https://doi.org/10.46335/IJIES.2024.9.6.13 e-ISSN: 2456-3463
Vol. 9, No. 6, 2024, PP. 55-61

International Journal of Innovations in Engineering and Science, www.ijies.net

Utilizing Deep Learning for The Detection and Tracking of Plant Diseases

Prof. Himanshu V. Taiwade[1], Rakesh Nagrikar[2], Diksha Narekar[3], Prashik Nagrale[4]

[1]Assistant Professor, [2,3,4] Student Department of Computer Science and Engineering, Priyadarshini College of Engineering, Nagpur, India

***Received on**: 5 May,2024* ***Revised on**: 29 June,2024* ***Published on**: 01 July ,2024*

Abstract: *This paper introduces the plant disease detection system that utilizes Deep Learning (DL) methodologies to tackle the obstacles experienced by farmers. The frontend infrastructure of the system uses React JS to create user-friendly web interfaces, while the backend is configured with TensorFlow Serving and FastAPI for effective model deployment and communication. The main part of the model is a TensorFlow-implemented Convolutional Neural based Network, which is enhanced for better generalization using data augmentation methods. In addition to providing important answers for agricultural and food security measures, this methodical approach seeks to revolutionize the identification of potato diseases. This holds potential to transform crop disease detection, ultimately enhancing food security and agricultural productivity.*

Keywords: CNN**,** DenseNet- 121, Tensor-flow, Machine Learning, Deep Learning, Early bright, Late blight.

I. INTRODUCTION

Agriculture serves as a cornerstone of sustenance and economic prosperity, particularly in countries like India, where approximately 18% of the national income hinges on agricultural activities. The cultivation of crops, such as potatoes, plays a pivotal role in fulfilling food requirements and bolstering economic stability [6]. However, this vital sector faces significant threats from crop diseases, which pose a formidable risk to agricultural productivity and livelihoods. The repercussions of these diseases extend beyond mere yield reduction, encompassing diminished crop quality and financial losses, thereby jeopardizing the nation's food security and economic well-being [2][4].

In the context of potato cultivation, farmers encounter daunting challenges in effectively combating crop diseases. Conventional methods reliant on visual inspection by experts are not only time-consuming and costly but also prone to inaccuracies. With the emergence of algorithms in deep learning, particularly Convolutional Neural based Networks, there has been a paradigm shift in disease identification methodologies [2]. These sophisticated models leverage complex image processing techniques to accurately diagnose plant infections based on distinctive patterns or spots observed on leaves. Moreover, the application of transfer learning, wherein pre-trained models like DenseNet-121 are fine-tuned for specific tasks, contributes to the increased efficacy and precision of disease detection systems.

By make the use of power of deep learning and transfer learning techniques, farmers can now access nonintrusive, real-time, and cost-effective solutions for early disease identification in potato crops [11]. The adoption of such technology promises to empower farmers with timely information, enabling them to implement necessary interventions swiftly and optimize crop protection measures.

II. RELATED WORK

The suggested expert method, according to Kalita, Hemanta, Shikhar Kr Sarma, and Ridip Dev Choudhury [3], is intended to help in the diagnosis of common illnesses that affect rice fields. Users engage with the system by selecting "YES" or "NO" in response to questions about different disease symptoms. Typically, a farmer with a crop disease issue inputs their problem through the user interface. The system then compares these responses to its knowledge base, which is represented as a set of rules, to identify the likely disease affecting the crop.

Himanshu V. Taiwade, Ketan D. Bodhe, Virendra P. Yadav, Nikesh V. Aote [8] the research project focuses on detecting and diagnosing diseases affecting cotton leaves. It includes using template matching techniques to create a prototype Android mobile application. The Vidharbha region's various cotton farms provide the photographs required for this application. Farmers may easily detect diseases in cotton fields by using smartphones with internet access. The suggested approach can help accomplish early disease pattern recognition so that timely intervention can be implemented.

Mr. Ramachandra Hebbar, Shima Ramesh, P. V. Vinod, Neveditha M, Pooja R, Prasad Bhat N, and Shashank N etc.[5] The Random Forest approach is utilized in this research to differentiate between healthy and unhealthy leaves. The implementation process involves multiple stages, such as dataset creation, feature extraction, classifier training, and image classification. A Random Forest classifier is trained using the datasets that include both healthy and diseased leaf images. The trained classifier is then utilized to distinguish between the two types of images. The Histogram of Oriented based Gradients approach is used to extract image features.

