Project 2 Report Paper:

Harmful Brain activity Classification with KerasCV

Joseph Martínez

Old Dominion University, jmart130@odu.edu

*Abstract* - The Plant Pathology 2021 - FGVC8 Kaggle competition addresses the imperative for automated identification and classification of foliar diseases in apple trees. Manual scouting for disease diagnosis in apple orchards is not only time-consuming but also financially burdensome. Leveraging computer vision-based models holds promise for disease identification; however, challenges such as variations in visual symptoms across cultivars and environmental factors hinder accuracy. Building upon insights from the previous Plant Pathology 2020-FGVC7 challenge, the 2021 competition presents a comprehensive dataset of approximately 23,000 high-quality RGB images of apple foliar diseases. Our objective is to develop machine learning-based models capable of accurately classifying leaf images and identifying individual diseases. Utilizing SAYANTIKA NAG's CNN architecture as a foundation, modifications were made to adapt the model for local GPU training. Key adjustments included reducing image size to 64x64 pixels, increasing image saturation, and simplifying the architecture to three convolutional layers. Training employed a focal loss function, an Adam optimizer, and a reproducibility feature. Results indicate an accuracy of 30.65%, falling short of top-performing models on the Kaggle leaderboard. Challenges such as limited data and computational resources, simplified model architecture, and high-class imbalance impacted performance. However, the project underscores the importance of data quantity, model complexity, and iterative development in enhancing disease diagnosis. In conclusion, while the ultimate goal of achieving top-tier performance was not met, the project provides valuable insights for future advancements. Addressing limitations through dataset augmentation, exploring sophisticated architectures, and refining training techniques can lead to more accurate solutions for plant pathology diagnosis, contributing to sustainable agriculture practices.

*Index Terms* – Computer Vision, CNN, foliar disease, image recognition.

Problem Definition

The Plant Pathology 2021 - FGVC8 Kaggle competition [1] aims to address the pressing need for automated identification and classification of foliar diseases in apple trees. Currently, disease diagnosis in apple orchards relies heavily on manual scouting by humans, which is not only time-consuming but also financially burdensome.

Despite the potential of computer vision-based models in disease identification, there exist significant challenges that must be overcome. Variations in visual symptoms of diseases across different apple cultivars, as well as environmental factors such as leaf color, morphology, and lighting conditions during image capture, pose substantial obstacles for accurate disease classification.

Building upon the success and lessons learned from the Plant Pathology 2020-FGVC7 challenge, 2021's competition presents a more extensive dataset consisting of approximately 23,000 high-quality RGB images of apple foliar diseases. The dataset includes diverse backgrounds, leaf maturity stages, and varying focal camera settings, providing a more comprehensive representation of real-world scenarios.

The primary objective of this competition is to develop machine learning-based models capable of accurately classifying leaf images into specific disease categories and identifying individual diseases amidst multiple symptoms present on a single leaf image.

By leveraging the provided dataset and advancing state-of-the-art machine learning techniques, participants are tasked with creating models that can significantly enhance the efficiency and effectiveness of disease diagnosis in apple orchards. Through this competition, we aim to foster innovation in agricultural technology and contribute to the sustainable management of plant diseases.

Methods

Code *HMS-HBAC: KerasCV Starter Notebook[[1]](#footnote-1)* from by the Keras team.

We intended to train a SOTA CNN based on the code from SAYANTIKA NAG, but since the model is computationally heavy, we realized and we will list several modifications that we needed to do in order to run in a local GPU of 8GB of RAM in a considerable time.

SAYANTIKA NAG's solution presented for the Plant Pathology 2021 - FGVC8 Kaggle competition employs a convolutional neural network (CNN) architecture for the classification of foliar diseases in apple trees. This section outlines the methodology, including data preprocessing, model architecture, training process, and evaluation.

## Data Preprocessing:

- Image Resizing: The input images are resized to a fixed dimension of 128x128 pixels using bicubic interpolation to maintain image quality.

