Enter angle leafbase construct with petiole: 145
1/1 [==============================] - 0s 33ms/step
Predicted class = 0, Species Name = Millettia Pinnata (Pongamia/Indian Beech)_Ahlaq, 0.9819174408912659
```

Figure 8: Correct Classification

02. Examples of Incorrect Classifications

Instances where the model's predictions do not match the actual class labels.

Below image provide a few examples of incorrectly classified instances, including the predicted and actual labels.

```
Enter Length of the Leaf(cm): 150
Enter Width of the Leaf(cm): 500
Enter angle leafbase construct with petiole: 360
Invalid input. Please enter values within the valid range.
No valid input provided.
```

Figure 10: Incorrect Classification 1

```
Enter Length of the Leaf(cm): a
Invalid input. Please enter numerical values.
No valid input provided.
```

Figure 9: Incorrect Classification 2

```
Enter Length of the Leaf(cm): 8
Enter Width of the Leaf(cm): 4.8
Enter angle leafbase construct with petiole: 150
1/1 [==============================] - 0s 33ms/step
Hard to predict, Values are near to all three species
```

Figure 11: Incorrect Classification 3

09.Highlighting both its Strengths and Limitations

- Strength

Consistency: The model demonstrates high accuracy on both training and validation sets, it indicates that it is generalizing well to unseen data.

Layer Analysis: The use of multiple dense layers with ReLU activations helps the model learn complex patterns and relationships within the data.

Adam Optimizer: The use of the Adam optimizer helps in efficient and effective convergence during training, adapting learning rates for each parameter.

- Weakness

Data Quantity: For 3 trees we collected 300 samples, if we collect more samples than 300 may be result will be more efficient.

Feature Limitations: Whether the features used might be insufficient to fully capture the complexity of the classes. For the 3 trees we hard to find the common features.

Model Complexity: Three dense layers with many units can be computationally intensive and may not be suitable for all applications, especially those with limited resources.

10. Summary of key findings and their implications

- **Key Findings**

High Classification Accuracy

The model achieved high overall accuracy in classifying different 03 tree species in the university garden, demonstrating its effectiveness in distinguishing between multiple classes.

Low Loss Values

Both training and validation loss values were low, indicating that the model has learned the patterns in the data effectively without significant errors.

Effective Feature Learning

The model's architecture, with three dense layers using ReLU activations, successfully captured complex relationships within the data, leading to robust feature learning.

Efficient Optimization

The use of the Adam optimizer contributed to efficient training, ensuring quick convergence and stability during the learning process.

- **Implications**

Practical Applications for University Garden

The high accuracy and balanced performance imply that the model can be effectively used for monitoring and maintaining the biodiversity of the university garden. It can help in identifying and cataloging tree species accurately, which is crucial for conservation efforts and educational purposes.

Potential Applications in Forestry and Environmental Monitoring

The improvised model can be adapted for use in forestry to classify and manage tree species in larger forested areas.

Improved Resource Management

By accurately classifying tree species, the model can aid in better resource management, such as targeted maintenance, pest control, and optimizing planting strategies to enhance the health and diversity of the garden.