International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 05 | May - 2023 SJIF 2023: 8.176 ISSN: 2582-3930

Pomegranate Fruit Disease Detection using Deep Learning

Mukund Nemane[1], Digvijay Gaikwad[2], Saurabh Sathe[3], , Prof. Priya Deshpande[4]

Dept. of Electronics and Telecommunication Engineering
PVG's College of Engineering and Technology GKPIOM Pune, Maharashtra, India

---

1)

Bacterial blight is a plant disease caused by various species of bacteria that affect a wide range of crops, including vegetables, fruits, and ornamental plants. It is characterized by the presence of dark, water-soaked lesions on the leaves, stems, and sometimes fruits or flowers of affected plants. The bacteria responsible for bacterial blight usually enter the plant through natural openings or wounds, such as leaf stomata or injuries caused by insects, wind, or other factors. Once inside the plant, they multiply and spread, causing damage to the plant tissues.

Fig -1: Bacterial Blight

2) Cercospora:

Cercospora fruit spot is a common fungal disease that affects pomegranate trees. It is caused by the fungus Cercospora punicae. This disease primarily affects the fruit, causing dark, circular spots to develop on the surface. The symptoms of Cercospora fruit spot on pomegranate include the formation of small, reddish-brown lesions on the fruit. As the disease progresses, the lesions enlarge and become sunken. The affected areas may also develop a whitish-gray or pinkish fungal growth. In severe cases, the spots can merge, leading to fruit cracking or rotting.

Fig -2: Cercospora

3) Thrips:

Pomegranate thrips (Eurythrips sp.) primarily act as pests that feed on pomegranate trees (Punica granatum) and cause direct damage to the leaves, flowers, and fruits. However, they are not known to transmit specific diseases to pomegranate trees.While pomegranate thrips themselves do not cause diseases, their feeding activity can weaken the tree, making it more susceptible to other pathogens or stress-related issues. For example, the wounds created by thrips feeding can provide entry points for secondary fungal or bacterial infections. These infections may lead to diseases such as fungal leaf spots, fruit rots, or bacterial blights. In such cases, it is the secondary pathogens that cause the diseases, not the thrips themselves.

Fig -3: Thrips

4.METHODOLOGY

1)Data Requirements :

Data is in the form of images. We have to collect the images of the diseases occurs on the pomegranate fruit. Image is in the form of png , jpeg.

2) Data Collection:

Data pre-processor is the process in which we collect the images of different diseases occurs on the pomegranate fruit.

3) Data Augemetation:

Data augmentation is a technique commonly used in deep learning to artificially increase the size and diversity of a training dataset by applying various transformations or modifications to the existing data. It helps to improve model performance and generalization by exposing the model to a wider range of variations and reducing overfitting.

4) Data Annotation:

Data annotation is the process of labeling or tagging data with relevant and meaningful information to make it usable for training machine learning models. In the context of deep learning, data annotation involves annotating the training dataset with annotations or labels that represent specific features or characteristics of the data. Data annotation can be performed manually by human annotators who carefully analyze the data and assign appropriate labels or annotations. In some cases, automated or semi-automated annotation tools may be used to speed up the annotation process.

5) Data Pre-processing:

Data preprocessing is an essential step in preparing the data before feeding it into a deep learning model. It involves transforming and organizing the data in a way that facilitates effective model training and improves overall performance. Various data preprocessing techniques help create a clean, standardized, and suitable dataset for training deep learning models. Each preprocessing step is applied to ensure the data is in a format that the model can effectively learn from, leading to improved model performance and accurate predictions.

6) Model Training:

Model training in deep learning involves optimizing the model's parameters using a labeled training dataset. The steps include data preparation, selecting a model architecture, initializing parameters, defining a loss function, choosing an optimization algorithm, iterating over the training data in mini-batches, updating parameters using backpropagation, validating the model's performance, repeating the process for a fixed number of epochs, and evaluating the final model on a separate test dataset. Fine-tuning and hyperparameter tuning can be performed to improve the model's performance.

