AGRI-LEARNING: DIAGNOSING PLANT DISEASES USING CONVOLUTIONAL NEURAL NETWORKS

Kamakshi Bali
Presidium High School
MehtA+

Alex Xie
Gilman Upper School
MehtA+

Maahir Doshi
Haileybury
MehtA+

July 28, 2022

ABSTRACT

Agriculture plays an imminent role not just for farmers, but for the entire world. One of the major problems farmers face is disease attacks which lead to plant loss and reduction in plant growth. These losses can be reduced by monitoring the health of the crop plant and keeping track of any disease the plant is susceptible to or has developed. Image recognition is the most recent and optimal method to detect any plant diseases. This project explores the usage of convolutional neural networks to predict diseases carried by the plant using pictures of leaves. We compare the performances of three models: a simple CNN model, a VGG-16 model, and a VGG-16 model pre-trained on ImageNet. The CNN and pre-trained VGG-16 model had the best validation accuracy of around 89 percent. We used Grad-CAM to understand the salient features that the model uses to classify the leaf images.

1 Introduction

Reaching high-income status is one of the primary goals of every developing country. Agriculture plays a critical role in helping countries reach this goal, by not only transforming economies but also achieving other important developmental goals like ensuring food security and reducing malnutrition rates. Crop plant disease, however, is a major obstacle in increasing crop production and ensuring crop quality [1]. Plant diseases also exacerbate the current deficit of food supply, in which at least 800 million people are inadequately fed. [2] Crop losses due to pests and pathogens are around 20 and 40 percent of the total production [3], which further reduces the productivity of crop plants by 10-95 percent [4].

It is imperative for farmers to be able to identify the disease and disease-causing agents to improve their disease control measures. Failure to do so can not only lead to wastage of time and money, but also increase plant losses [5].

Although it is possible to detect diseases through manual inspection or with the aid of botanic experts, this task proves to be extremely laborious and time-consuming. Hence, the need arises for machine-based methods to detect plant diseases with better efficacy and in less time [6]. This is exactly what we strive to achieve through this model.

2 Related Work

Several research studies for image-based plant disease detection using machine learning have been reported. In summary, the majority of them were based on traditional classification techniques such as Support Vector Machines (SVM'S), Naive Bayes, and K-nearest neighbors (K-NN). However, most research in plant disease detection has preferred the use of the SVM learning algorithm over all other classifiers [7].

Camargo and Smith were early researchers to set forward their work based on pattern recognition using SVM. Diseased regions such as spots and lesions were identified and fed as inputs to the classifiers. With 117 images to train on, they recorded an accuracy of 93 percent. [8]. Bernardes et al. also incorporated the SVM model to detect the cotton foliar disease wherein the wavelet transform was used as the feature extractor [9].

Possibly the first application of DL to plant disease detection was presented by Mohanty et al. and Sladojevic et al who used transfer learning and gathered images that constituted a larger dataset [10, 11]. Using AlexNet and GoogLeNet, and training from scratch, they achieved an accuracy of almost 95 percent.

In this paper, however, we abandon the conventional approach, turning to the VGG-16 model and train it on a much larger PlantVillage Dataset and aim to achieve higher accuracy. A work reported by Xu et al. demonstrated a test accuracy of 90.4 percent using a VGG-16 model which was trained with transfer learning on a different dataset[12]. Nonetheless, this accuracy could have been increased by using the pre-trained version or building a VGG-16 model from scratch- the two approaches that we have implemented in this paper. We also train our dataset on a CNN model and compare it to VGG-16.

3 Methodology

3.1 Dataset

The PlantVillage dataset consists of 54303 healthy and unhealthy leaf images that are further divided into 38 categories by the species name and disease. Since the original dataset is not available from the original source, we used another dataset from Kaggle that had almost 20641 leaf images belonging to the species of Tomato, Pepper Bell, and Potato (figure 1) that were either healthy or unhealthy. The unhealthy images were organized in various directories by different diseases that the leaves were affected by.

