Tomato Blight Detector

Shahjalal University of Science and Technology

Raisa Fairooz Md Adith Mollah Md Sirazul Islam
Md Ashraful Islam Siddique Badhon

February 9, 2023

Abstract

Farmers growing tomatoes , face economic losses every year due to various diseases that occur to tomato plant.Early blight and late blight are two common diseases caused by respectively fungus and micro-organism.if these diseases can be detected early and be applied appropriate treatment then it can save a lot of waste and prevent economic loss eventually contributing to the agricultural domain to a great extent.Image detection and recognition using Convolutional Neural Network (CNN)[1] through mobile devices can be a solution in this regrard.

To detect whether a tomato plant leaf is healthy or defected with late blight or early blight a mobile application is being developed where users can upload selected picture and get results with confidence rate. Behind the scene, the system uses a deep learning CNN[1] model to identify classes.The neural network model being trained , validated and tested over 4490 images eventually providing test data score of 95% accuracy.

Contents

List of Figures

List of Tables

Introduction

Tomato late blight and early blight are two common diseases that can significantly impact on production and economic sector. These diseases are caused by fungi that infect the foliage and fruit of tomato plants,leading to visible symptoms such as leaf yellowing, defoliation, and fruit rot. Effective detection and management of these diseases is crucial for the success of tomato production.

One potential approach for detecting tomato blights is the use of image recognition techniques, specifically Convolutional Neural Networks (CNNs) [1]. CNNs are a type of artificial neural network that are particularly well-suited for image analysis tasks, such as identifying patterns and features in images. By training a CNN on a large dataset of images of defected and healthy tomato plants, it is possible to develop a model that can accurately identify the presence of late blight and early blight based on visual characteristics. This approach has the potential to greatly improve the efficiency and accuracy of disease detection, compared to traditional methods that rely on visual inspection by trained personnel.

Here , over 4490 frequency of images taken from kaggle which were [3] being used for training , validating and testing purpose.80% of the whole dataset being used for training and the rest 20% being utilized for validating and testing purpose. All the images resized to 256 by 256 dimension. after fitting or training about 50 epoch an accuracy of 95% received.Users receive result by uploading pictures on the application which get the outcome from backend server connected with the trained model itself.

Overall, image recognition using CNN[1] for detecting tomato late blight and early blight is a promising approach that could enhance the ability to effectively manage these diseases and protect tomato crops.

Problem Definition

The problem of tomato late blight and early blight is a significant one that affects tomato production world wide. Late blight is caused by the fungus Phytophthora infestans which is a highly destructive disease that can spread rapidly and cause significant losses in tomato production. Leaf symptoms of late blight first appear as small, water-soaked areas that rapidly enlarge to form purple-brown, oily-appearing blotches. On the lower side of leaves, rings of grayish white mycelium and spore-forming structures may appear around the blotches.

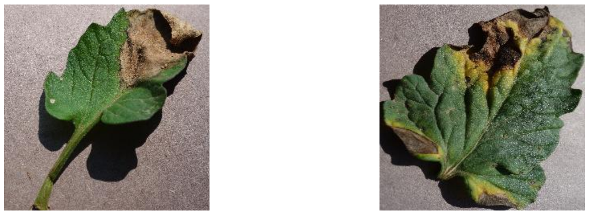

(a) Late Blight (b) Early Blight

Figure 1: Tomato late blight and early blight leaves

Early blight , caused by the fungus Alternaria solani , is also a serious threat to tomato crops , particularly in warm and humid environments.Tomato plants infected with early blight develop small black or brown spots, usually about 0.25 to 0.5 inch (6–12 mm) in diameter, on leaves, stems, and fruit. Leaf spots are leathery and often have a concentric ring pattern which usually appear on older leaves first.

Current methods for detecting and managing these diseases include visual inspection by trained personnel, laboratory testing, and the use of chemical pesticides. However, these methods can be time-consuming, costly, and potentially harmful to the environment. In this regard, an image recognition technology can play a vital role to prevent previous methods demerits.

