# Using Deep Learning to Determine Honeysuckle Bark

Brett Huffman - CSCI 5390 - Main Project Phase VI

**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 6 Objective

In Phase 6, several pretrained models are evaluated against the baseline model. Pretrained model's classifiers are systematically placed on top of the dense layers of the existing model.

The pretrained models studied are:

- VGG16;
- DenseNet121;
- and TensorHub Plant Model V1

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.

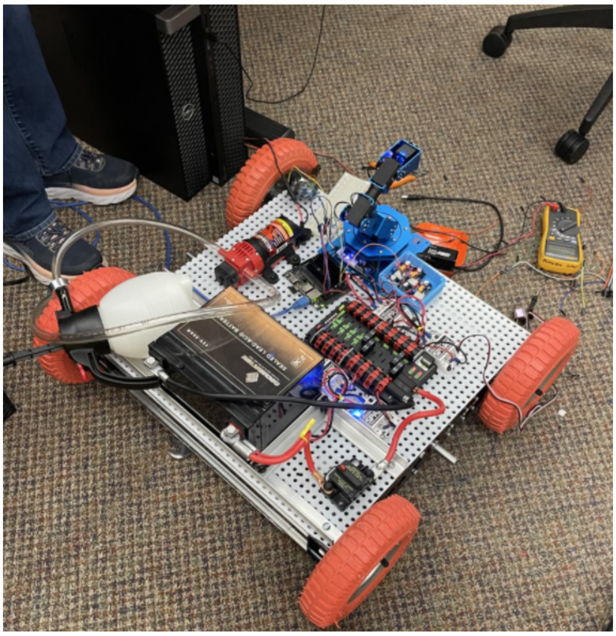


Figure 1: A view of the early rover prototype

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 ([Web20a]).

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 6 Procedure and Results

Taking the best model from Phase 4, a baseline was established of both Accuracy and F1 scores. This baseline shown in Table 1 will be used to evaluate all models in this study.

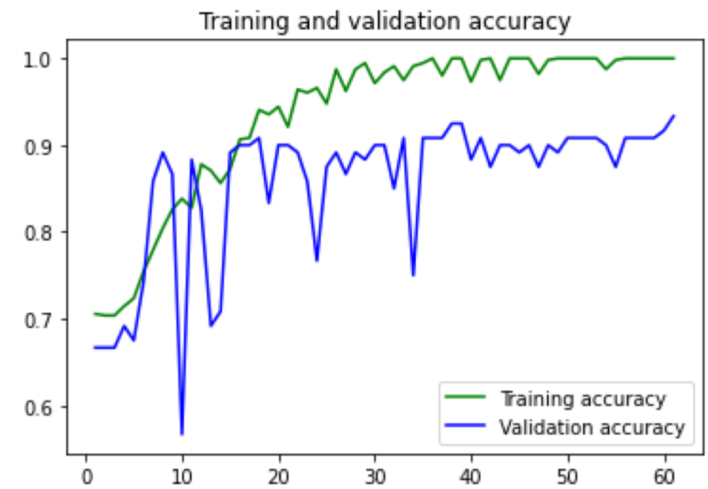
| Training Notes | Visualization |
| --- | --- |
| • Baseline training time<br>• Trained in 98 epochs<br>• Reached 94% Accuracy |  |

Table 1: Baseline Analysis

After establishing this baseline, each pre-trained model is evaluated in an attempt to find the best pre-trained model.

Details of each attempt will be found in the following subsections:

## 3.1 VGG16

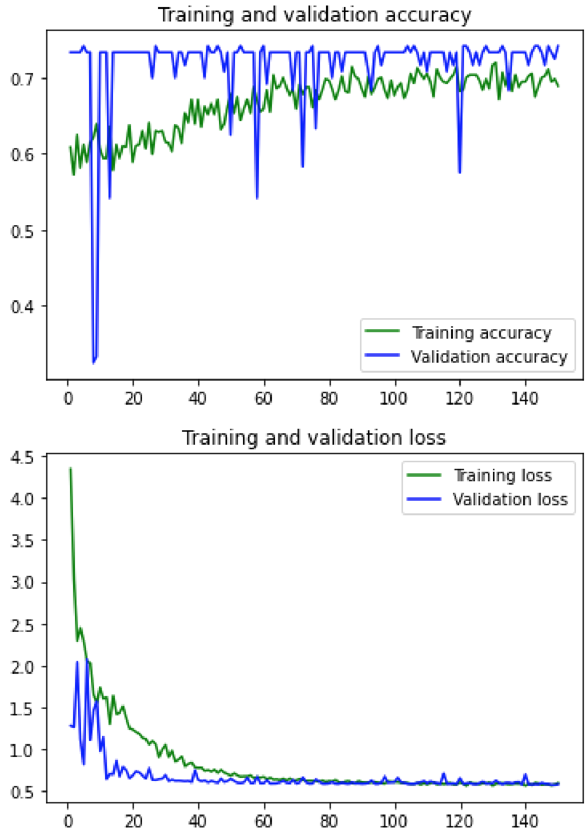
The VGG16 Classifier was utilized first. Since the Honeysuckle project uses Data Augmentation, the technique described in Feature Extraction Together with Data Augmentation was built [Cho21].

