**Plant Disease Detection Using Leaf Image Classification**

**Assem Zhaishylyk, Gaini Negizbayeva BDA-2202**

This project focuses on developing a classification model to detect sugarcane leaf diseases using the Sugarcane Leaf Disease Dataset, which includes 2526 images in five different categories: Healthy, Mosaic, RedRot, Rust, and Yellow. The goal is to build a robust model using transfer learning techniques, enhance it with additional layers like Batch Normalization and Dropout, and demonstrate improved performance over a baseline model.

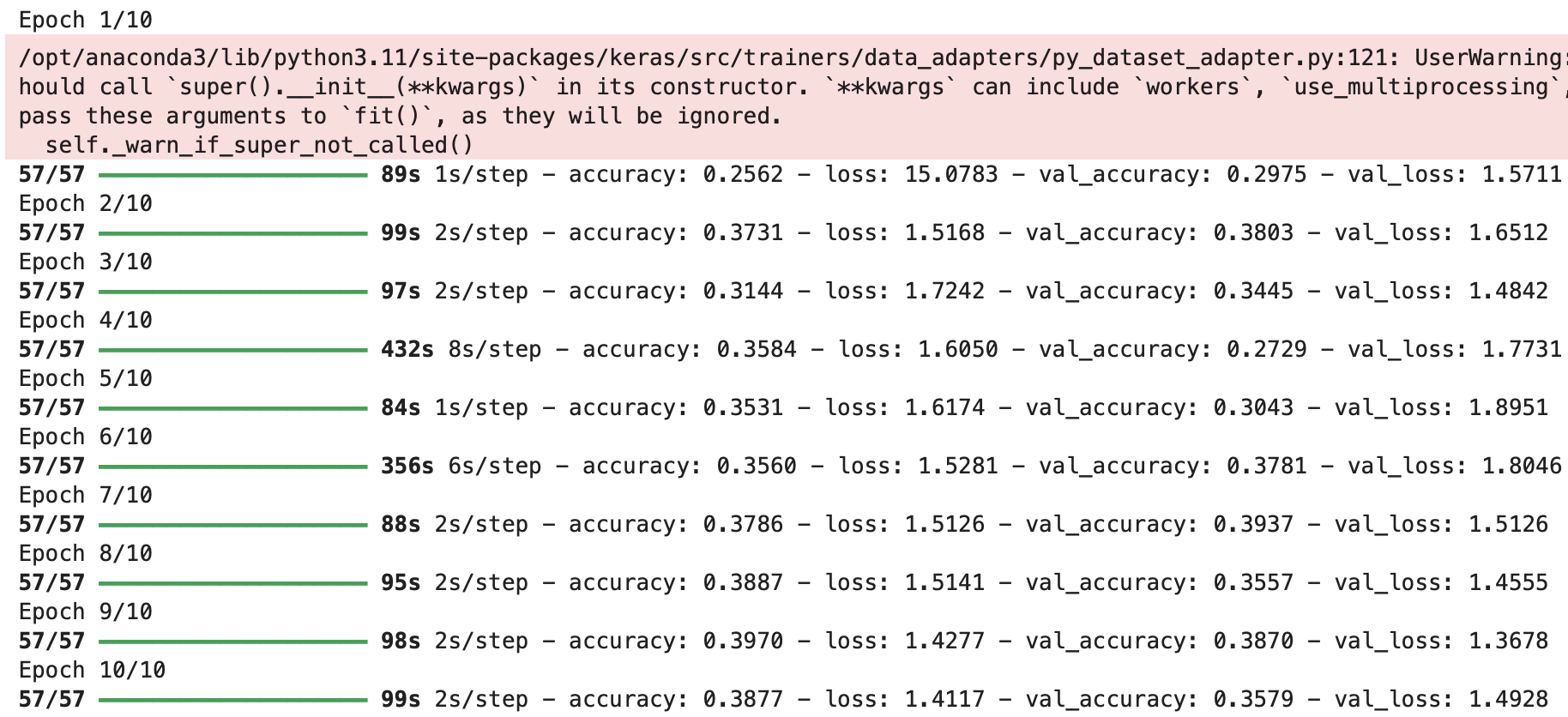
The dataset contains 5 classes (Healthy, Mosaic, RedRot, Rust, and Yellow), with images stored in separate folders for each class. The dataset was split into a training set (80%) and a validation set (20%) using Keras’ ImageDataGenerator. To avoid overfitting and improve model generalization, we applied several data augmentation techniques such as rotation, zoom, shear, and horizontal flipping.



