Potato Leaf Disease Detection Using Machine Learning

Vanshika Varshney
Computer Science
Illinois Institute of Technology
vvarshney@hawk.iit.edu

Shikha Sharma
Computer Science
Illinois Institute of Technology
ssharma47@hawk.iit.edu

Abstract*— The dynamic nature of agricultural environments and the intricate patterns associated with leaf diseases present challenges in accurate and timely detection. Leveraging the wealth of available data and advanced machine learning algorithms, there is an opportunity to develop models capable of effectively identifying patterns associated with leaf diseases. This report emphasizes the significance of addressing plant health, particularly in the context of leaf diseases, and explores how machine learning models can be trained using historical data to forecast potential instances of such diseases. Despite the complexities involved in the agricultural domain, where disease outbreaks can vary widely, our machine learning models demonstrate the potential to offer reliable predictions for the early detection of leaf diseases.

I. INTRODUCTION

In recent years, advancements in machine learning technology have opened new possibilities for addressing critical challenges in agriculture. One such pressing issue is the early detection and management of plant diseases, which significantly impact crop yield and food security. The utilization of machine learning techniques for leaf disease detection has gained prominence due to its potential to revolutionize traditional methods. Unlike conventional approaches that rely heavily on manual observation and expertise, machine learning models can analyze vast datasets, identify subtle patterns, and provide accurate predictions. In this context, the application of machine learning for leaf disease detection is not only a technological leap but also a necessity for sustainable agriculture. With the increasing demand for food production to meet the needs of a growing global population, the ability to swiftly and accurately identify plant diseases can significantly enhance crop management practices.

II. PROBLEM STATEMENT:

A. *The problem*

Despite the critical importance of early disease detection in maintaining crop health and ensuring food security, traditional methods of identifying leaf diseases in agriculture are often labor-intensive, time-consuming, and reliant on subjective human expertise. The challenge lies in the need for a more efficient and accurate approach to detect leaf diseases promptly. This underscores the urgency of developing robust machine learning models capable of analyzing visual cues from plant leaves, recognizing disease patterns, and providing timely insights to farmers. The goal is to create a scalable and accessible solution that empowers agricultural practitioners with the tools needed to address leaf diseases effectively, thereby contributing to sustainable and resilient farming practices.

B. *Related Work*

Several studies and initiatives have explored the application of machine learning techniques for leaf disease detection in agriculture. The convergence of computer vision, image processing, and machine learning has led to promising developments in automating the identification and diagnosis of plant diseases. Here are some notable contributions in the related domain: Deep Learning Approaches using CNN, TransferLearning, Disease Classification Models, Remote Sensing IoT Integration, Explainable AI for Agriculture

III. PROPOSED MODEL

To address the challenges of leaf disease detection in agriculture, we propose an integrated machine learning framework that leverages the strengths of various components for robust and accurate identification. The proposed model comprises several key elements:

I. *Convolutional Neural Network (CNN)* that CNN serves as the foundational feature extractor.
II. *Data Augmentation* to enhance model generalization these techniques are applied during training.
III. *Adam Optimizer* combines the benefits of stochastic gradient descent.
IV. *Sparse Categorical Cross entropy* is a suitable loss function for classification problems where each input belongs to a single class.
V. *Early-Stopping* implements early stopping during the training of a neural network, preventing overfitting, and ensuring that the model generalizes well to new data.

The integration of these components creates a comprehensive and adaptable deep learning framework for leaf disease detection. Through rigorous training, validation, and fine-tuning processes, the model aims to provide accurate, timely, and interpretable predictions, empowering farmers with actionable insights for effective crop management.

IV. DATA PROCESSING AND VISUALIZATION

The code begins by loading the image dataset from the "PlantVillage" directory. It loads and organizes the data, ensures randomness through shuffling, standardizes image sizes, and prepares the data in batches for efficient model training. The extracted class names and batch information further contribute to the understanding and validation of the processed dataset. This prepared dataset is now ready for use in training a machine learning model for leaf disease detection.

