Morels

Elaina Rohlfing
University of Missouri-St. Louis

05/10/2024

Contents

Abstract

The aim of this project is to build and train a Convolutional Neural Network (CNN) that can identify the presence or absence of Morel(s) in a typical forest landscape. This project also explores the benefits of leveraging transfer learning and various off-the-shelf-tools for this task. Morels were of interest due to their popularity amongst the foraging community. Morels most likely earned their high status because of their wide distribution, ease of identification, and highly prized flavor. Combined, these features make them an accessible and enticing option for both beginner and expert foragers. Their popularity makes them an engaging subject to study and, most interestingly, the distinct honeycomb patterning that makes them easy to identify also makes them particularly challenging for novice foragers to spot.

With these things in mind, this project investigates whether a CNN faces similar challenges as a novice forager or if this problem is particularly well-suited to deep learning. If a high performing enough model can be developed, it would be useful to deploy as a mobile app that can assist foragers who have trouble detecting these delicacies hiding in plain sight.

Chapter 1

Dataset

A new dataset was created for this project. The goal of the project was to mimic the type of input a model would receive if it were scanning the forest floor in search of partially hidden mushrooms. To achieve this goal, I went directly to foraging communities to gather exactly that type of image that a forager would generate, themselves.

1.1 Data Collection

1.1.1 Sources

The majority of images were collected from four different Facebook communities with publicly available media. A small percent of the images (roughly 10%) were collected from google image search. To save time, images were downloaded in batches of 300 and then manually filtered to ensure high resolution quality and to remove any irrelevant data points. List of Image Sources:

- Missouri Morel Mushroom Hunting[1]
- Mid Missouri Morel Hunting (and wild mushroom identification) [2]
- Morel Mushroom Sightings[3]
- Google image search

After filtering, the image collection process yielded 398 high quality images. It is worth noting, although most of the data collection was from regional sources, the addition of images from the Morel Mushroom Sightings community boosted the image collection spread just enough to ensure the presences of at least one representative for each species of True Morels. Ideally, this dataset would be larger and more varied, but this is our starting point.

1.1.2 Image Processing

To create two classes ("morel" and "none") each image was sampled into multiple, smaller images that could represent each class. See figure 1.1 for an example of the process.

Figure 1.1: Creating samples from a single image.

1.2 Characteristics of Final Dataset

The final dataset consisted of 792 samples split evenly between the "morel" and "none" classes, resulting in a baseline accuracy of 50% Figure 1.2 for a few examples of images in the final dataset.

Figure 1.2: A few samples from the completed dataset.

1.3 Challenges

The dataset used in this project was the second iteration. The first dataset consisted of only 467 images split unevenly, for a baseline accuracy of 62%. Images were sampled with a uniform pixel-size of 300 px by 300 px. With the expectation that this method would ensure uniform resolution quality. Unfortunately this was not the case and many images were eliminated from the initial collection. Additionally, after sampling the images were first shuffled and then separated into separate train, validate, and test sets. After some testing and experimentation it was realized that allowing sampled images to be sorted into all three development sets was most likely one of the reasons for the model's high variance. The high similarity in samples cut from the same image is comparable to having duplicate entries that leak data into the validation and training sets. These issues were resolved with two key steps. First, by enforcing a higher standard for image resolution during filtering. Second, by shuffling the raw images (prior to sampling). Performing shuffling earlier, allowed the images to be distributed evenly for the

splitting process, while also ensuring the samples from each image remained grouped together. This made it simple to ensure image samples were not mixed inappropriately during splitting.

