Final (hopefully) Project Report: Comparative Analysis of Machine Learning and Deep Learning Models

Project Overview

This project aimed to evaluate and compare the performance of various machine learning (ML) and deep learning (DL) models on a given dataset. The goal was to determine which model achieves the highest accuracy and how traditional ML methods contrast with state-of-the-art DL architectures, including custom Convolutional Neural Networks (CNNs) and pre-trained models.

Dataset and Preprocessing

Data Preprocessing Steps

Top 10 Labels: Images were filtered to include only the top 10 most frequent labels.

Image Resizing: All images were resized to a uniform dimension of 224x224 pixels.

Label Encoding: Labels were encoded using LabelEncoder and transformed into a categorical format.

Data Splitting: The dataset was divided into training (80%) and testing (20%) subsets.

Normalization: Pixel values were normalized to the range [0, 1].

Data Augmentation

To enhance the robustness of the models, the following data augmentation techniques were employed on the training set:

- Rotation
- Width Shift
- Height Shift
- Horizontal Flip
- Zoom
- Shear

Model Implementations and Performance

Custom CNN

The custom CNN was designed with multiple convolutional layers, each followed by max pooling for dimensionality reduction. The architecture included dense layers with

ReLU activation and a final softmax layer for classification. The model was optimized using the Adam optimizer and trained with categorical cross-entropy loss.

Accuracy: 80.00%

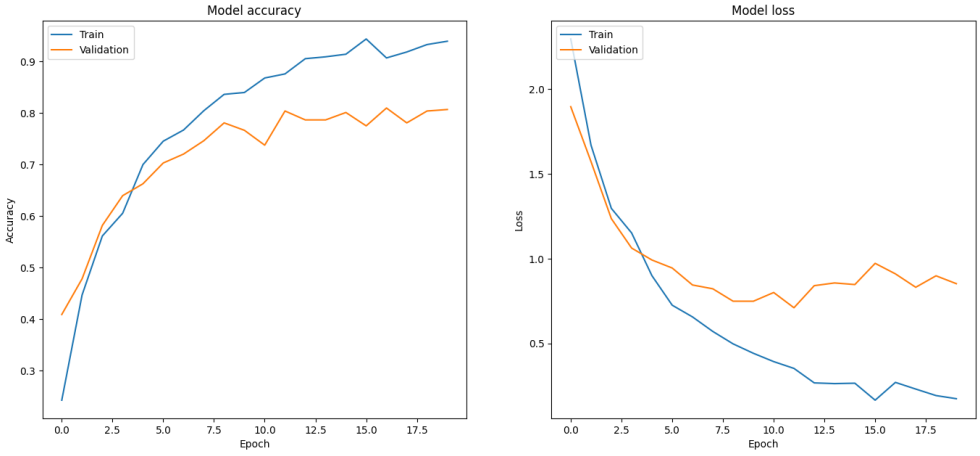

Figure 1, Confusion Matrix for Custom CNN

MobileNetV2

MobileNetV2 is a lightweight and efficient model suitable for mobile and embedded vision applications. It uses depthwise separable convolutions to reduce computational cost, making it fast and resource-efficient.