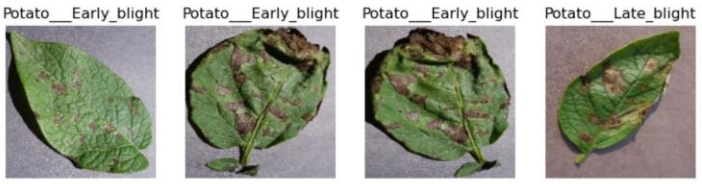
```
dataset_simple = tf.keras.preprocessing.image_dataset_from_directory(
    "PlantVillage",
    seed=123,
    shuffle=True,
    image_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
class_names = dataset_simple.class_names
class_names
for image_batch, labels_batch in dataset_simple.take(1):
    print(image_batch.shape)
    print(labels_batch.numpy())
```

It loads images from a directory ("PlantVillage) and organizes them into a dataset. Subdirectories within "PlantVillage" likely represent different classes, such as various plant diseases and a healthy state. The images in the dataset are resized to a consistent size (IMAGE_SIZE, IMAGE_SIZE) to ensure uniformity. The shape of the batch of images is printed, giving information about the dimensions of the images.

Data Visualization

Visualize some of the images from our dataset

```
plt.figure(figsize=(10, 10))
for image_batch, labels_batch in dataset_simple.take(1):
    for i in range(12):
        ax = plt.subplot(3, 4, i + 1)
        plt.imshow(image_batch[i].numpy().astype("uint8"))
        plt.title(class_names[labels_batch[i]])
        plt.axis("off")
```



V. MODEL TRAINING

The preprocessed data is then trained using different Machine Learning Models and the results and snippets are given below for each model used in the project.

1. **CNN coupled with SoftMax activation:** The methodology leaf disease detection using CNN and SoftMax activation involves the following steps:

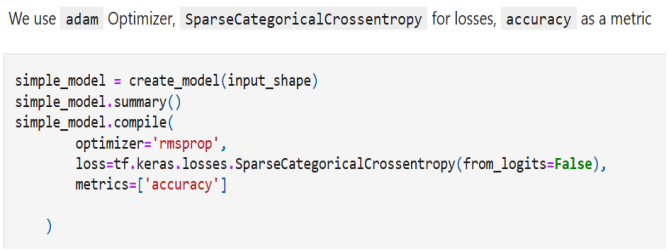
i. Design the architecture of CNN using TensorFlow's Keras API: Include convolutional layers, max-pooling layers, and dense layers. Use the SoftMax activation function in the output layer for multi-class classification. Compile the model with an appropriate optimizer (e.g., Adam), categorical cross entropy loss, and evaluation metric (e.g., accuracy). Apply data augmentation techniques to the training dataset to increase diversity and improve the model's generalization. Train the model specifying the number of epochs and validation data. Evaluate the trained model on the test set to assess its accuracy and performance. Use the trained model to make predictions on new data, such as unseen images of plant leaves. In this code, SoftMax activation is output layer for multiclass activation. It is used to transform raw output scores into probabilities. Cross-entropy is used with SoftMax for training classification models and measures the difference between predicted probability and true class labels.

ii. Adam Optimizer, Sparse Categorical cross entropy for losses, accuracy as a metric:

In the model compilation step, specify to use Adam optimizer. Set loss = 'sparse_categorical_crossentropy' when compiling the model, especially when dealing with integer labels.

Specify metrics=['accuracy'] during model compilation to monitor accuracy during training.

These steps are integrated into the model training pipeline and help guide the optimization process, measure the effectiveness of the model, and ensure appropriate handling of classification tasks with sparse labels.



iii. Data Augmentation:

The methodology leaf disease detection using Data Augmentation involves the following steps:

Configure a data augmentation generator using tools provided by the chosen machine learning framework. For example, in TensorFlow, you can use the Image Data Generator class. Use the configured data augmentation generator to load and augment the training dataset. Design a Convolutional Neural Network (CNN) architecture using the chosen machine learning framework. This architecture typically includes convolutional layers, max-pooling layers, and dense layers with appropriate activation functions. Compile the CNN model with an optimizer (e.g., Adam), a loss function (e.g., categorical cross entropy), and an evaluation metric (e.g., accuracy). Train the model using the augmented training dataset. Evaluate the model on the validation or test set to assess its performance. Once the model is trained and fine-tuned, use it to make predictions on new and unseen leaf images. By following these steps, data augmentation contributes to a more robust and generalized machine learning model for leaf disease detection.