7) Model Testing:

Model testing in deep learning is the process of evaluating the performance and generalization ability of a trained model using a separate test dataset . Model testing provides an unbiased assessment of the trained model's performance on unseen data, helping to measure its effectiveness and reliability. It allows researchers and practitioners to understand the model's limitations, make informed decisions, and iterate on the training process to improve the model's performance.

8) Model Review:

Model review in deep learning refers to the process of critically examining and assessing the performance, strengths, and weaknesses of a trained model. Model review is a crucial step to ensure the reliability,

effectiveness, and fairness of deep learning models. It helps identify areas for improvement, validate the model's performance, and make informed decisions for further iterations or deployment.

9) Model Deployment:

Model deployment in deep learning refers to the process of making a trained model available and operational for use in real-world applications. Model deployment involves transitioning a trained deep learning model from the development environment to a production-ready state. It requires careful considerations of infrastructure, performance, security, and ongoing maintenance to ensure the model's successful integration and usability in real-world applications.

5.MASK IMAGE AND ITS GENERATION

Masking using U-Net is a popular approach for image segmentation and mask generation tasks. U-Net is a convolutional neural network architecture that consists of an encoder-decoder structure with skip connections, which helps to preserve spatial information during up sampling. Here's a step-by-step overview of how you can use U-Net for mask generation:

1.Dataset Preparation:
Gather a dataset that includes input images and corresponding masks. The masks should represent the regions of interest or objects you want to generate masks for.

2. Data Pre-processing:
Pre-process the input images and masks as necessary. Common pre-processing steps include resizing, normalizing pixel values, and converting them to a suitable format for deep learning frameworks.

3. Architecture Design:
Define the U-Net architecture for mask generation. The U-Net consists of an encoder, which down samples the input image and learns high-level features, and a decoder, which up samples the learned features to generate the final mask. The encoder and decoder are connected by skip connections that concatenate feature maps of the same spatial size.

4. Training:
Split your dataset into training and validation sets. Train the U-Net model using the training set by minimizing a suitable loss function, such as binary cross-entropy, between the predicted masks and the ground truth masks. Use an optimization algorithm like stochastic gradient descent (SGD) or Adam to update the network weights during training.

5. Prediction:
After training, you can use the trained U-Net model to generate masks for new input images. Pass the input image through the trained model, and the decoder part of the network will generate the mask.

4. Post-processing: Depending on the specific task, you may need to apply post-processing techniques to refine the generated masks. This could involve thresholding the mask to obtain binary values, applying morphological operations like erosion or dilation, or using other domain-specific techniques to improve the quality of the masks. It's worth noting that U-Net can be modified and customized based on the specific requirements of your task. You can adjust the architecture, add regularization techniques, or incorporate additional components as needed for implementing U-Net for mask generation typically involves using deep learning frameworks such as TensorFlow, PyTorch, or Keras, which provide pre-defined layers and utilities to facilitate the development and training process.