Accuracy: 83.57%

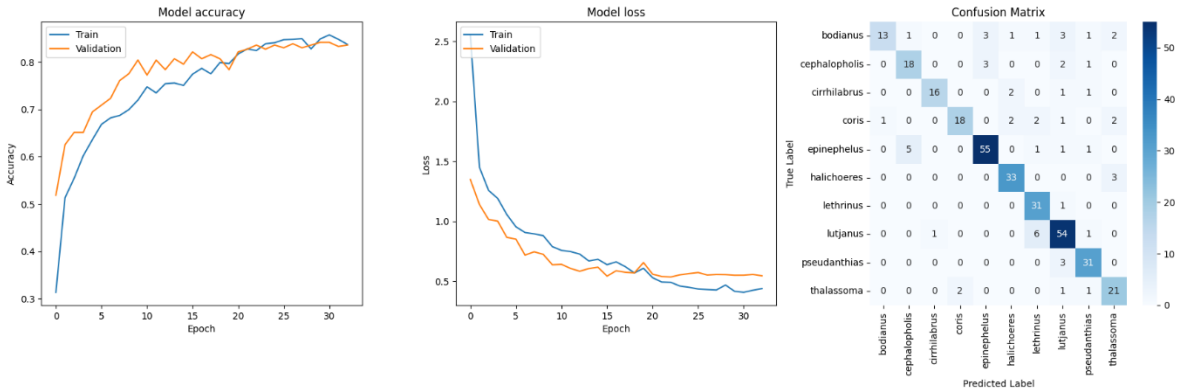

Figure 2, Confusion Matrix for MobileNetV2

VGG16

VGG16 is a deep convolutional network known for its simplicity and depth, consisting of 16 layers. It uses small convolution filters (3x3) and has a large number of parameters, which makes it powerful for image classification tasks.