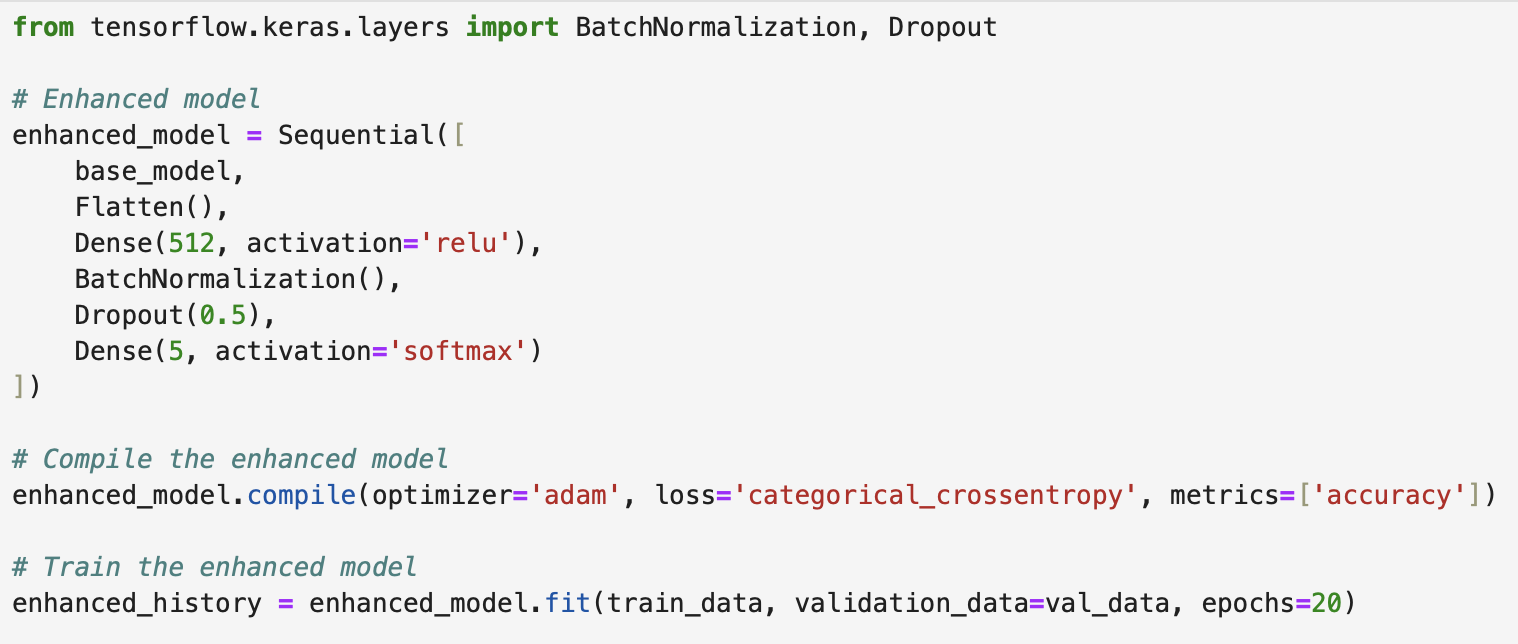
**Classification Project**

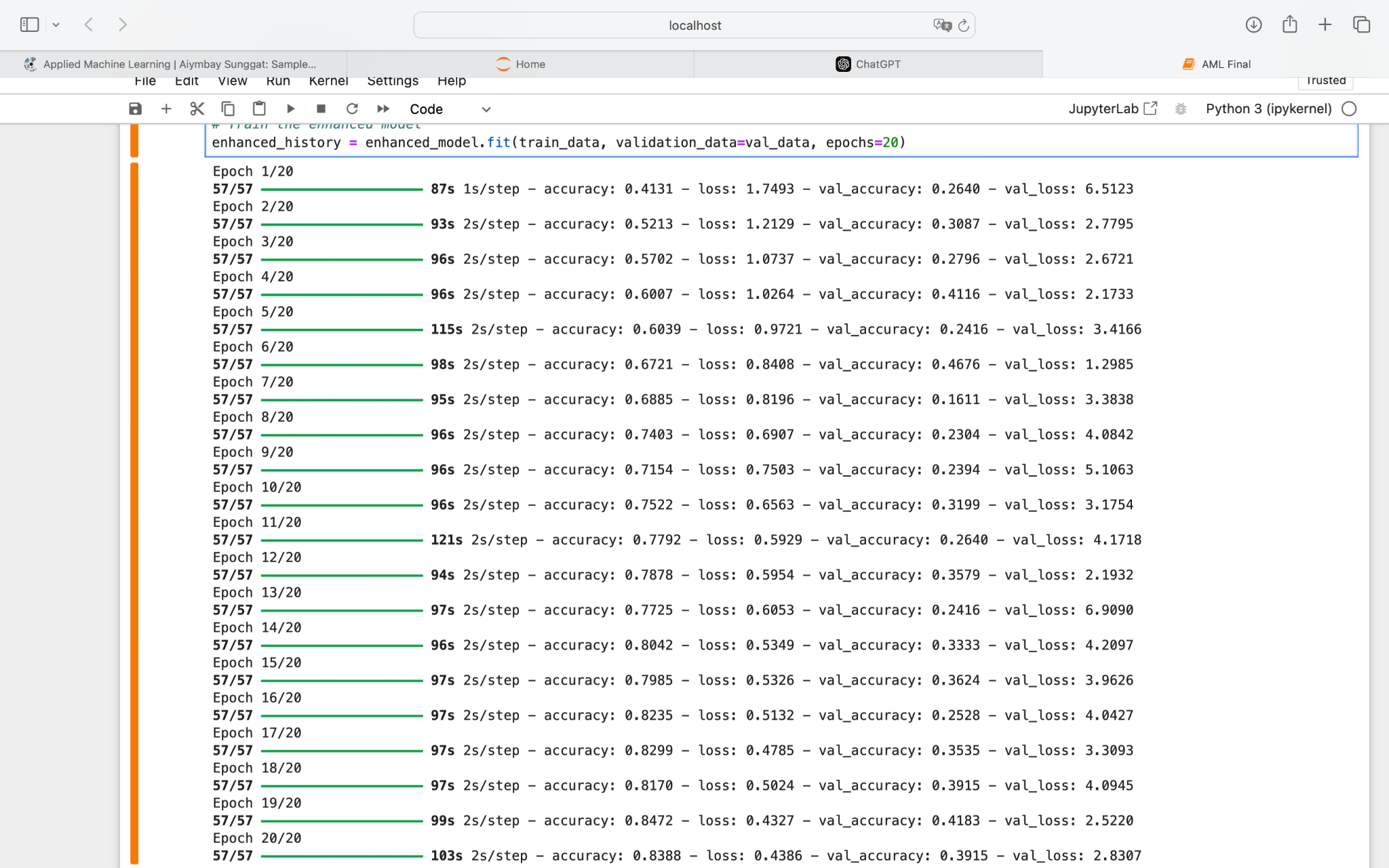
The baseline model was built using a pretrained ResNet50 network, with the final layer replaced to match the 5 output classes. All other layers were frozen to leverage the pretrained weights while only training the final classification layer. The baseline model achieved 38.77% of accuracy on the training data and 35.79% accuracy on the test data.