iv. L2 Regularization:

The methodology leaf disease detection using Lasso Regression involves the following steps:

Assemble a labeled dataset of leaf images with classes representing different diseases and a healthy state. Split the dataset into training and testing sets. Import necessary libraries, including TensorFlow (or another deep learning framework) and NumPy for numerical operations. Set constants like image size, batch size, and the number of classes based on the characteristics of the dataset. Design the CNN architecture using TensorFlow's Keras API. Include convolutional layers, max-pooling layers, and dense layers with SoftMax activation for classification. Compile the model with an appropriate optimizer (e.g., Adam), loss function (e.g., categorical cross entropy), and evaluation metric (e.g., accuracy). Apply data augmentation techniques to the training dataset to increase diversity and improve the model's generalization. Train the model using the fit function, specifying the number of epochs and validation data. Evaluate the trained model on the test set to assess its accuracy and performance. Use the trained model to make predictions on new data, such as unseen images of plant leaves.

v. Stratified K-Fold for Cross Validation:

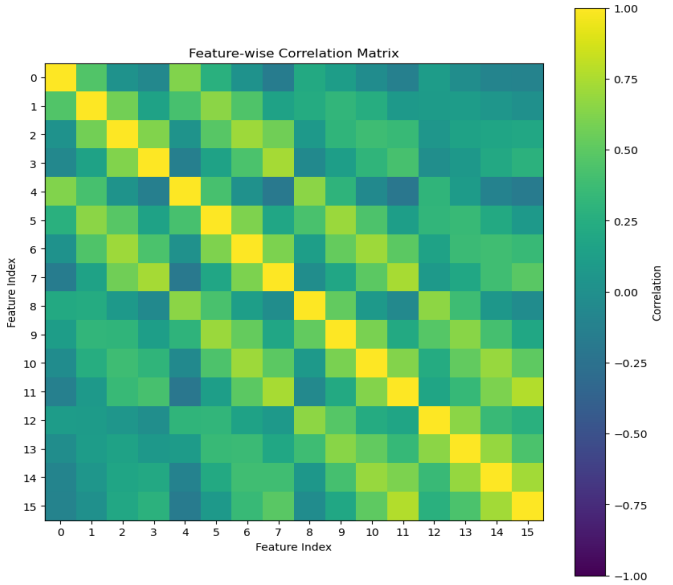
The methodology leaf disease detection using k-fold cross validation involves the following steps:

Load the dataset and split it into features and labels. Create an instance of Stratified KFold with the desired number of folds. Use the split method of StratifiedKFold to generate indices for training and validation sets for each fold. Train your machine learning model on the training set and evaluate its performance on the validation set for each fold. Aggregate and analyze the performance metrics obtained from each fold to assess the overall effectiveness of the model.

vi. Feature vise correlation matrix:

In leaf disease detection using machine learning, the Feature-Wise Correlation Matrix is a tool for analyzing the relationships between different features in the dataset. The methodology leaf disease detection using feature vise correlation matrix involves the following steps:

Calculate the Feature-Wise Correlation Matrix, which shows the correlation coefficients between each pair of features. Examine the correlation matrix to identify relationships between features.

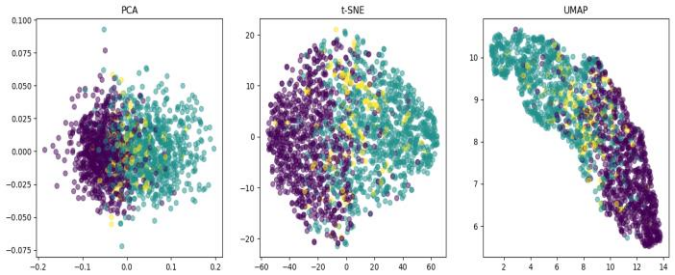


In this matrix, diagonal line of matrix represents correlation of each feature with itself which is always 1 that is why it is of same color. In this the light-yellow color indicates weak positive correlation i.e., weak tendency to move in same dimension. The dark yellow indicates strong positive correlation i.e., as one variable increases, other also increases. The dark green indicates negative correlation i.e., as one variable increases, another decreases. The light cells indicate weak correlation i.e., cells have weak tendency to move in opposite direction.