Background

Deep learning is a subfield of machine learning which involves the use of artificial neural networks with composing multiple layers to learn complex patterns in data.Nowadays these algorithms are used for image recognition tasks.Deep learning algorithms are trained using large datasets and powerful computing resources, and are able to automatically extract useful features from raw data.

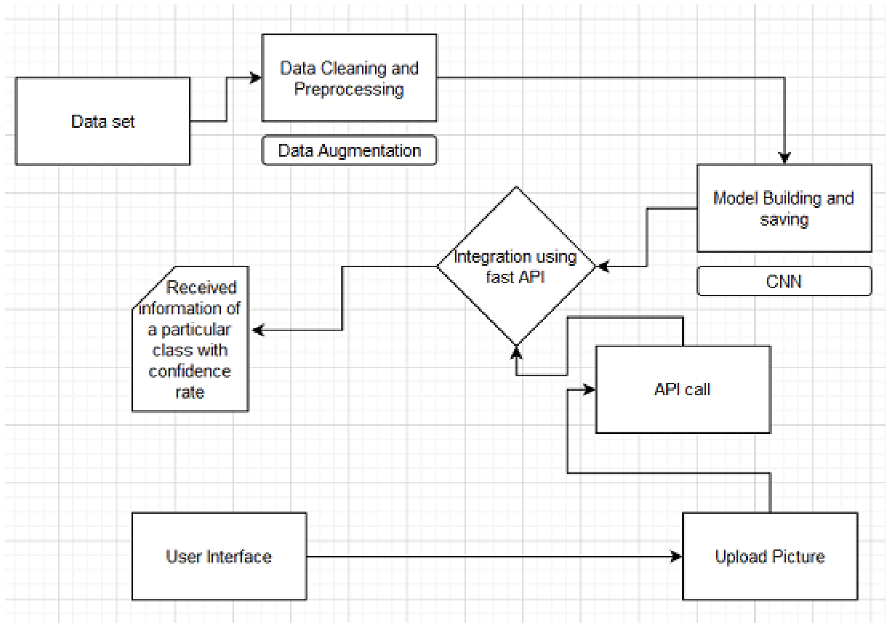
A Convolutional Neural Network (CNN)[1] is a type of artificial neural network that is specifically designed for image analysis tasks. It is composed of multiple layers of interconnected neurons that apply mathematical operations known as convolutions to process and analyze images. CNNs have been successful in tasks such as image classification, object detection, and segmentation, and have achieved state-of-the-art results in many such tasks.

Flutter[4] is an open-source software development kit (SDK) used for developing applications for mobile, web, and desktop platforms. It is developed by Google and is based on the Dart programming language. It provides a fast and intuitive way to build user interfaces, with a wide range of customizable widgets and flexible layouts.It has gained popularity due to its ease of use and ability to create natively compiled applications for cross platforms from a single codebase.

FastAPI[9] is an open-source, modern, fast, and high-performance web framework for building APIs with Python 3.6+ based on standard Python type hints. It is built on top of Starlette, a lightweight ASGI framework, and utilizes the latest Python async i/o capabilities to provide fast and reliable request handling. FastAPI includes a variety of features that make it easy to develop and deploy APIs, including dependency injection,unlimited plugins, automatic validation of request data, automatic generation of OpenAPI documentation, and support for web sockets. Its focus on performance and ease of use has made FastAPI a popular choice for building scalable and reliable APIs.

Methodology

Workflow



Data set
Data Cleaning and Preprocessing
Data Augmentation
Model Building and saving
CNN
Integration using fast API
Received information of a particular class with confidence rate
API call
User Interface
Upload Picture

Figure 2: System workflow

The above figure depicts the workflow of the whole system.Here fast API[9] is the main integral point of the back-end and front-end part.Front-end connects through API call where back-end connects through model injection with fast API.