The results were less-than-spectacular. The promise of utilizing a professional, pre-trained model seemed to be the answer to raise the model's accuracy. However, the model seemed to be unable to adapt to the new classifier's output. Below are the best results obtained from the VGG16 model.

<table>
<tr><th>VGG16 Results</th><th>Visualization</th></tr>
<tr><td>

- Accuracy: 0.718
- Precision: 0.359
- Recall: 0.5
- F1: 0.418

</td><td>




</td></tr>
</table>

Table 2: VGG16 Results

The model utilized with VGG16 was is listed below:

```
_________________________________________________________________
Layer (type)                  Output Shape               Param #
=================================================================
input_2 (InputLayer)          [(None, 224, 224, 3)]      0
sequential (Sequential)       (None, 224, 224, 3)        0
tf.__operators__.getitem (S    (None, 224, 224, 3)       0
licingOpLambda)
tf.nn.bias_add (TFOpLambda)    (None, 224, 224, 3)       0
vgg16 (Functional)            (None, None, None, 512)    14714688
flatten (Flatten)             (None, 25088)              0
dense (Dense)                 (None, 32)                 802816
dense_1 (Dense)               (None, 16)                 512
dropout (Dropout)             (None, 16)                 0
dense_2 (Dense)               (None, 2)                  34
=================================================================
Total params: 15,518,050
Trainable params: 803,362
Non-trainable params: 14,714,688
_________________________________________________________________
```

## 3.2 DenseNet121

Densenet121 turned out to be immensely helpful in improving model accuracy. The model suddenly popped up to 96% accuracy. The highest accuracy obtained since the debut of the project.

Much like the VGG16 model, the DenseNet121 was added to the project via the conv_base method to support Data Augmentation. It obtained amazing results in just 46 epochs.

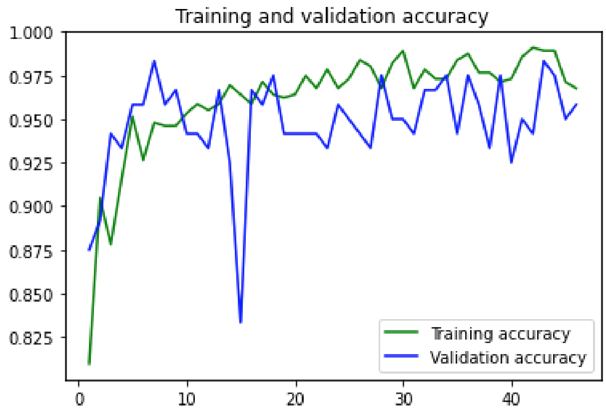
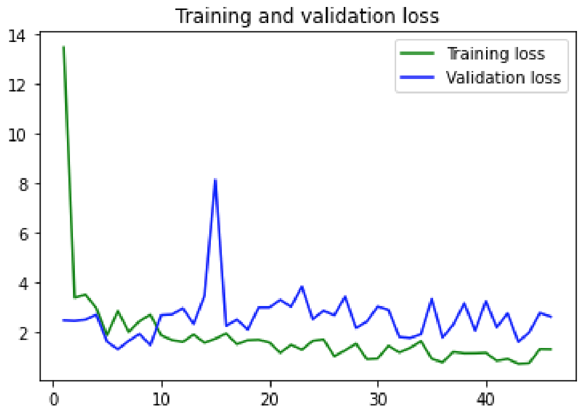
| VGG16 Results | Visualization |
| --- | --- |
| • Accuracy: 0.968<br>• Precision: 1.0<br>• Recall: 0.9375<br>• F1: 0.9667 | <br><br><br> |