The suggested system, according to S. Ponni Alias Sathya a, S. Ramakrishnan a, M.I. Shafreen a, R. Harshini a, and P. Malini [8], is made to recognize plant leaf illnesses by the processing of leaf photos. The process involves image acquisition, followed by preprocessing steps like quality assessment, smoothing, and shape analysis to pinpoint the affected areas. For component segmentation, genetic algorithms are utilized, and for image classification, the Support Vector Machine, also known as the SVM, Classifier. In addition to precisely locating and displaying the plant disease's source, the system can recognize healthy leaf regions and provide an accuracy evaluation through the use of a fuzzy approach.

Sk Mahmudul Hassan, Khwairakpam Amitab, etc. [10] This suggested solution offers a thorough summary of the current state of research on plant disease diagnosis, encompassing both cutting-edge deep-learning-based methods and conventional handmade feature methods. It draws attention to the difficulties with handmade feature techniques as well as the positive aspects of deep learning. Even if deep learning models with greater feature extraction capabilities, such as GoogleNet and InceptionV3, may have performance problems when used with diverse or real-world datasets. The review also identifies ongoing challenges in achieving effective plant disease identification.

Kaushika , Ishita Sharmad , Isha Jindale , Vaishali Deshwalf [11] The goal of the research is to use deep learning for picture recognition to both detect plant diseases and minimize financial losses. It explores three neural network architectures: Faster R-CNN, R-CNN (SSD), and single-shot multi-box detector. The proposed technique efficiently handles diverse scenarios and achieves a promising 94.6% accuracy in validation, indicating the potential of convolutional neural networks for complex problem-solving with AIbased deep learning.

J., A.; Eunice, J.; Popescu, D.E.; Chowdary, M.K.; Hemanth, this paper explores the use of pre-trained convolutional neural based network models, including VGG-16, DenseNet-121 and Inception V4, for efficient and accurate plant disease recognization. Among which DenseNet-121, has achieved maximum training accuracy of 99.87%.

M. A. Khan, T. Akram, M. Sharif, M. Awais, K. Javed, H. Ali, and T. Saba, [9] In particular, this study uses architectures such as AlexNet and VGG16 to offer a revolutionary visualization method that blends correlation coefficients with models based on deep learning. To detect grape diseases, Kerkech et al. used the LeNet model in conjunction with vegetation indices in color space. A number of techniques, including gradient time input, boot back-propagation, depth Taylor decomposition, integration gradient layered related transmission, and significant figure, were examined in a distinct article in order to analyze deep learning models. Several biological and nonbiological soybean stressors were used to train DenseNet121 to identify them. He studies discovered that interpretability techniques were useful in emphasizing infected leaf regions as critical characteristics in accurately categorized photos.

III. POTATO LEAF DISEASES

Potato leaf diseases encloses a wide range of fungal, bacterial, and viral infections that can affects with impactful plant health and its production. The most well-known of these, Phytophthora infestans, is the source of the late blight disease; nevertheless, it is impossible to predict how severely it may harm a population. This fungus aggressively grows well in cool, moist conditions, quickly spreads across fields and affects potato crops in a large count. And another is a hallmark symptom include dark, water-soaked lesions on leaves, stems, and tubers, readily gets together by a white, fuzzy fungal growth under humid conditions. Late blight played a historic role in the Irish Potato Famine and remains a significant threat to potato production globally.

https://doi.org/10.46335/IJIES.2024.9.6.13 e-ISSN: 2456-3463
Vol. 9, No. 6, 2024, PP. 55-61

Another common disease is early blight, caused by Alternaria solani. This disease is specified as small, dark lesions with concentric rings, early blight disease can destroy the leaves of plants and reduce tuber quality. Blackleg, caused by bacteria like Pectobacterium and Dickeya species, leads to blackened, necrotic lesions at the base of stems, eventually causing plant collapse readily. Leaf roll viruses, such as Potato leafroll virus (PLRV), induce leaves to curl downward and thicken, impacting both production and tuber quality.

