Apple tree foliar disease recognition and classification

By

Venu Madhavi Dusanapudi

Amrutha Mulinti

Srinija Vaibhavi Boggavarapu

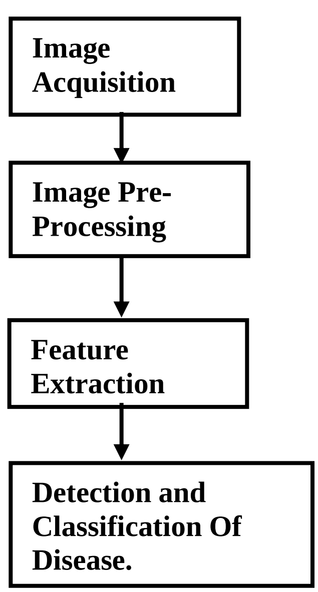
Abstract

Utilizing a convolutional neural network and CV2, we've created a predictive model for identifying diseases in apple tree leaves, addressing the longstanding issue of reduced crop quality and production in agriculture. The user-friendly front-end application allows seamless uploading of apple leaf images, providing instant predictions on the specific type of disease affecting them.

1. Introduction

Agriculture, a pivotal sector in numerous economies, relies heavily on the import and export of various fruits, with apples being particularly high in demand globally. Unfortunately, apple fields often fall prey to unavoidable diseases, impacting crop yields. Despite concerted efforts to enhance agricultural productivity, millions still face hunger due to challenges in accurately diagnosing crop-affecting diseases. Misdiagnoses can lead to detrimental consequences, including increased input costs and the emergence of resistant pathogen strains. Notably, major diseases like Rust, scab, and others significantly impede apple plant yields, necessitating early and accurate disease classification for effective prevention. This project focuses on classifying apple leaf diseases, utilizing a dataset of 1821 images and employing a convolutional neural network (CNN) and OpenCV to enhance disease detection and contribute to increased production rates.

2. Methods



2.1 Image Acquisition:

The process of obtaining an image from an outside source in order to process it further is known as image acquisition. Getting a picture is the first stage in the workflow since without it, no procedure can begin.

➔ **Categorization of Apple Leaves in the Dataset:**

- **Rust** is a plant disease caused by certain types of fungi. These fungi are quite particular, as they only target living plants. The infection process kicks off when a spore finds its way to a plant's surface, takes root, and starts invading the host. The infection primarily affects various parts of the plant, including leaves, petioles, tender shoots, stems, and fruits. Plants dealing with a severe rust infection might exhibit stunted growth, yellowing (chlorosis), or show visible signs of infection like rust-colored fruiting bodies.
- Apple **scab**, brought about by the fungus Venturia inaequalis, has an interesting life cycle. During winter, the fungus hangs out on fallen leaves affected by the disease. When spring arrives, these fungi release spores into the air. The wind then carries these spores to fresh leaves, flowers, fruits, or green twigs that are just starting to grow. To tackle apple scab, you can use a fungicide as part of an all-purpose spray for fruit trees—no need for insecticides, just focus on the fungicide component, and you're good to go.
- When we talk about the "**multiple diseases**" class, it means the plant is dealing with both scab and rust infections simultaneously.
- The "**healthy**" category indicates that the leaf is free from any type of disease.

2.2 Image Pre-Processing:

Image preprocessing is a crucial step in preparing leaf images for model training and inference. This process transforms raw input datasets of leaf images into a more refined and standardized format, enhancing the overall quality of the images. It involves addressing issues like undesired distortions, handling missing data, and correcting inconsistencies within the dataset.

Large datasets can pose challenges in terms of storage space and computational complexity due to varying feature dimensions. To tackle this, data reduction techniques are employed to streamline the dataset, ultimately improving the efficiency of image processing. By reducing the volume of data, the performance of image processing is enhanced, making it more manageable.

In essence, preprocessing techniques play a key role in readying datasets for the identification of leaf diseases through the analysis of leaf images. This ensures that the data used for model training is well-organized, free from unwanted artifacts, and optimized for effective disease recognition.

2.3 Feature Extraction:

Feature extraction is a crucial step in image processing, serving as a foundational phase that sets the stage with optimal conditions. In a CNN-based detection framework, the feature extractor plays a key role in capturing important characteristics of leaf diseases. This technique dives into the analysis of properties like color, shape, and texture in leaf images, providing a convenient way to distill meaningful information. By delving into the specifics of a leaf image, the feature extraction method becomes adept at identifying distinctive aspects such as lesion shapes and colors associated with different leaf diseases. This process essentially lays the groundwork for accurate classification, ensuring that the unique features of various diseases are effectively captured and utilized. In essence, feature extraction acts as a critical component in unraveling the nuanced details of leaf images for robust disease classification.