vii. Dimensionality Reduction Techniques:

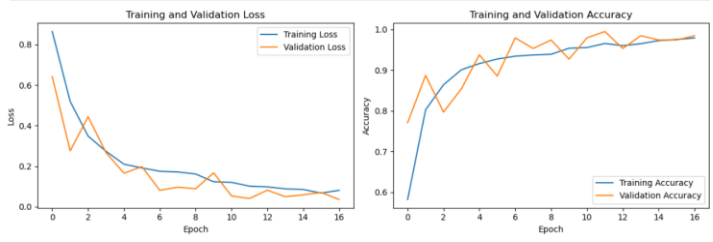
Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), and Uniform Manifold Approximation and Projection (UMAP) are employed for dimensionality reduction and visualization. The methodology leaf disease detection using dimensionality reduction techniques involves the following steps:

Extract relevant features from leaf images, resulting in a high-dimensional feature space. Apply PCA, t-SNE, or UMAP to reduce the feature space while preserving essential information. Visualize the reduced-dimensional space to explore patterns, clusters, and relationships among leaf images. Use the reduced-dimensional feature representations for training machine learning models, potentially improving model efficiency and performance.



In PCA, the purple dots show that data is clustered after being projected into first 2 principal components. In t-SNE, green dots show that data is clustered after being embedded into high-dimensional space using t-SNE. In UMAP, yellow dots show how data is clustered after being embedded into 2d space using UMAP. Points with same color represent same class/category. Separation indicates how well the dimension reduction technique distinguishes the different classes or groups. In the above graph, PCA graph is most spread out, meaning data is high-dmensional.t-SNE and UMAP are more compact, meaning they have captured underlying structure of data better than PCA.

The accuracy achieved by the model is **94.5%**, showcasing its effectiveness in making correct predictions."




```
scores = model_instance.evaluate(test_ds)
print(f"Test Loss and Accuracy with modified model: {scores}")
```

```
8/8 [==============================] - 4s 191ms/step - loss: 0.1529 - accuracy: 0.9453
Test Loss and Accuracy with modified model: [0.15292838215827942, 0.9453125]
```

Below are some results for predictions by our model:

Conclusion:

The proposed model not only streamlines the detection process but also contributes to the overall sustainability and efficiency of crop management practices. The comprehensive data processing and visualization pipeline ensures that the model is trained on a diverse and representative dataset, fostering its ability to generalize to real-world scenarios. As agriculture continues to face the growing demands of a global population, the proposed model stands as a technological cornerstone, offering a scalable and efficient solution for timely disease detection. The continuous learning mechanism ensures the model's adaptability to evolving disease patterns, reinforcing its relevance in dynamic agricultural landscapes. In essence, the strides made in the development of this deep learning framework not only contribute to the advancement of precision agriculture but also hold the potential to significantly impact global food security.

Future Work:

While the proposed machine learning framework for leaf disease detection represents a significant advancement, there are several avenues for future research and improvements to further enhance its efficacy and applicability in agricultural settings: Multi-Species Extension, Dynamic Disease Patterns, Real-Time Monitoring, Transfer Learning for New Diseases User-Friendly Interfaces, Edge Computing Optimization, Ensemble Learning Enhancements Privacy and Ethical Considerations Collaboration with Agricultural Communities Climate-Responsive Adaptation. By exploring these avenues for future work, researchers and practitioners can contribute to the ongoing evolution of machine learning applications in agriculture, with a focus on improving accuracy, scalability, and real-world impact.

Project GitHub Repository:

The training dataset, testing dataset, code repository and results of the project can be found on the below GitHub repository – https://github.com/vvanshika09/Leaf_disease_detection/blob/main/Leaf_disease_detection_simple_model.ipynb