Powdery mildew, caused by various Erysiphales fungi, manifests as white powdery patches on leaves, hindering photosynthesis and weakening plants. Early dying, often initiates causing by soilborne pathogens like Verticillium and Rhizoctonia, triggers premature yellowing and death of lower leaves, results in a decrease of yield potential.

Effective management strategies involve a combination of cultural practices, including crop rotation, sanitation, and the use of resistant cultivars. Additionally, timely application of fungicides and bactericides, along with monitoring for early symptoms, can help mitigate the impact of these diseases on potato crops. Vigilance and proactive management are essential to safeguarding potato plants against these pervasive leaf diseases.

IV. SYSTEM OVERVIEW

The system architecture for the automated plant disease detection system comprises a backend having a lot of experience with the trained data models and frontend infrastructure with the basic features and services that are necessary for the detection of the diseases. The backend side of the system uses TensorFlow Serving for model deployment and FastAPI for effective communication with the client side and server side. The frontend side of the system uses React JS for the web interface useful for the user. The construction of the model involves the developing of a Convolutional Neural based Network which is a deep learning algorithm using TensorFlow, with data augmentation techniques for improved generalization of pre-trained datasets. The user interface provides an understandable design to have proper accessibility and ensures seamless communication with the backend. This methodical technique provides a solid solution that transforms the diagnosis of potato diseases and significantly increases the efficacy of agricultural and food solutions and security measures.

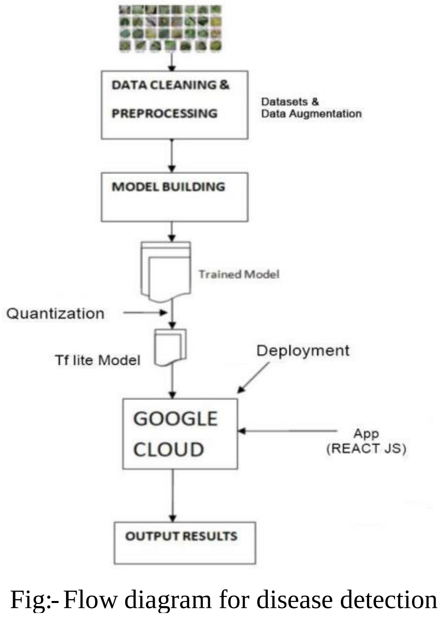


Fig:- Flow diagram for disease detection

V. IMPLEMENTATION

The system is consisting of following modules:

I. Data Collection and Preprocessing Module: This module involves gathering a diverse dataset of plant images depicting various diseases and healthy conditions. These images can be sourced from publicly available datasets, agricultural research institutions, or collected through field surveys. Preprocessing steps include resizing images to a consistent resolution, converting them to a suitable format (e.g., JPEG), and labeling them according to disease type or healthy status. Additionally, data augmentation techniques such as rotation, flipping, and scaling can be applied to augment the dataset and improve model generalization.

II. Data Training and Model Building Module: In this module, the preprocessed dataset is divided into various parts like:

Data Preprocessing: To have the raw dataset available for training, preprocessing is done. Image resizing, normalization, data augmentation (if required), and dividing the dataset into training, validation, and testing sets are some examples of the tasks that may be included in this step.

Model Architecture Selection: In this step, a CNN architecture is chosen. In your case, DenseNet-121 is selected. DenseNet is a popular CNN architecture known for its efficient use of parameters and strong performance on various image classification tasks.

TensorFlow Setup: TensorFlow, a popular deep learning framework, is set up for building and training the CNN model.

Transfer Learning with Pre-trained Weights: Transfer learning is employed using pre-trained weights from ImageNet. With this method, performance on the target task can be enhanced with comparatively fewer datasets by utilizing the knowledge acquired from training on a large dataset (ImageNet).

Model Training: Using the training dataset, the convolutional neural network model is trained. Hyperparameters like learning rate, batch size, and optimizer selection are adjusted during training to maximize model performance. The training process involves forward and backward passes through the network, adjusting weights using techniques like backpropagation.

Validation Set Evaluation: After each training epoch or a certain number of iterations, the model's efficacy is evaluated on the validation set. In order to evaluate the model's performance and pinpoint areas in need of development, metrics such as accuracy, precision, recall, and F1 score are calculated.