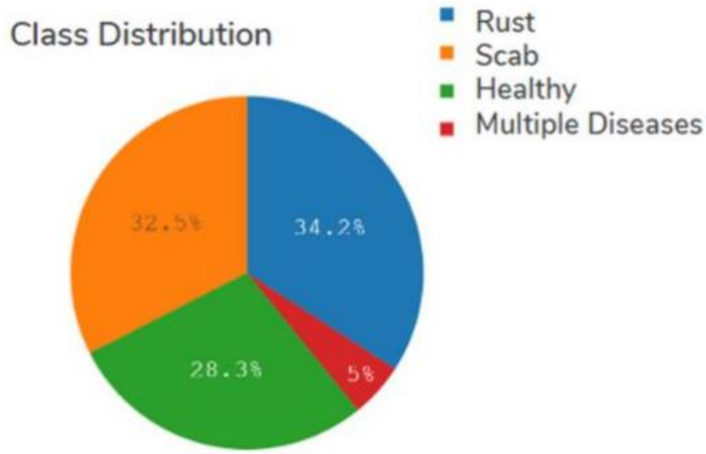
2.4 Detection and Classification of Apple Leaf Diseases Using CNN:

This project focuses on using Convolutional Neural Networks (CNNs) to detect and categorize diseases in apple leaves. CNNs were chosen for their effectiveness in image classification and recognition, boasting high accuracy. One key advantage is their reduced reliance on extensive preprocessing, minimizing the need for manual intervention and streamlining functionality development. Additionally, CNNs are known for their simplicity, making them easy to comprehend, and their quick implementation further contributes to their appeal in this context.

3. Experiments

3.1 Dataset:

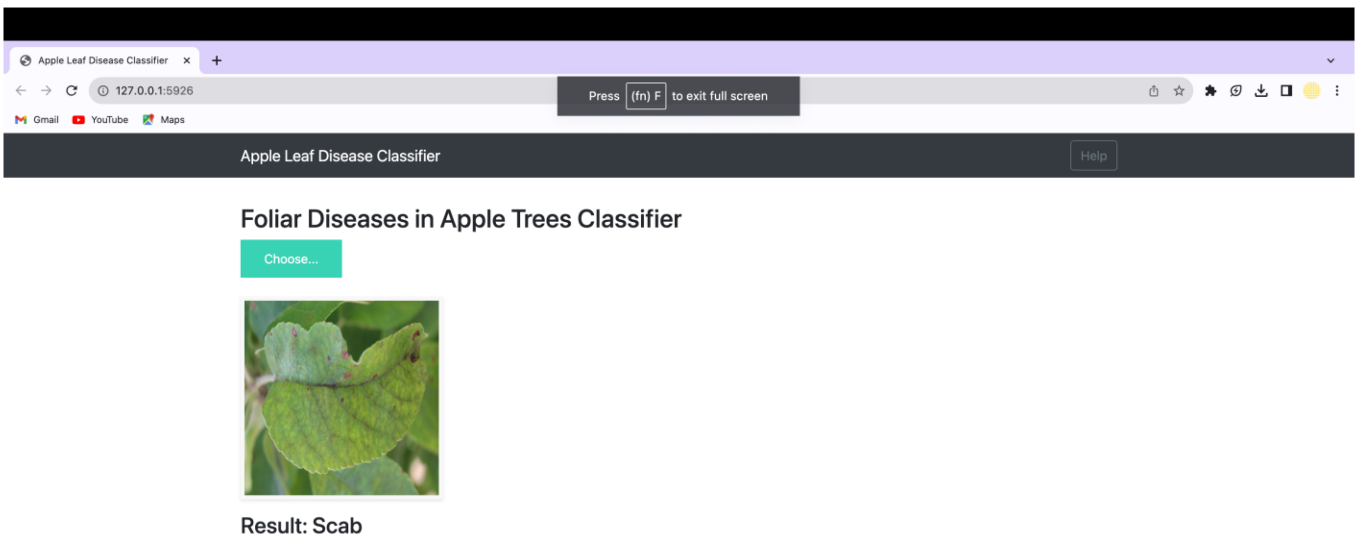
The plant pathology 2020 Kaggle dataset is what we utilized. There are 1821 test photos without labels and 1821 captioned training images of apple tree leaves. Four classes—healthy, scab, rust, and numerous diseases—are applied to the photos.