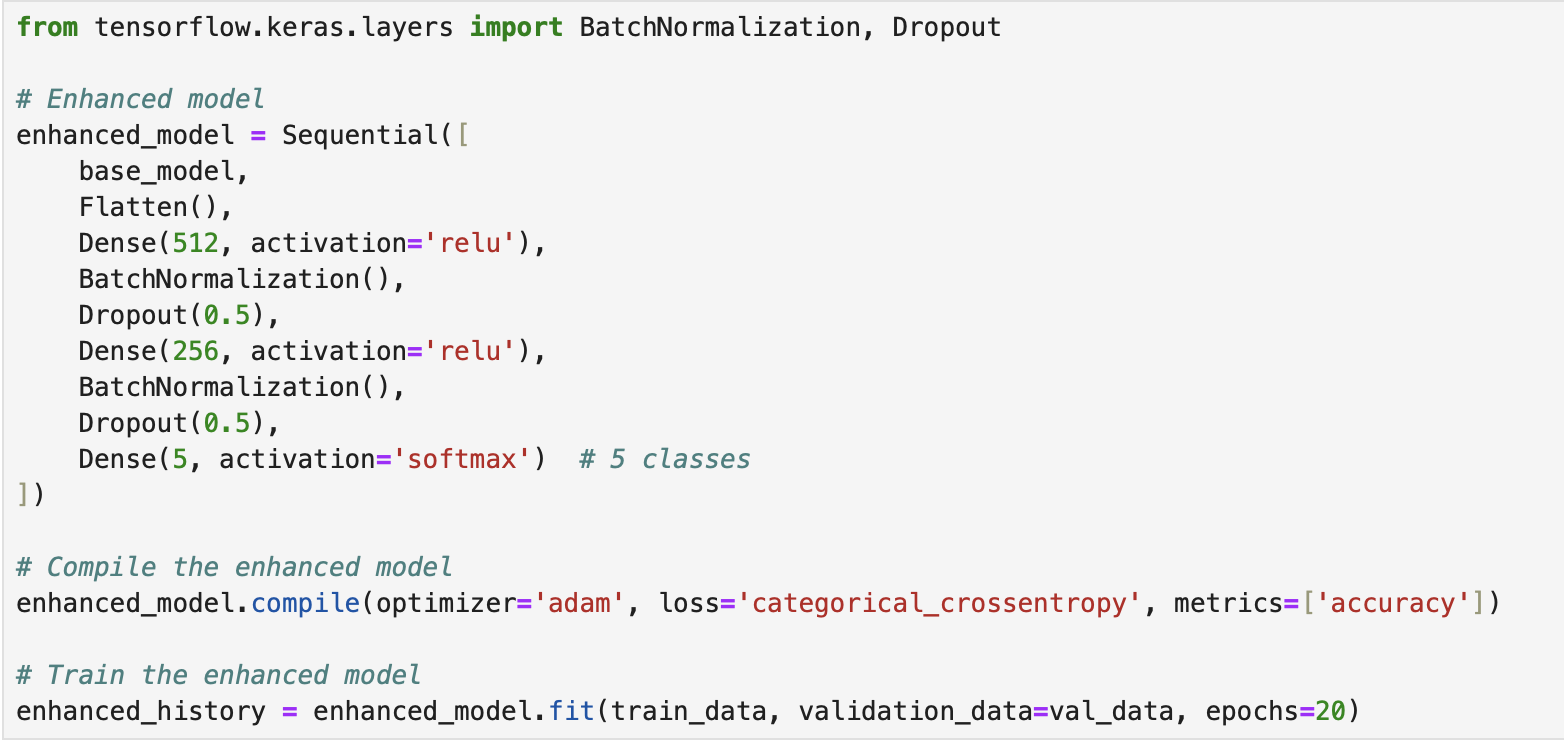


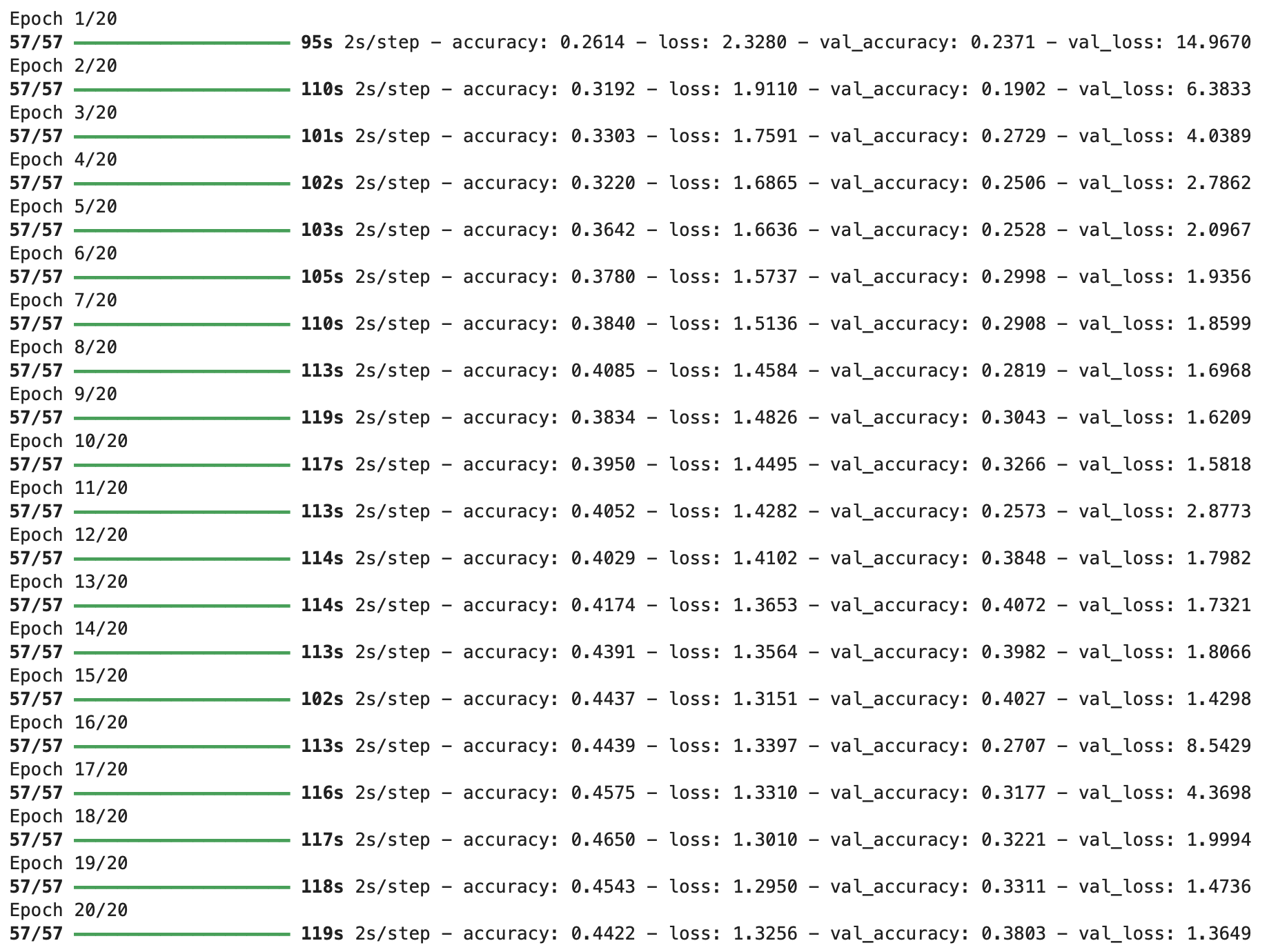
To improve performance, we added a Dense layer with 512 units, followed by Batch Normalization and Dropout layers to reduce overfitting. The enhanced model showed a higher training accuracy but required regularization techniques due to signs of overfitting on the validation set. This Model achieved accuracy of 83.88% on the training data and only 39.15% on the test data.



