Using Deep Learning to Determine Honeysuckle Bark

Brett Huffman - CSCI 5390 - Main Project Phase III

Abstract

The objective of this project is to build a convolutional neural network which can accurately classify based on a learned images set.

This specific project will build a model capable of spotting several species of invasive Honeysuckle in the wild. The model will try to determine which species is invasive out of the many desired forest plants in the Illinois/Missouri habitat.

1 Phase 3 Objective

There are multiple objectives for Phase 3:

- To experiment first with the Mnist dataset, then our own model, to obtain the highest possible accuracy;
- To split the data into training, validation and test sets;
- Implement Earlystopping and Model Checkpointing on the validation set;
- Build an architecture that yields the highest accuracy on the validation set.

The results of all these activities are discussed in this paper.

2 Overall Problem To Be Solved

The Engineering and Biology Departments at Principia College are teaming up to build an autonomous rover that will poison unwanted species of plants.

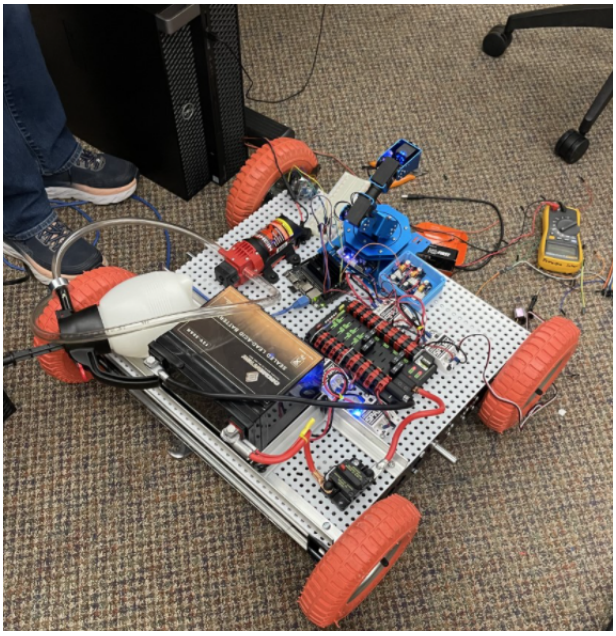

Figure 1: A view of the early rover prototype

After a year of work, they have demonstrated the ability to maneuver around a space, then when manually activated, chemically treat an unwanted plant.

The Biology department has identified a herbicide that is only poisonous to Honeysuckle – the main plant which they want to eradicate.

The problem with the herbicide is that it must be delivered into the stem. Thus, to treat a plant, the rover lowers a grinder boom which takes some of the bark off the plant. Next, a few drops of the herbicide is sprayed into the plant. Correctly applied, the plant dies within days ([**?**]).

The last big problem for the team to solve is how to autonomously determine if the plant is a target Honeysuckle.

This project is an attempt to see if the species of plant can be accurately identified from other plants in the target area.

2.1 Honeysuckle

Honeysuckle is an invasive species brought into the United States in the early 1900's as an ornamental plant. It has been used for erosion control, but quickly became invasive to many other species of native plants. It invades areas that have been disturbed such as forest fire scorched areas and flood plains. It rapidly out competes native plants for nutrients and sunshine ([**?**]).

Further, Honeysuckle produces a thick canopy that prevents sunlight from getting to lower levels of the forest and effectively chokes off new growth.

For these reasons, eradication of the honeysuckle in wild areas is an important goal for botanists ([**?**]).

3 Phase III Procedure and Results

After experimenting with the Mnist dataset, attention was focused on the Honeysuckle project. Early Stopping and Model Checkpointing were added to the model per the project requirements.

Many models were tested, as the Validation dataset results were so abjectly different than that of the Training dataset results. In every case, the best results that could be achieved with the Validation dataset was 64% accuracy.

Several models are shown here as examples of the best examples found.

3.1 Model with Best Validation Accuracy

After many iterations of different models, a best-case accuracy model was chosen. However, its accuracy is still in-question as it only ever achieved approximately 64% with Validation data.