Accuracy: 87.61%

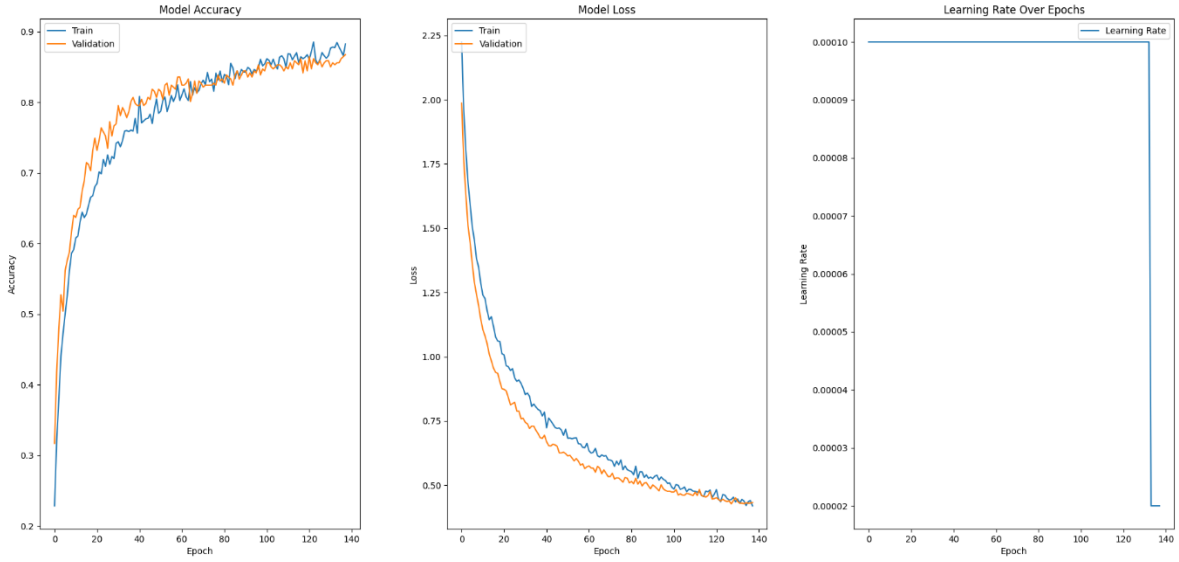

Figure 3, Accuracy and Loss for VGG16

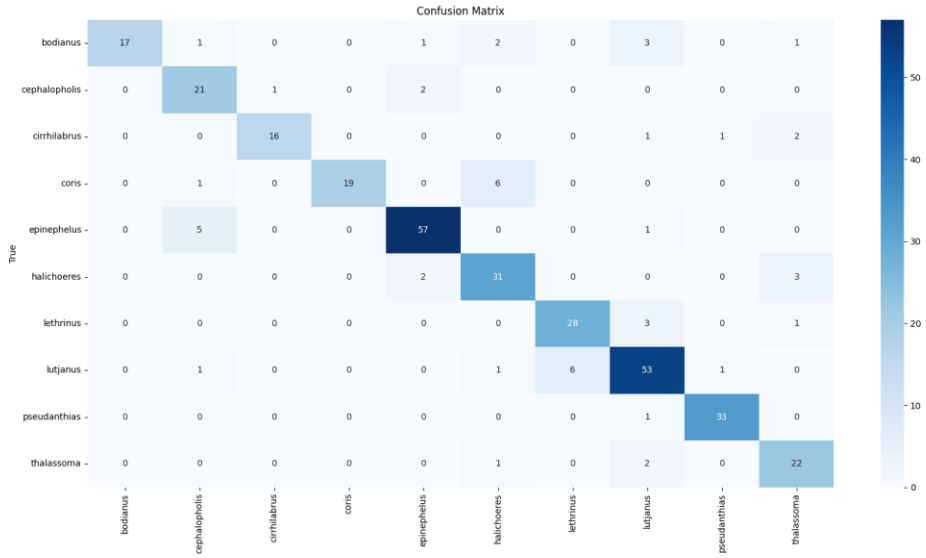

Figure 4, Confusion Matrix for VGG16

DenseNet121

DenseNet121 connects each layer to every other layer in a feed-forward fashion, which allows for improved information flow and reduced vanishing gradients. This architecture is known for its efficiency and ability to achieve high accuracy with fewer parameters.

Accuracy: 90.20%

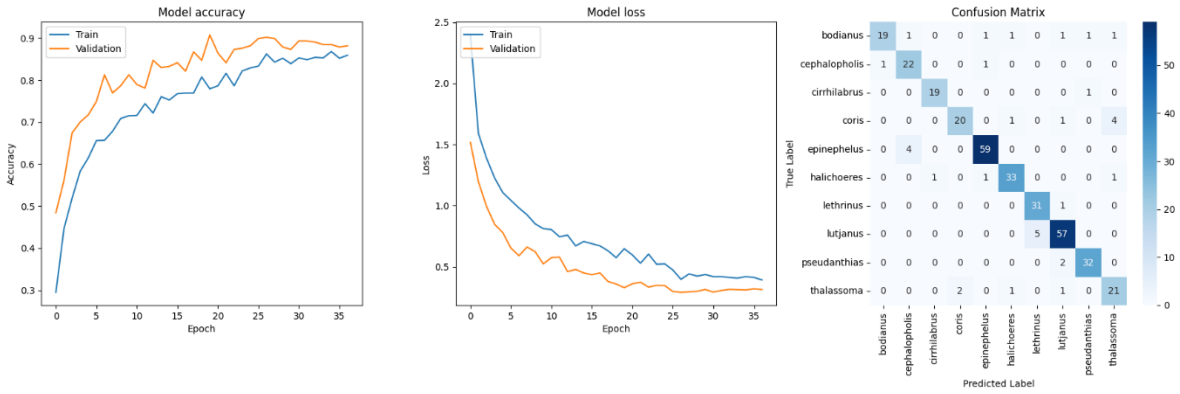

Figure 5, Confusion Matrix for DenseNet121

Xception

Xception stands for "Extreme Inception" and is an extension of the Inception architecture. It replaces the standard Inception modules with depthwise separable convolutions, leading to a more efficient network with fewer parameters.