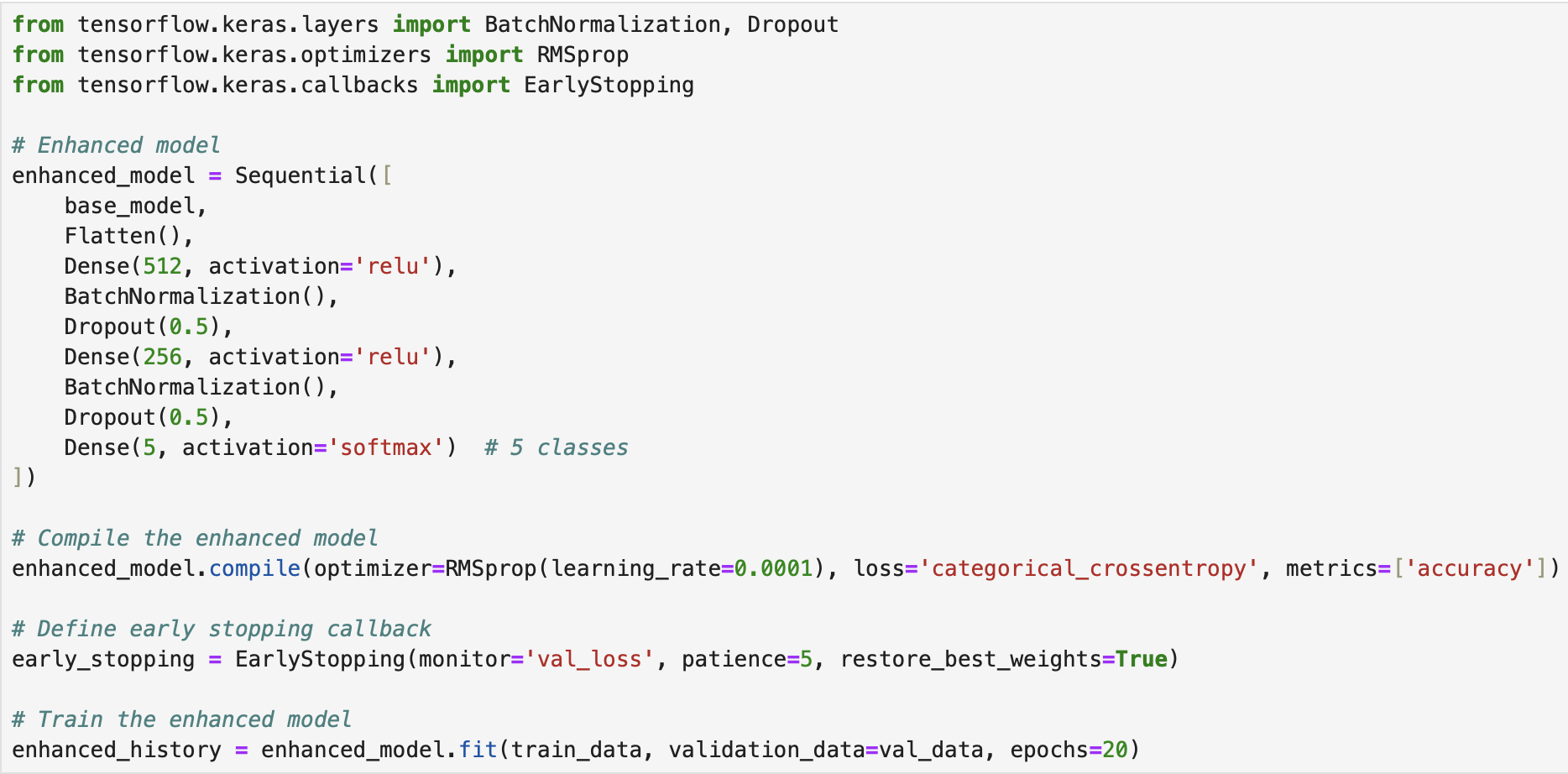


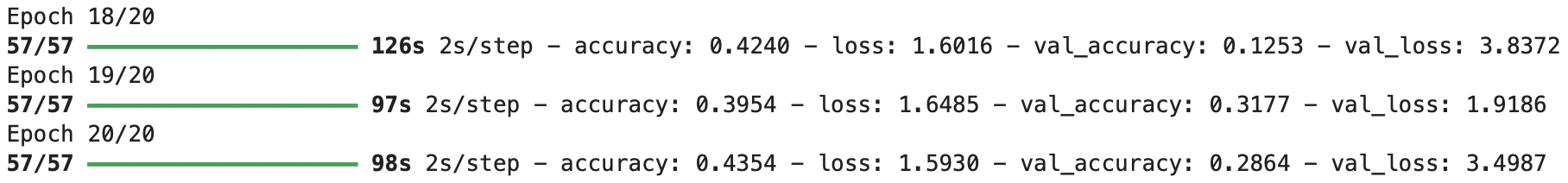
To solve the problem of overfitting more data augmentation techniques were added to make the model more robust and help it generalize better. Also, the Dropout rate was increased and the placement of Batch Normalization layers was adjusted to prevent the model from overfitting. With all these changes we could solve the overfitting problem, but the enhanced model didn’t perform much better than the baseline model.