Chapter 2

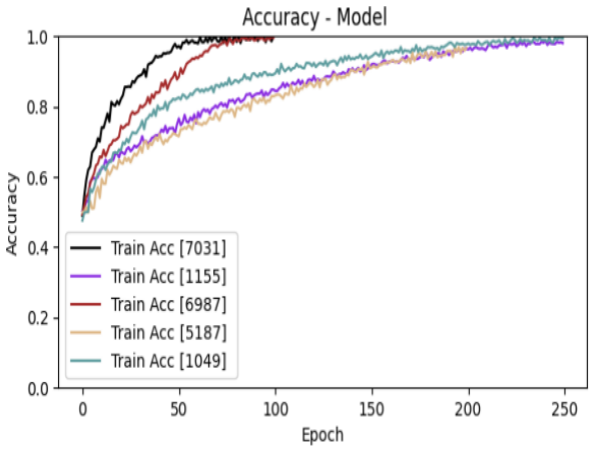
Phase I

2.1 Building the smallest possible Overfitting model

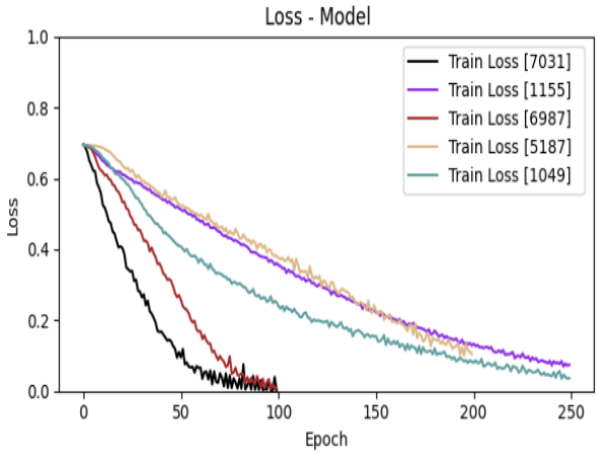
2.1.1 Results

Table 2.1: Comparison of Model Performance By Parameters

Training		Performance Metrics				
Number of Parameters	Number of Epochs	Loss	Accuracy	Recall	Precision	F1
7031	100	0.0019	1.0000	1.0000	1.0000	1.0000
6987	130	0.0000	1.0000	1.0000	1.0000	1.0000
5187	200	0.0836	0.9816	0.9918	0.9707	0.9812
1155	250	0.0661	0.9816	0.9644	1.0000	0.9819
1049	250	0.0367	0.9987	0.9974	1.0000	0.9987



(a) Comparison of Accuracy for all Models

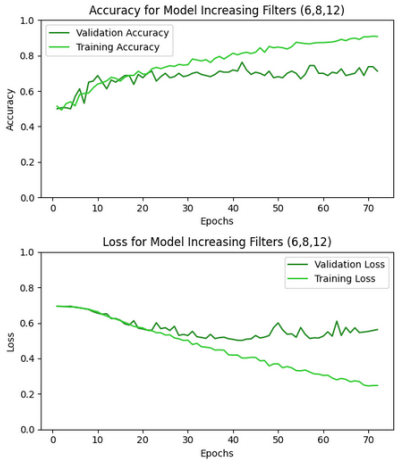


(b) Comparison of Loss for all Models

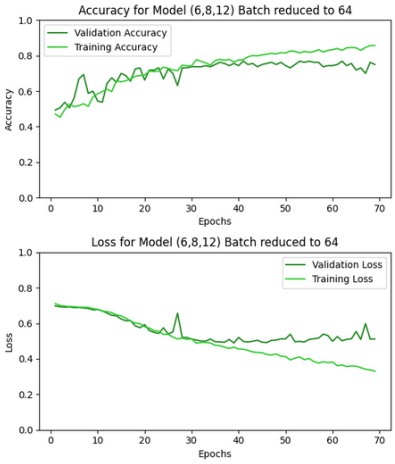
Figure 2.1: Learning Curves For Overfitting Model

