Model Optimization and Tuning Phase Template

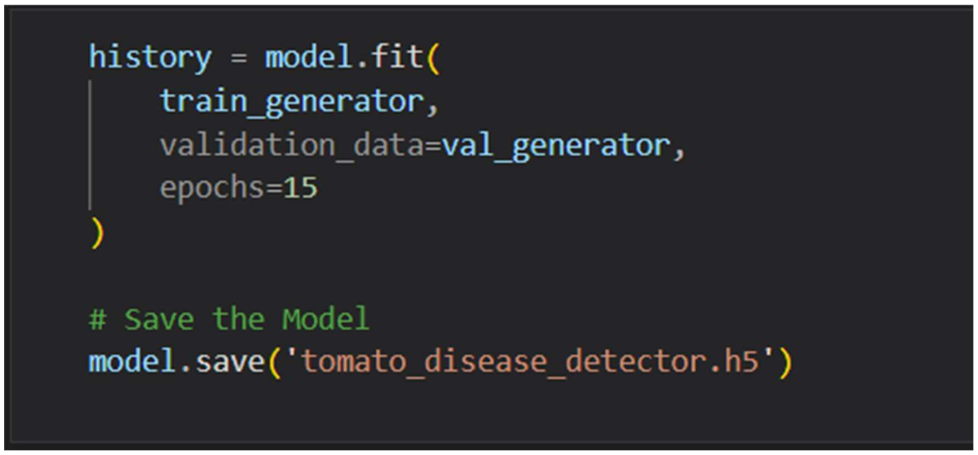
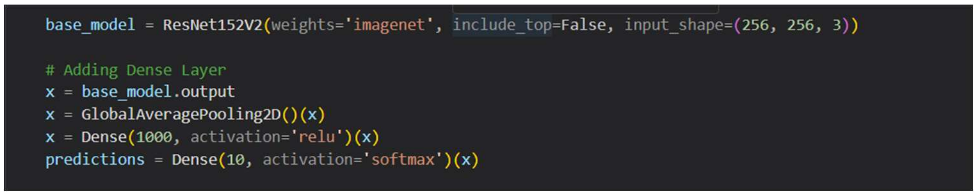
Date	11 july 2024
Team ID	SWTID1719992739
Project Title	Visual Diagnostics: Detecting Tomato Plant Diseases through Leaf Image Analysis
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

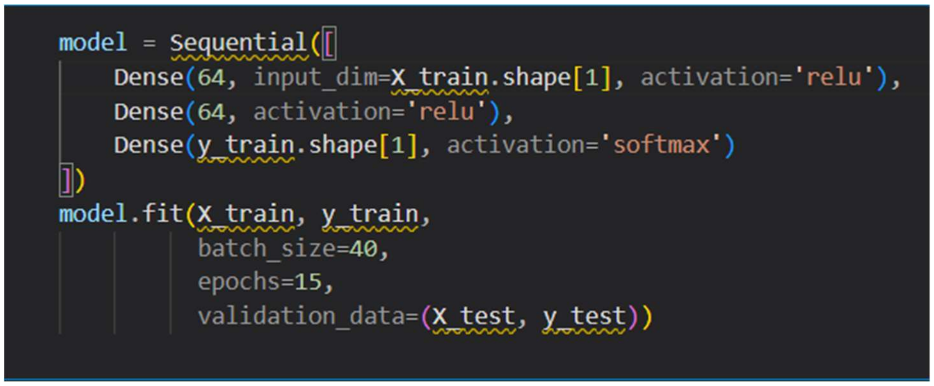
The Model Optimization and Tuning Phase involves refining neural network models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (8 Marks):