Preprocessing The PlantVillage dataset consists of 54303 healthy and unhealthy leaf images that are further divided into 38 categories by the species name and disease. Since the original dataset is not available from the original source, we used another dataset that had almost 20641 leaf images belonging to the species of Tomato, Pepper Bell, and Potato that were either healthy or unhealthy. The unhealthy images were organized in various directories by different diseases that the leaves were affected by. According to figure 2, there were 3 categories for Potato, 2 for Pepper Bell, and the rest 10 for Tomato.

This dataset had 4 redundant images that were neither in the JPG or PNG format and were removed entirely in order to prevent any misclassifications or errors in the code.

3.2 Model

We implemented 3 models: VGG-16 from scratch, the Pre-trained VGG-16 model, and a simple CNN model.

3.2.1 Simple CNN Model

The simplest CNN model passes the image through 8 layers. The images are resized to 255 pixels and trained on 100 epochs at the 0.001 learning rate using the Adam optimizer. The Cross-Entropy Loss is used as the loss function, which is given by: [13]

$$l = -(ylog(p) + (1 - y)log(1 - p)) \tag{1}$$

3.2.2 VGG-16 Models

The VGG-16 model is a type of CNN architecture and has an important distinguishing element that made us choose it. There are other famous architectures like AlexNet and ZFNet that have filter sizes as large as 11x11 and 7x7 with strides 4 and 2 respectively. The VGG-16 model, on the other hand, uses a comparatively smaller filter size of 3x3 with a stride of 1 pixel. These small-size convolution filters allow the model to have a larger number of weight layers, which leads to a better performance in transfer learning.

The VGG-16 model has 13 convolutional layers, 5 Max Pooling layers (these 2 are consistently arranged through the architecture), and 3 Dense layers which equal to 21 layers, out of which only 16 of them are weight layers. It takes an input tensor of size (224,224) with 3 RGB channels. Convolutional Layer 1 has 64 filters, Layer 2 has 128, Layer 3 has 256, and Layer 4 and 5 have 512 filters each. Three dense layers follow these convolutional layers. The first two have 4096 channels each, the third has 1000 channels, and the final layer is the soft-max layer (figures 3, 5). The model is trained on a subset of the ImageNet dataset, which has over 14 million images belonging to 22,000 categories. However, the VGG-16 model can also be trained from scratch on a custom dataset, which was our first approach. We later opted to implement the pre-trained model.

Figure 1: Examples of leaf images in the dataset. top left: Potato early blight, top right: pepper bell bacteria, bottom left: tomato leaf mold, bottom right: tomato bacterial spot

VGG-16 Model using the custom dataset The VGG-16 model implemented from scratch was trained on the PlantVillage images.

The model was first trained using the Adam optimizer, while the activation functions were reLU and softmax (figure 4). Afterward, we also decided to use the RMSprop optimizers to see if the accuracies would be any different, for which the activation functions were left untouched.

Pre-trained VGG-16 Model For the pre-trained VGG16 model, we decided to go solely with the RMSprop optimizer. The activation function used was softmax (figure 6).

4 Results

4.1 Simple CNN Model

The simple CNN model had the best validation accuracy score. The accuracy for trained images was 99.38 percent, while for validation images, it was around 89.5 percent. The evaluation performance graph 7 clearly depicts the increase

