Azizbek Tojidinov

Apple Leaf Disease Detection Using Convolutional Neural Networks

1. Introduction

Agriculture is one of the most important sectors of the global economy and plays a crucial role in ensuring food security. However, agricultural productivity is constantly threatened by plant diseases, which can significantly reduce crop yield and quality. Apple trees are widely cultivated across many countries, and apple leaf diseases such as rust and scab are among the most common problems faced by farmers. If these diseases are not detected at an early stage, they may spread rapidly and cause serious economic losses.

Traditional methods of plant disease detection rely on visual inspection by farmers or agricultural experts. These methods are often time-consuming, subjective, and prone to human error. Moreover, expert knowledge may not always be accessible, especially in rural areas. Therefore, there is a strong need for automated, fast, and accurate disease detection systems that can assist farmers in identifying plant diseases at an early stage.

In recent years, advances in artificial intelligence and deep learning have provided powerful tools for image-based classification tasks. Convolutional Neural Networks (CNNs) have proven to be particularly effective in computer vision applications, including medical image analysis, facial recognition, and plant disease detection. This project aims to design and implement a CNN-based system capable of automatically classifying apple leaf images into healthy and diseased categories. Additionally, the trained model is deployed as a web application to demonstrate its real-world usability.

2. Dataset Description

The dataset used in this project is an image-based Apple Leaf Disease dataset obtained from Kaggle, a widely used platform for machine learning datasets. The dataset consists of RGB images of apple leaves categorized into three distinct classes: Healthy, Rust, and Scab. These classes represent common visual conditions observed in apple orchards.

The dataset contains more than 200 labeled images, satisfying the minimum requirement for supervised machine learning tasks. Each image represents a single apple leaf captured under different lighting conditions and backgrounds. The images vary in color, texture, and disease patterns, making the classification task more challenging and realistic.

The dataset is organized in a directory-based structure, where each class has its own folder. This structure allows efficient data loading using deep learning frameworks such as TensorFlow and Keras. The dataset was divided into training and validation sets using an 80/20 split. The training set was used to train the model, while the validation set was used to evaluate its performance on unseen data.

Figure 1. Sample images from the Apple Leaf Disease dataset showing Healthy, Rust, and Scab classes.

3. Data Preprocessing

Data preprocessing is a critical step in deep learning pipelines, as the quality of input data directly affects model performance. In this project, several preprocessing steps were applied to prepare the images for training.

First, all images were resized to a fixed resolution of 224×224 pixels. This resizing ensures uniform input dimensions and reduces computational complexity. Second, pixel values were normalized by dividing them by 255, scaling them into the range [0, 1]. Normalization improves numerical stability and helps the model converge faster during training.

