Detecting Mould on Mushrooms using CNN

1st Puneeth Nagarajaiah
Student, Machine Learning
York University
Toronto, Canada
puneeth@my.yorku.ca

2nd Imdad
Lead Data Scientist
Afiniti
Toronto, Canada
imdad0266@gmail.com

3rd Kwangjin Park
Firmware Engineer
AMD
Toronto, Canada
kwanpark@amd.com

***Abstract*—The goal of our project is to use a Pre-trained Convolutional Neural Network (CNN) model to detect mould on mushrooms grown at Setas Mushrooms. Organic mushrooms, which are grown without the use of synthetic agents or pesticides, are prone to exposure to mould spores, which usually affect about 25% of all mushrooms grown at Setas. A CNN model will be able to help Setas in the timely detection of mould, thus eliminating the need for manual inspection and also containing its spread. We used two models: VGG19 and MobileNet V2 and both achieved 90+% Validation accuracy. In our final step, we developed a mobile app that can capture images and give out predictions on the spot.**

***Index Terms*—CNN, Mould, Mushrooms, VGG19, MobileNet V2**

I. Background

Max Justice is the CEO of Setas Mushroom Farm. He inquired us about a machine learning solution to detect mould on mushrooms grown at his farm. Mould is a type of fungus that can grow on and consume mushrooms, which can lead to reduced yields or even complete crop failure. This can lead to economic losses for mushroom farmers and impact the availability and cost of mushrooms in the market.

Fig. 1. Mushrooms grown inside Setas Mushroom Farm

Setas grows organic mushrooms and organic mushroom farming can be more susceptible to moulds than conventionally grown mushrooms. This is because organic farming methods typically rely on natural inputs such as compost, manure, and other organic materials that can contain spores and bacteria, increasing the likelihood of mould growth.

The type of mould we are detecting is called Trichoderma. Trichoderma is a genus of soil-dwelling fungi found worldwide. They are fast-growing, highly adaptable fungi that form symbiotic relationships with plant roots, making them ideal for fungicidal use in agricultural settings.

II. Business Problem

Detecting mould on mushrooms can be challenging for several reasons:

- Visual similarity: Some moulds can look very similar to the natural discoloration or bruising of mushrooms, making it difficult to detect mould growth through visual inspection alone.
- Diversity of moulds: There are many different types of moulds, each with different characteristics and growth patterns. Identifying the specific type of mould can be challenging, and may require laboratory analysis.
- Location of mould: Mould growth can occur on the surface of the mushrooms or within the gills, stem or cap, making it difficult to detect without close inspection.
- Sporadic growth: Mould growth can be sporadic, meaning that it may not be present in all areas of the mushroom crop, and may occur at different times during the growing cycle.
- Cost and time: Laboratory analysis to identify mould species can be costly and time-consuming, which may not be feasible for small-scale mushroom farmers.

To overcome these challenges, it is important to regularly monitor mushrooms for any signs of mould growth and to remove any affected mushrooms promptly to prevent the spread of mould. Conventionally, detecting mould on mushrooms is done through visual inspection. This is manual, time-consuming and prone to error.

III. Dataset

The dataset/images provided by Setas has two classes, mushrooms with Trichoderma mould and without Trichoderma mould. At present, we have 90 images of uncontaminated blocks and 90 contaminated blocks. These images are taken using smartphone and displays high resolution, which is critical in identify small mould area. Images are in colour to better identify discolouration on mushroom.

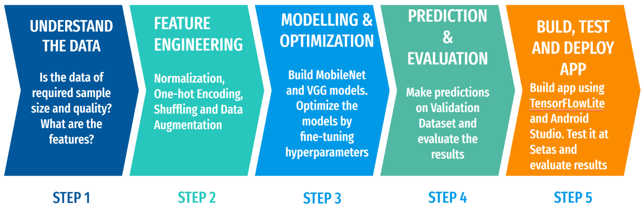

Fig. 2. Our steps to solving the problem

Fig. 3. With Mould

IV. Solution

With the images currently available, we initially propose to build a VGG19 model to classify the mushrooms. If the results are encouraging and we obtain more high quality images, we propose to build an Android app that contains a MobileNetV2 model to automate the detection of mouldy mushrooms.

Fig. 4. Without Mould

Fig. 5. Image Prediction with VGG19

Fig. 6. Image Prediction with MobileNet V2

V. Methodology

A. Model

Our model uses the convolutional neural network (CNN) methodology. CNN is a deep learning algorithm specializing in image analyses. The specific feature of this algorithm lies in the first of the two standard layers of Neural Network architecture (”feature learning” and ”classification”). To minimize the requirements of pre-processing and computational power, this approach conducts a series of ”convolution” operations for feature extraction. After accepting the input, the feature learning step is organized into groups of convolutional layers integrated via a pooling layer.