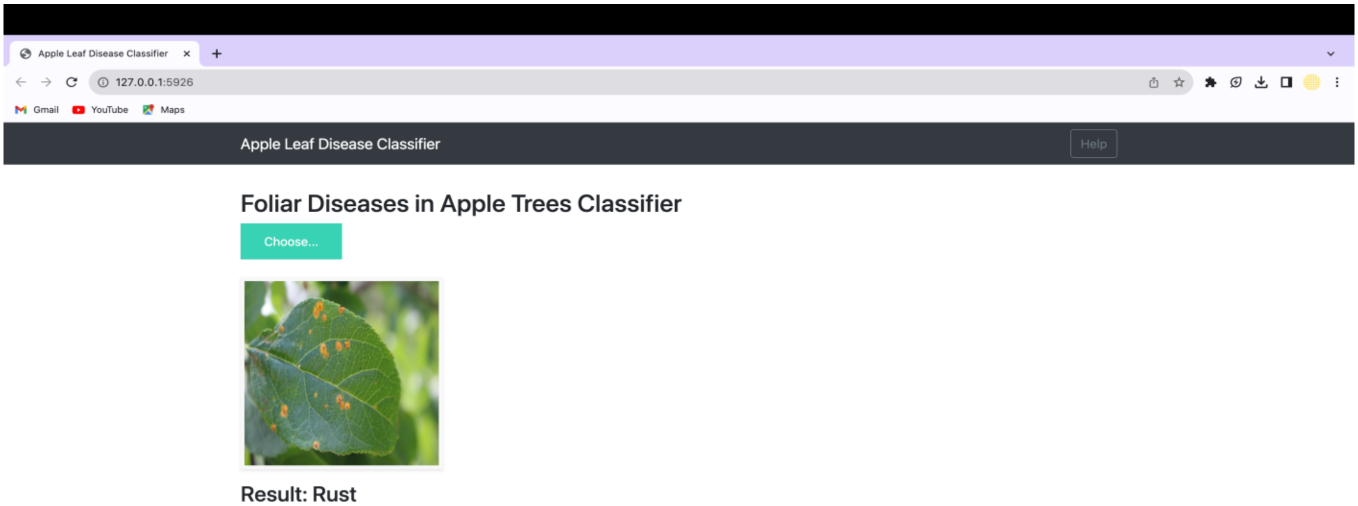


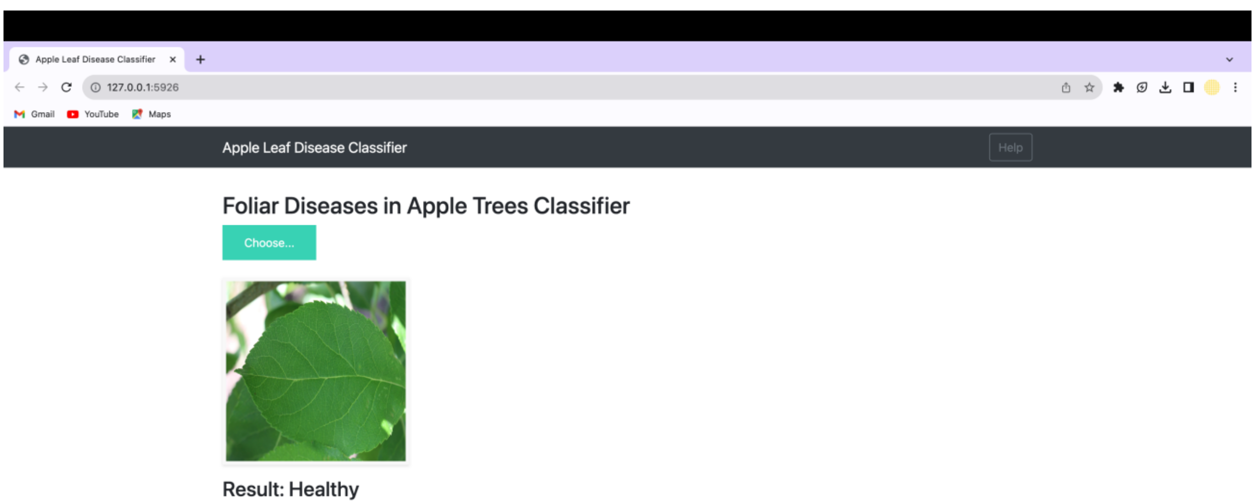
3.2 Evaluation metrics:

