

# Using Deep Learning to Determine Honeysuckle Bark

Brett Huffman - CSCI 5390 - Main Project Phase V

## **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 5 Objective**

In Phase 5, the following data regularization techniques were explored and the main model's accuracy were studied:

- Batch Normalization;
- Dropout;
- L1 and L2 Regularization

The results of all these Regularization techniques are discussed in this paper. In addition, a model combining all these techniques was explored.

## **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.

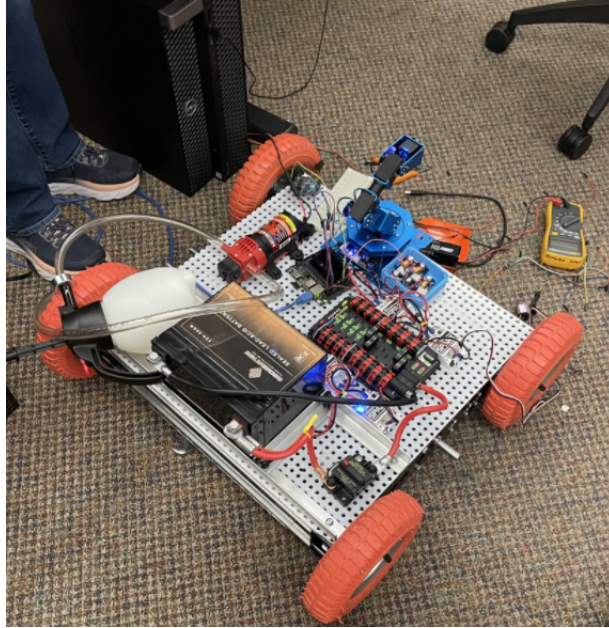


Figure 1: A view of the early rover prototype

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 ([Web20]).

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 ([Wik22]).

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 ([oC20]).

### 3 Phase V Procedure and Results

Taking the best model from Phase IV, a baseline was established of both Accuracy and F1 scores. This baseline shown in Table 1 will be used to evaluate all Regularization techniques in this study.

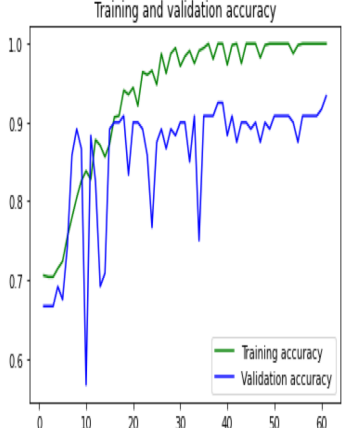
Value Studied	Training Notes	Visualization
No normalization techniques added	<ul style="list-style-type: none"> <li>• Baseline training time</li> <li>• Trained in 98 epochs</li> <li>• Reached 94% Accuracy</li> </ul>	 <p>The graph displays two metrics over 60 epochs. The green line represents training accuracy, which begins at approximately 0.7, increases steadily to about 0.95 by epoch 20, and then continues to rise with minor fluctuations, reaching nearly 1.0 by epoch 60. The blue line represents validation accuracy, starting at approximately 0.65, showing significant initial volatility with a sharp dip to 0.6 around epoch 10, then rising to about 0.9 by epoch 20. It remains relatively stable between 0.85 and 0.95 for the remainder of the training process.</p>

Table 1: Baseline Analysis

After establishing a baseline, each Regularization technique was accomplished against the model in an attempt to find the best parameters of each. Details of each attempt will be found in the following subsections:

### 3.1 Batch Normalization

Batch Normalization is the technique of adding a special layer behind either Dense, Convolutional (or both) layers. It must be applied before the activation function. Interestingly, it also takes the place of having to include the Bias vector. The code for one layer of Batch Normalization is seen in figure ??.

```
x = layers.Conv2D(filters=32, kernel_size=(3, 3), use_bias=False)(x)
x = layers.BatchNormalization()(x)
x = layers.Activation("relu")(x)
```