2.2 Building a Model For Best Fit

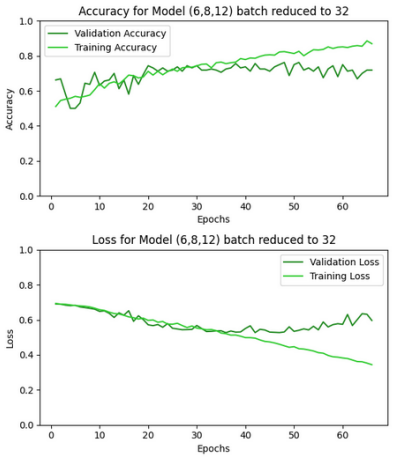
2.2.1 Results



(a) Performance with largest batch size

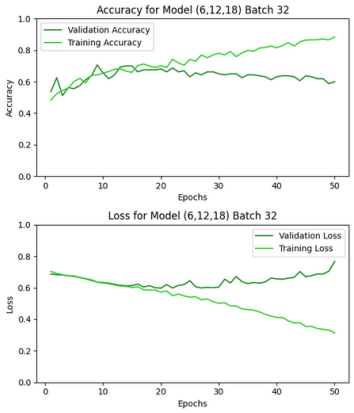


(b) Performance with batch size 64

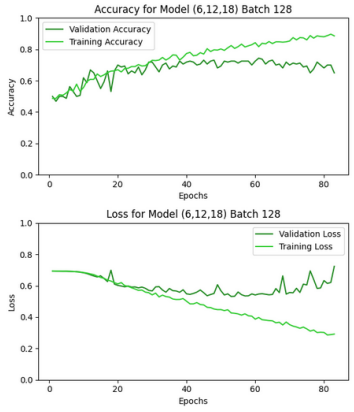


(c) Performance with smallest batch size

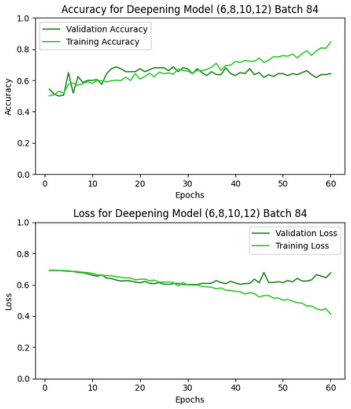
Figure 2.2: Comparison of Learning Curves as Batch Size is Decreased



(a) Performance with additional filters (Batch 32)



(b) Performance with additional filters (Batch 128)



(c) Performance with additional Layers (Batch 128)

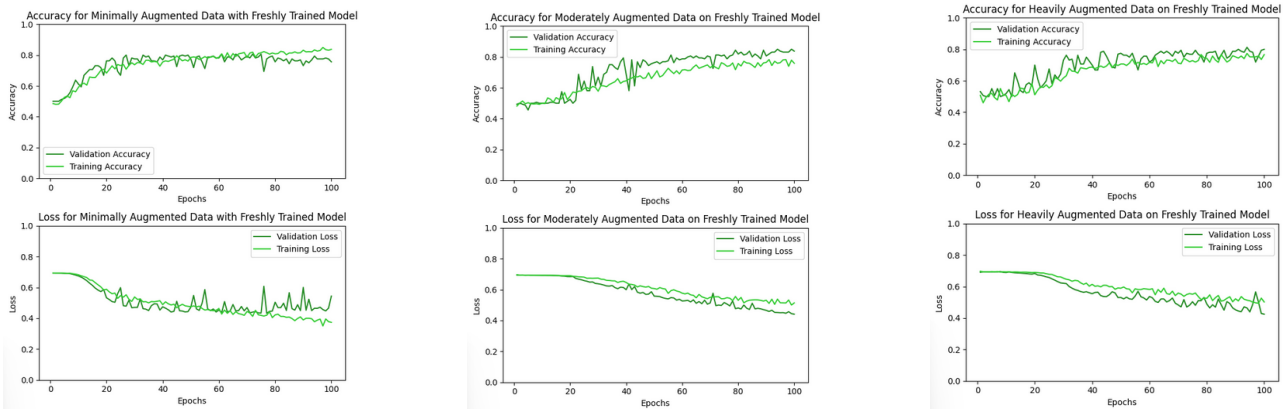
Figure 2.3: Comparison of Learning curves on Models with Larger Architecture

Chapter 3

Phase II

3.1 Applying Augmentation

3.1.1 Results



(a) Minimal augmentation (b) Moderate augmentation (c) Heavy augmentation

Figure 3.1: How Augmenting the Dataset Impacts Learning

3.2 Exploring off-the-shelf Tools and Transfer Learning

For this project, I explored two additional models.