Accuracy: 87.61%

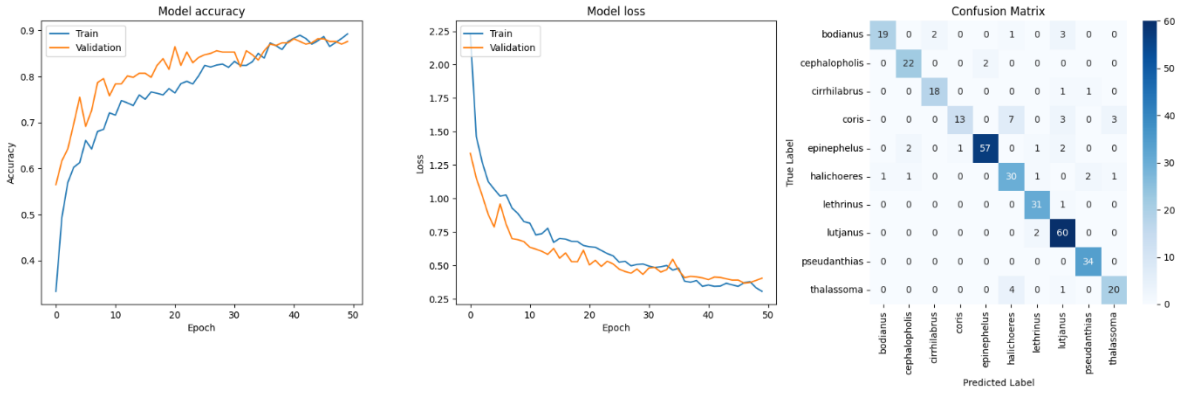

Figure 6, Confusion Matrix for Xception

InceptionV3

InceptionV3 is part of the Inception family of networks, designed to optimize computational efficiency. It uses a series of convolutions, pooling, and concatenations to create a powerful model with relatively fewer parameters.

Accuracy: 83.72%

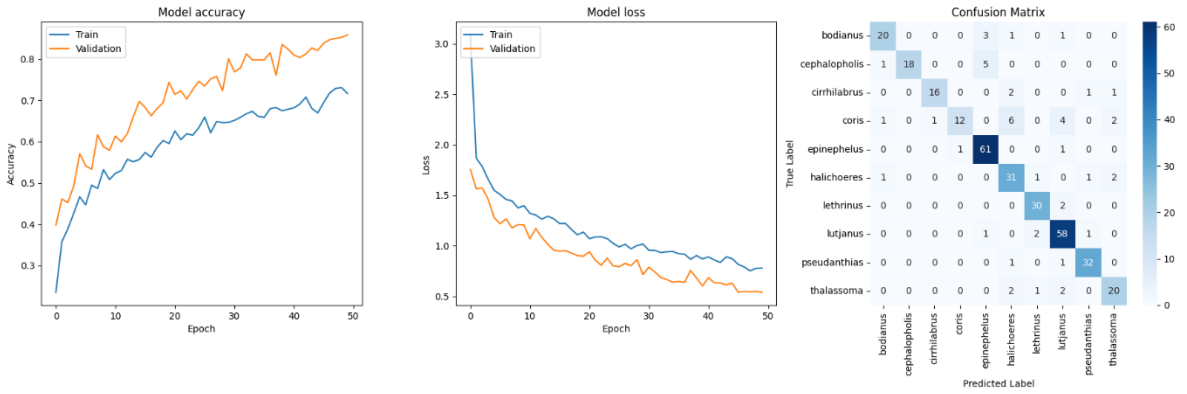

Figure 7, Confusion Matrix for InceptionV3

EfficientNet

EfficientNet is a family of models that scale up in depth, width, and resolution systematically. Despite being resource-efficient, the performance of EfficientNet on this particular dataset was lower compared to other deep learning models.

