Model Development Phase Template

Date	10 Feb 2026
Project Title	Greenclassify: Deep Learning-Based Approach For Vegetable Image Classification

Model Selection Report

In the model selection report for future deep learning and computer vision projects, various architectures, such as CNNs or RNNs, will be evaluated. Factors such as performance, complexity, and computational requirements will be considered to determine the most suitable model for the task at hand.

Model Selection Report:

Model	Description
Custom CNN	A custom-designed CNN architecture, tailored to the specifics of this dataset and task. This model was chosen for its flexibility and potential to optimize performance for our specific dataset.
Pre-trained Models (considered but not used):	Models such as ResNet, Inception, MobileNet were initially considered. However, these were ultimately deemed less suitable because our dataset is relatively small and training a full pre-trained model may lead to overfitting or require substantial computational resources. Fine-tuning pre-trained models on the dataset remains an option for future work.

Model Used: Custom CNN

Justification:

A custom CNN architecture was selected due to several factors:

- Dataset Size: The dataset, while sizeable, is not large enough to effectively utilize a very deep pre-trained model without substantial risk of overfitting.
- Flexibility: A custom CNN provides the flexibility to tailor the architecture to the specific characteristics of the vegetable image dataset and to control the model's complexity.
- Computational Resources: The custom model's parameters are carefully designed to strike a balance between performance and computational requirements, making it efficient for training with accessible hardware.

Model Architecture Summary:

The selected custom CNN architecture comprises multiple convolutional blocks, each consisting of convolutional layers, batch normalization, max-pooling, and dropout layers. The architecture concludes with fully connected (dense) layers leading to a softmax output layer for multi-class classification. A detailed architecture summary is provided below:

Model Architecture Details:

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 150, 150, 32)	896
batch_normalization (BatchNormalization)	(None, 150, 150, 32)	128
max_pooling2d (MaxPooling2D)	(None, 75, 75, 32)	0
dropout (Dropout)	(None, 75, 75, 32)	0
conv2d_1 (Conv2D)	(None, 75, 75, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 75, 75, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 37, 37, 64)	0
dropout_1 (Dropout)	(None, 37, 37, 64)	0
conv2d_2 (Conv2D)	(None, 37, 37, 128)	73,856

batch_normalization_2 (BatchNormalization)	(None, 37, 37, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 128)	0
dropout_2 (Dropout)	(None, 18, 18, 128)	0
conv2d_3 (Conv2D)	(None, 18, 18, 256)	295,168
batch_normalization_3 (BatchNormalization)	(None, 18, 18, 256)	1,024
max_pooling2d_3 (MaxPooling2D)	(None, 9, 9, 256)	0
dropout_3 (Dropout)	(None, 9, 9, 256)	0
flatten (Flatten)	(None, 20736)	0
dense (Dense)	(None, 512)	10,617,344
dropout_4 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 15)	7,695
Total params:		10,996,219

This custom CNN architecture achieved satisfactory performance during training and validation. Further optimization and hyperparameter tuning will be explored in subsequent phases.