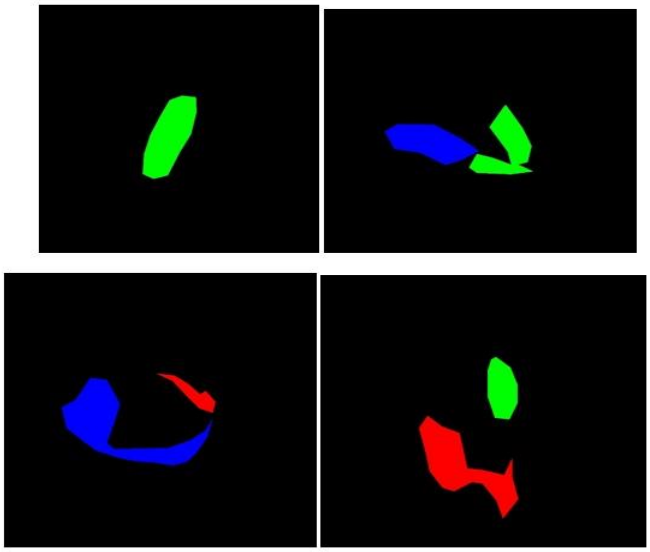

Fig -4: Mask Images

6.ARCHITECTURE DIAGRAM

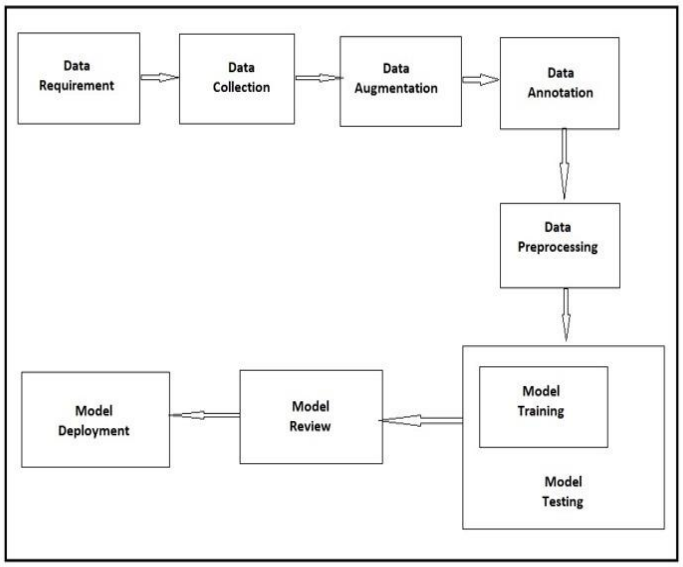

Fig -5: Architecture

7. ALGORITHM

1.UNET Algorithm:

The U-Net architecture gets its name from its U-shaped network structure. It consists of an encoder path and a decoder path. The encoder path is responsible for capturing the context and extracting high-level features from the input image, while the decoder path up samples the features to generate a segmentation map of the same size as the input image.

Here's a high-level overview of the U-Net architecture:

1. Encoder Path: The encoder path follows a typical convolutional neural network (CNN) architecture. It consists of a series of convolutional layers followed by non-linear activation functions (such as ReLU) and max-pooling layers. Each convolutional layer increases the number of feature channels while reducing the spatial dimensions of the input.

2. Bridge: The last layer in the encoder path connects to the decoder path, preserving the learned high-level features.

3. Decoder Path: The decoder path takes the feature maps from the encoder path and up samples them to the original image size. Each up sampling step consists of a combination of up sampling (e.g., transposed convolution) and concatenation with feature maps from the corresponding encoder path. This process allows the network to recover spatial information lost during the down sampling in the encoder path.

4. Skip Connections: The U-Net architecture incorporates skip connections, which connect feature maps from the encoder path to the decoder path at corresponding levels. Skip connections help in the transfer of detailed information from the encoder to the decoder, allowing the network to combine low-level and high-level features effectively.

5. Output: The final layer of the decoder path uses a 1x1 convolution followed by a suitable activation function (e.g., sigmoid for binary segmentation or soft max for multi-class segmentation) to generate the segmentation map.

The U-Net architecture has been successful in various image segmentation tasks, particularly in medical imaging applications such as cell segmentation, tumor detection, and organ segmentation. Its ability to capture both local and global contextual information, along with the skip connections, makes it effective in accurately segmenting objects in images.

Since its introduction, several variations and improvements have been proposed to enhance the U-Net architecture, such as adding more layers, using different types of convolutions, incorporating attention mechanisms, and applying post-processing techniques for better segmentation results.