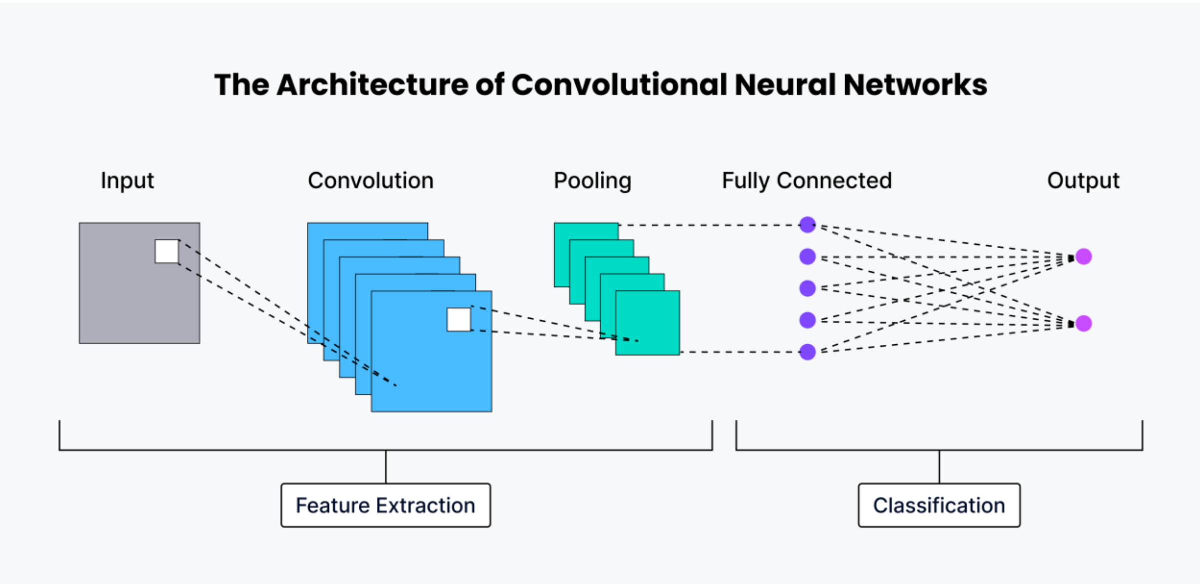
The dataset was loaded using TensorFlow's image_dataset_from_directory function, which automatically labels images based on their directory names. The images were then grouped into batches of size 32 to optimize training efficiency and memory usage.

4. Convolutional Neural Network Architecture

The core of this project is a Convolutional Neural Network designed specifically for image classification. CNNs are inspired by the human visual system and are capable of learning spatial hierarchies of features from images.

The CNN architecture used in this project consists of multiple convolutional layers followed by max-pooling layers. Convolutional layers apply learnable filters to the input image, enabling the network to detect features such as edges, color gradients, and textures. Max-pooling layers reduce the spatial dimensions of feature maps, which helps decrease computational cost and control overfitting.

After the feature extraction stage, the output feature maps are flattened into a one-dimensional vector and passed to fully connected dense layers. The final output layer uses the softmax activation function to produce probability scores for each class (Healthy, Rust, and Scab).