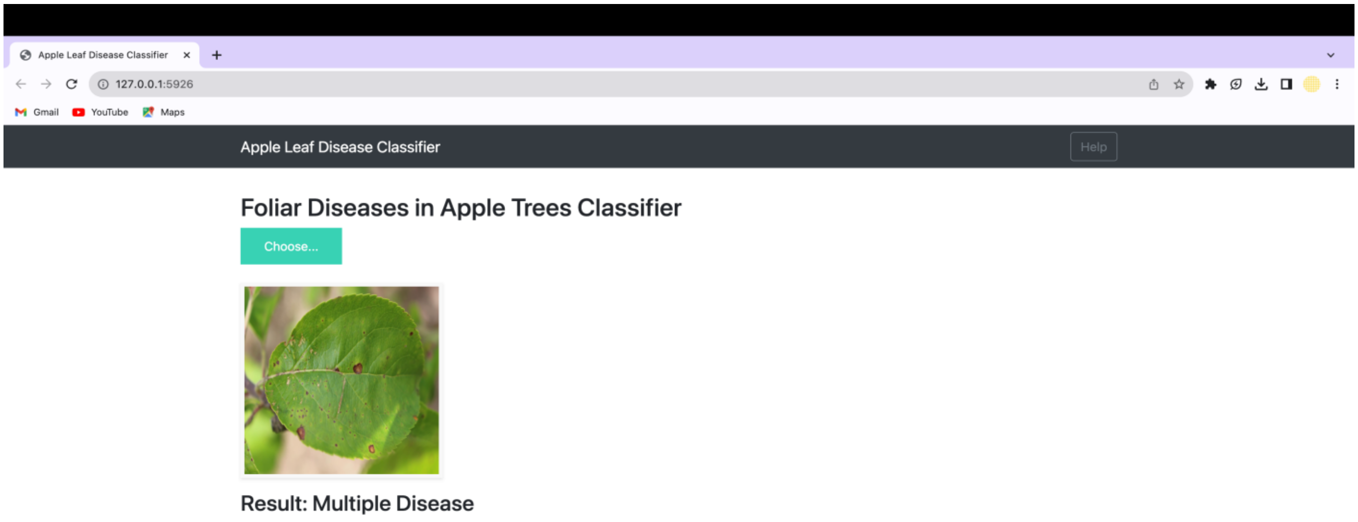
Effective evaluation metrics are crucial for assessing the performance of models in image classification tasks, such as identifying diseases in apple leaves. The accuracy metric gives an overall snapshot of the model's correctness in predictions. For a more detailed understanding, precision, recall, and the F1-score come into play, offering insights into the equilibrium between true positives, false positives, and false negatives. These metrics are particularly valuable for discerning how the model behaves regarding positive and negative classifications. The confusion matrix further dissects the model's predictions, delineating true positives, true negatives, false positives, and false negatives, providing a nuanced understanding of its overall performance. Metrics like the Receiver Operating Characteristic (ROC) curve and Area Under the Precision-Recall Curve (AUC-PR) are especially beneficial for gauging the model's discrimination capabilities between different classes, particularly in scenarios with imbalanced datasets. Collectively, these evaluation metrics paint a comprehensive picture of the model's efficacy in accurately categorizing various leaf diseases.