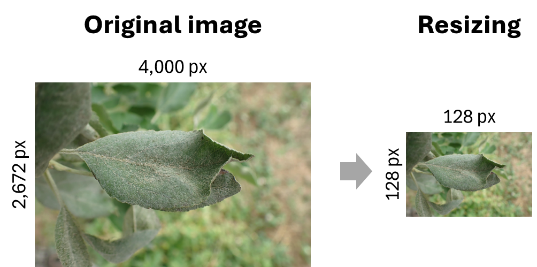


Figure I

Image resizing

## Model Architecture:

- Convolutional Neural Network (CNN): The model architecture consists of three convolutional layers followed by max-pooling layers. Each convolutional layer is activated using the Rectified Linear Unit (ReLU) activation function to introduce non-linearity. The output of the final convolutional layer is flattened and fed into a fully connected layer, followed by a softmax activation function to produce class probabilities. Below is the architecture:

CNN\_model(

(conv1): Conv2d(3, 16, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu1): ReLU()

(maxpool1): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(conv2): Conv2d(16, 32, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu2): ReLU()

(maxpool2): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(conv3): Conv2d(32, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu3): ReLU()

(maxpool3): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(conv4): Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu4): ReLU()

(maxpool4): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(fc): Linear(in\_features=8192, out\_features=12, bias=True)

)

Figure II

Original model architecture

## Training Process:

- Dataset Splitting: The whole dataset was used and splitted into training and validation sets with an 80:20 ratio, ensuring model generalization and performance assessment.

- Focal Loss Function: A custom Focal Loss function is utilized as the optimization criterion. This loss function addresses class imbalance by down-weighting well-classified examples, focusing more on hard-to-classify examples.

- Adam Optimizer: The Adam optimizer is employed with a learning rate of 0.001 to update the model parameters during training.

- Training Loop: The model is trained over multiple epochs. In each epoch, the training dataset is iterated through batches of images and their corresponding labels. The model's outputs are compared with ground truth labels to compute the Focal Loss, which is then minimized using backpropagation.

## Evaluation:

- Validation Loop: After each training epoch, the model's performance is evaluated on the validation set. The accuracy metric is calculated by comparing the predicted labels with the ground truth labels. This provides insights into the model's ability to generalize to unseen data.

- Model Saving: Upon completion of training, the trained model parameters are saved locally for future inference.

Additional Considerations:

- Label Mapping: To facilitate model training, a mapping between text labels and numerical indices is created. This enables the conversion of text labels to numerical representations compatible with the model's output layer.

- Device Usage: The model is trained on the available hardware accelerator (GPU) if present, otherwise, it falls back to using the CPU for computation.

The implementation follows best practices in deep learning for image classification tasks, aiming to achieve high accuracy in identifying foliar diseases in apple trees while addressing challenges such as data variation and class imbalance.

Changes to Original Architecture

In addition to SAYANTIKA NAG's original architecture, several modifications were made to adapt the model for training on a local GPU with limited computational resources and to enhance its performance in recognizing foliar diseases in apple trees.

## Data Preprocessing:

• Image Size Reduction: The size of the input images was reduced from 128x128 to 64x64 pixels. This reduction in image size helps reduce computational complexity while maintaining essential features for disease recognition.

• Saturation Increment: Saturation of the images was increased to augment the dataset. This augmentation technique enhances color variation in the images, making it easier for the model to recognize subtle differences in leaf coloration, particularly dark spots indicative of disease presence.



Figure III

Image resizing and saturation increase

## Model Architecture:

• Reduction of Convolutional Layers: The original architecture included four convolutional layers. However, to mitigate computational overhead, one convolutional layer was removed, resulting in a simplified architecture with three convolutional layers.

• Adjustment in Image Size: With the reduction in image size, the model's architecture was adapted to accommodate the smaller input dimensions of 64x64 pixels, ensuring compatibility between the input size and the network architecture. Below is the architecture:

CNN\_model(

(conv1): Conv2d(3, 16, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu1): ReLU()

(maxpool1): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(conv2): Conv2d(16, 32, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu2): ReLU()

(maxpool2): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(conv3): Conv2d(32, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(relu3): ReLU()

(maxpool3): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(fc): Linear(in\_features=4096, out\_features=12, bias=True)

)

Figure IV

Modified model architecture

## Training Process:

• Batch Size Increase: The batch size was significantly increased to 2048 to leverage GPU parallelization effectively. This adjustment helps optimize GPU memory usage and accelerates training without compromising model performance.

• Data Subset Selection: Only 10% of the original dataset was utilized for training, validation, and testing. From this subset, 80% was allocated for training, 10% for validation, and 10% for testing, ensuring sufficient data for model training and evaluation while reducing computational overhead.