8.FLOWCHART

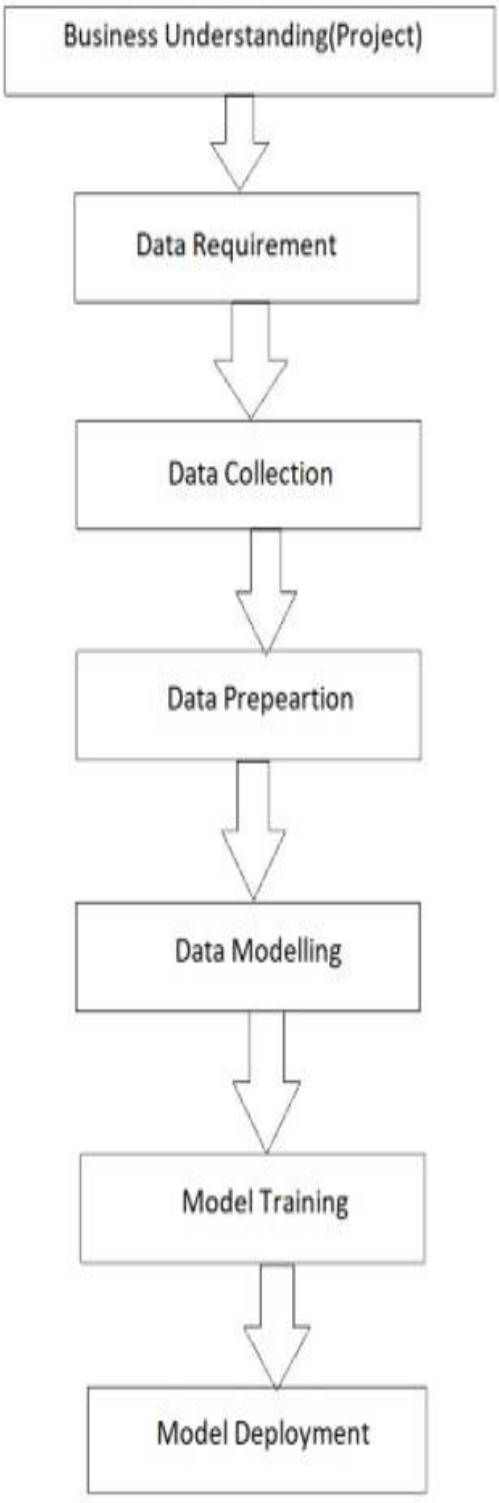

Fig -2: Flowchart

9. EXPECTED RESULT

The current study implements a CNN LSTM model using Python to detect the diseased pomegranate and classify them into normal and abnormal. The input image is first preprocessed, then its features are extracted on three parameters namely- color, morphology, and CCV then, training and classification of the same are done. The proposed system provides two methods for the user to check the disease infection for the input pomegranate image as- with intent search and without intent search. Experimental results display different accuracy levels of disease detection based on the input image quality and the stages of the disease. Thus, this system takes one step towards promoting the farmers to do the smart farming and allowing them to take decisions for a better yield by making them capable to take the necessary preventive, corrective action on their pomegranate crop. In future, the system can be improved with the new features incorporated as- training the system to detect diseases for other fruits, increase dataset size to improve the overall system performance to detect diseases more accurately.