- MobileNetV2 (available through tesorflow.keras.applications)
- Teachable Machines (An off-the-shelf tool by Google)

3.2.1 MobileNetV2

MobileNetV2 is a pretrained model that can be used with any image dataset. I decided to explore this model specifically because a model trained to identify morels could have useful applications as a mobile app.

Method

Finetuning MobileNetV4 with the small morel dataset required the following steps. Initially, the weights of the model were frozen to prevent disrupting everything the model already knows. Adding an augmentation layer on the bottom of the model was crucial as well. The dataset is quite small and so it must be artificially increased to give the model a better chance of learning and not memorizing. Augmentation values were chosen based on the best performing transformations found in the earlier augmentations step. Additionally, a very small learning rate of 0.0005 was used to further reduce overfitting risks.
With these settings in place, the model trained to convergence within 30 epochs. Final validation accuracy was 0.94 and validation loss was 0.17. (See figure 3.2a)
At this point it is possible to unfreeze the layers and perform finetuning. For this experiment, I chose to unfreeze the top 90 layers of the model. The bottom-most layers were left frozen, to continue benefiting from their highly generalized knowledge. Before training, the learning rate was reduced even further to 0.00001 to mitigate overfitting.
After training for only 20 more epochs, the model reached a validation accuracy of 0.96 and a validation loss of 0.09. (See figure 3.2b) Pretty good!

3.2.2 Teachable Machine

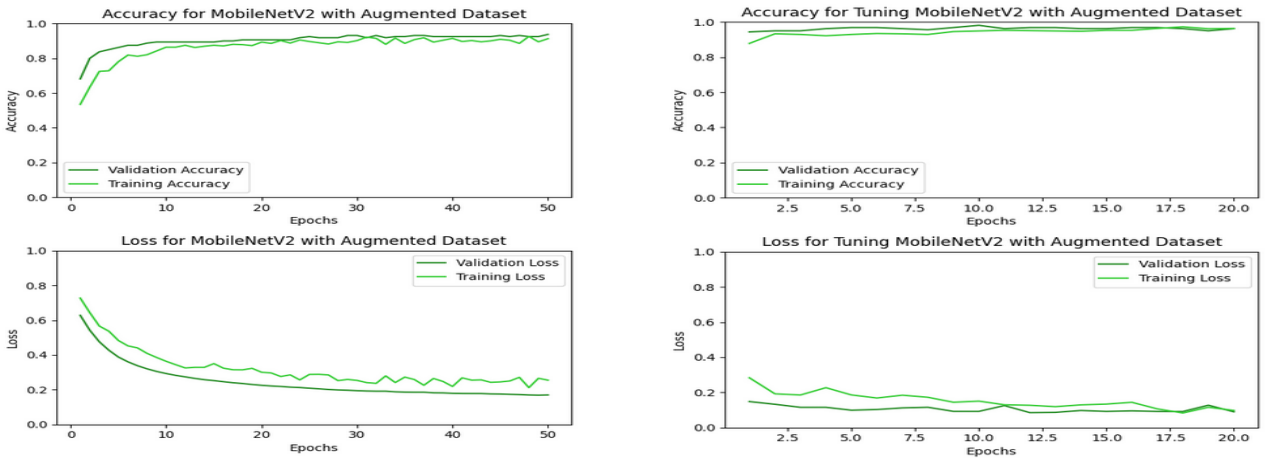
Teachable Machine is a powerful pretrained tool that can (often) produce a very well trained model with only a limited amount of data and within a very short time period. Unlike the pretrained models offered in keras, you cannot fine-tune Teachable Machine.

Method

The biggest benefit of using Teachable Machine is its simplicity. For this experiment, I simply loaded my training and validation images into the model and allowed it to train for 20 brief epochs.
Those results were fairly good, producing an accuracy in the low 90s. Conveniently, Teachable Machines allows the user to adjust a few parameters, batch, learning rate, and epochs. With a little fine tuning I was able to produce a much higher performing model, see figure3.3 in 120 (very fast) epochs.

The best performing parameters were a batch size of 64 and learning rate of 0.00002. These values produced the smoothest learning curves while also delaying overfitting for the longest possible time (40 epochs).