Iterative Training: Iterative model training is carried out until the validation set shows acceptable performance. This may involve adjusting hyperparameters, changing the architecture, or incorporating regularization techniques to prevent overfitting.

Testing: Once the model get trained and validated, it is evaluated on the testing set to assess its generalization performance on unseen data This stage makes sure the model functions properly in practical situations.

III. Backend Server Implementation Module: This module involves setting up a backend server using

TensorFlow Serving Setup: TensorFlow Serving is set up to deploy your trained model. TensorFlow Serving is a system for serving machine learning models, especially TensorFlow models, in production environments. It allows for efficient and scalable deployment of models for inference.

FastAPI Integration: FastAPI is integrated to develop API endpoints for communication between the frontend and backendFastAPI is a cutting-edge, quick web framework that uses standard Python type hints to develop APIs with Python 3.7+. It's particularly suitable for this task due to its performance and ease of use.

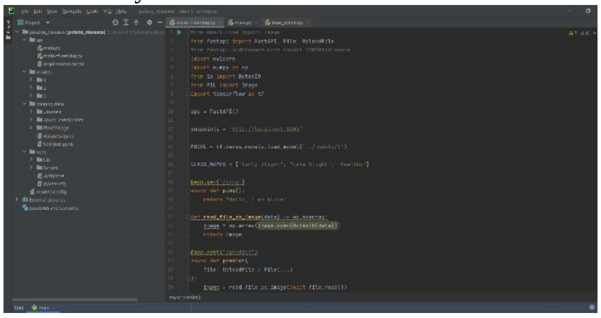
Google Cloud Platform (GCP) Hosting: The backend server is hosted on the Google Cloud Platform (GCP). GCP offers scalability, reliability, and accessibility, making it an ideal choice for hosting production services. You can leverage services like Compute Engine, Kubernetes Engine, or Cloud Run to deploy and manage your backend server.

API Endpoint Implementation: Using FastAPI, API endpoints are implemented to handle requests from the frontend. These endpoints receive input data (e.g., images) from the frontend, perform inference using the deployed model, and return the results back to the user interface.

Scalability and Reliability: The backend server architecture is developed to be productive and reliable. This involves considerations such as load balancing, auto-scaling, monitoring, and logging to ensure that the service can handle varying levels of traffic and maintain high availability.

Security Considerations: Security measures such as authentication, authorization, and encryption may be implemented to protect the backend server and the data it handles.

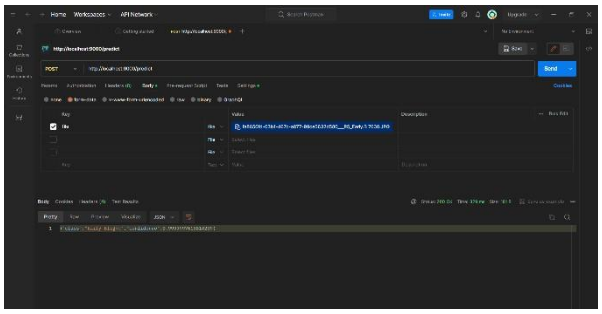
Testing and Monitoring: The backend server is tested to make ensure its functionality and performance. Additionally, monitoring tools are set up to track metrics such as response time, error rate, and resource utilization to identify and address any issues in real-time.



```
from email.mime import image
from fastapi import FastAPI, File, UploadFile
from fastapi.middleware.cors import CORSMiddleware
import uvicorn
import numpy as np
from io import BytesIO
from PIL import Image
import tensorflow as tf

app = FastAPI()

endpoints = "http://localhost:8501"

MODEL = tf.keras.models.load_model("../models/1")

CLASS_NAMES = ["Early Blight", "Late Blight", "Healthy"]

@app.get("/ping")
async def ping():
    return "Hello, I am alive"

def read_file_as_image(data) -> np.ndarray:
    image = np.array(Image.open(BytesIO(data)))
    return image

@app.post("/predict")
async def predict(
    file: UploadFile = File(...)
):
    image = read_file_as_image(await file.read())
```

Fig: - Backend Server with tf-serving

https://doi.org/10.46335/IJIES.2024.9.6.13 e-ISSN: 2456-3463
Vol. 9, No. 6, 2024, PP. 55-61



```
POST  http://localhost:9000/predict
file  File  [illegible]_RS_Early.B 7030.JPG

1  {"class":"Early Blight","confidence":0.99[illegible]}
```

Fig: - Result after integration of FastAPI

IV. Model Optimization with Quantization Module: The Module aims in order to improve the effectiveness of the trained model, making it suitable for deployment on resource-constrained devices like smartphones.