There also was an attempt to train the model using the RMSprop optimizer with a learning rate of 0.001. Early stopping was used to avoid overfitting, with training stopped if the validation loss did not improve after 5 epochs. However, the accuracy of the trained data has not changed (43.54%), on the contrary, the model performed worse on the test data (28.64%).





The best result that the baseline model achieved was a training accuracy of 39.70% and a validation accuracy of 38.70%. The enhanced model improved the training performance but still suffered from overfitting, with the best final validation accuracy of 40.72%.

| Model | Training Accuracy | Validation Accuracy | Training Loss | Validation Loss |
| --- | --- | --- | --- | --- |
| Baseline | 0.3970 | 0.3870 | 1.4277 | 1.3678 |
| Enhanced | 0.4650 | 0.4072 | 1.3010 | 1.3649 |

One major challenge was the model overfitting the training data while performing poorly on the validation set. This was addressed by incorporating more data augmentation and increasing the dropout rate in the enhanced model. However, the limited dataset size may still be a factor in the model's generalization performance.

**Metrics and Validation**

The evaluation metrics include precision, recall, F1-score, and support for each class in the sugarcane disease detection model.

#### 

Precision measures the proportion of true positive predictions among all positive predictions made for a given class. In other words, it shows how many of the predicted positive samples were actually correct.

* RedRot: Out of all the images classified as RedRot, only 29% were correct.
* Mosaic: Only 19% of the Mosaic predictions were correct.
* Healthy: The model correctly identified 27% of the images it classified as Healthy.
* Rust: Very low precision, with only 14% correct Rust predictions.
* Yellow: 25% of the predicted Yellow leaf disease images were correct.

Recall (or sensitivity) measures the proportion of actual positive cases that were correctly identified by the model. It shows how many of the true positive samples were detected.

* RedRot: The model correctly identified 44% of actual RedRot images.
* Mosaic: Only 8% of the Mosaic images were correctly classified, showing poor performance for this class.
* Healthy: The model identified only 4% of actual Healthy images correctly.
* Rust: Only 11% of actual Rust cases were correctly detected.
* Yellow: The model performed best on Yellow, correctly identifying 51% of the actual cases.

#### 

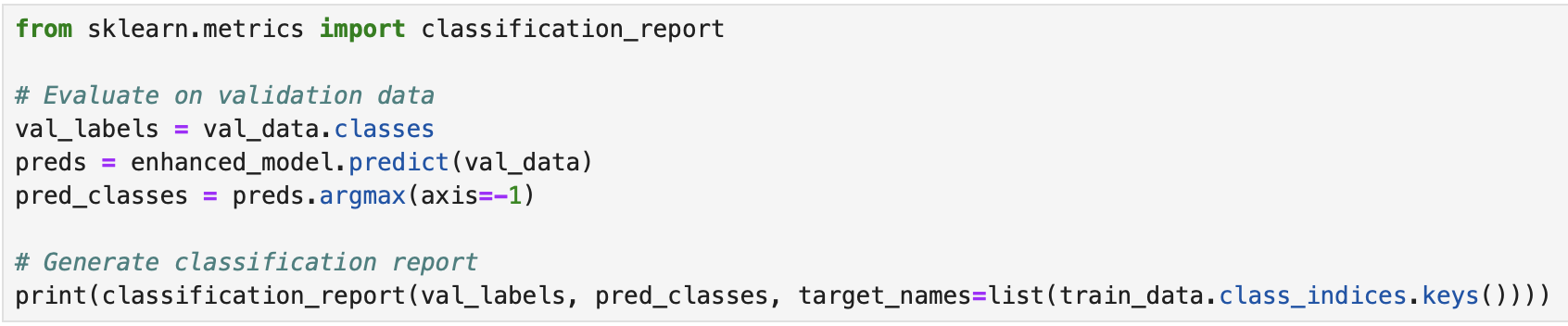
The F1-score is the harmonic mean of precision and recall, providing a balanced measure of the model’s accuracy for each class.

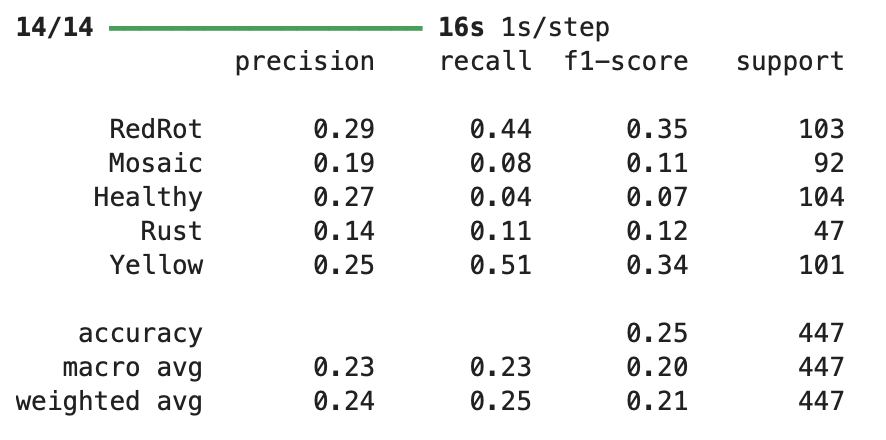
* RedRot: F1-score shows a balanced but moderate performance for the RedRot class.
* Mosaic: Very poor performance, with both precision and recall being low.
* Healthy: The model struggles significantly with the Healthy class.
* Rust: Rust also has a low F1-score, indicating poor detection.
* Yellow: Yellow has the highest F1-score, but still underperforms compared to ideal results.