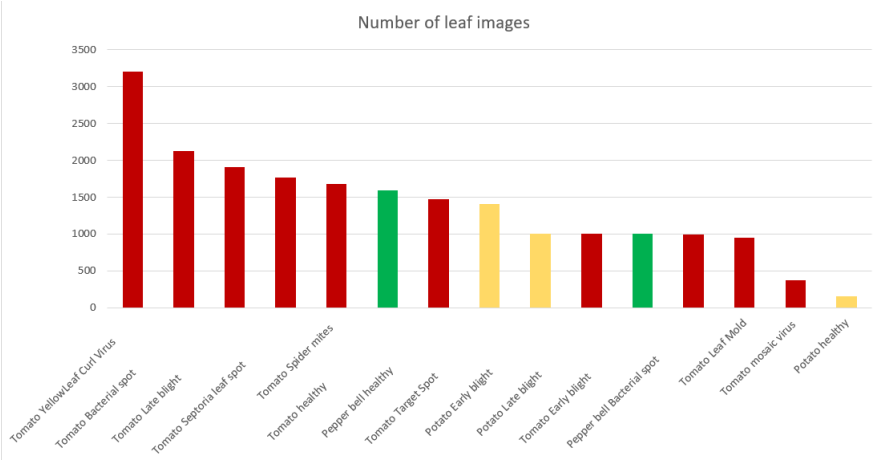

Figure 2: Number of images in each class

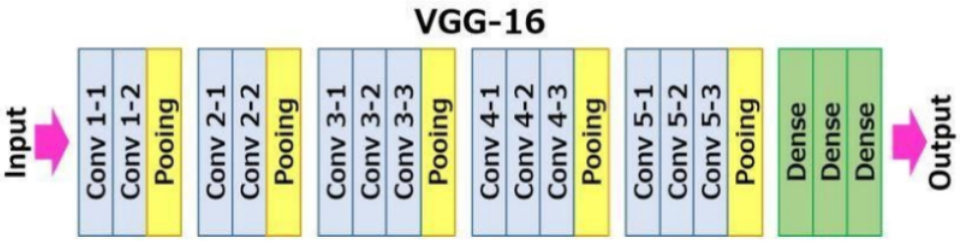

Figure 3: VGG-16 Architecture 1

in both accuracies before the 20th epoch itself, and although they are subjected to fluctuation, the training accuracy stays between 90 to 100, and validation accuracy between 80 to 90.

4.2 VGG-16 Model using the custom dataset

The VGG-16 model that was built from scratch gave the lowest accuracy. It was first trained on the Adam optimizer for 100 epochs for which the training accuracy remained fixated between 14 to 17 percent, as can be seen in figure 8.

However, on changing the optimizer to RMSprop, the accuracy increased slightly by 1-2 percent.

4.3 Pre-trained VGG-16 Model

Therefore, while implementing the pre-trained VGG-16 model, we decided to use only the RMSprop optimizer. This model successfully returned an accuracy of 99.91 percent on training images and 88.11 percent on the validation images when trained for 50 epochs (figure 9).

Clearly, the VGG-16 model trained by using only the PlantVillage dataset fails to correctly predict any leaf disease, potentially because of the small number of training and validation images to train on. The pre-trained model, on the other hand, has a much larger accuracy as it has been trained on a very large dataset provided by ImageNet, which is a large image dataset with nearly 14 million annotated images. We are able to achieve a high accuracy with the pre-trained version, majorly because of transfer learning between the ImageNet and PlantVillage dataset. The simple CNN model has very similar accuracies to that of the pre-trained VGG-16 model (comparison graph figure 10).

5 Interpretability using Grad-CAM

We implemented Grad-CAM to determine exactly which areas of the leaf were used by the model in its prediction. Gradient-weighted Class Activation Mapping (Grad-CAM) uses the gradients flowing into the final convolutional

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 224, 224, 64)	1792
conv2d_1 (Conv2D)	(None, 224, 224, 64)	36928
max_pooling2d (MaxPooling2D)	(None, 112, 112, 64)	0
conv2d_2 (Conv2D)	(None, 112, 112, 128)	73856
conv2d_3 (Conv2D)	(None, 112, 112, 128)	147584
max_pooling2d_1 (MaxPooling2	(None, 56, 56, 128)	0
conv2d_4 (Conv2D)	(None, 56, 56, 256)	295168
conv2d_5 (Conv2D)	(None, 56, 56, 256)	590080
conv2d_6 (Conv2D)	(None, 56, 56, 256)	590080
max_pooling2d_2 (MaxPooling2	(None, 28, 28, 256)	0
conv2d_7 (Conv2D)	(None, 28, 28, 512)	1180160
conv2d_8 (Conv2D)	(None, 28, 28, 512)	2359808
conv2d_9 (Conv2D)	(None, 28, 28, 512)	2359808
max_pooling2d_3 (MaxPooling2	(None, 14, 14, 512)	0
conv2d_10 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_11 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_12 (Conv2D)	(None, 14, 14, 512)	2359808
max_pooling2d_4 (MaxPooling2	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 4096)	102764544
dense_1 (Dense)	(None, 4096)	16781312
dense_2 (Dense)	(None, 15)	61455