Quantization Techniques Selection: Quantization techniques are chosen based on the requirements and constraints of the deployment environment. In this case, post-training quantization is selected as it offers a balance between model size reduction and preservation of performance.

Post-Training Quantization: Post-training quantization is a technique that preserves accuracy by transforming the weights of the trained model to lower precision forms, like 8-bit integers. This loss of precision contributes to a reduction in the computational complexity and memory footprint of the model, which makes it more appropriate for deployment on devices with constrained resources.

Model Evaluation: After quantization, the optimized model undergoes thorough evaluation to ensure that its performance remains satisfactory for the target application. To ensure that there was no appreciable loss in model quality due to the quantization procedure, metrics including accuracy, precision, recall, and F1 score are evaluated.

Efficiency Metrics Assessment: The optimized model is evaluated against efficiency metrics relevant to real-time inference on resourceconstrained devices. These metrics may include inference speed, memory footprint, and energy consumption. Making sure the quantized model satisfies the efficiency criteria for a smooth deployment on smartphones or other edge devices is the main objective.

Fine-Tuning (Optional): In some cases, fine-tuning or further optimization techniques may be applied to the quantized model to improve its performance or efficiency. This step involves iterative experimentation and refinement to achieve the desired balance between model size, inference speed, and accuracy.

Validation on Target Devices: Once the optimized model passes evaluation and efficiency metrics assessment, it is validated on target devices such as smartphones to confirm its compatibility and performance in real-world scenarios.

V. Frontend Development and Deployment to GCP: The Frontend Development and Deployment to GCP module focuses on creating user-friendly interfaces for both web and mobile platforms and deploying them to the Google Cloud Platform (GCP).

Frontend Development with React JS and React Native: User interfaces for both web and mobile platforms are developed using React JS for web applications and React Native for mobile applications. React JS and React Native are popular JavaScript libraries for building interactive and responsive user interfaces.

User Interface Design: Users can upload photographs of plant leaves and get real-time disease diagnosis findings with ease thanks to the user-friendly and intuitive interfaces. Principles of user interface (UI) and user experience (UX) design are used to produce visually appealing and functional interfaces.

Integration with Backend: The frontend applications are integrated with the backend server to enable communication for uploading images and receiving disease detection results. API endpoints provided by the backend server are utilized to facilitate this communication.

Deployment to GCP: Once the frontend applications are developed and tested locally, they are deployed to the Google Cloud Platform (GCP) for hosting. Services like Google App Engine or Firebase Hosting are leveraged for deploying web and mobile applications respectively.

Continuous Integration and Deployment (CI/CD): Continuous integration and deployment pipelines are set up to automate the deployment process. This involves configuring CI/CD tools such as Google Cloud Build or Jenkins to automatically build, test, and deploy frontend applications whenever changes are pushed to the repository. This ensures seamless updates and scalability of the frontend applications.

VI. ADVANTAGES

1. Deep Learning offers a more accurate and precise method for diagnosing crop diseases compared to traditional approaches.
2. The CNN approach enables early detection of crop disease symptoms, potentially preventing widespread outbreaks and minimizing yield losses.
3. By automating the disease detection process, farmers can save time and resources, leading to increased productivity and profitability.
4. The proposed system, accessible via smartphones, makes disease identification more convenient for farmers, even in remote areas with limited resources.
5. The deep learning framework allows for the identification of both symptomatic and presymptomatic cases, providing comprehensive coverage of crop diseases.
6. Evaluation of the model's performance across various metrics ensures its efficacy and reliability in disease detection, providing farmers with confidence in the system.
7. Integration of DenseNet-121 into the framework achieves impressive classification accuracy, surpassing traditional methods and ensuring more reliable results.
8. Crop disease detection could be revolutionized by this method, improving global agricultural output and food security.