Each convolutional layer extracts features via sliding a kernel/filter of a specific size with a stride equal to 1 (one-step moves) computing dot product across the specified window. All neurons in a given convolutional layer share the same parametrization (weights and bias). This means that they respond to the same features. Each layer is therefore focused on detecting a different set of features (starting with lower-level features).

Our model replicates the VGG19 pre-trained example of CNN architecture developed by Simonyan and Zisserman (Simonyan and Zisserman, 2014). VGG19 is a 19 layers CNN operating with five groups of convolutional layers (ReLu activation function, grouping via max-pooling) complemented by three fully connected layers for higher-complexity feature extraction (see figure 2).

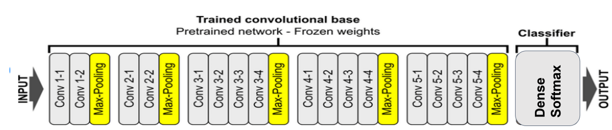

Fig. 7. VGG19 Architecture

The filter size is 3x3. It is a multi-group classification model allowing up to 1000 potential groups (classifying using Soft-Max classification techniques). The model is pre-trained on more than 1 million pictures from the ImageNet database.

B. Model Training

Let us quickly explain the major difference between traditional machine learning techniques and transfer learning.

In the past, to train a model, you would require a large amount of training data set. However, the knowledge acquired

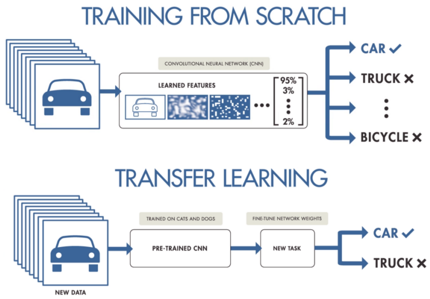

Fig. 8. Traditional Approach VS Transfer Learning

by the first model is not retained. So to train a model on a new data set, you need another large data set for the second model. Thanks to transfer learning, the knowledge acquired from the previous training is retained, so if you want the model to learn new knowledge from the new data set, you can train it faster and with less training data (Baheti, 2023).

We only about a hundred images of two classes of mushrooms, so we will utilize transfer learning in the training of our model. The base model we will be using will be VGG19, but we removed the dense layer and replaced it with ours, so the model can classify mushrooms with or without mould in the end.

VI. Mobile App

A. Building an Android App

Fig. 9. A graphical depiction of steps to build an Android App

There are three steps to building an app [3]:

- **Step 1**: Train the classifier model using Teachable Machine
- **Step 2:** Download the TensorFlow Lite version of the Model
- **Step 3:** Sync the model with the Android App using Android Studio

In the first step, we train a model using the Teachable Learning Platform (TM). At present, TM only supports the MobileNetV2 model. In the second step, we will download a TensorFlow Lite version of this model in the normal floating point and the quantized versions. In the third step, we will sync this model into an Android Java-based app using Android Studio.

VII. Results

A. Training and Validation Accuracy

The first time we were training our model, we froze the base layers from the pre-trained model. This is because we want to update just the new layer we added at the end of our new classification. The freezing allows the model to retain the knowledge acquired previously from training through the ImageNet data set. After freezing the base layers, we trained the model over 20 epochs.

As you can see in the graphs, we achieved quite a high training and validation accuracy (Figure 10). However, we also noticed that the validation accuracy became stale around the 14th epoch but validation loss kept decreasing until the last epoch as shown in figure 11

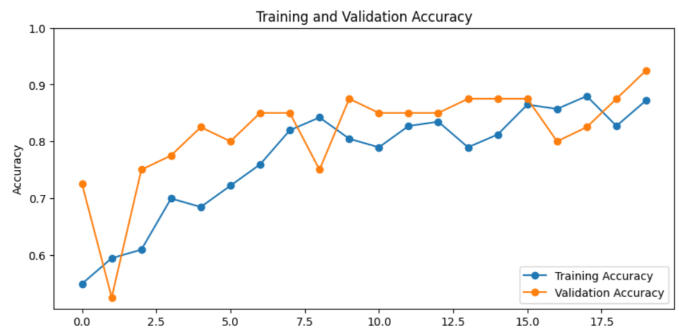

Fig. 10. Initial Training and validation accuracy

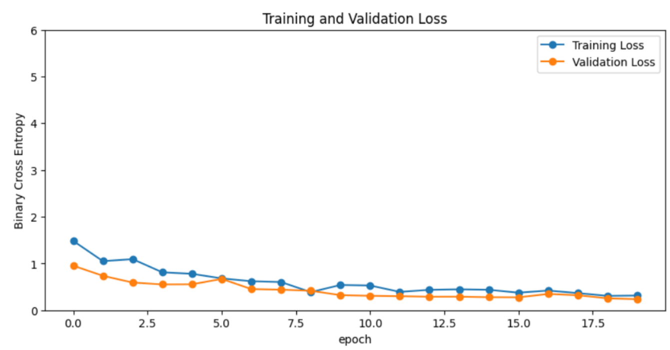

Fig. 11. Initial Training and validation loss

With regard to the app, the following results were obtained.