Value Studied	Training Notes	Visualization
Batch Normalization added to all layers except dense	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 189 epochs</li> <li>• Reached 91% Accuracy</li> </ul>	<p>The graph shows training accuracy (green line) rising from approximately 0.7 to 1.0 within the first 25 epochs and remaining stable. Validation accuracy (blue line) starts at 0.7, drops to 0.4, then rises to fluctuate between 0.8 and 0.9 for the remainder of the 189 epochs.</p>
Batch Normalization added to all layers except dense	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 109 epochs</li> <li>• Reached 94% Accuracy</li> </ul>	<p>The graph shows training accuracy (green line) rising from approximately 0.7 to 1.0 within the first 10 epochs and remaining stable. Validation accuracy (blue line) starts at 0.7, drops to 0.6, then rises to fluctuate between 0.85 and 0.95 for the remainder of the 109 epochs.</p>

Table 2: Batch Normalization Analysis

## 3.2 Dropout Regularization

Dropout was experimented on the base case model to find it's overall effect. Dropout removes sporadic output values within layers. This has the effect of increased learning due to the sparse representation.

Dropout had a large impact on the learning accuracy of the model by increasing the accuracy from between 3-4%. Results can be seen in Table 3.

Value Studied	Training Notes	Visualization
Dropout = .25	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 60 epochs</li> <li>• Reached 91% Accuracy</li> </ul>	<p>Training and validation accuracy</p>
Dropout = .5	<ul style="list-style-type: none"> <li>• Much slower than normal training time</li> <li>• Trained in 71 epochs</li> <li>• Reached 94% Accuracy</li> </ul>	<p>Training and validation accuracy</p>
Conv Dropout = .5 and Dense Dropout = .5	<ul style="list-style-type: none"> <li>• Much slower than normal training time</li> <li>• Trained in 127 epochs</li> <li>• Reached 94% Accuracy</li> </ul>	<p>Training and validation accuracy</p>

Table 3: Dropout Regularization Analysis



### 3.3 L1 / L2 Regularization

L1 and more specifically, L2 Regularization seemed to have the biggest impact on the model accuracy. Several different models were studied and L2 set to .001 achieved 94.8% accuracy. For a single Regularization factor, this seemed to have the biggest impact. However, L2 Regularization seemed to be the slowest training and took the most epochs.

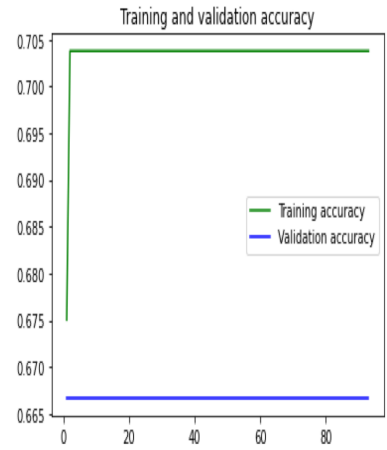
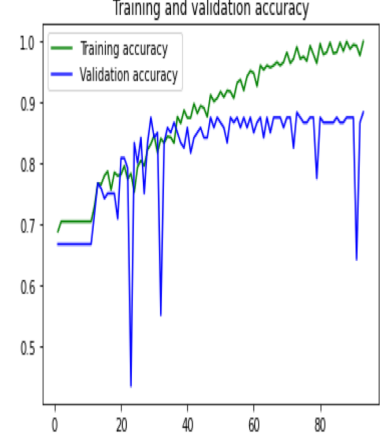
Value Studied	Training Notes	Visualization
L1 Regularization = .001	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 77 epochs</li> <li>• Reached 67% Accuracy</li> </ul>	 <p>The graph for L1 Regularization = .001 shows training accuracy (green line) starting at approximately 0.675 and rising sharply to about 0.74 by epoch 10, then remaining stable. Validation accuracy (blue line) starts at approximately 0.665 and remains flat throughout the 80 epochs shown.</p>
L1 Regularization = .002	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 93 epochs</li> <li>• Reached 88% Accuracy</li> </ul>	 <p>The graph for L1 Regularization = .002 shows both training (green) and validation (blue) accuracy starting around 0.7. Training accuracy fluctuates and generally increases, reaching nearly 1.0 by epoch 80. Validation accuracy also fluctuates and increases, reaching approximately 0.85 by epoch 80.</p>

Table 4: L1 Regularization Analysis

Value Studied	Training Notes	Visualization
L2 Regularization = .001	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 180 epochs</li> <li>• Reached 94% Accuracy</li> </ul>	<p>Training and validation accuracy</p>
L2 Regularization = .002	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 65 epochs</li> <li>• Reached 93% Accuracy</li> </ul>	<p>Training and validation accuracy</p>
L2 Regularization = .003	<ul style="list-style-type: none"> <li>• Slower than normal training time</li> <li>• Trained in 63 epochs</li> <li>• Reached 92% Accuracy</li> </ul>	<p>Training and validation accuracy</p>

Table 5: L2 Regularization Analysis

## 4 New Model with Regularization

As a final step in exploration of Regularization, the best model from Phase 4 was changed to include Batch Normalization, two Dropout layers with 0.5, and an L2 Regularization of 0.002.