9. By addressing gaps in reliable disease detection methods, the proposed CNN approach contributes to closing the existing loopholes in plant disease management, leading to more sustainable agricultural practices.

VII. CONCLUSION

An important development in agricultural technology is the use of Deep Learning, namely Convolutional Neural based Networks, in the diagnosis of plant diseases. The use of CNN to crop disease detection, particularly in potato plants, is the focus of this work, which has great potential for the agricultural economy.

Potato, being one of the staple crops in many countries, plays a crucial role in ensuring food security and economic stability. By effectively detecting and managing diseases in potato plants, farmers can mitigate yield losses and safeguard the economic benefits associated with potato cultivation.

The proposed CNN approach demonstrates several advantages, including enhanced precision, early disease detection, improved efficiency, and accessible technology via smartphones. Moreover, the integration of DenseNet-121 into the framework showcases superior classification accuracy, providing reliable results for disease identification.

The diseases that can be effectively detected using this approach include but are not limited to: 1. Early Blight (Alternaria solani)
2. Late Blight (Phytophthora infestans)

By effectively detecting these diseases, farmers can implement timely interventions, such as targeted pesticide applications or crop rotations, to manage and control disease outbreaks, ultimately safeguarding potato yields.

REFERENCES

[1] *Suresh, V., Krishnan, M. S., Hemavarthini, M., Jayanthan, K. S., & Gopinath, D. (2020). Plant Disease Detection using Image Processing. International Journal of Engineering Research and Technology, V9(03). https://doi.org/10.17577/ijertv9is030114*

[2] *Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM,60(6),84–90. https://doi.org/10.1145/3065386*

[3] *Bashish, D. A., Braik, M., & Bani-Ahmad, S. (2010). A framework for detection and classification of plant leaf and stem diseases. IEEE. https://doi.org/10.1109/icsip.2010.5697452*

[4] *Jiang, P., Chen, Y., Liu, B., He, D., & Liang, C. (2019). Real-Time detection of Apple leaf diseases using deep learning approach based on improved convolutional neural networks. IEEE Access, 7, 59069–59080. https://doi.org/10.1109/access.2019.2914929*

[5] *Maniyath, S. R., Vinod, P., Niveditha, M. K., Pooja, R. C., N, P. B., Shashank, N., &Hebbar, (2018). Plant Disease Detection Using Machine Learning. IEEE. https://doi.org/10.1109/icdi3c.2018.00017*

[6] *Khan, M. A., Akram, T., Sharif, M., Awais, M., Javed, K., Ali, H., & Saba, T. (2018). CCDF:*

Automatic system for segmentation and recognition of fruit crops diseases based on correlation coefficient and deep CNN features. Computers and Electronics in Agriculture, 155,220–236. https://doi.org/10.1016/j.compag.2018.10.013

[7] *Recent Technology and Engineering, 9(1), 909–914. Suresh, V., Krishnan, M. S., Hemavarthini, M., Jayanthan, K. S., & Gopinath, D. (2020b). Plant Disease Detection using Image Processing. International Journal of Engineering Research and Technology, V9(03).https://doi.org/10.17577/ijertv9is0301 14*

[8] *Bodhe, K. D., Taiwade, H. V., Yadav, V. P., &Aote, N. (2018). Implementation of Prototype for Detection & Diagnosis of Cotton Leaf Diseases using Rule Based System for Farmers. 2018 3rd International Conference on Communication and Electronics Systems (ICCES). https://doi.org/10.1109/cesys.2018.8723931*

[9] *Kaushika, N., Reddy, K. S., & Kaushik, K. (2016). Sustainable Energy and the Environment: a Clean Technology approach. In Springer eBooks. https://doi.org/10.1007/978-3-319-29446-9*

[10] *Vasavi, P., Punitha, A., & Rao, T. V. N. (2023). Chili crop disease prediction using machine learning algorithms. Revue D'intelligenceArtificielle, 37(3), 727–732. https://doi.org/10.18280/ria.370321*

[11] *International Chohan, M., Khan, A., Chohan, R., Katpar, S. H., & Mahar, M. S. (2020). Plant Disease Detection using Deep Learning. Journal of https://doi.org/10.35940/ijrte.a2139.059120*