Interestingly, as seen in image 2, while the accuracy has topped out, the Validation loss continues to climb. That indicates overfitting has occurred.

To counter overfitting, in later models the number of Convolutional Layers was increased and number of Dense Layers were decreased. This helped to put the Training and Validation

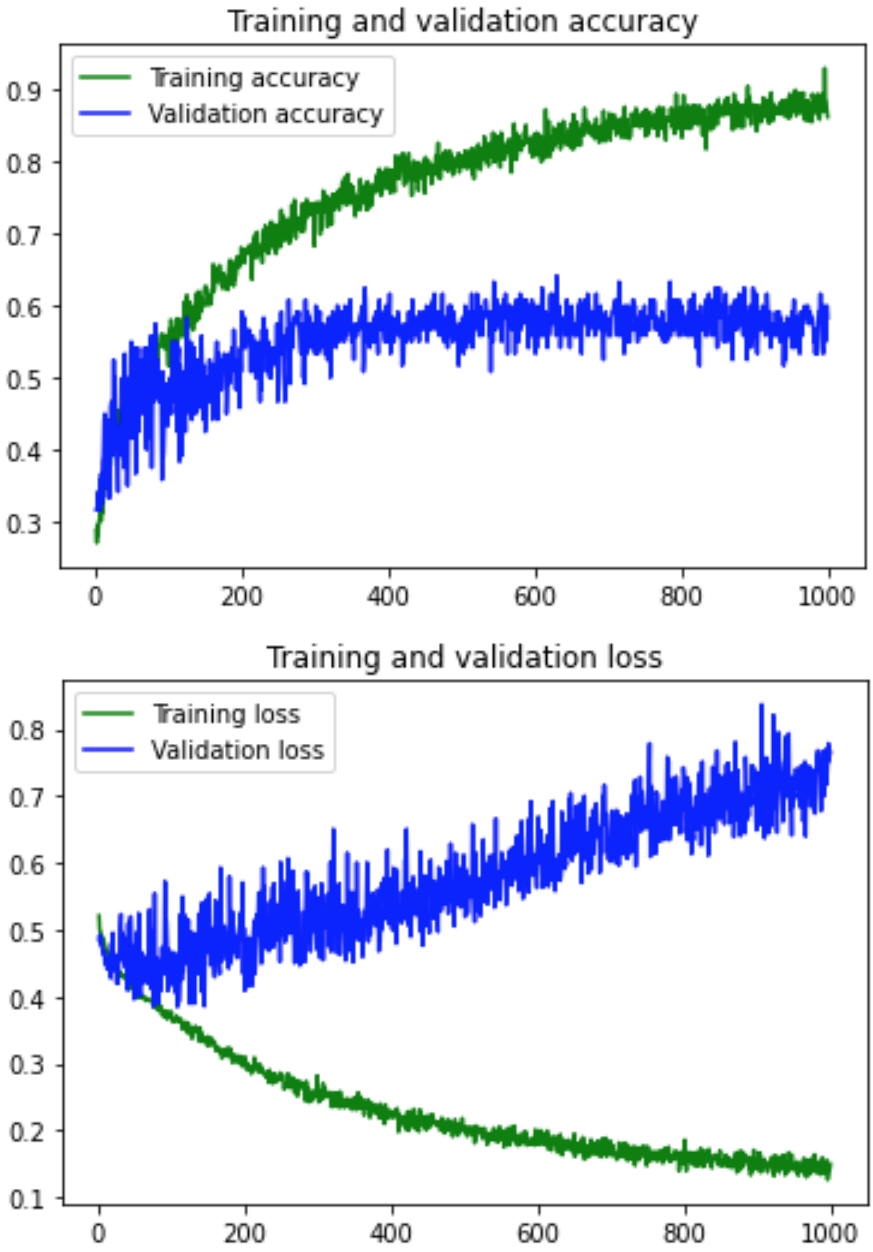

Figure 2: Model with Best Validation Accuracy

Loss in closer harmony with each other. However, the overall accuracy was severely hit.