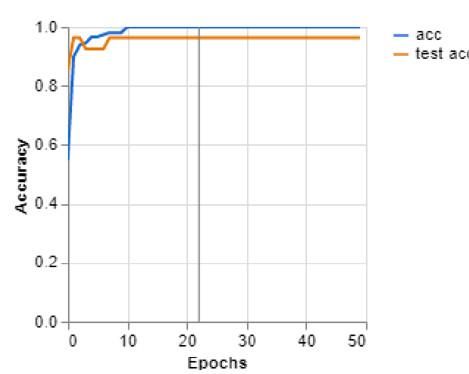

Fig. 12. Training and Validation Accuracy of MobileNetV2 model used in the Android App

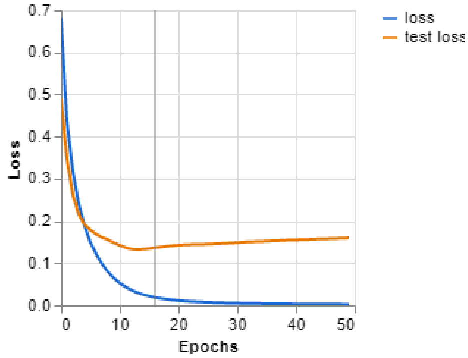

Fig. 13. Training and Validation Loss for MobileNetV2 model used in the Android App

From the above graphs, we can see that the model has attained a validation accuracy of approximately 95%.

Final Results for VGG19 model: Once our model is trained to predict whether the mushroom has a mould or not, we wanted to perform fine-tuning by unfreezing the base model. We will then train it again with a very small learning rate of 0.00001. The base pre-trained layers take very small steps and adjust slightly, which improves the model by a small amount. We retrained the model for 10 epochs. We also reduced the learning rate by 10% if two consecutive epochs did not result in validation loss reduction. As you can see in the graphs, our training and validation accuracy improved to 97.5% (see figure 14). The validation loss was also reduced further (see figure 15).

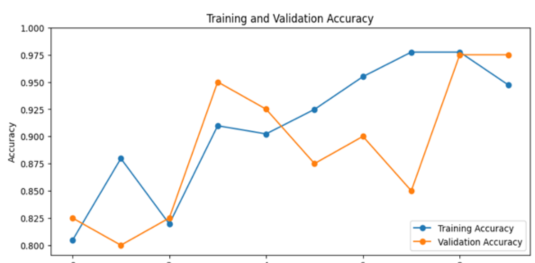

Fig. 14. Final Training and validation accuracy

Fig. 15. Final Training and validation loss

B. Predictions and Insights

After training the model, we used it to predict random mushroom images as is shown in Figure 16. Here you can see small light green spots of protruding mould which the model was able to detect perfectly

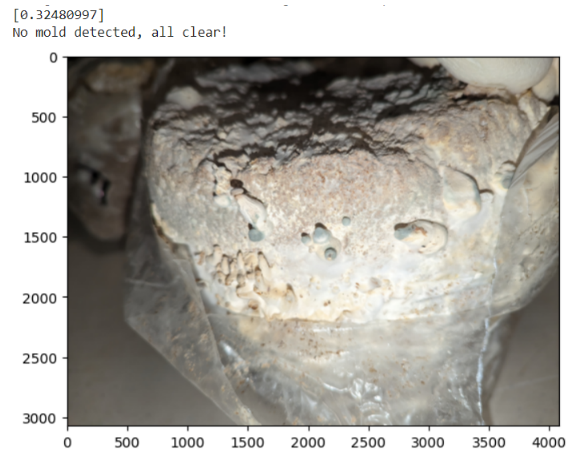

Fig. 16. Correct prediction of mold

We noticed some cases where the predictions were incorrect. That led us to look deeper into our training data and understand how the model identifies a mold. As an example (see figure 17), we gave a healthy image of a substrate to it and it classified it as mold.