Accuracy: 75.50%

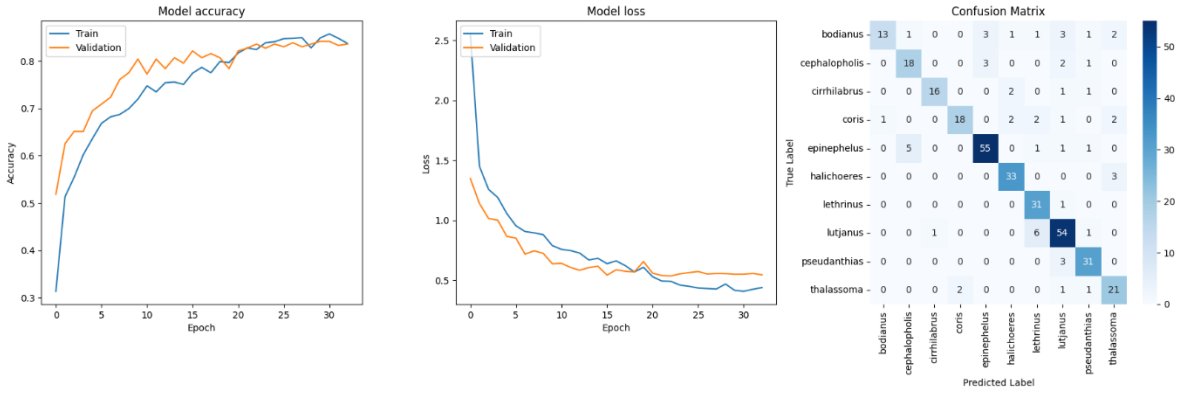

Figure 8, Confusion Matrix for EfficientNet

ResNet50

ResNet50 introduces residual connections that allow gradients to flow through the network more easily during backpropagation. This helps in training very deep networks by addressing the vanishing gradient problem.

Accuracy: 82.54%

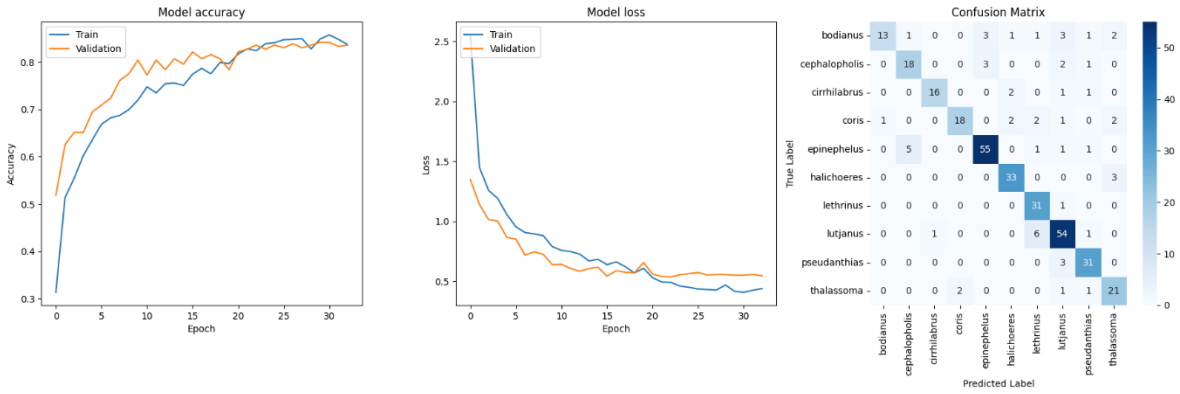

Figure 9, Confusion Matrix for ResNet50

Traditional Machine Learning Models

Support Vector Machine (SVM)

SVM is a powerful supervised learning algorithm used for classification. It finds the optimal hyperplane that maximizes the margin between different classes. Despite its effectiveness in certain scenarios, it was outperformed by deep learning models on this dataset.