Table 3: DenseNet121 Results

The best Densenet model's shape was as follows:

```
Layer (type)                     Output Shape                   Param #
=======================================================================
input_2 (InputLayer)             [(None, 224, 224, 3)]          0
sequential (Sequential)          (None, 224, 224, 3)            0
densenet121 (Functional)         (None, None, None, 1024)       7037504
flatten (Flatten)                (None, 50176)                  0
dense (Dense)                    (None, 64)                     3211328
dense_1 (Dense)                  (None, 32)                     2080
dropout (Dropout)                (None, 32)                     0
dense_2 (Dense)                  (None, 2)                      66
=======================================================================
Total params: 10,250,978
Trainable params: 3,213,474
Non-trainable params: 7,037,504
_______________________________________________________________________
```

### 3.3 TensorHub CropNet Model V1

In addition to the famous models in favor now, a lesser-known, but plant-specific model was tested. It was the CropNet Model V1 from TensorHub. CropNet was built on top of the iNaturalist and plants subset of ImageNet-21K ([Web20b]). The model has been built with 2.15M plant images and 8864 classes.

CropNet takes the exact same size images as VGG16 and DenseNet121, so it made adapting it easy.

The results were not as good as expected. They were better than VGG16, but only achieved a mid-80/

The best model of the CropNet classifier was as follows:

```
 Layer (type)                  Output Shape              Param #
=================================================================
 input_3 (InputLayer)          [(None, 224, 224, 3)]     0
 equential_2 (Sequential)     (None, 224, 224, 3)       0
 keras_layer (KerasLayer)      (None, 1280)              15581216
 flatten_2 (Flatten)           (None, 1280)              0
 dense_7 (Dense)               (None, 32)                40992
 dense_8 (Dense)               (None, 8)                 264
 dropout_2 (Dropout)           (None, 8)                 0
 dense_9 (Dense)               (None, 2)                 18
=================================================================
Total params: 15,622,490
Trainable params: 41,274
Non-trainable params: 15,581,216
_________________________________________________________________
```

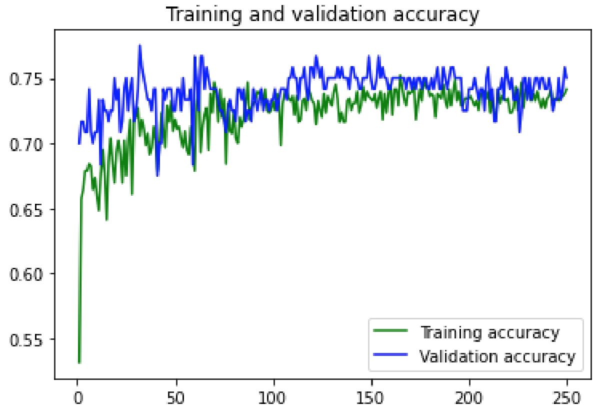
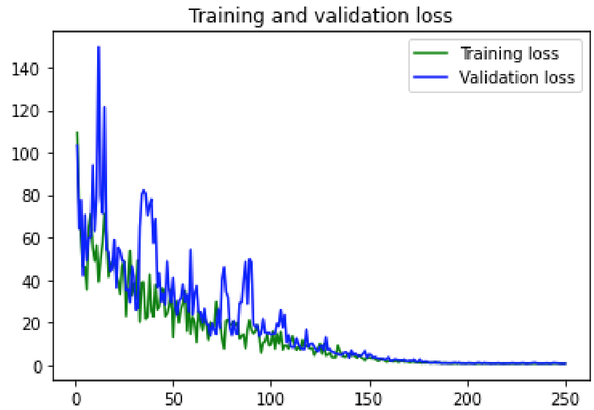
| VGG16 Results | Visualization |
|---|---|
| • Accuracy: 0.788<br>• Precision: .58<br>• Recall: 0.521<br>• F1: 0.569 | <br> |

Table 4: CropNet Results

# 4 Conclusion

In the end, the DenseNet121 proved a very formidable addition to the network. It added a full 2% accuracy above the best-trained, most-tweaked version of the Honeysuckle model. It showed the use of pre-trained models can be an easy way to improve model accuracy as long as studies are done to pick the very best classifier for the problem.

# References


[Cho21] Francois Chollet. *Deep learning with Python (2nd ed.)*. Manning Publications, 2021.

[oC20] Missouri Department of Conservation. Bush honeysuckle control. `https://mdc.mo.gov/trees-plants/invasive-plants/bush-honeysuckle-control`, 2020.

[Web20a] 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.

[Web20b] Tensorflow Hub Website. cropnet/feature$_vector/concat.$, 2020.

Wikipedia. Lonicera japonica. `https://en.wikipedia.org/wiki/Lonicera_japonica`, 2022. Invasive Honeysuckle Species Description.