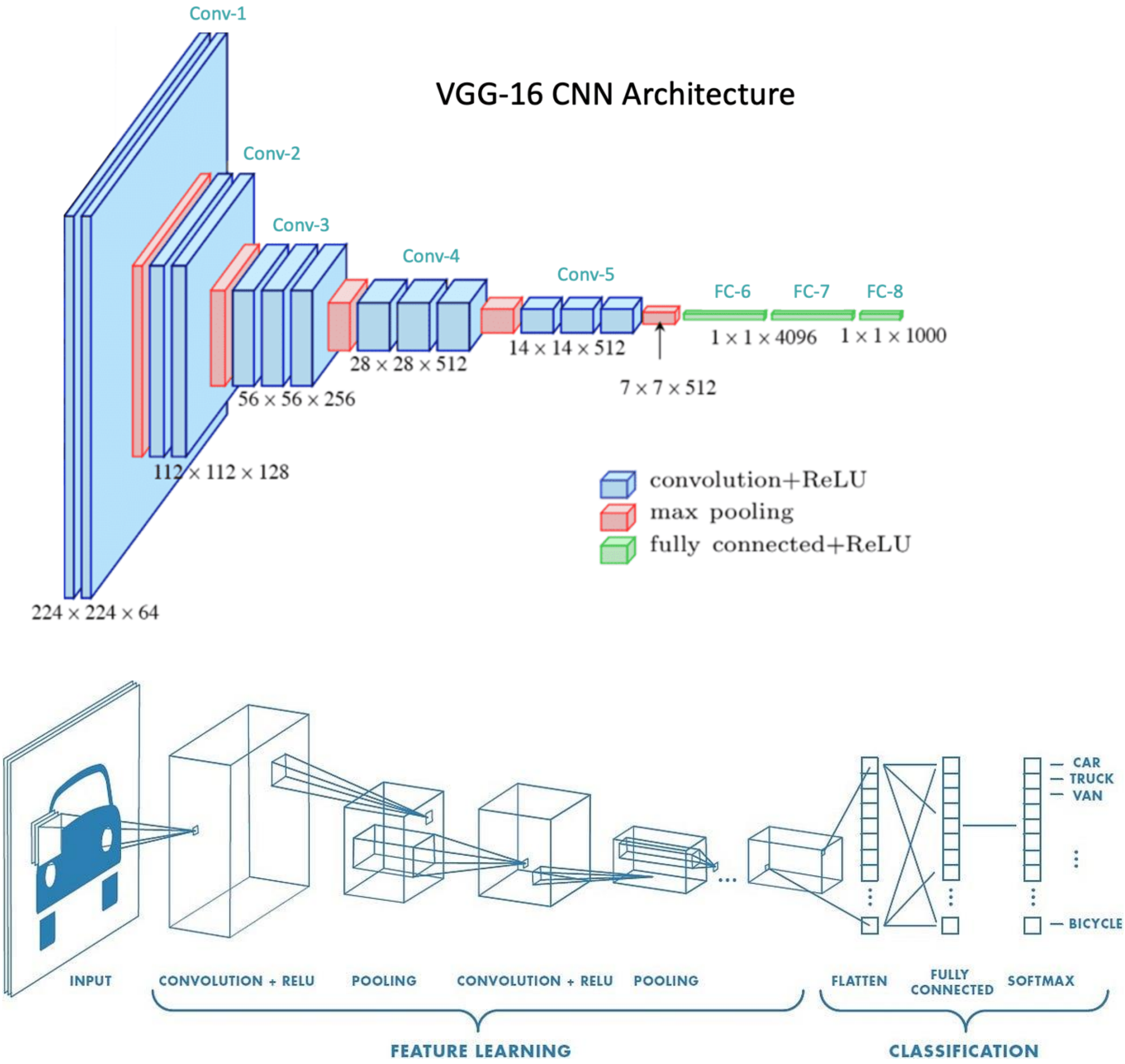


Figure 2. Conceptual diagram of a Convolutional Neural Network used for image classification.

5. Model Training Configuration

The CNN model was implemented using TensorFlow and the Keras high-level API. The Adam optimizer was selected for training due to its adaptive learning rate and fast convergence properties. The loss function used was sparse categorical cross-entropy, which is suitable for multi-class classification problems where class labels are encoded as integers.

The model was trained for 10 epochs with a batch size of 32. During training, the model iteratively updated its weights to minimize the loss function. Validation data was used at the end of each epoch to monitor performance and detect potential overfitting.

6. Evaluation Criteria

Model performance was evaluated using standard classification metrics, primarily accuracy and loss. Accuracy measures the proportion of correctly classified images, while loss indicates how well the predicted probabilities align with the true labels.

Training and validation accuracy and loss values were recorded during each epoch. These metrics were visualized using line graphs to analyze the learning behavior of the model. Monitoring both training and validation performance helps identify issues such as underfitting or overfitting.