Dataset Information and Preprocessing

From kaggle[3][10] 3 classes of images being collected. Frequency of 996 , 1588 and 1906 images respectively as early blight , healthy and late blight.All the images being resized to 256 by 256 dimension for training , validation and testing purpose.A total of 141 batches being created where each batch consist of 32 images along with 3 channels which are red , green and blue.80% of the whole dataset used for training and the rest 20% is divided between validation and testing operation.

Data Augmentation

Specific augmentation techniques being applied to the images before training.Images being randomly flipped to horizontal and vertical sides along with adjusting color contrast and random rotations.Main purpose of augmentation is to increase the diversity of the whole dataset.

Backend and Frontend

For back-end fastAPI[9] is used to integrate the model with the front-end through API call.Jupyter notebook [2] being used as an environment for developing the model.Flutter [4] is used for user interface showcasing various features built with widgets[5] such as uploading pictures.The selected pictures data moves to the model through API call which figures out exact class with confidence rate.The received information is displayed to users in a fancy text field form.

Model Building

For model building[8] a deep learning[**?**][7] Convolutional Neural Network (CNN) is being created which is a standard way for performing image classification.The architecture consisting of 2 types of activation functions such as rectified linear unit and softmax.Before moving to the dense layer part , a series of convolution with rectified linear unit and max pooling operations being applied for feature extraction.

The first layer having 32 neurons and the next 5 layers each having 64 neurons performing convolution with activation function as rectified linear unit.Each of these 6 layers max pooling operation is injected.Layers then get flattened before creating 64 neurons dense layer eventually leading to the final layer of 3 neurons for 3 classes providing probabilistic result which is normalized due to the activation function softmax.

Results and Analysis

Accuracy and Validation Scores

Metric	Score
Accuracy on trained data	98 %
Validation data accuracy	96 %
Accuracy on test data	95 %

Table 1: Performance evaluation metrics

Over 50 times the learning algorithm worked through the entire training dataset.Each time the model was calculating losses and accuracy of training and validation dataset.Finally , after 50 feed forward and back propagation , accuracy from trained and test data is respectively 98% and 95% .In case of validation accuracy , a score of 96% is being received.

Analysis on training and validation dataset

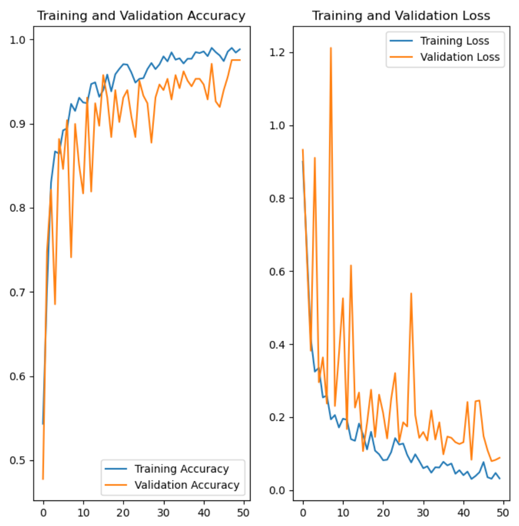


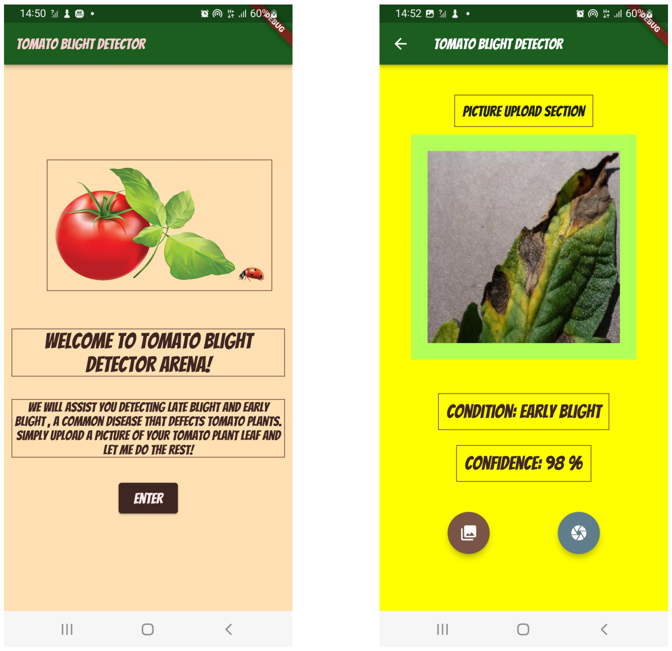
Figure 3: Analysis of training and validation