Accuracy: 73.20%

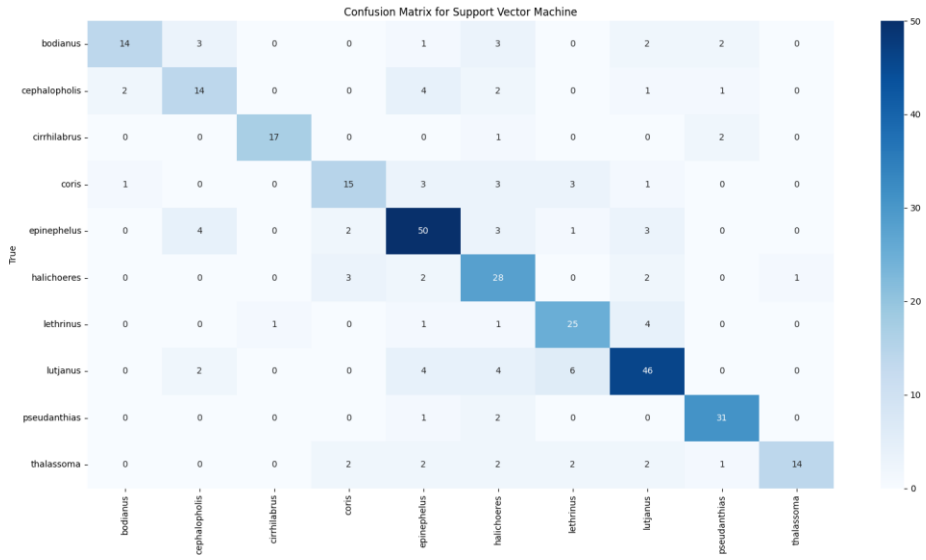

Figure 10, Confusion Matrix for Support Vector Machine

K-Nearest Neighbours (KNN)

KNN is a simple, instance-based learning algorithm that classifies data points based on the majority vote of their neighbours. It is intuitive but tends to perform poorly on large or complex datasets, as reflected in its low accuracy.

Accuracy: 56.48%

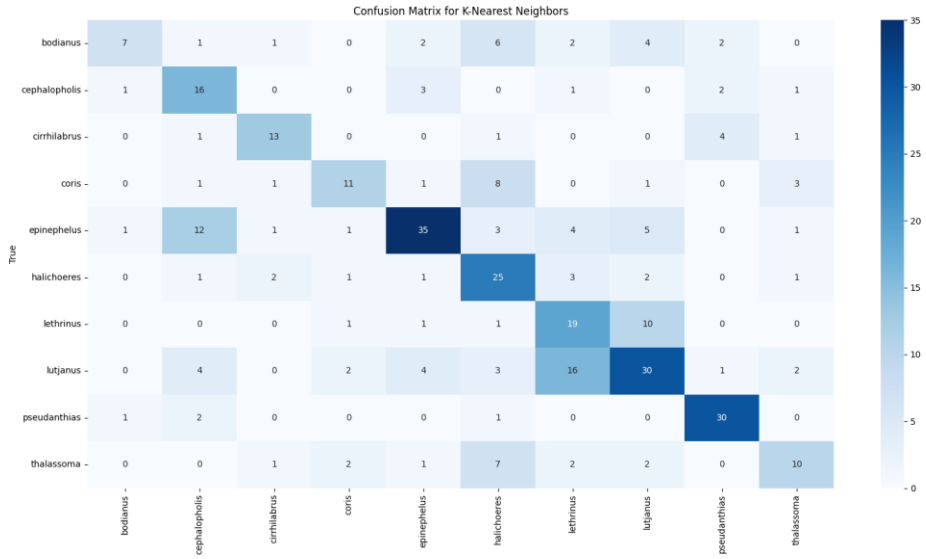

Figure 11, Confusion Matrix for K-Nearest Neighbours

Random Forest

Random Forest is an ensemble method that combines multiple decision trees to improve classification accuracy. However, it struggled with this particular dataset, possibly due to the complexity of the image data.