3.2.3 Results



(a) Training MobileNetV2 to Convergence

(b) Progression of Finetuning

Figure 3.2: Transfer learning

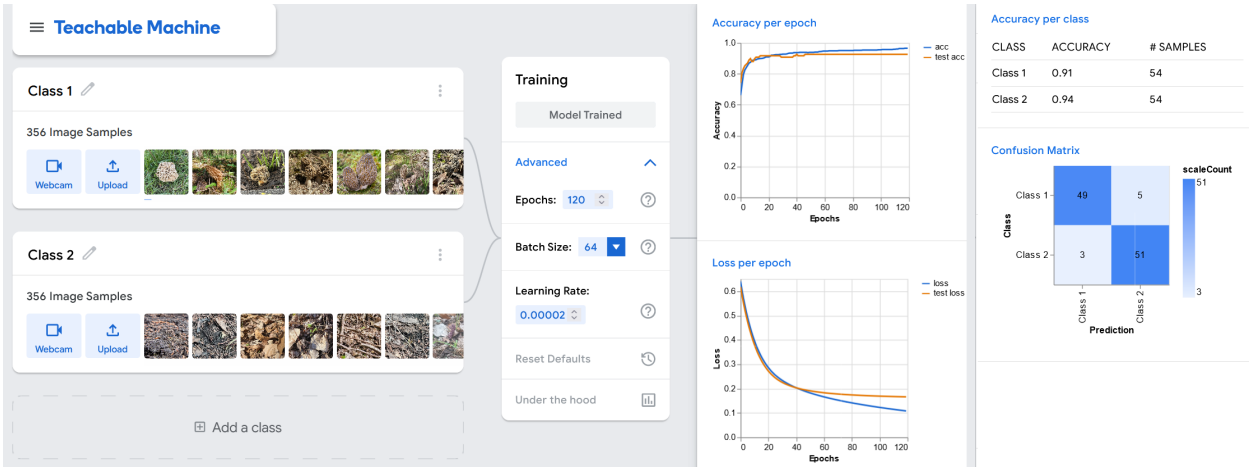


Figure 3.3: Teachable Machine UI and Training Results

I was able to get great results with both models. MobileNetV2 was trained to convergence in less than 30 epochs and Teachable Machine trained very quickly. Despite this, evaluating these models revealed that the models were not truely able to generalize to a real world situation.

Chapter 4

Conclusions

I believe that creating a more comprehensive dataset would greatly improve the model's learning capeabilities. The models explored in this project did quite well at first glance, but after spending sometime evaluating prediction capabilities with a more realistic dataset, such as images from the iNaturalist website [4] it is clear that the current dataset falls short of its goal. Considering the narrow range of data collected for this problem, it is not possible to produce a model that would be effective in any real-world situation.
Including a broader range of classes to represent a larger portion of the environments that morels inhabit would allow a model to be trained to a much more generalized understanding of what makes a morel and what does not.

Chapter 5

Challenges and Future Research

The biggest challenge I faced in this project was image sampling. Ultimately, two completely separate datasets were generated for this project. While the initial dataset performed acceptably in the overfitting stage, it was incredibly difficult to train once it was shuffled and split. Although creating a second dataset put me behind schedule, it showed immediate improvements on all tests. Where the first dataset struggled to get more than 5% above baseline, the new dataset was easily tuned to reach as high as 30% or more above baseline.
Ultimately there were still some major shortcomings with the datasets created, but for a very narrow question "Is there a morel or is there nothing?", the models made their predictions well. In the future, I would like to work with the iNaturalist dataset and build towards a much more robust model that may have even broader applications for young and hobbyist naturalists. I look forward to exploring this project further.

Bibliography

[1] Missouri mushroom hunters. https://www.facebook.com/groups/728992747972388. Accessed: April 30, 2024.

[2] Mid missouri morel hunting (and wild mushroom identification). https://www.facebook.com/groups/1133694627094651. Accessed: April 30, 2024.

[3] Morel mushroom sightings. https://www.facebook.com/groups/2051074861594288. Accessed: April 30, 2024.

[4] Inaturalist. https://www.inaturalist.org/. Accessed: May 8, 2024.