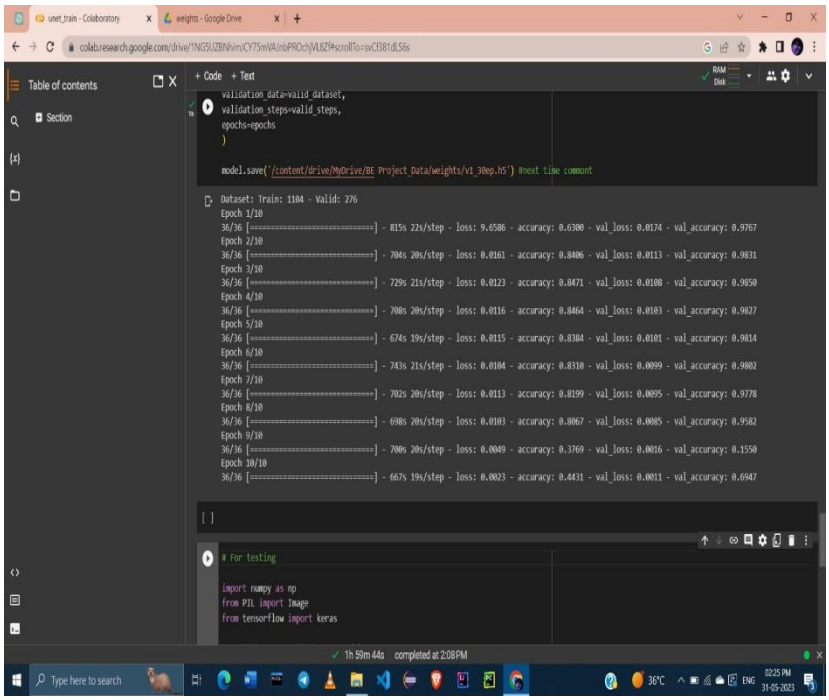

Fig -6: Trained Model

10. CONCLUSION AND FUTUREWORK

By training a UNET model on a comprehensive dataset of pomegranate leaf images, we achieved accurate segmentation and identification of diseased regions. The UNET algorithm proved to be a powerful tool in automating disease detection processes and facilitating timely interventions for disease management in pomegranate cultivation. This research contributes to the development of efficient and reliable techniques for pomegranate disease detection, ultimately benefiting farmers and improving crop productivity.
In future work, there are several directions to explore for further advancements in pomegranate disease detection using the UNET algorithm. Expanding the dataset with more diverse samples and including a wider range of pomegranate diseases would enhance the model's performance and generalization. Additionally, developing a real-time disease detection system and integrating it with precision agriculture technologies would enable on-field monitoring and prompt decision-making. The exploration of transfer learning techniques and adaptation of pre-trained models from related crops or diseases can also accelerate model training and improve performance. These future research endeavors will contribute to the continued development of accurate and efficient disease detection systems for pomegranate crops, promoting sustainable cultivation practices and benefiting the pomegranate industry as a whole.

11.REFERENCES

1.Automaton AI

2.Tejal Deshpande, Sharmila Sengupta and K.S. Raghuvanshi, "Grading identification of disease in pomegranate leaf and fruit", International Journal of Computer Science and Information Technologies, vol. 5, no. 3, pp. 4638-4645, August 2014.

3.Rashmi Pawar, Ambaji Jadhav, “Pomegranate disease detection and classification”, 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI).

4.Sharath D.M.; Akhilesh; Rohan M.G.; S . Arun Kumar; Pratap C. “Disease Detection in Pomegranate using Image Processing”, 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)

5.Sona Pawara, Dnyanesh Nawale, Kunal Patil, Rakesh Mahajan, “Early Detection of Pomegranate Disease Using Machine Learning and Internet of Things” 2018 3rd International Conference for Convergence in Technology (I2CT)

6.Kaur R, Kaushal S. Antimicrobial and antioxidant potential of pomegranate (Punica granatum L.) peel. International Journal of Chemical Studies . 2018;3441(3449).

7.Prajwal TM, Pranathi A, SaiAshritha K, Chittaragi NB, Koolagudi SG. Tomato Leaf Disease Detection Using Convolutional Neural Networks. In: and others, editor. 2018 Eleventh International Conference on Contemporary Computing (IC3). Noida. 2018;p. 1–5. doi:10.1109/IC3.2018.8530532.

8.Militante SV, Gerardo BD. Detecting Sugarcane Diseases through Adaptive Deep Learning Models of Convolutional Neural Network. In: and others, editor. 2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS). 2019;p. 1–5. doi:10.1109/ICETAS48360.2019.9117332.