Total params: 134,321,999
Trainable params: 134,321,999
Non-trainable params: 0

Figure 4: VGG-16 model from scratch summary

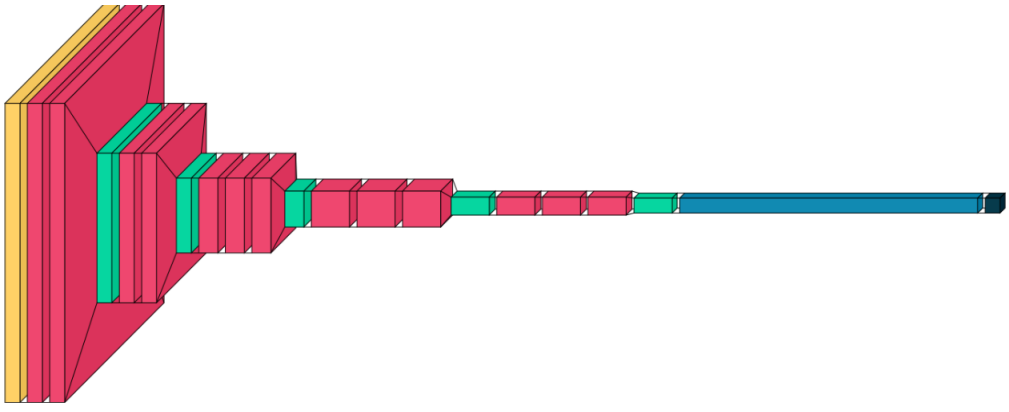

Figure 5: VGG-16 Architecture 2

layer to produce a localization map to depict the important regions of the image that influences the decision-making algorithm of the machine learning model [14]. Grad-CAMs are essentially the top-rated methods when it comes to model evaluation metrics [15]. Although Grad-CAM for our model did not give very strong results for all images, it was helpful in concluding that the model was still using the right areas for most images, as shown by the leaf images in figure 11.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 100, 100, 3)]	0
block1_conv1 (Conv2D)	(None, 100, 100, 64)	1792
block1_conv2 (Conv2D)	(None, 100, 100, 64)	36928
block1_pool (MaxPooling2D)	(None, 50, 50, 64)	0
block2_conv1 (Conv2D)	(None, 50, 50, 128)	73856
block2_conv2 (Conv2D)	(None, 50, 50, 128)	147584
block2_pool (MaxPooling2D)	(None, 25, 25, 128)	0
block3_conv1 (Conv2D)	(None, 25, 25, 256)	295168
block3_conv2 (Conv2D)	(None, 25, 25, 256)	590080
block3_conv3 (Conv2D)	(None, 25, 25, 256)	590080
block3_pool (MaxPooling2D)	(None, 12, 12, 256)	0
block4_conv1 (Conv2D)	(None, 12, 12, 512)	1180160
block4_conv2 (Conv2D)	(None, 12, 12, 512)	2359808
block4_conv3 (Conv2D)	(None, 12, 12, 512)	2359808
block4_pool (MaxPooling2D)	(None, 6, 6, 512)	0
block5_conv1 (Conv2D)	(None, 6, 6, 512)	2359808
block5_conv2 (Conv2D)	(None, 6, 6, 512)	2359808
block5_conv3 (Conv2D)	(None, 6, 6, 512)	2359808
block5_pool (MaxPooling2D)	(None, 3, 3, 512)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 15)	69135

Total params: 14,783,823
Trainable params: 69,135
Non-trainable params: 14,714,688