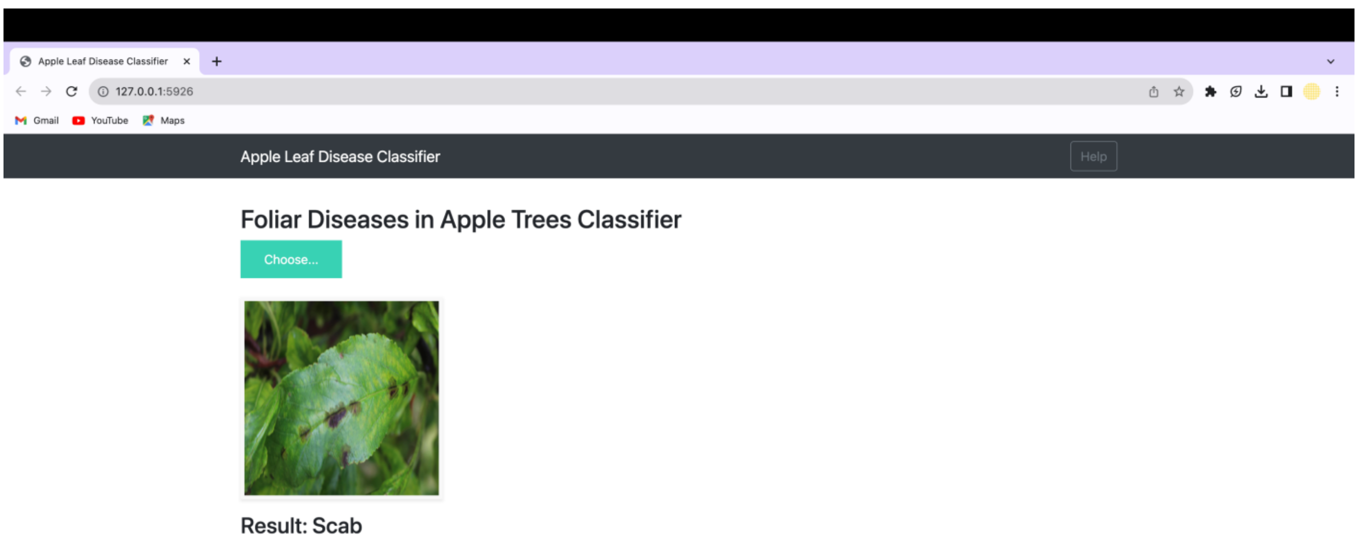
3.3 Results:

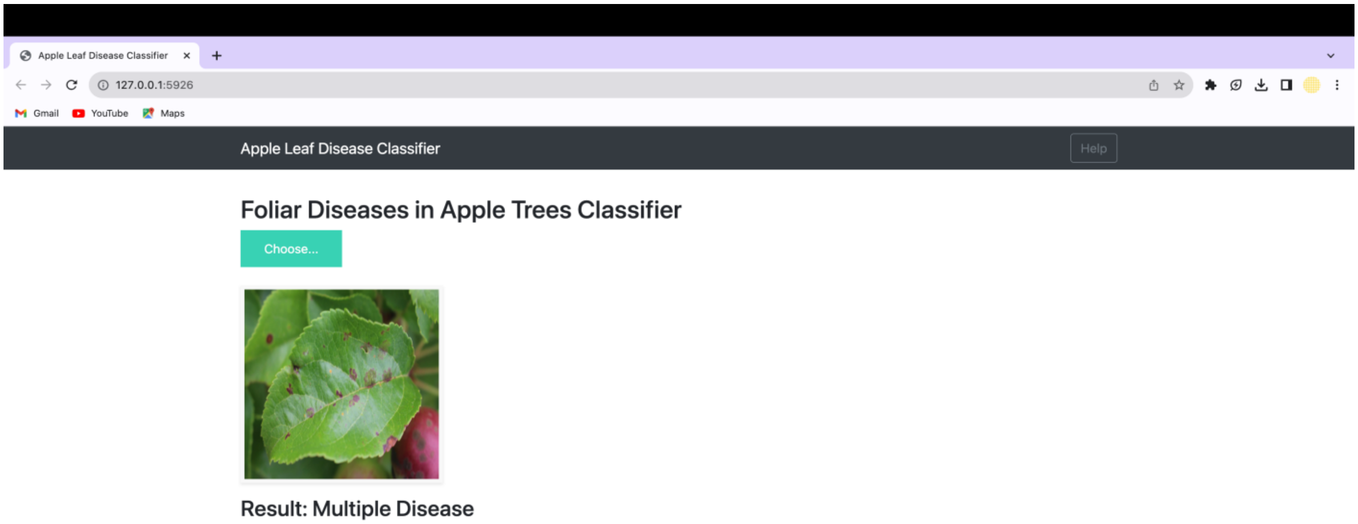


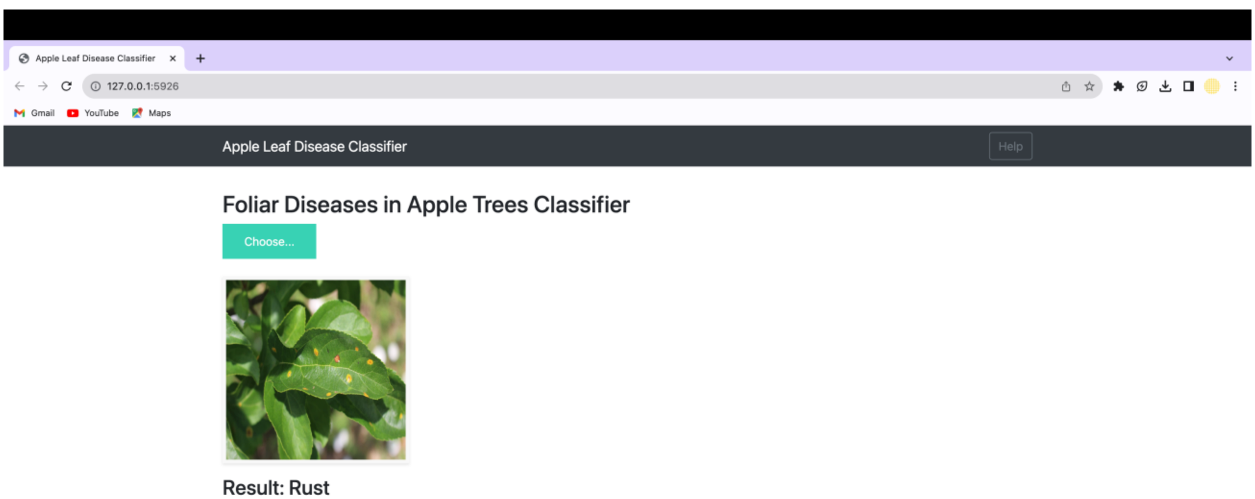

Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Rust

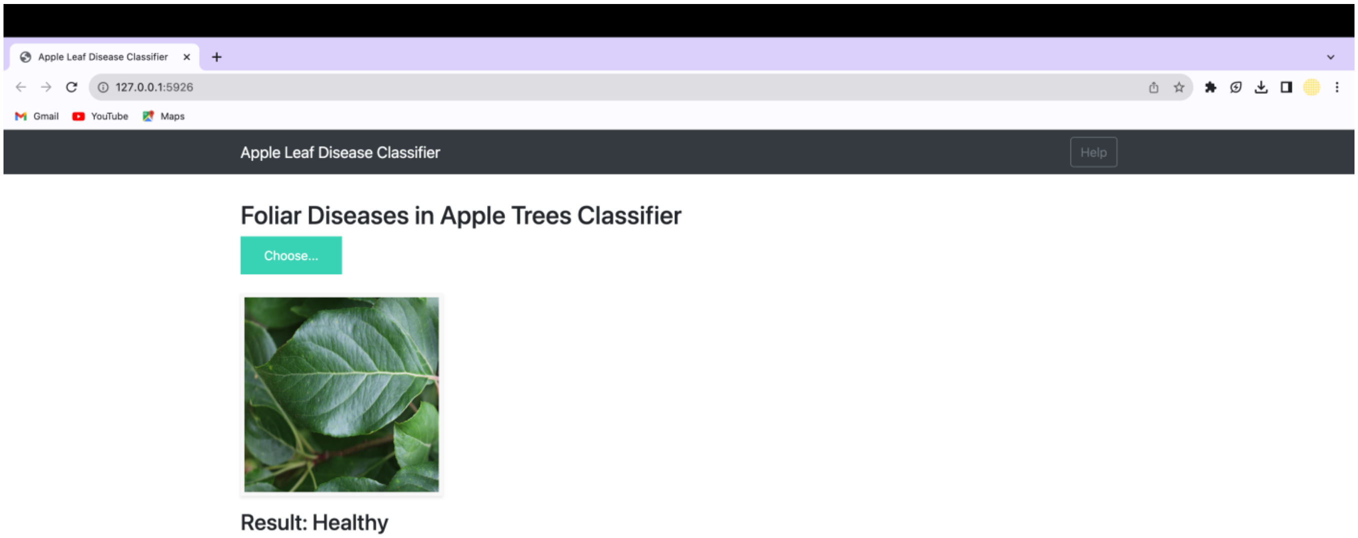

Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Healthy


Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Multiple Disease


Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Scab


Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Multiple Disease


Apple Leaf Disease Classifier
127.0.0.1:5926
Gmail
YouTube
Maps
Apple Leaf Disease Classifier
Help
Foliar Diseases in Apple Trees Classifier
Choose...
Result: Rust



3.4: Analysis and Discussion:

Results and Analysis

Our Convolutional Neural Network (CNN) model yielded promising results. During the training phase, we observed a steady decrease in the loss curve and a simultaneous increase in the accuracy curve. This indicated that our model was learning effectively from the training data. However, we noticed that beyond a certain number of epochs, the validation loss started to increase, hinting at overfitting. The implementation of early stopping criteria proved useful in preventing this.
A series of experiments were conducted to observe the change in performance when altering the model architecture and parameters. Similarly, tuning the batch size and learning rate during training had a noticeable impact on the model's performance.