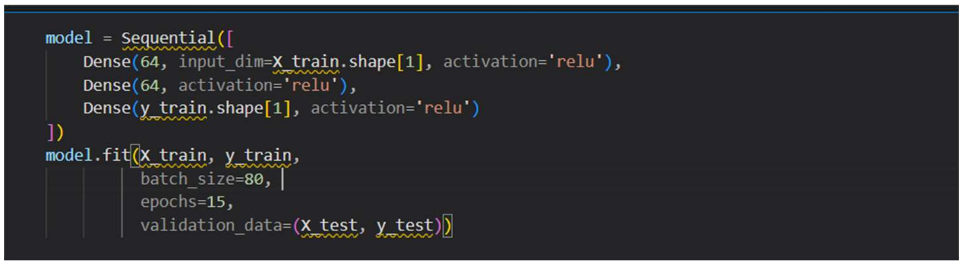
Model	Tuned Hyperparameters
ResNet152V2	Batch size: It refers to the number of training examples utilized in one iteration. It determines how many samples from the training dataset will be used to compute the error and update the model parameters. Epochs: It is one complete pass through the entire training dataset. During one epoch, the learning algorithm works through all the training samples once, and in larger datasets, this is often divided into several batches. The number of epochs determines how many times the learning algorithm will work through the entire dataset. More epochs can lead to better learning but also increases the risk of overfitting.

	Activation Function: An activation function in a neural network defines the output of a node given an input or set of inputs. It introduces non-linearity into the model, allowing the network to learn complex patterns. Without activation functions, the neural network would behave like a linear regression model, incapable of solving complex tasks.
	
	`history = model.fit(`
	`    train_generator,`
	`    validation_data=val_generator,`
	`    epochs=15`
	`)`
	
	`# Save the Model`
	`model.save('tomato_disease_detector.h5')`
	
	`base_model = ResNet152V2(weights='imagenet', include_top=False, input_shape=(256, 256, 3))`
	
	`# Adding Dense Layer`
	`x = base_model.output`
	`x = GlobalAveragePooling2D()(x)`
	`x = Dense(1000, activation='relu')(x)`
	`predictions = Dense(10, activation='softmax')(x)`

Mobilenetv2	Batch size: It refers to the number of training examples utilized in one iteration. It determines how many samples from the training dataset will be used to compute the error and update the model parameters. Epochs: It is one complete pass through the entire training dataset. During one epoch, the learning algorithm works through all the training samples once, and in larger datasets, this is often divided into several batches. The number of epochs determines how many times the learning algorithm will

	work through the entire dataset. More epochs can lead to better learning but also increases the risk of overfitting.
	
	Activation Function: An activation function in a neural network defines the output of a node given an input or set of inputs. It introduces non-linearity into the model, allowing the network to learn complex patterns. Without activation functions, the neural network would behave like a linear regression model, incapable of solving complex tasks
	
	
	`model = Sequential([`
	`    Dense(64, input_dim=X_train.shape[1], activation='relu'),`
	`    Dense(64, activation='relu'),`
	`    Dense(y_train.shape[1], activation='softmax')`
	`])`
	`model.fit(X_train, y_train,`
	`          batch_size=40,`
	`          epochs=15,`
	`          validation_data=(X_test, y_test))`

DenseNet	Batch size: It refers to the number of training examples utilized in one iteration. It determines how many samples from the training dataset will be used to compute the error and update the model parameters. Epochs: It is one complete pass through the entire training dataset. During one epoch, the learning algorithm works through all the training samples once, and in larger datasets, this is often divided into several batches. The number of epochs determines how many times the learning algorithm will work through the entire dataset. More epochs can lead to better learning but also increases the risk of overfitting. Activation Function: An activation function in a neural network defines the output of a node given an input or set of inputs. It introduces non-linearity

	into the model, allowing the network to learn complex patterns. Without activation functions, the neural network would behave like a linear regression model, incapable of solving complex tasks. 

```
model = Sequential([
    Dense(64, input_dim=X_train.shape[1], activation='relu'),
    Dense(64, activation='relu'),
    Dense(y_train.shape[1], activation='relu')
])
model.fit(X_train, y_train,
          batch_size=80,
          epochs=15,
          validation_data=(X_test, y_test))
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
ResNet152V2	We chose this model because we got an accuracy of 99.52% in a training time of around 180 minutes. We then tested the model on various images and it worked great without any overfitting whereas the model we made using dense net offered more accuracy (99.64%) but took around 240 minutes to train and was overfitting on some cases while testing on the images. Again, MobileNetV2 offered the same accuracy as ResNet152V2 but considering that it was meant for low power devices we rejected that and proceeded with ResNet152V2 As our final model.