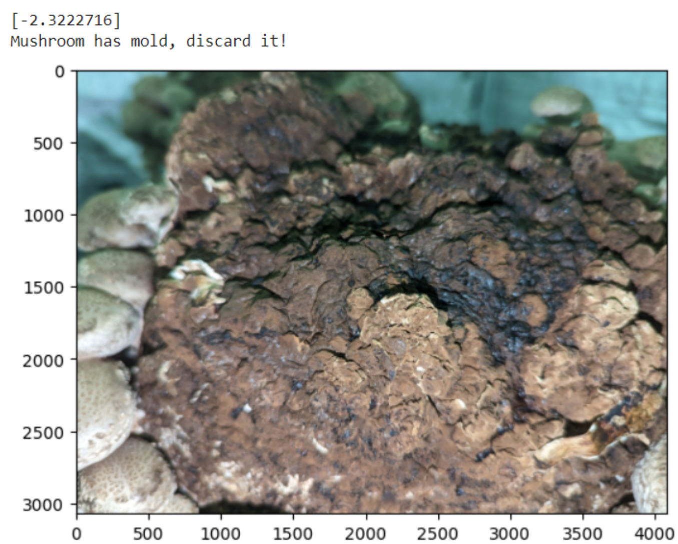

Fig. 17. Incorrect prediction of mold

Although not obvious at first, there are few insights into the internal workings of the model through these wrong predictions. It appears that model is identifying mold with the help of its color, if the image has predominantly green background then the model sometimes misclassifies a healthy image

Similarly, when we gave it an image of healthy substrate but it had green shadow towards it right side, the model misclassified it as containing mold (see figure 18.

To work around these mistakes, we either need to pre-process the image background before feeding it to the model or we need a model which is agnostic to the colour or type of background a mushroom image can have. This will further improve our predictive accuracy.

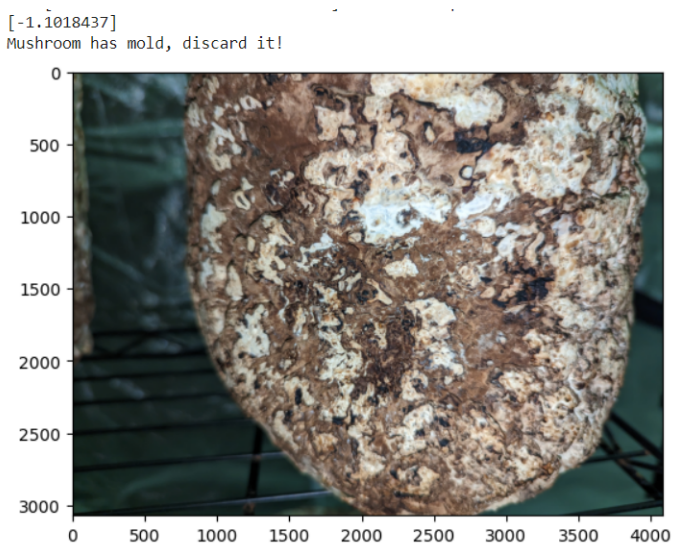

Fig. 18. Incorrect prediction of mould

With regard to the app, the following predictions were obtained.

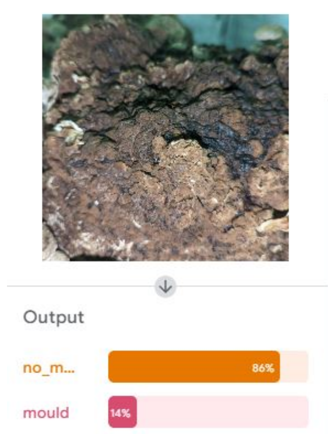

Fig. 19. Predictions by App-based MobileNetV2 model

We can see that the results are a percentage of mould present and not present in an image. The model suggests the presence of mould wherever there is a slight discolouration or green colour in an image.

VIII. Conclusion

Our model used mushroom data set to identify the mould on the mushroom. Our base model was VGG19 and it was trained to make binary classification. We also used MobileNet V2 for mobile app version of the model. On top of basic training, we used fine tuning to slightly improve the model's accuracy, and we were able to achieve 97.5 % training accuracy and 97% validation accuracy at the end.

Our model performed fairly well on predicting if the mushroom has the mould or not, but it did make some mistakes. These mistakes stem from the small size of new data and possibly, some similarity in colours between different images.

So where can we go from here? We could train our model to identify different stages of mould based on the mould size. We could also try getting even more data for mushroom images, which could greatly reduce the mistakes in classification during prediction.

From CSML1030 Group Project we learned the following:

- How to use CNN to predict images, use our own data set on the base model to create a highly predictive model
- Explore and get practical experience in basic CNN pipeline structure
- Improve training and validation accuracy through fine-tuning our model and experimenting with different parameters
- Deploy a mobile app that can make prediction on real-time image data

We really enjoyed the course and appreciate all the support and teaching from Saber. Thank you.

References

[1] Simonyan, Karen, and Zisserman, Andrew. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[2] Baheti, Pragatic. "A Newbie-Friendly Guide to Transfer Learning." Retrieved from *https://www.v7labs.com/blog/transfer-learning-guide* (2023).

[3] Google for Developers *https://developers.google.com/learn/pathways/get-started-image-classification*