Two of the above figures represent training and validation datasets loss and accuracy metric with respect to epochs from 1 upto 50.Here epoch represents the learning algorithms working through the entire training dataset.

A bit fluctuating pattern noticed on both loss and accuracy part.At the beginning of the working phase , validation and training scores are detached from each other.Around 9 epoch the difference between validation accuracy and training accuracy is around 17% which justifies the detaching phase.Training and validation loss figure showing similar behaviour for the beginning epoch phases.

But as the frequency of epoch increases the difference between validation and training part decreases along with better accuracy and lower loss.

Frontend Results



WELCOME TO TOMATO BLIGHT DETECTOR ARENA!

WE WILL ASSIST YOU DETECTING LATE BLIGHT AND EARLY BLIGHT , A COMMON DISEASE THAT DEFECTS TOMATO PLANTS. SIMPLY UPLOAD A PICTURE OF YOUR TOMATO PLANT LEAF AND LET ME DO THE REST!

PICTURE UPLOAD SECTION

CONDITION: EARLY BLIGHT

CONFIDENCE: 98 %

(a) Initial page (b) Executed page

Figure 4: User-end experience

The first figure depicts about the initial page after opening the application which consist of some text widgets,image field.

The second figure successfully classified tomato early blight leaf with a confidence rate of 98%.Through the assistance of fast API[9] the information is being received from the model.

Conclusion

Tomato blight detector[6] application proposes a solution to detect whether a tomato leaf is healthy or it is defected with late blight or early blight.In this regard , a Convolutional Neural Network (CNN)[1] architecture is being used for image recognition task.To train the model a combined dataset of images consisting of 3 classes is used.Several preprocessing and augmentation techniques has been applied to the images.Specific frontend features being applied for users to navigate the application conveniently. Promising results received after the training process.

This approach has the potential to accurately identify the presence of these diseases at an early stage.Overall , this application has the potential to enhance the economic efficiency and environmental sustainability of production and protect against the devastating impacts of late blight and early blight.

References

[1] Rahul Chauhan, Kamal Kumar Ghanshala, and R.C Joshi. Convolutional neural network (cnn) for image detection and recognition. In *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pages 278–282, 2018.

[2] Brian Granger Fernando Pérez. Jupyter notebook. https://jupyter.org/, 2014.

[3] Anthony Goldbloom. Kaggle. https://www.kaggle.com/, 2010.

[4] Google. Flutter. https://flutter.dev/, May 2017.

[5] Google. Widget catalog. https://docs.flutter.dev/development/ui/widgets, May 2017.

[6] Raisa Fairooz Md Sirazul Islam Siddique Badhon Md Adith Mollah, Md Ashraful Islam Shanto. Tomato blight detector. https://github.com/Adith082/Tomato_Blight_Detector, 2022.

[7] Dhaval Patel. Codebasics. https://www.youtube.com/@codebasics, 2015.

[8] Dhaval Patel. Model building. https://youtu.be/ZN6P_GEJ7lk, 2021.

[9] Sebastián Ramírez. Fast api. https://fastapi.tiangolo.com/, 2018.

[10] Arjun Tejaswi. Dataset of various plants. https://www.kaggle.com/datasets/arjuntejaswi/plant-village, 2019.