Accuracy: 0.65625

Precision: 0.6416666666666666

Recall: 0.5566666666666668

F1: 0.5497360703812316

3.2 Model with Good Loss Profile

Another interesting model was one which did not yield as good of accuracy, but the Validation Loss diagram looks much better. It follows the Training Loss closely at least for the first fifty epochs. This behavior is shown in Figure 5.

The resulting validation metrics for this model were:

Accuracy: 0.4916666666666667

Precision: 0.4666666666666667

Recall: 0.31666666666666665

F1: 0.27999999999999997

Obviously, with early stopping the model never gets a chance to extend out to achieve a better accuracy. If that feature is turned off, the resulting 400-epoch model is somewhat more favorable:

Accuracy: 0.53125

Precision: 0.6178571428571429

Recall: 0.47174603174603175

F1: 0.488515406162465

3.3 Representative Model

In most every other case, the models produced were worse than the other two presented here. They always overfit very early and had very poor performance on Test data.

A good representative is this model which has 5 Convolution Layers followed by 1 Dense Layer. The metrics for this model were not as good as prior models:

Best Accuracy Model

Layer (**type**)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
sequential (Sequential)	(None, 150, 150, 3)	0
conv2d (Conv2D)	(None, 148, 148, 4)	112
max_pooling2d (MaxPooling2D)	(None, 74, 74, 4)	0
conv2d_1 (Conv2D)	(None, 72, 72, 8)	296
max_pooling2d (MaxPooling2D)	(None, 36, 36, 8)	0
conv2d_2 (Conv2D)	(None, 34, 34, 16)	1168
max_pooling2d (MaxPooling2D)	(None, 17, 17, 16)	0
conv2d_3 (Conv2D)	(None, 15, 15, 32)	4640
max_pooling2d (MaxPooling2D)	(None, 7, 7, 32)	0
dropout (Dropout)	(None, 7, 7, 32)	0
flatten (Flatten)	(None, 1568)	0
dense (Dense)	(None, 5)	7845