The model yielded the highest accuracy of the entire semester. Accuracy was 95% trained over 200 epochs. Final validation loss was .28. The results are shown in figure .

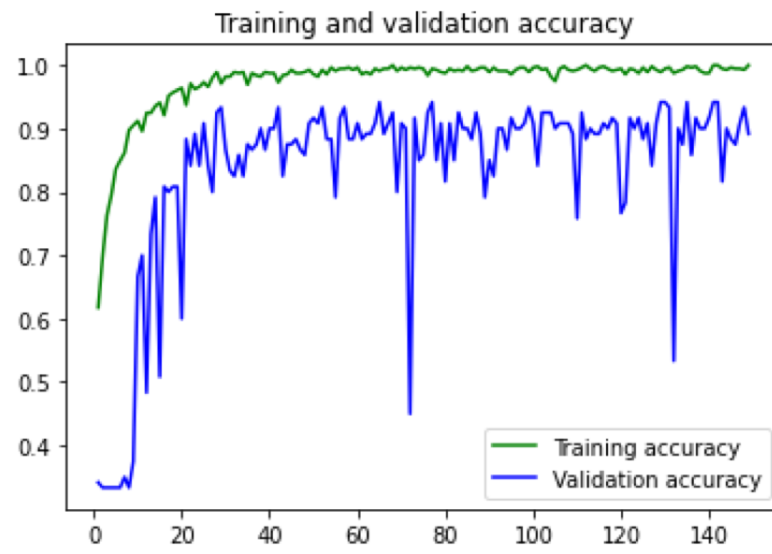


Figure 2: New Model Accuracy

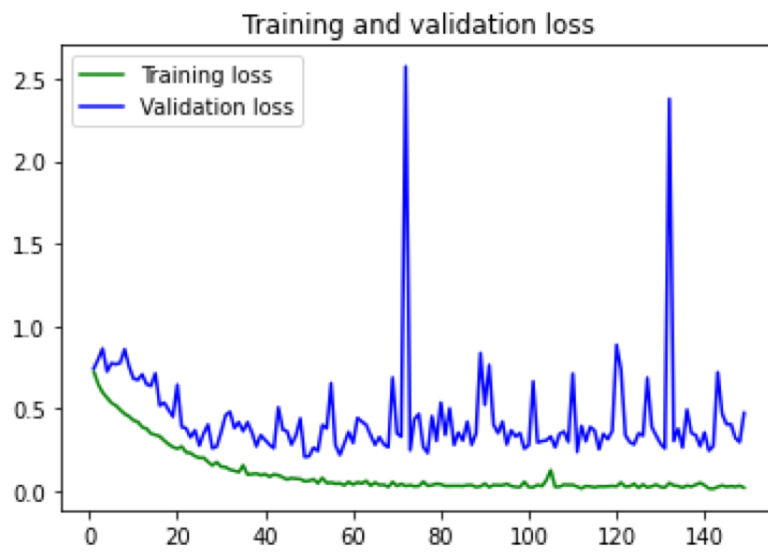


Figure 3: New Model Loss

## 5 Conclusion

Data Augmentation definitely had a large impact on improving the accuracy of the Honeysuckle model. It will definitely be used in the later phases of the project.

## References

- [oC20] Missouri Department of Conservation. Bush honeysuckle control. <https://mdc.mo.gov/trees-plants/invasive-plants/bush-honeysuckle-control>, 2020.
- [Web20] Integrated Pest Management Website. Weed of the month: Bush honeysuckle—an ornamental gone wrong. <https://ipm.missouri.edu/ipcm/2015/9/Weed-of-the-Month-Bush-honeysuckle-an-ornamental-gone-wrong/>, 2020.
- [Wik22] Wikipedia. *Lonicera japonica*. [https://en.wikipedia.org/wiki/Lonicera\\_japonica](https://en.wikipedia.org/wiki/Lonicera_japonica), 2022. Invasive Honeysuckle Species Description.