#### 

Support refers to the number of actual occurrences of each class in the validation dataset: RedRot - 103 images, Mosaic - 92 images, Healthy - 104 images, Rust - 47 images, Yellow - 101 images.

Overall accuracy for the validation set is 25%, meaning that only 25% of all images were classified correctly across all classes. This is quite low, and the gap between the precision, recall, and accuracy shows that the model is struggling with generalization.





The model shows low overall performance with poor precision, recall, and F1-scores across most classes, particularly struggling with the Mosaic, Healthy, and Rust categories. The Yellow and RedRot classes have slightly better metrics but still exhibit significant room for improvement. Given the low validation accuracy of 25%, this suggests that the model is not generalizing well to unseen data and is likely overfitting the training data.

**Detection and Segmentation Project**

### **Step 1: Dataset Preparation**

To train and evaluate the models, we used a dataset organized into folders representing different plant disease categories (e.g., Yellow, Rust, RedRot, Mosaic, and healthy). The dataset was split into training and validation sets.

#### **Actions:**

* **Data Augmentation**: Applied transformations to increase the variability of the training data. Techniques included:
  + Rotation, flipping, zooming, and shifting.
  + Rescaling pixel values to the range [0, 1].
* **Validation Data Preprocessing**: Only rescaling was applied to ensure consistency during evaluation.



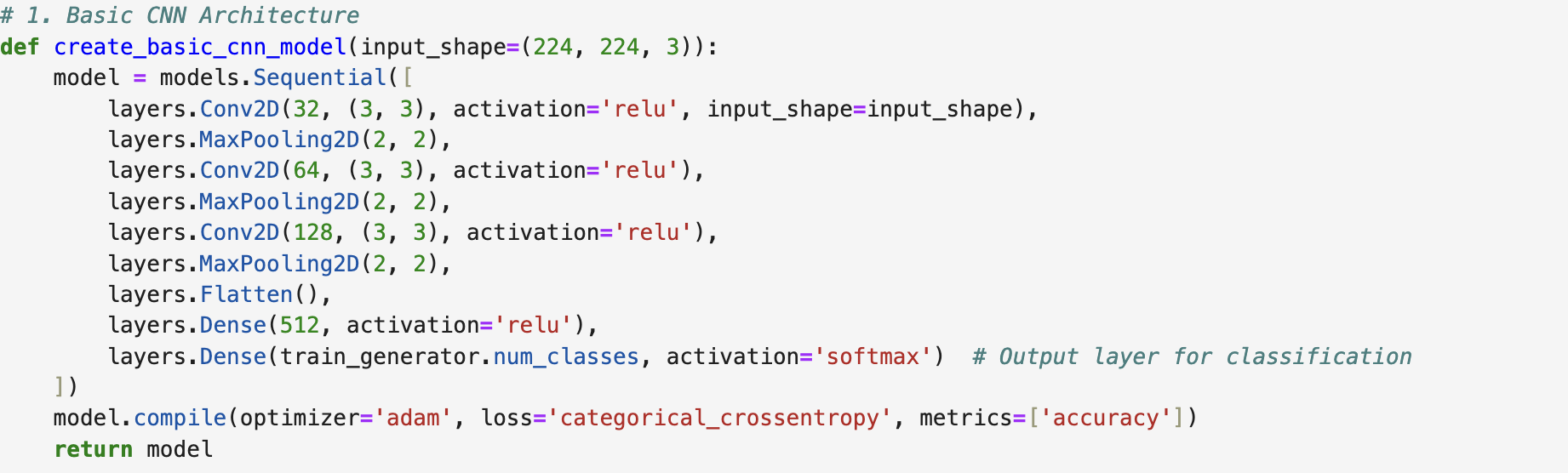
### **Step 2: Designing the Models**

Two architectures were chosen to analyze and compare their performance:

#### **2.1. Basic CNN Architecture**

A simple convolutional neural network was designed with:

* Three convolutional layers (32, 64, and 128 filters).
* Max-pooling layers after each convolutional layer.
* A dense layer with 512 neurons and a ReLU activation.
* A softmax output layer for multi-class classification.



#### **2.2. ResNet50 Architecture**

A pre-trained ResNet50 model was fine-tuned by:

* Freezing the base model layers to preserve its pre-trained knowledge.
* Adding a global average pooling layer.
* Adding a dense layer with 512 neurons for feature extraction.
* Adding a softmax layer for multi-class classification.