• Reproducibility Feature: A reproducibility feature was incorporated to ensure that the model's results can be replicated across different runs. This feature includes setting random seeds for various components of the training process, such as data loading, model initialization, and optimization, ensuring consistent behavior across different executions.

These modifications aim to streamline the training process, optimize computational resources, and enhance the model's ability to accurately identify foliar diseases in apple trees. By fine-tuning the architecture and training parameters, the model achieves a balance between performance and computational efficiency, making it suitable for deployment in resource-constrained environments.

Results

The performance of the modified convolutional neural network (CNN) model in classifying foliar diseases in apple trees was evaluated based on several metrics. The obtained results are as follows:

TABLE I

Performance metrics

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy  Precision  Recall  F1 Score | 0.3065  0.0939  0.3065  0.1438 |

For comparison, Table II shows the scored of the top 5 leaderboard of the competition.

TABLE II

Performance of leaderboard top 5 of the competition

|  |  |  |
| --- | --- | --- |
| # | Team | Score |
| 1  2  3  4  5 | baseline  Luminide  Data Oriented  Team Why  mknzfr | 0.88336  0.87560  0.87469  0.87287  0.86851 |

Comparing these metrics with the top performers on the Kaggle leaderboard, it is evident that the model's performance falls significantly short in terms of accuracy and other evaluation metrics. The model achieved an accuracy of 30.65%, which is substantially lower than the top leaderboard entries, which achieved accuracies above 86%.

The relatively low performance of the model can be attributed to several factors:

1. Limited Data and Reduced Training Size: Utilizing only 10% of the original dataset for training, validation, and testing might have resulted in insufficient data for the model to learn complex patterns effectively. This reduction in training data could have led to poor generalization and lower performance.

2. Simplified Model Architecture: The reduction in the number of convolutional layers and the removal of one layer from the original architecture might have resulted in a loss of representational capacity, limiting the model's ability to extract intricate features from the input images.

3. High Class Imbalance: The dataset might have exhibited significant class imbalance, with certain disease categories being overrepresented compared to others. This imbalance could have affected the model's ability to learn from minority classes, leading to lower precision and recall values.

4. Limited Computational Resources: Despite efforts to optimize the model for training on a local GPU with 8GB of RAM, the computational resources might still have been insufficient to train a complex CNN architecture effectively. This limitation could have hindered the model's capacity to learn intricate patterns from the data.

## Lessons Learned

This project highlights several key learnings:

1. Importance of Data Quantity and Quality: Adequate data quantity and diversity are essential for training robust machine learning models, particularly in complex tasks such as image classification. Future iterations of the project should focus on acquiring a larger and more diverse dataset to improve model performance.

2. Model Complexity vs. Computational Resources: Balancing model complexity with available computational resources is crucial. While simplifying the model architecture can improve training speed and resource utilization, it should be done judiciously to avoid compromising performance.

3. Continuous Iteration and Experimentation: Machine learning projects often require iterative development and experimentation. Different model architectures, hyperparameters, and training strategies should be explored systematically to identify the most effective approach.

Conclusion

In conclusion, while the modified CNN model exhibited shortcomings in achieving competitive performance levels compared to the top-performing models on the Kaggle leaderboard, the project has offered valuable insights into the intricacies of disease classification within agricultural settings. The constrained computational resources and the limited training data subset posed significant challenges in training a highly accurate model. Despite these obstacles, the project underscores the importance of continued exploration and innovation in the realm of plant pathology diagnosis.

While the ultimate goal of achieving state-of-the-art performance was not met, the project lays a foundation for future advancements in the field. Moving forward, addressing the limitations observed in this iteration, such as augmenting the dataset with a larger and more diverse collection of images, exploring more sophisticated model architectures, and leveraging advanced training techniques, can lead to the development of more robust and accurate solutions for plant pathology diagnosis. By iteratively refining methodologies and incorporating lessons learned from this endeavor, future iterations of the project hold the promise of realizing the overarching goal of enhancing disease detection and management in agricultural systems.

Acknowledgment

Special thanks to my family for their constant support and to my friends who helped me troubleshoot Pytorch errors, making this project possible.

References

1. Thapa, *Plant Pathology 2021 - FGVC8*. Kaggle.

Author Information

**Joseph Martínez,** Modeling and Simulation Engineering Master’s student at Old Dominion University (ODU).

1. <https://www.kaggle.com/code/awsaf49/hms-hbac-kerascv-starter-notebook> [↑](#footnote-ref-1)