Accuracy: 61.67%

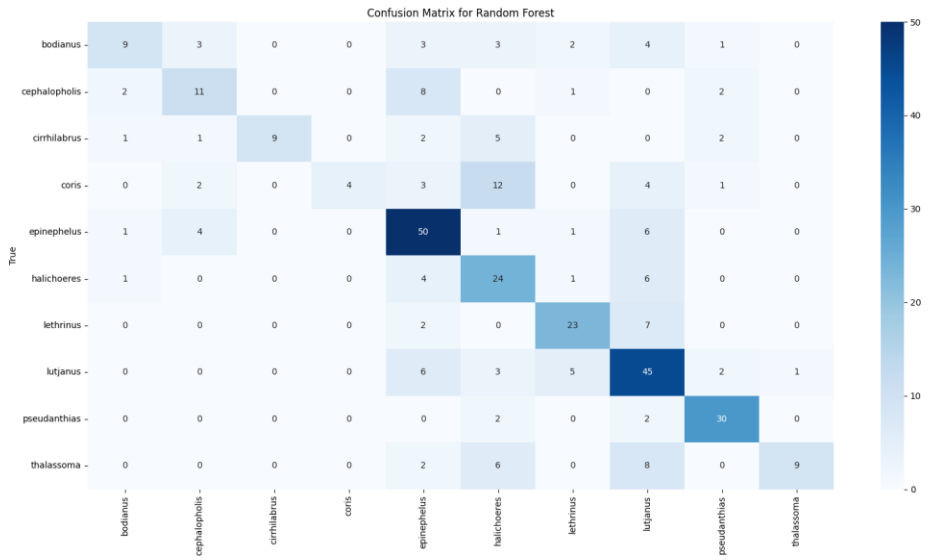

Figure 12, Confusion Matrix for Random Forest

Decision Tree

Decision Tree is a straightforward model that splits data based on feature values. While easy to interpret, its performance on this dataset was poor, likely due to overfitting and the inability to capture complex patterns.

Accuracy: 34.01%

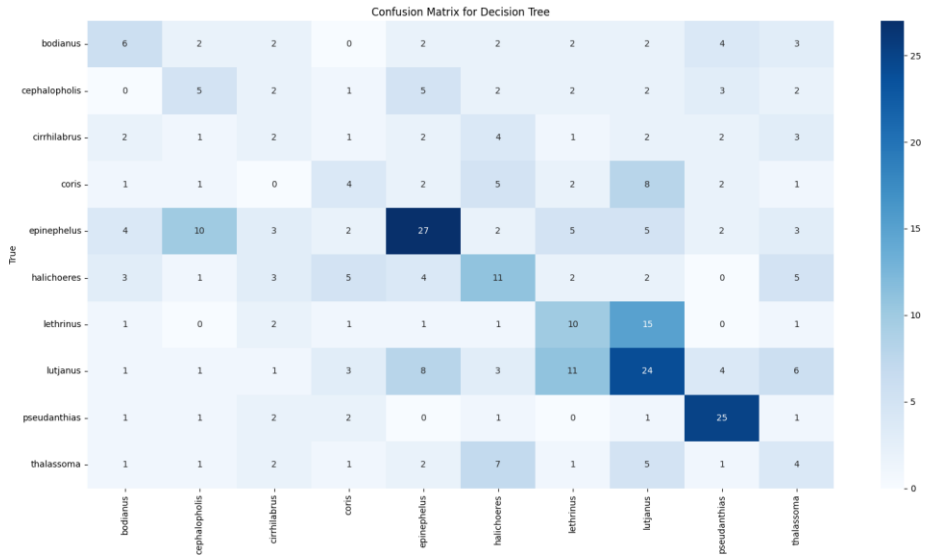

Figure 13, Confusion Matrix for Decision Tree

Naive Bayes

Naive Bayes is a probabilistic classifier based on applying Bayes' theorem. It assumes feature independence, which often doesn't hold true for image data, leading to its low accuracy in this case.