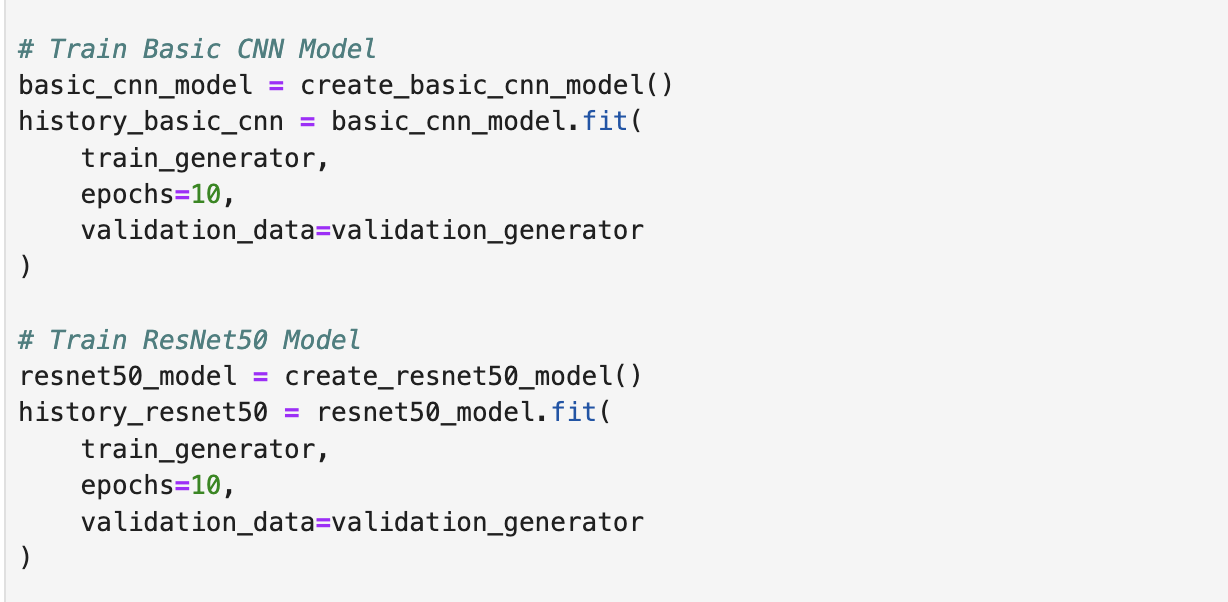


### **Step 3: Training the Models**

Both models were trained using the same dataset for 10 epochs.

#### **Process:**

1. **Model Training**: Each model was trained separately using the fit method.
2. **Validation**: Performance was monitored using validation accuracy.

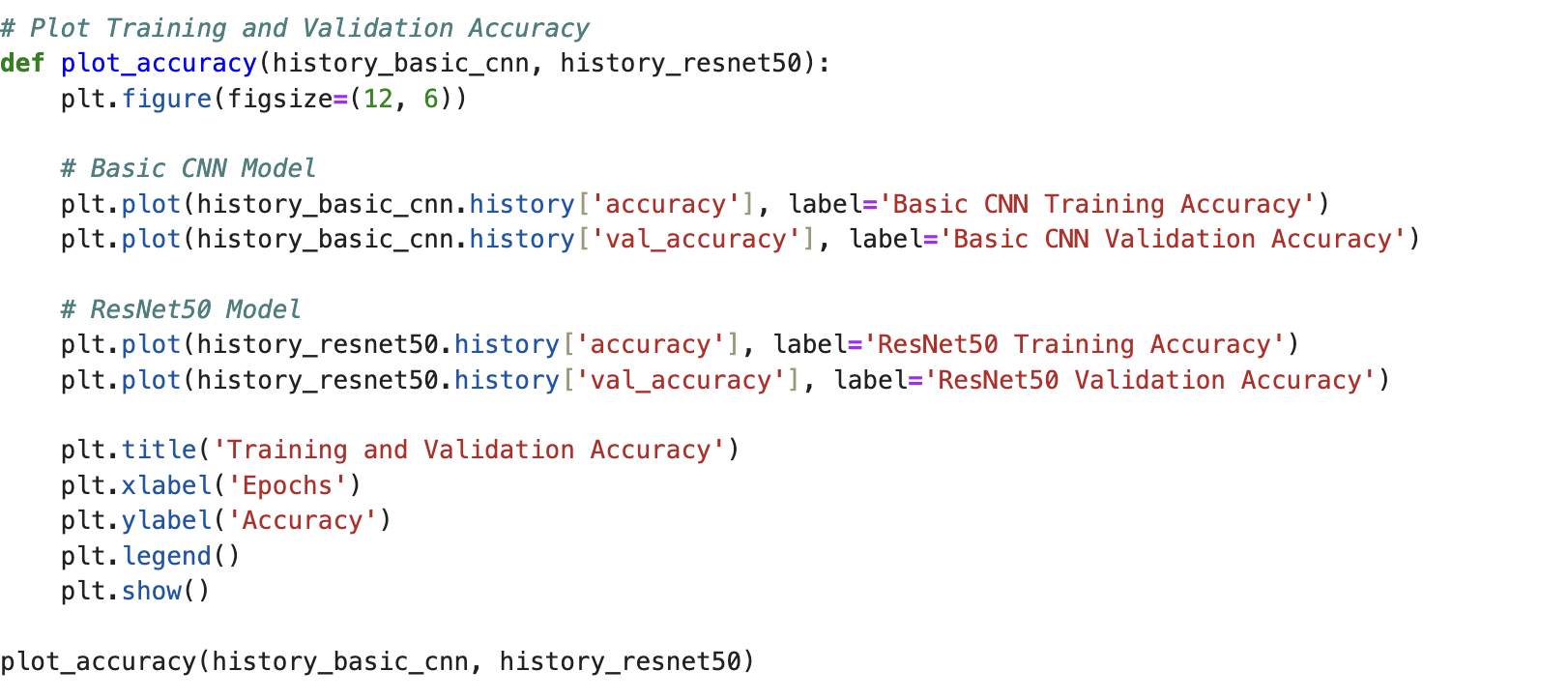


### **Step 4: Performance Comparison**

We compared the performance of both models based on validation accuracy and visualized the results.

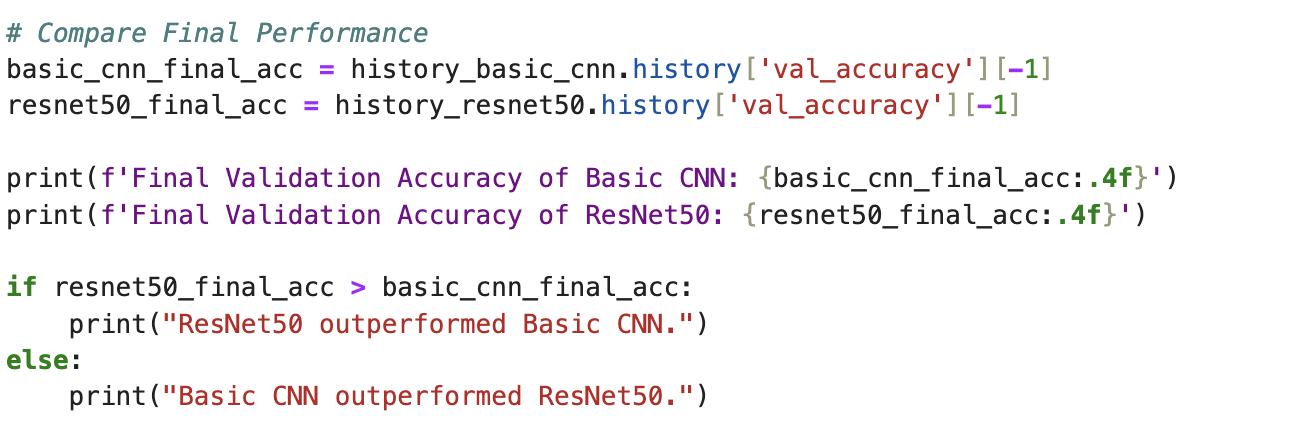
#### **4.1. Accuracy Visualization**

The training and validation accuracy for both models were plotted across all epochs to observe trends and stability.



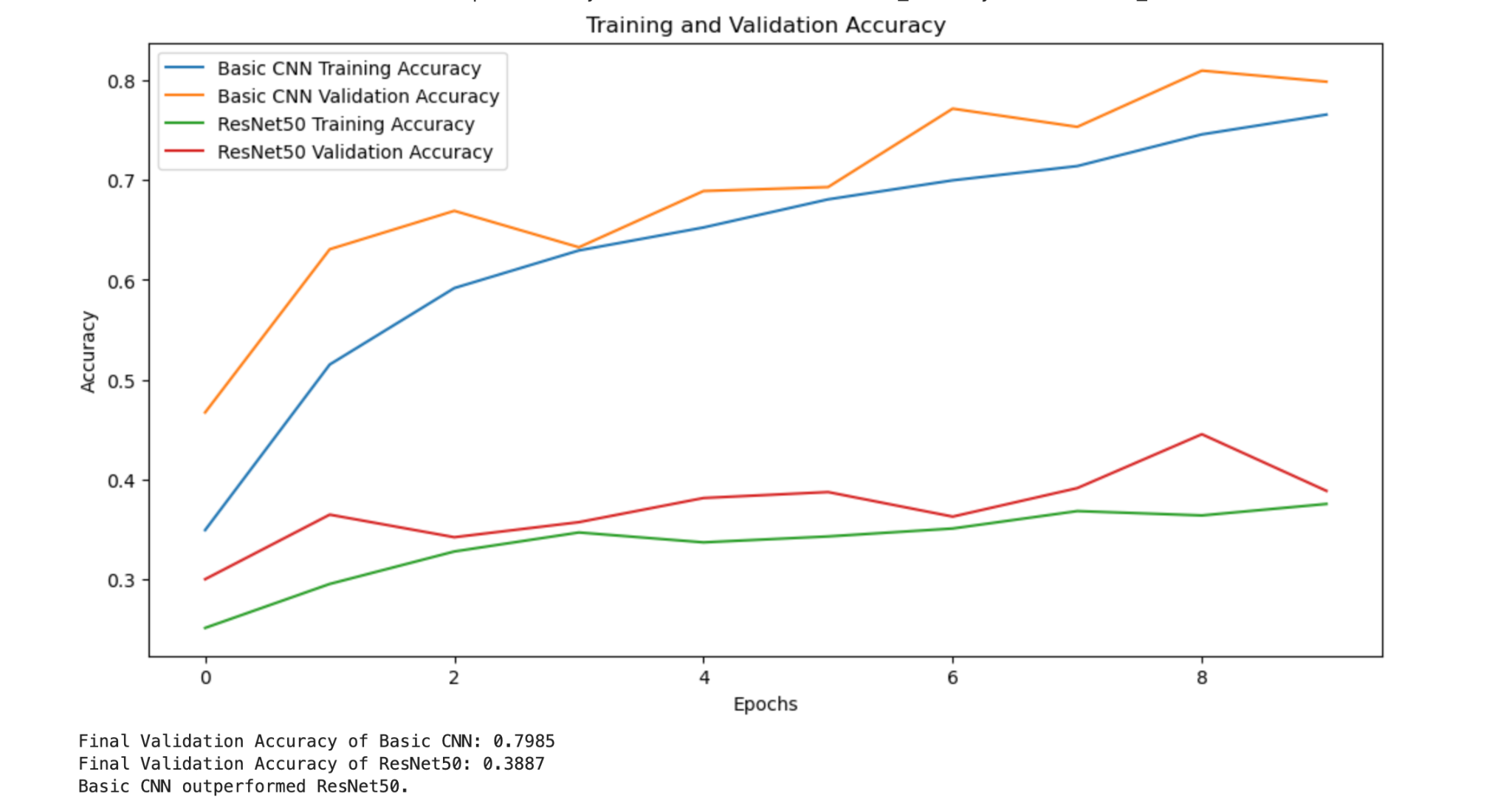
#### **4.2. Final Validation Accuracy**

The final validation accuracy of both models was extracted and compared.



### **Results and Observations:**

* The **Basic CNN** was better suited to the dataset, achieving higher validation accuracy. This indicates that the simpler architecture worked better for the given task and dataset size.
* **ResNet50** underperformed, possibly due to:
  + Limited dataset size for fine-tuning.
  + The frozen layers in ResNet50 might not have aligned well with the dataset features.
  + Overfitting or inadequate feature generalization.



### 

### 

### **Conclusion:**

1. **Model Selection**: The Basic CNN was the more effective model for this task, highlighting that simpler architectures can sometimes outperform more complex, pre-trained models.
2. **Insights on ResNet50**:
   * ResNet50's poor performance suggests the need for further tuning, such as unfreezing specific layers or using a different optimizer.
   * The dataset size and diversity may not have been sufficient to fully leverage ResNet50's pre-trained capabilities.
3. **Future Improvements**:
   * Fine-tune ResNet50 by gradually unfreezing more layers for better alignment with the dataset.
   * Experiment with other pre-trained models like EfficientNet or MobileNet.
   * Use techniques like learning rate scheduling and regularization to avoid overfitting.

This outcome emphasizes the importance of model evaluation and tuning for specific tasks, as complex models do not always guarantee superior results.