Total params: 14,061

Trainable params: 14,061

Non-trainable params: 0

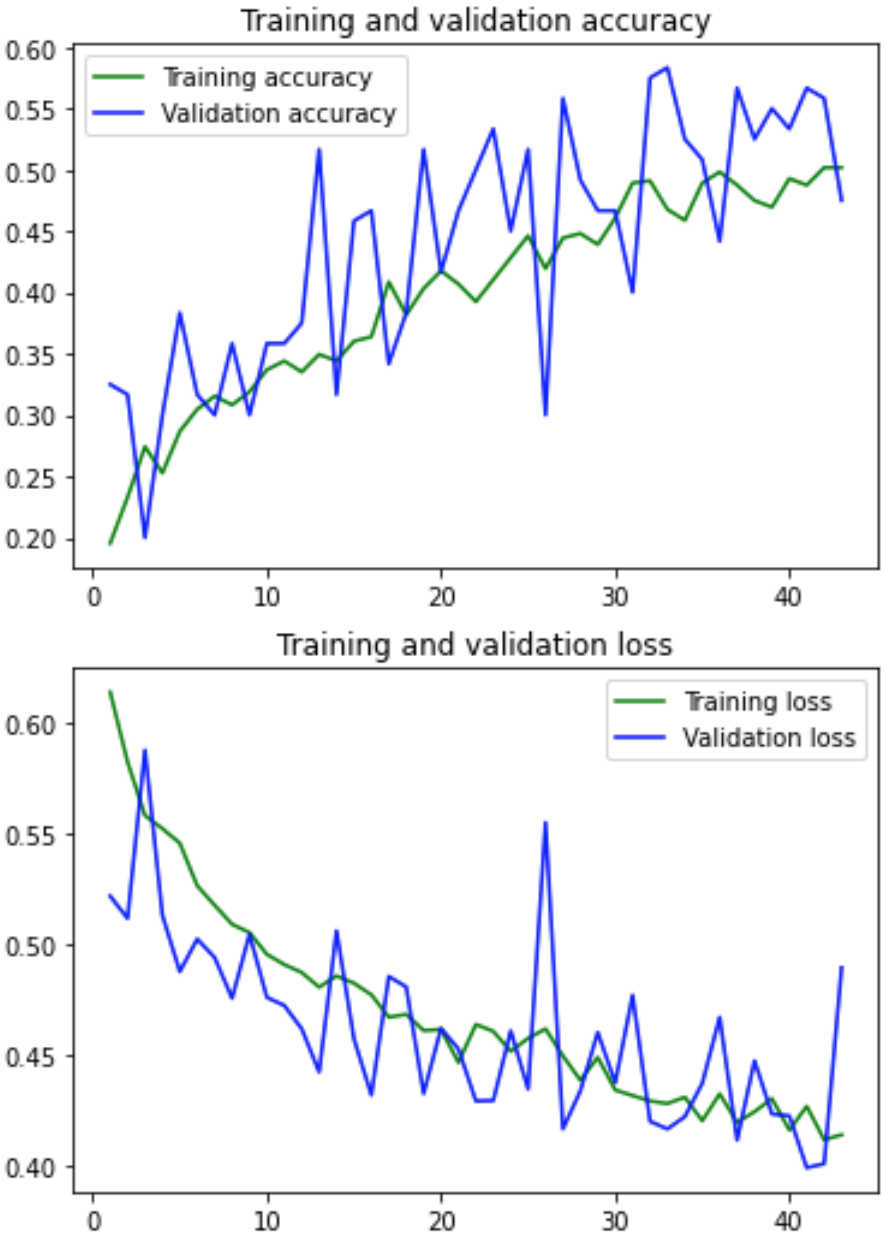

Figure 3: Model 2 Validation Accuracy and Loss

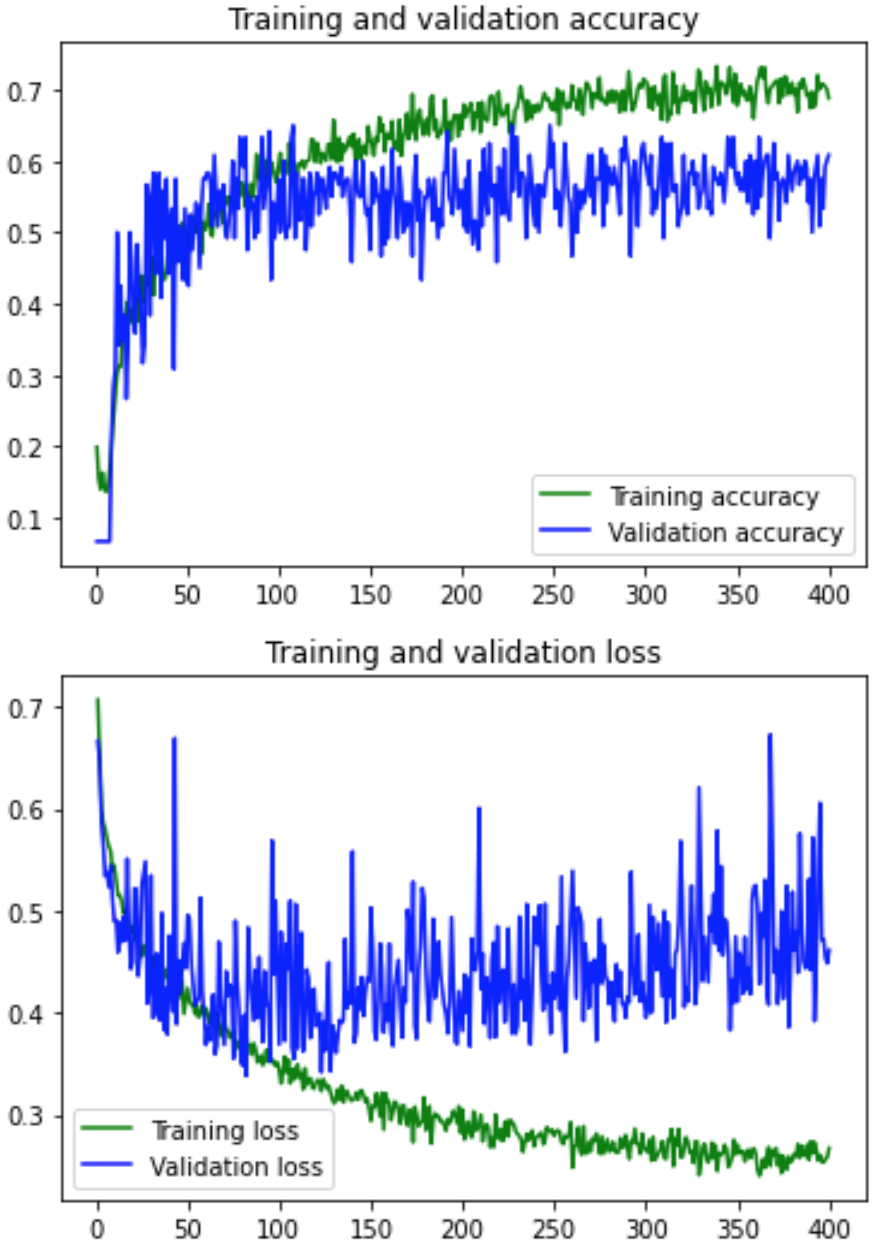

Figure 4: Model 2 400 Epoch Validation Accuracy and Loss

Best Validation Loss Model

Layer (**type**)	Output Shape	Param #
input_3 (InputLayer)	[(None, 150, 150, 3)]	0
sequential_2 (Sequential)	(None, 150, 150, 3)	0
conv2d_7 (Conv2D)	(None, 148, 148, 16)	448
max_pooling2d (MaxPooling2D)	(None, 74, 74, 16)	0
conv2d_8 (Conv2D)	(None, 72, 72, 32)	4640
max_pooling2d (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_9 (Conv2D)	(None, 35, 35, 64)	8256
max_pooling2d (MaxPooling2D)	(None, 17, 17, 64)	0
dropout_3 (Dropout)	(None, 17, 17, 64)	0
flatten_2 (Flatten)	(None, 18496)	0
dense_3 (Dense)	(None, 16)	295952
dropout_4 (Dropout)	(None, 16)	0
dense_4 (Dense)	(None, 5)	85