A couple of foliar diseases had symptoms very similar to each other, making it challenging for the model to distinguish between them accurately in the early stages of disease development. We have discarded such diseases and classified diseases based on their distinct symptoms contributing to successful prediction of the diseases.

Discussions

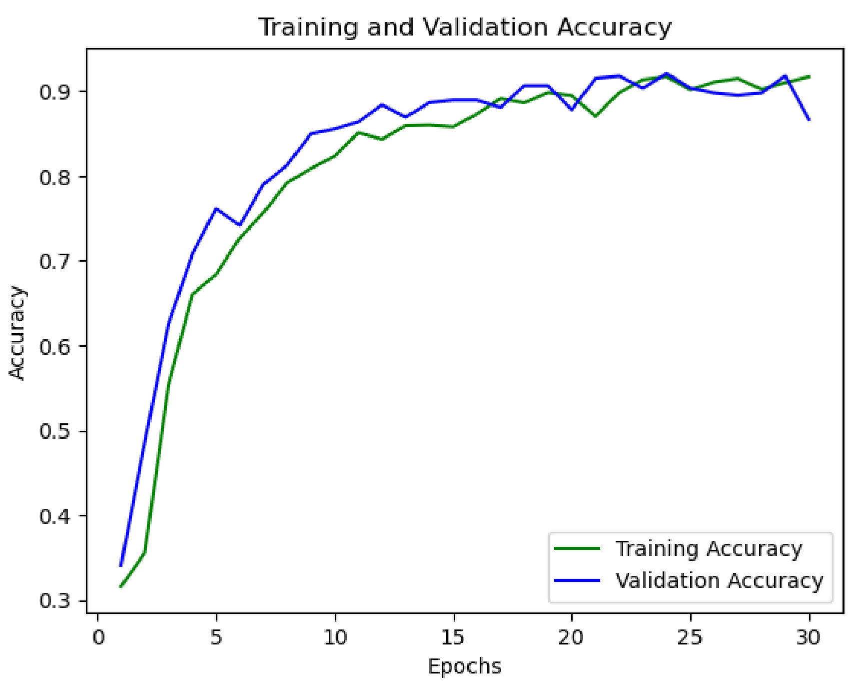
In terms of visualizations, the processed images and sample predictions provide interesting insights. The edge detection and other preprocessing techniques significantly improved the quality of the images for the CNN. Some sample predictions were spot-on, correctly identifying features in the images, while others were off-mark, indicating areas for improvement.

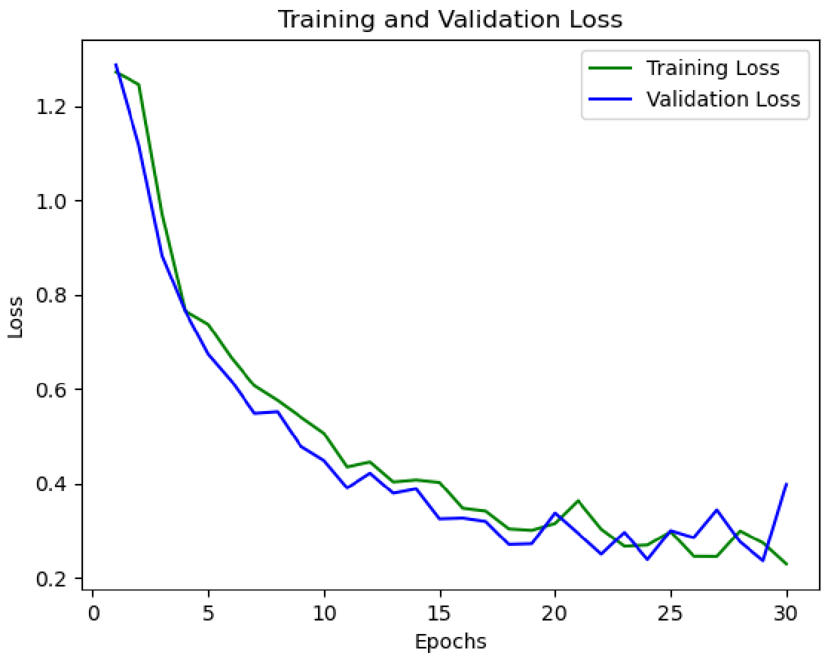
Considering failure cases, our model struggled with images that had high levels of noise or those where the features were not distinct enough due to lack of contrast or poor lighting. For instance, you could use an Autoencoder to reduce the dimensionality of your data and then feed the compressed data into a CNN for classification. This can be particularly useful when you have high-dimensional data. Alternatively, if your data is very noisy, you could use a denoising Autoencoder to clean up the noise before feeding the data into a CNN. These cases provide valuable information for future refinement of our preprocessing techniques.

Metrics

The confusion matrix was a crucial tool for evaluating our model. It revealed that our model had a high true positive rate for most classes but also a relatively high false-positive rate for some classes. This suggests that while our model was good at identifying correct features, it also misclassified a significant number of instances.

The model's performance was also influenced by the quality of the input images. High-resolution images with good contrast between the diseased areas and healthy regions led to better classification results, while images under poor lighting conditions or low resolution resulted in lower accuracy. In terms of practical applications, this study shows promising results for the early detection of apple tree diseases, which could significantly aid in prompt and effective disease management. Below graphs show training and testing accuracy and loss.





4. Conclusion:

From this project, we've learned a great deal about the practical application of autoencoders in image processing tasks. It was particularly interesting to see how the autoencoder could be used as a preprocessing step to denoise and standardize the images before feeding them into a separate convolutional neural network (CNN). The project demonstrated the significance of properly aligning the dimensions of input and output layers in a neural network. The ValueError encountered during the initial stages of the project was a prime example. It was a good reminder that the target output dimensions must match the actual output dimensions of the model.
Working with the Keras API and TensorFlow backend proved to be efficient and flexible. The ability to create custom generators for training data gives a lot of freedom in how the data is prepared and presented to the model. This project drove home the importance of understanding the data flow from preprocessing all the way to model training. It was also fascinating to explore the concept of unsupervised learning in the form of the autoencoder. Seeing how an autoencoder can learn to reconstruct inputs without any explicit target was a powerful demonstration of the capabilities of neural networks. In conclusion, this project has been a valuable learning experience. It has provided a deeper understanding of autoencoders, data preprocessing, and the Keras API. It has also underscored the importance of careful design and debugging in developing robust machine learning models. We look forward to applying these insights to future projects.

5. Contribution:

5.1 Code contribution:

In the scope of this project, I played a critical role in the data preparation and execution of advanced image processing techniques, including the implementation of the Canny Edge Detection algorithm. I initiated my tasks with the loading of train and test datasets, and subsequently, I enriched the train dataset by adding a 'label' column to capture the classification information. I then developed advanced image processing functions such as canny edge detection for edge detection, contrast enhancement, and eliminate noise from the images. Following this, I created distinct directories for training and testing data. I not only transferred the relevant images to these directories based on their labels but also applied the aforementioned image processing functions to each image. This process involved converting the images to grayscale, applying a Gaussian filter for noise reduction, and using the Canny function for edge detection. In the final stage, I performed image resizing to standardize the input for our model, ensuring that each image was of the size 224x224. I utilized matplotlib to visualize the impact of these transformations on a sample image. Throughout these processes, I worked in close alignment with the team, and my work served as the foundation for subsequent tasks in the project. My contributions in data preparation and image processing were pivotal in establishing a well-prepared dataset, unlocking the potential for sophisticated image analysis and driving us towards our project objectives.

5.2 Report contribution:

I contributed in outlining our approach to detecting and classifying apple leaf diseases using Convolutional Neural Networks (CNNs). In the "Experiments" section, I detailed our use of the plant pathology 2020 Kaggle dataset for effective model training and testing. In the "Evaluation Metrics" section, I explained key metrics like accuracy, precision along with the utility of a confusion matrix for in-depth analysis of model predictions. At the end I have displayed the respective results.

6.References:

[1] " Cotton Plant disease detection using Deep Learning " - Navina Pandhare, Vrunali Panchal, Shivam S. Mishra, Mrs. Darshna Tambe https://www.irjmets.com/uploadedfiles/paper/issue_4_april_2022/21202/final/fin_irjmets1650794528.pdf

[2] "Foliar Apple Tree Disease Classification"- Bingze Dai, Tian Qiu, Kai Ye http://noiselab.ucsd.edu/ECE228/projects/

[3] Data set - https://www.kaggle.com/c/plant-pathology-2020-fgvc7

[4] CNN - https://bdtechtalks.com/2020/01/06/convolutional-neural-networks-cnnconvnets/