Figure 6: Pre-trained VGG-16 model summary

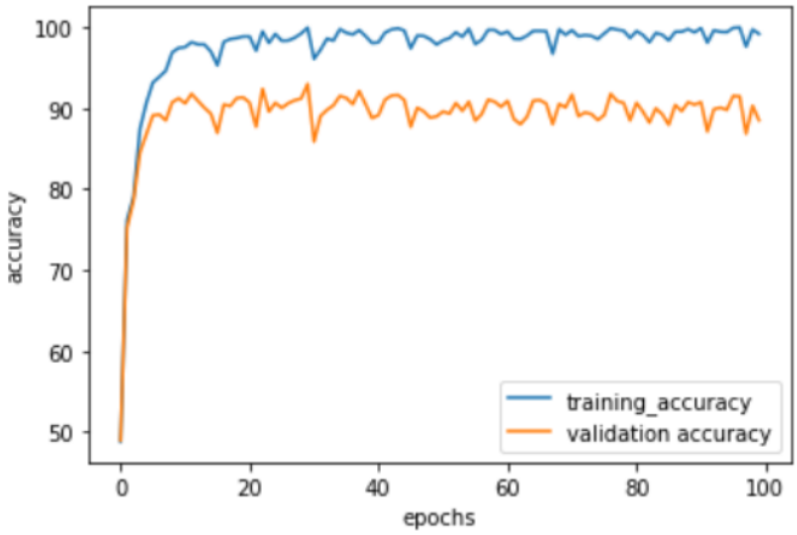

Figure 7: Simple CNN model accuracy

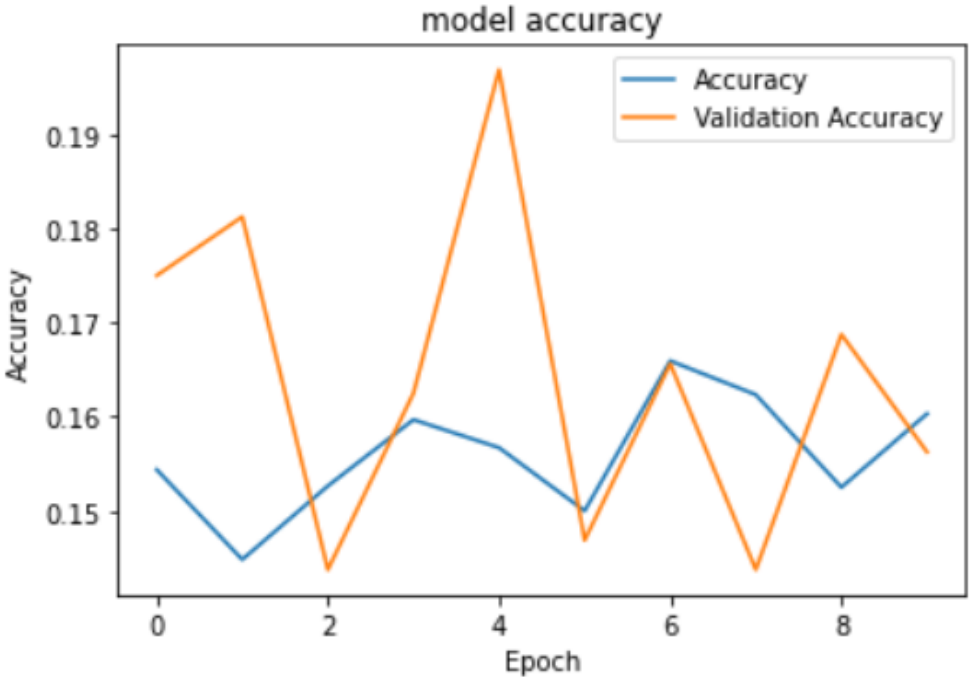

Figure 8: VGG-16 model from scratch accuracy

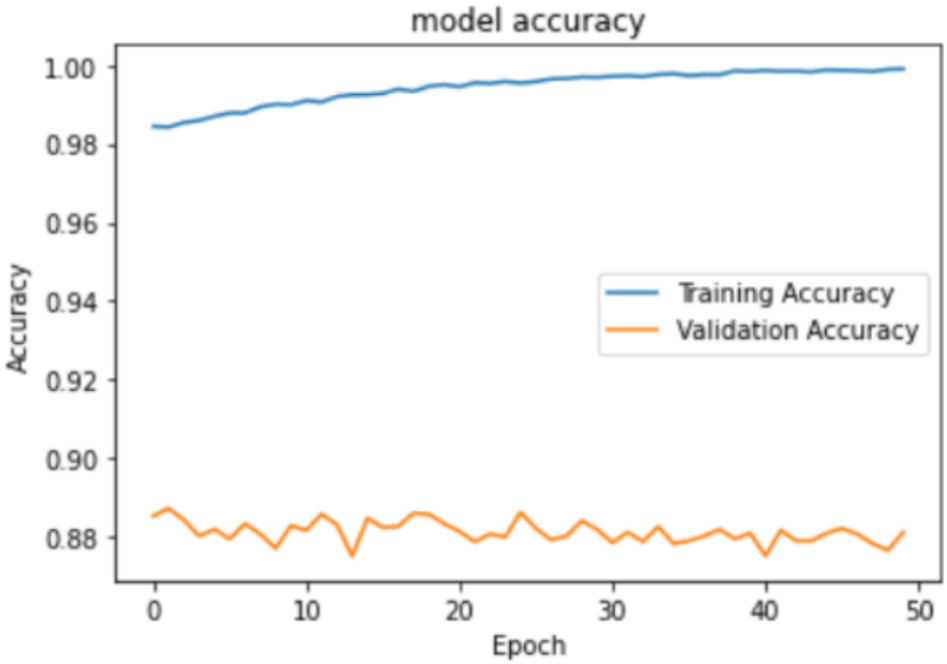

Figure 9: Pre-trained VGG-16 model accuracy