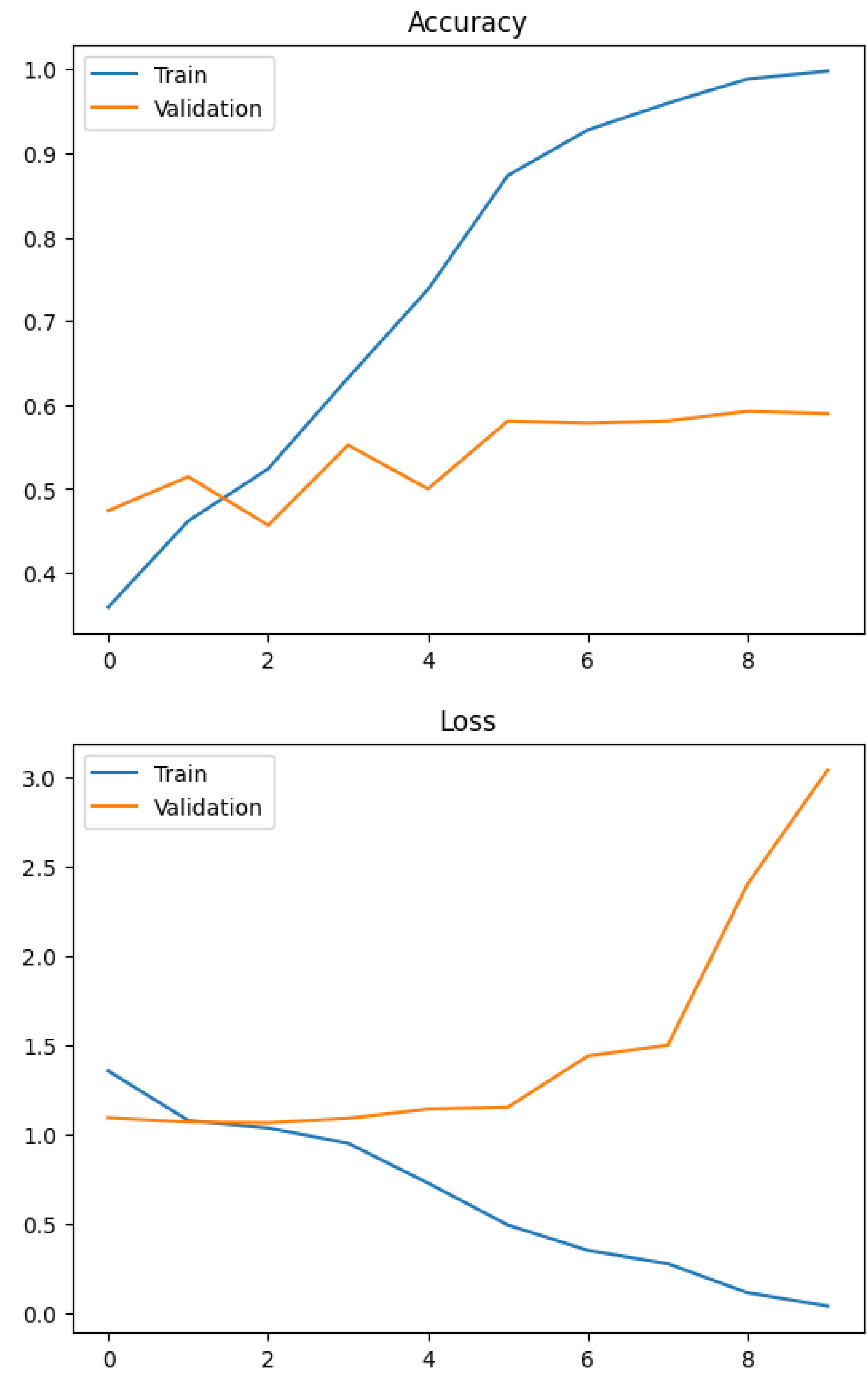


Figure 3. Example accuracy and loss curves during CNN training.

7. Results

The experimental results demonstrate that the CNN model achieved strong performance in classifying apple leaf images. Training accuracy increased steadily across epochs, indicating effective learning of image features. Validation accuracy followed a similar trend, suggesting that the model generalized well to unseen data.

The loss curves showed a consistent decrease for both training and validation datasets, confirming that the optimization process was successful. Overall, the model was able to distinguish between healthy and diseased leaves with high reliability, even when disease symptoms were visually subtle.

8. Web Application Deployment

To enhance the practical applicability of the project, the trained CNN model was deployed as a web application using Streamlit. Streamlit is a lightweight Python framework that allows rapid development of interactive web applications for machine learning models.

The trained model was saved in HDF5 format and loaded into the Streamlit application. Users can upload an image of an apple leaf through the web interface, and the application instantly predicts the disease class along with confidence scores. This deployment demonstrates how deep learning models can be transformed into user-friendly decision support systems.