Accuracy: 33.72%

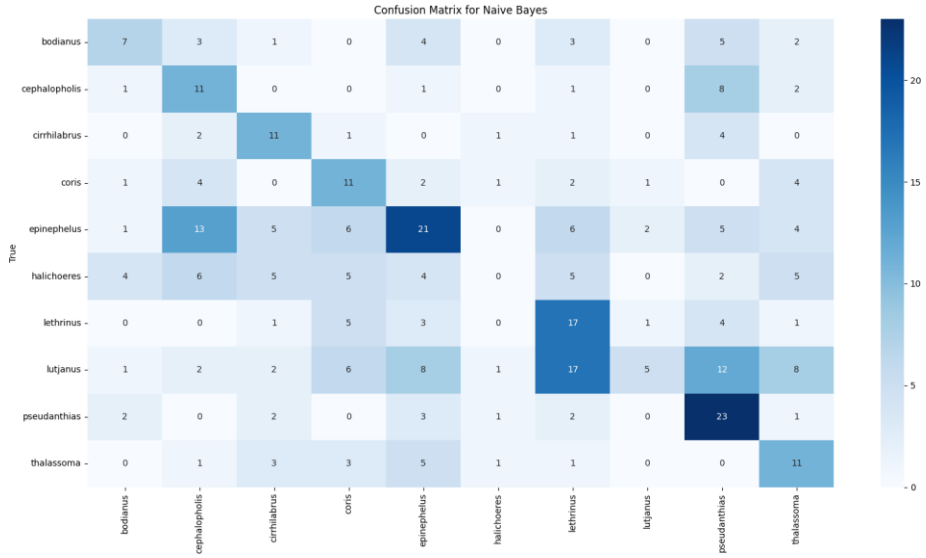

Figure 14, Confusion Matrix for Naive Bayes

Logistic Regression

Logistic Regression is a linear model used for binary classification, extended here for multiclass classification using softmax. Although simple, it performed relatively well compared to other traditional methods.

Accuracy: 72.05%

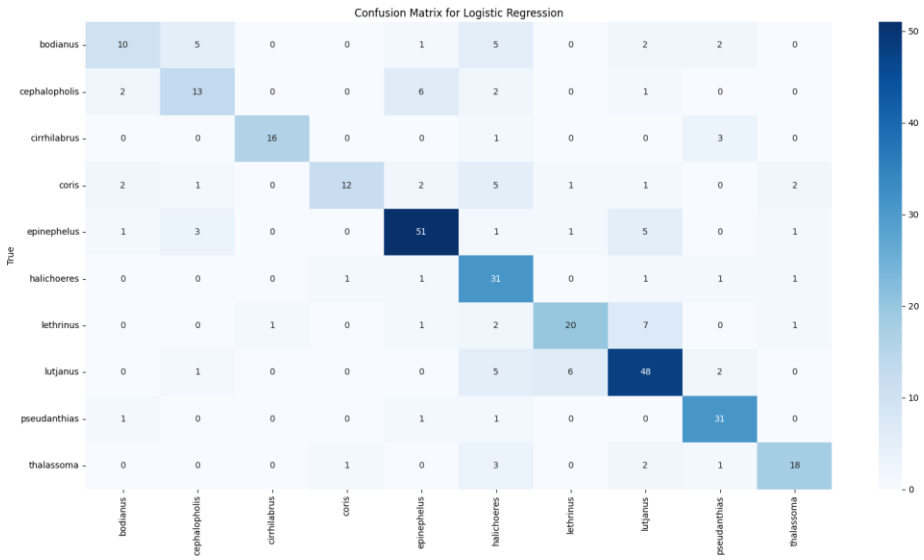

Figure 15, Confusion Matrix for Logistic Regression

Conclusion

The results clearly indicate that deep learning models, particularly DenseNet121, excel in classification tasks on this dataset. Transfer learning, which leverages pre-trained models, significantly outperforms traditional ML approaches. The custom CNN, although not as accurate as the top-performing pre-trained models, still demonstrated strong performance, underscoring the importance of carefully designed architectures.

Future Work

Hyperparameter Tuning: Further tuning of learning rates, batch sizes, and optimizers could enhance the performance of both custom and pre-trained models.

Architecture Exploration: Exploring deeper and more complex architectures, such as variations of ResNet and DenseNet, as well as emerging models like Vision Transformers.

Ensemble Methods: Combining the strengths of multiple models through ensemble techniques might further improve overall accuracy.