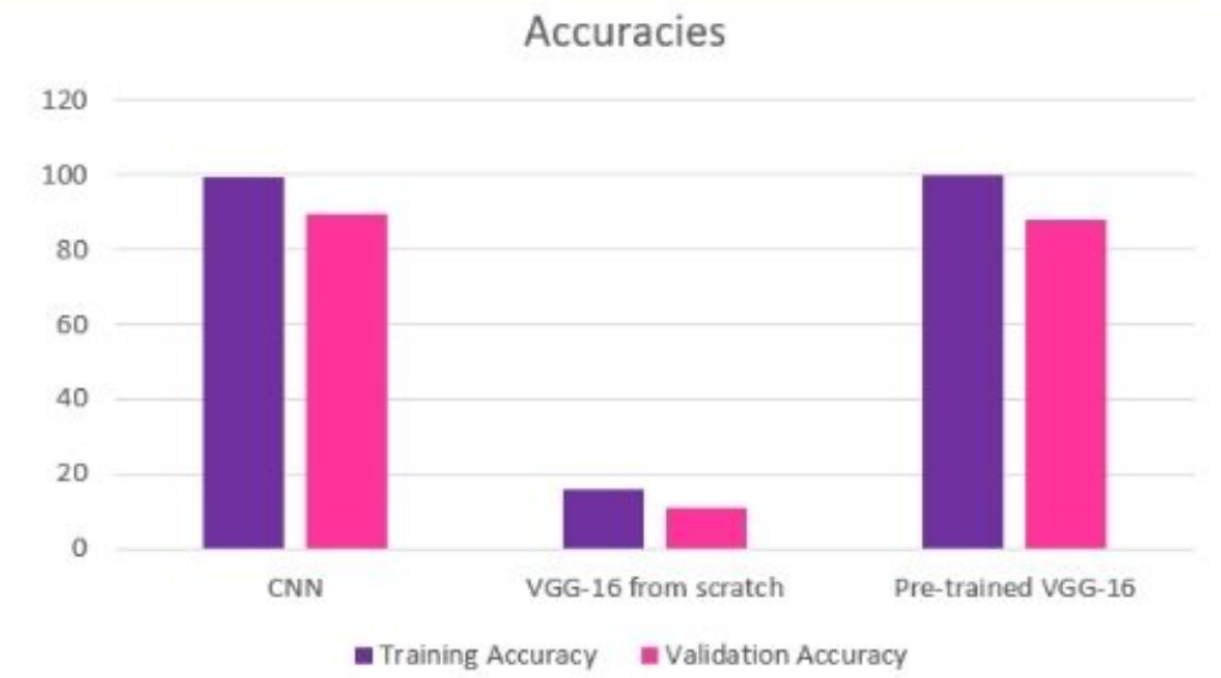

Figure 10: Comparing the accuracies

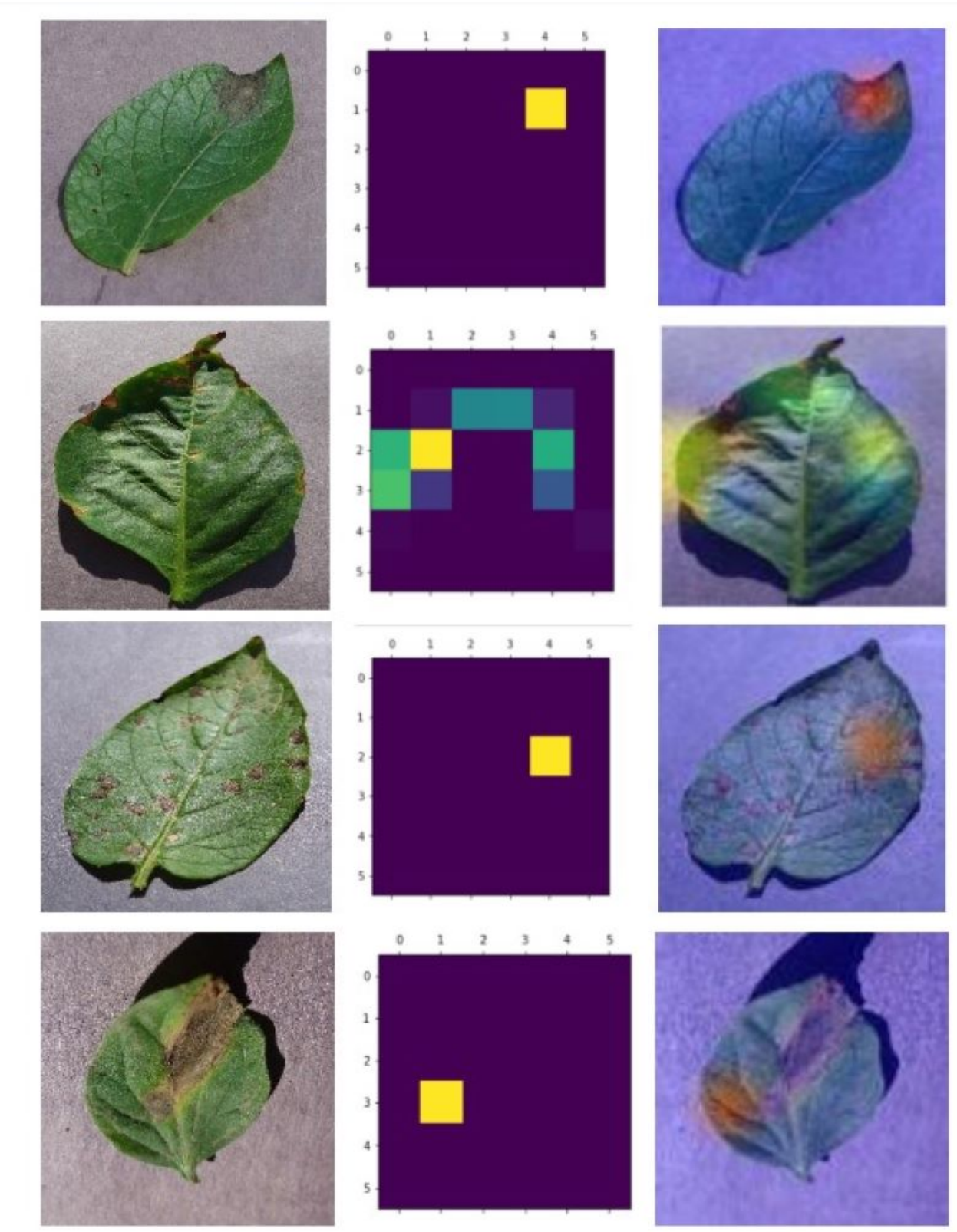

Figure 11: Grad-CAM on Pre-trained VGG-16 model

6 Future Work

Although this model is helpful in determining whether a plant is healthy or not, it is limited to very few diseases and just 3 of the major crop plants. Future work would involve gathering a larger dataset having more crop plants and a wider range of diseases that affect them so as to increase its versatility and also improve its performance.