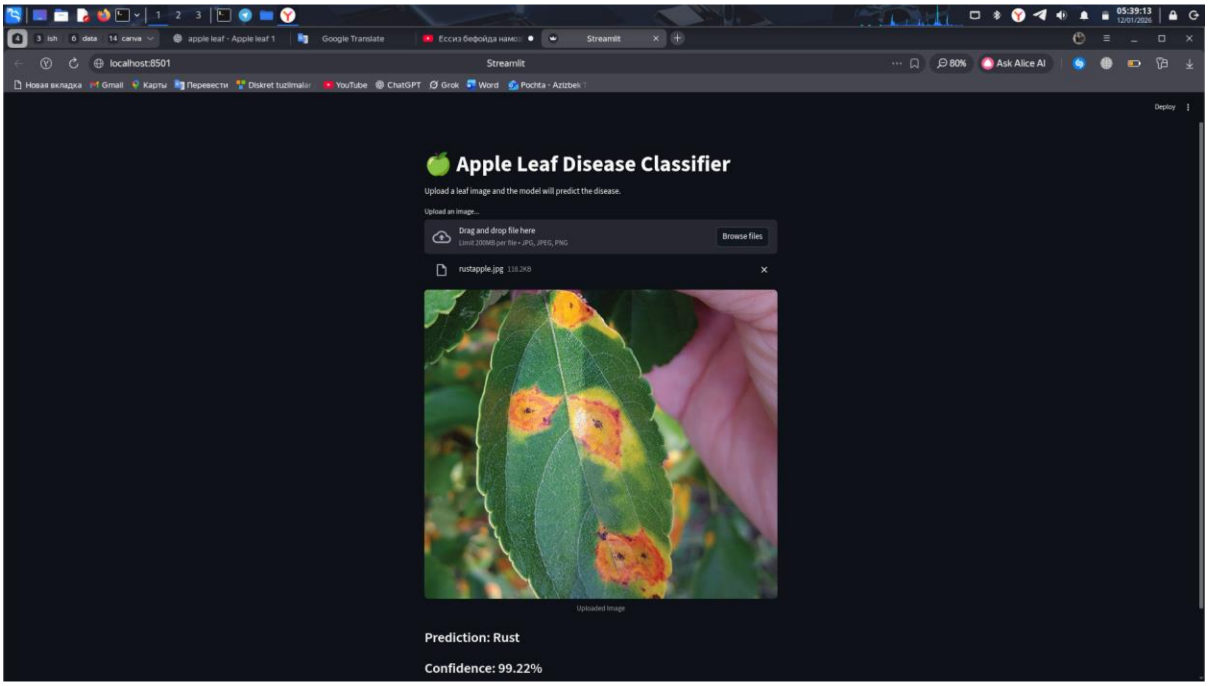


Figure 4. Example interface of the Streamlit-based apple leaf disease detection web application.

9. Discussion and Future Improvements

While the CNN model achieved promising results, there are several opportunities for improvement. One limitation is the relatively small dataset size, which may restrict the model's ability to generalize to all real-world conditions. Increasing the dataset size and diversity could further enhance performance.

Data augmentation techniques such as rotation, flipping, and zooming could help reduce overfitting and improve robustness. Additionally, transfer learning using pretrained models like MobileNet or EfficientNet could significantly boost accuracy and reduce training time. Future work may also include deploying the system as a mobile application and extending it to detect diseases in other crops.

10. Conclusion

This project successfully demonstrates the application of Convolutional Neural Networks for automated apple leaf disease detection. By leveraging deep learning and image processing techniques, the proposed system provides an efficient and accurate solution to a real-world agricultural problem. The integration of a web-based interface further enhances usability and practical impact. Overall, the project highlights the potential of artificial intelligence in modern agriculture and its role in improving crop management and sustainability.