Total params: 309,381

Trainable params: 309,381

Non-trainable params: 0

Accuracy: 0.5416666666666666

Precision: 0.5366666666666667

Recall: 0.5428571428571429

F1: 0.5162337662337662

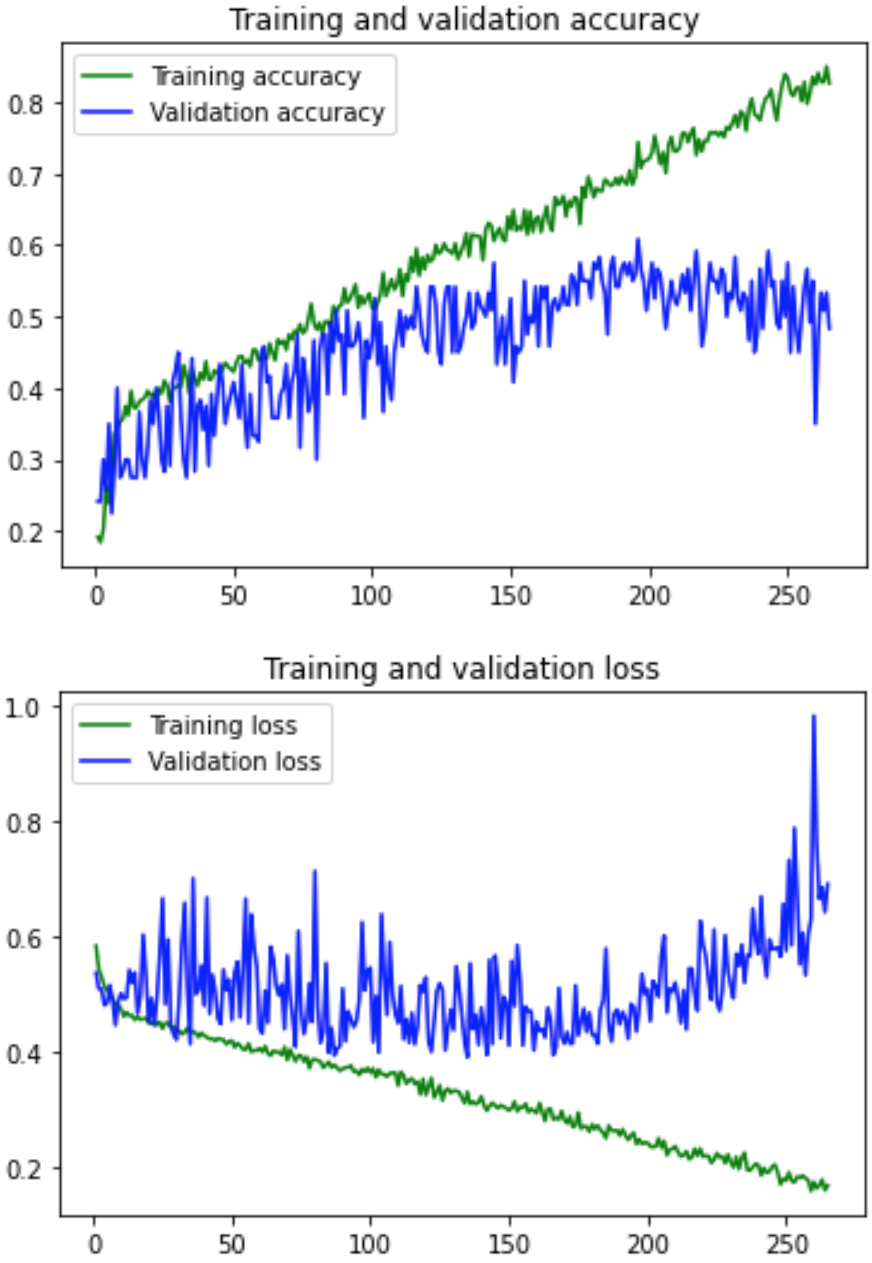

Figure 5: Representative Model Accuracy and Loss

Representative Model

Layer (**type**)	Output Shape	Param #
input_7 (InputLayer)	[(None, 150, 150, 3)]	0
sequential_6 (Sequential)	(None, 150, 150, 3)	0
conv2d_25 (Conv2D)	(None, 148, 148, 8)	224
max_pooling2d_23 (MaxPool2D)	(None, 74, 74, 8)	0
conv2d_26 (Conv2D)	(None, 72, 72, 16)	1168
max_pooling2d_24 (MaxPool2D)	(None, 36, 36, 16)	0
conv2d_27 (Conv2D)	(None, 34, 34, 32)	4640
max_pooling2d_25 (MaxPool2D)	(None, 17, 17, 32)	0
conv2d_28 (Conv2D)	(None, 15, 15, 64)	18496
max_pooling2d_26 (MaxPool2D)	(None, 7, 7, 64)	0
conv2d_29 (Conv2D)	(None, 5, 5, 128)	73856
dropout_7 (Dropout)	(None, 5, 5, 128)	0
flatten_4 (Flatten)	(None, 3200)	0
dense_7 (Dense)	(None, 8)	25608
dense_8 (Dense)	(None, 5)	45

Total params: 124,037

Trainable params: 124,037

Non-trainable params: 0

4 Conclusion

The third phase of this project was by-far the most challenging as getting the model to produce good validation accuracy results were unobtainable. It is believed that one main problem is that more training images are needed.

Currently, there are 840 images to be shared between training, validation and testing. The goal will be to nearly double the number of training images in the coming days. A college biologist will be taking the author on a campus-wide trip to identify trees for pictures.