Further, this model could be made more useful and productive if it was able to detect the reason behind the growth of the particular disease. This would be a complicated task to achieve, but could possibly be implemented if additional data like the area/region of growth of the plant, the precipitation and/or humidity level it was exposed to, other weather conditions it was grown in, type of soil, etc. were to accompany the image datasets. Acquiring this information along with more crop plant images would prove to be a great way to modify the model from unimodal to multimodal.

7 Conclusion

We implemented 3 models: A simple CNN model, a VGG-16 model built from scratch, and a pre-trained VGG-16 model to detect the disease a potato, tomato, or pepper bell plant may have by the image of their leaf. The results of the study show that while both the conventional CNN model and pre-trained VGG16 model were effective in reaping a high training and validation accuracy and hence are the best models to implement, it was the VGG-16 model trained on the custom dataset of leaf images that was ineffective in predicting almost any disease accurately. In addition, the use of Grad-CAM makes the models more interpretable by depicting the salient features that are used for the classification of the leaf image.

To view our code, click here

8 Division of Labor

We divided the work as follows:

- Inputting, filtering, and processing the data: Kamakshi
- Training the CNN model: Kamakshi, Maahir Doshi
- Training the VGG-16 models: Kamakshi
- Performance metric graphs: Kamakshi
- Grad-CAM: Kamakshi
- Research paper: Kamakshi
- Poster: Kamakshi, Alex Xie

9 Acknowledgements

We would sincerely like to thank our instructors, Ms. Haripriya Mehta, Ms. Andrea Jaba, Mr. Bhagirath Mehta, and Ms. Anna Li for guiding and helping us throughout the camp and this project.

References

[1] Qi A. Fitt B.D.L. Richard, B. Control of crop diseases through integrated crop management to deliver climate-smart farming systems for low- and high-input crop production. https://doi.org/10.1111/ppa.13493.

[2] Scott PR Strange RN. Plant disease: a threat to global food security.

[3] Serge Savary Andrea Ficke Jean-Noël Aubertot Clayton Hollier. Crop losses due to diseases and their implications for global food production losses and food security. 1 June 2012.

[4] ASMH Bari-E Hossain F Ahmed, HA AI-Mamun. Classification of crops and weeds from digital images: A svm approach. Elsevier-2012.

[5] M.R. Williamson Riley, M.B. and O. Maloy. Plant disease diagnosis. DOI: 10.1094/PHI-I-2002-1021-01.

[6] Wang X Liu, J. Plant diseases and pests detection based on deep learning: a review. https://doi.org/10.1186/s13007-021-00722-9.

[7] R. Ehsani S. Sankaran, A. Mishra and C. Davis. A review of advanced techniques for detecting plant diseases. https://dx.doi.org/10.1007/s11263-019-01228-7 10.1007/s11263-019-01228-7.

[8] A. Camargo and J. Smith. Image pattern classification for the identification of disease-causing agents in plants.

[9] J. M. R. S. Tavares A. A. Bernardes and Eds. Dordrecht R. M. Natal Jorge. Identification of foliar diseases in cotton crop. 2017.

[10] D. P. Hughes S. P. Mohanty and M. Salathé. Using deep learning for image-based plant disease detection. 2016.

[11] A. Anderla D. Culibrk S. Sladojevic, M. Arsenovic and D. Stefanovic. Deep neural networks based recognition of plant diseases by leaf image classification. 2016.

[12] Y. Guo H. Yang R. Zhang P. Xu, G. Wu. Automatic wheat leaf rust detection and grading diagnosis via embedded image processing system. 2017.

[13] Saurav Maheshkar. Weights biases. what is cross entropy loss- a tutorial with code.

[14] Abhishek Das Ramakrishna Vedantam Devi Parikh Dhruv Batra Ramprasaath R. Selvaraju, Michael Cogswell. Grad-cam: Visual explanations from deep networks via gradient-based localization. https://dx.doi.org/10.1007/s11263-019-01228-7 10.1007/s11263-019-01228-7.

[15] Cogswell M. Das A. Vedantam R. Parikh D. Selvaraju, R. R. and D Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization.