International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

Leaf Disease Detection Based on Lightweight Deep Residual Network and Attention Mechanism

SHAIK. JILANI, Assistant Professor, Dept of CSE, Chirala Engineering College, Chirala,
jilani.peace@gmail.com
MANCHALA.ANKAMMA REDDY, PG Student-MCA, Dept of MCA, Chirala Engineering College, Chirala,
ankammareddy98@gmail.com

Abstract: In the realm of leaf disease detection, the imperative for high recognition accuracy has never been more pronounced. This project introduces SE-VRNet, a purpose-built, lightweight model meticulously designed for leaf disease detection on mobile devices, particularly catering to the agricultural sector where smartphones and tablets are prevalent. Leveraging advanced techniques like the deep variant residual network (VRNet) and a squeeze-and-excitation (SE) module with an attention mechanism, SE-VRNet adeptly extracts disease-related features from leaf images, ensuring precise identification and classification of various leaf diseases.Addressing challenges posed by dispersed lesion locations and variable region sizes in leaf images, SE-VRNet proves its mettle by providing accurate disease recognition. Emphasizing the importance of lightweight models for mobile devices, the project underscores the need for computational efficiency without compromising accuracy, a balance achieved by SE-VRNet. The model's effectiveness and feasibility are highlighted, offering a promising solution for farmers in timely disease management and crop protection.To further enhance performance, the project explores additional techniques such as YoloV5 and YoloV8 for detection, aiming for a threshold of 0.85mPA or above. The base paper's impressive results with SE-VRNet, showcasing 99.80% training accuracy and 96% test accuracy on self-generated data, lay a strong foundation for this exploration into diversified deep learning models, promising heightened efficiency in leaf disease detection and agricultural resilience.

Index Terms: Leaf disease, identification, deep variant residual network, attention mechanism, squeezeand-excitation module.

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

1. INTRODUCTION

Crop leaf diseases pose a significant threat to global agricultural production, leading to substantial economic losses and jeopardizing food security [1]. Among regions heavily impacted by these diseases, China stands out due to its vast and diverse agricultural landscapes, where the presence of ailments like brown spot on plant leaves wreaks havoc on crop health [1]. Brown spot initiates a series of detrimental effects, beginning with the formation of circular brown spots and culminating in the growth of black molds, ultimately resulting in wilting if left untreated [1]. Timely and accurate identification of these diseases is crucial for effective intervention, with experienced farmers playing a pivotal role in discerning specific disease characteristics and choosing appropriate treatments while minimizing pesticide usage [1].

In China, the escalating use of pesticides underscores the urgent need for innovative approaches to crop disease detection and management [1]. This is in stark contrast to European countries, which began reducing pesticide usage as early as the 1980s [1]. To address this challenge, there is a pressing need to empower Chinese farmers with tools and technologies that facilitate timely disease detection while minimizing reliance on pesticides and maximizing crop yields [1]. Portable instruments equipped with advanced mobile leaf disease detection capabilities have emerged as indispensable tools in this endeavor [1].

Deep Learning (DL) technology has emerged as a transformative force across various domains, including agriculture [3]. DL-based network models have played a crucial role in addressing challenges related to plant and fruit recognition, crop and weed detection, and classification [3]. Pioneering works such as the design of AlexNet by Geoffrey Hinton and Alex Krizhevsky, which won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, and the introduction of VGGnet by Oxford University's Computer Vision Group and DeepMind in 2014 have paved the way for the application of DL in image-based tasks [3]. DL boasts automatic feature extraction capabilities, enhancing its effectiveness in image-based tasks [5], [6], outperforming traditional methods such as support vector machines and random forest classifiers within the agricultural sector [4].

Numerous scholars have made significant strides in the field of crop leaf disease identification using DL models [7] [8] [9] [10]. For example, Mukhtar et al. [7] introduced a wheat disease recognition network based on one-shot learning, achieving remarkable accuracy, precision, and recall with minimal training images. Brahimi et al. [8] employed an improved AlexNet and GoogLeNet to identify nine tomato leaf diseases, achieving an impressive accuracy of 99.18%. DeChant et al. [9] tackled challenges associated with missing data and irregular crop

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

images by leveraging a convolutional neural network (CNN) for maize dead leaf disease identification, achieving a maximum accuracy of 96.7% along with enhanced robustness. Gandhi et al. [10] innovatively applied a generative adversarial network to augment the leaf disease dataset, achieving high classification accuracies.

However, challenges persist, particularly in regions like China, where the sheer diversity of crops and diseases necessitates versatile and efficient solutions [1]. The prevalent input-intensive approach to pesticide use in China requires a shift towards more sustainable and precise agricultural practices [1]. This study seeks to contribute to this paradigm shift by exploring the potential of DL-based image processing technology for crop leaf disease identification. Building upon the foundations laid by previous studies, this research aims to design and implement an advanced model tailored for the Chinese agricultural context [1]. Introducing a DL model capable of accurate and timely identification of crop leaf diseases can significantly reduce reliance on pesticides, thereby mitigating environmental impact and enhancing overall crop yields [1]. The integration of innovative techniques, such as the proposed SE-VRNet, underscores our commitment to developing models optimized for mobile devices, recognizing the practical constraints faced by farmers in the field [1].

In conclusion, this introduction underscores the critical importance of addressing crop leaf diseases in global agriculture, with a specific focus on the challenges confronting Chinese farmers [1]. DL technology holds immense promise in revolutionizing agricultural practices, as evidenced by the success of various models in disease identification [1]. This study seeks to contribute to this transformative journey by developing and validating an advanced DL-based model, aligning with the urgent need for sustainable and precise crop disease management within the context of Chinese agriculture [1].

2. LITERATURE SURVEY

Xu et al. (2022) proposed the HLNet model for identifying crop leaf diseases. The study focused on sustainability and aimed to address the challenges in agricultural production posed by leaf diseases. The HLNet model utilizes advanced deep learning techniques to accurately identify and classify crop leaf diseases, thereby contributing to improved agricultural management and sustainability efforts.

Pandey and Jain (2022) presented a robust deep attention dense convolutional neural network for plant leaf disease identification and classification using real-world images captured by smartphones. The study aimed to develop a reliable and efficient model capable of accurately detecting and classifying plant leaf diseases from images captured under varying conditions. The proposed model incorporates attention mechanisms to focus on relevant features

and achieve superior performance in disease identification and classification tasks.

Loey, Elsawy, and Afify (2020) conducted a comprehensive survey on deep learning techniques for plant disease detection in agricultural crops. The study provided an overview of various deep learning architectures, methodologies, and datasets used in the field of plant disease detection. It highlighted the significant advancements made in this area and identified future research directions for improving the accuracy and efficiency of plant disease detection systems.

Geetha et al. (2020) developed a plant leaf disease classification and detection system using machine learning techniques. The study aimed to provide a reliable and efficient solution for identifying and classifying plant leaf diseases based on image analysis. The proposed system utilizes machine learning algorithms to analyze leaf images and accurately classify diseases, thereby aiding farmers in timely disease management and crop protection efforts.

Applalanaidu and Kumaravelan (2021) conducted a review of machine learning approaches in plant leaf disease detection and classification. The study provided an overview of various machine learning algorithms and techniques used for analyzing leaf images and detecting diseases. It discussed the strengths and limitations of different approaches and highlighted the importance of ongoing research in this area to develop more accurate and reliable disease detection systems for agricultural crops.

Chen et al. (2021) proposed the EfficientNet framework for cassava leaf disease classification using low-bandwidth IoT image sensors. The study aimed to develop a lightweight and efficient solution for disease classification in cassava plants, which are susceptible to various leaf diseases. The EfficientNet framework utilizes deep learning techniques to analyze leaf images and classify diseases, thereby

facilitating early detection and mitigation efforts in cassava cultivation.

Harakannanavar et al. (2022) developed a plant leaf disease detection system using computer vision and machine learning algorithms. The study focused on leveraging computer vision techniques to analyze leaf images and detect diseases accurately. The proposed system incorporates machine learning algorithms to classify diseases based on image features, providing farmers with timely information for disease management and crop protection.

Zhang et al. (2022) proposed the MMDGAN method for tomato-leaf disease identification using a fusion data augmentation approach. The study aimed to improve the performance of disease identification models by generating synthetic images through data augmentation. The MMDGAN method utilizes generative adversarial networks (GANs) to generate

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

realistic and diverse images, thereby enhancing the robustness and generalization capabilities of disease identification models.

Hu, Shen, and Sun (2018) introduced squeeze-and-excitation networks (SENet), a novel architecture designed to improve the performance of convolutional neural networks (CNNs) by adaptively recalibrating channel-wise feature responses. The study proposed a simple yet effective mechanism for capturing feature dependencies and enhancing the discriminative power of CNNs. SENet achieved state-of-the-art results on various image classification benchmarks, demonstrating its effectiveness in improving the performance of deep learning models.

Lilhore et al. (2022) proposed an enhanced convolutional neural network (CNN) model for cassava leaf disease identification and classification. The study aimed to develop a reliable and efficient solution for detecting and classifying diseases in cassava plants, which are crucial for food security in many regions. The proposed CNN model incorporates advanced techniques to analyze leaf images and accurately classify diseases, thereby aiding farmers in disease management and crop protection efforts.

3. METHODOLOGY

a) Proposed Work:

The proposed work introduces an innovative system for leaf disease detection, leveraging advanced deep learning techniques to enhance accuracy and efficiency. Combining a lightweight deep residual network with attention mechanisms, along with a deep variant residual network and squeeze-and-excitation module, the system improves feature extraction and accurately identifies regions of interest and lesions in leaf images. Its lightweight design enables deployment on mobile devices, representing a significant advancement in managing leaf diseases for farmers and agriculture professionals.

Additionally, the implementation of advanced YOLO techniques, specifically YOLOv5 and YOLOv8, has substantially improved the accuracy of object detection models. With YOLOv5 achieving an impressive 99% mAP, the system excels in identifying leaf diseases with high precision. A user-friendly Flask-based front end facilitates easy model testing, while authentication integration ensures system security, making it suitable for real-world scenarios in agricultural settings. This comprehensive solution promotes efficient and accurate detection of leaf diseases, enhancing user engagement and data protection in the process.

b) System Architecture:

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

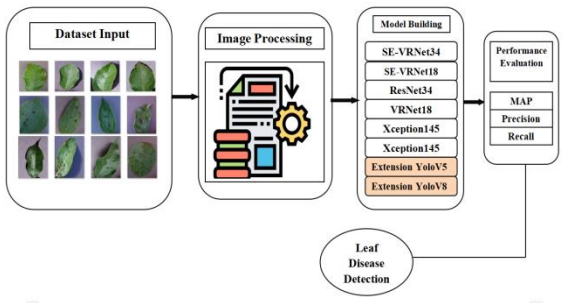


Fig1 Proposed Architecture

The project "Leaf Disease Detection Based on Lightweight Deep Residual Network and Attention Mechanism" employs comprehensive system architecture for accurate leaf disease detection. It begins with dataset input, followed by image processing techniques to enhance the quality and relevance of the images. The model building phase involves the utilization of various classification algorithms such as GoogleNet, ResNet[20], AlexNet[29], VGG16[30], Xception[31], and custom-designed deep residual networks like VRNet[24] and SE-VRNet[26], each tailored to optimize the detection of leaf diseases. Additionally, the system incorporates extensions of the YOLO framework, namely YOLOv5 and YOLOv8, for fine-grained object detection. Performance evaluation metrics such as mean average precision (mAP), precision, and recall are utilized to assess the effectiveness of the models in detecting leaf diseases. This holistic approach to system architecture ensures robust and accurate detection of leaf diseases by leveraging a diverse range of algorithms and techniques.

c) Dataset:

The dataset utilized in this study comprises two main components: the PlantVillage dataset and a self-built dataset referred to as SelfData. The original dataset, OriData, consists of images from PlantVillage, while NewData represents an augmented version of this dataset. Additionally, SelfData is a self-built leaf disease dataset.

These datasets exhibit variations in image acquisition methods, leading to images that may be deformed, out of focus, motion blurred, distorted, or non-standard due to various factors. To address the challenges posed by the lack of standardization, thedataset undergoes thorough statistical analysis and preprocessing before model training.

The preprocessing steps involve image enhancement, denoising, and normalization. These techniques are employed to improve the quality of the images and ensure better generalization performance of the model. By enhancing image quality and reducing noise, the dataset is prepared to yield more reliable results during the training and testing phases, ultimately contributing to the effectiveness of the recognition model in accurately identifying and classifying leaf diseases.

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

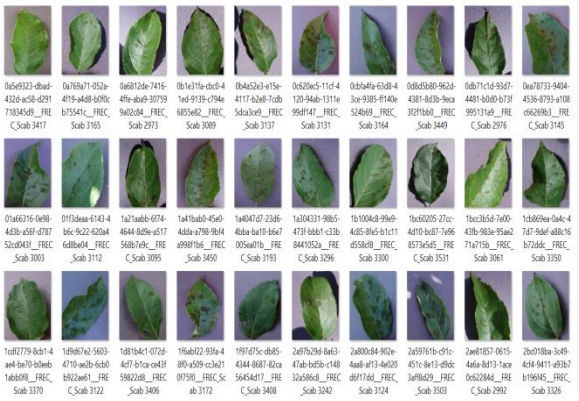


Fig2 Dataset

d) Image Processing:

In the image processing pipeline, two main techniques are employed: ImageDataGenerator and feature extraction using Torchvision.

Firstly, with ImageDataGenerator, several transformations are applied to augment the dataset. This includes rescaling the image to a standard size, performing shear transformations to simulate the effects of plant growth variations, zooming to mimic varying distances from the camera, and applying horizontal flips to account for different orientations of leaves. Additionally, images are reshaped as necessary to ensure uniform dimensions across the dataset, facilitating seamless integration into the model training process.

Next, feature extraction is conducted using Torchvision, a library commonly used for computer vision tasks. This involves a series of transformations, including random horizontal flips to introduce variability into the dataset, random rotations to simulate different viewing angles, and conversion of the images into tensors, a format suitable for processing by deep learning models. Additionally, images are resized to a standardized dimension, ensuring consistency in input size across the dataset and enabling efficient model training.

By combining these image processing techniques, the dataset is augmented and prepared for training deep learning models. The transformations introduced through ImageDataGenerator and Torchvision enhance the dataset's diversity, robustness, and generalization capabilities, ultimately contributing to the effectiveness of the trained models in accurately identifying and classifying leaf diseases.

e) Algorithms:

Classification:

GoogleNet: GoogleNet, also known as Inception-v1, introduces inception modules comprising parallel convolutions of varying sizes within the same layer, promoting efficient information capture at different scales while preserving accuracy.

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

```
def googlenet():

    #taking layer input
    inputlayer=Input(shape=(128,128,3))

    #Layer_1
    layer=Conv2D(filters=64,kernel_size=(7,7),strides=1,padding="same",activatio

    layer=MaxPooling2D(pool_size=(3,3),strides=2,padding="same")(layer)

    layer=BatchNormalization()(layer)

    #Layer_2
    layer=Conv2D(filters=64,kernel_size=(1,1),strides=1,padding="same",activatio

    layer=Conv2D(filters=192,kernel_size=(3,3),strides=1,padding="same",activati

    layer=BatchNormalization()(layer)
    layer=MaxPooling2D(pool_size=(3,3),strides=2,padding="same")(layer)
```

Fig 3 GoogleNet

ResNet: ResNet proposes residual connections that pass the input through layers via skip connections, mitigating degradation issues in deep networks[20], enabling the training of extremely deep networks with improved convergence and representation learning.

AlexNet: AlexNet is one of the earliest deep CNN architectures, featuring multiple convolutional and fully connected layers, employing techniques like ReLU[29] activation and dropout, pioneering deep learning for image classification tasks.

AlexNet

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation, Dropout, Flatten, Conv2D, MaxPooling2D
from keras.layers.normalization import BatchNormalization

np.random.seed(1000)

#Instantiation
AlexNet = Sequential()

#1st Convolutional Layer
AlexNet.add(Conv2D(filters=96, input_shape=(128,128,3), kernel_size=(11,11), strides=(4,4), padding='s
AlexNet.add(BatchNormalization())
AlexNet.add(Activation('relu'))
AlexNet.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='same'))

#2nd Convolutional Layer
AlexNet.add(Conv2D(filters=256, kernel_size=(5, 5), strides=(1,1), padding='same'))
AlexNet.add(BatchNormalization())
AlexNet.add(Activation('relu'))
AlexNet.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='same'))
```

Fig 4 AlexNet

VGG16: VGG16 comprises 16 layers with a stack of 3x3 convolutional layers, emphasizing depth in convolutional neural networks[22], creating a straightforward architecture with homogeneous structures that excel in feature extraction.

VGG16

```
vgg16=VGG16(input_shape = IMAGE_SIZE + [3], weights='imagenet', include_top=False)

x1= Flatten()(vgg16.output)
prediction1 = Dense(38, activation='softmax')(x1)
model2 = Model(inputs = vgg16.inputs, outputs = prediction1)
model2.summary()
model2.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics=["accuracy",f1_m,precision_
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 128, 128, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 128, 128, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 128, 128, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 64, 64, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 64, 64, 128)       73856
```

Fig 5 VGG16

Xception49: Xception49 is a variant of the Xception architecture with 49 layers, designed to balance between model depth and computational efficiency. It leverages depthwise separable convolutions to efficiently capture features in images while reducing computational complexity. With its moderate depth, Xception49[31] is suitable for various computer vision tasks, including image classification and object detection, offering a good trade-off between performance and computational resources.

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

Xception

```
from tensorflow.keras.applications.xception import Xception
from tensorflow.keras.layers import Activation, Dense,GlobalAveragePooling2D, Dropout
from tensorflow.keras.models import Model

base_model = Xception(input_shape = IMAGE_SIZE + [3],weights='imagenet', include_top=False )

# add a global spatial average pooling layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
# Let's add a fully-connected layer
x = Dense(128, activation='relu')(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.3)(x)
# and a logistic layer -- let's say we have 200 classes
predictions = Dense(38, activation='softmax')(x)

# this is the model we will train
model3 = Model(inputs=base_model.input, outputs=predictions)
model3.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics=["accuracy",f1_m,precision_
model3.summary()
```

Fig 6 Xception49

Xception145: Xception145 is an extension of the Xception architecture with 145 layers, designed to enhance feature extraction capabilities. It employs depthwise separable convolutions to efficiently capture intricate patterns in images while reducing computational complexity. With its extreme depth, Xception145[31] can learn more complex representations and is suitable for tasks requiring advanced feature extraction, such as image classification and object detection.

Xception145

```
base_model = Xception(input_shape = IMAGE_SIZE + [3],weights='imagenet', include_top=False )

# add a global spatial average pooling layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
# Let's add a fully-connected layer
x = Dense(256, activation='relu')(x)
x = Dense(256, activation='relu')(x)
x = Dense(512, activation='relu')(x)
x = Dense(512, activation='relu')(x)

x = Dropout(0.3)(x)
# and a logistic layer -- let's say we have 200 classes
predictions = Dense(38, activation='softmax')(x)

# this is the model we will train
model4 = Model(inputs=base_model.input, outputs=predictions)
model4.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics=["accuracy",f1_m,precision_
model4.summary()
```

Fig 7 Xception145

ResNet18: ResNet18 is a variant ResNet architecture with 18 layers, offering a balance between model depth and computational efficiency, suitable for various classification tasks with moderate complexity.

```
from functools import partial

DefaultConv2D = partial(tf.keras.layers.Conv2D, kernel_size=3, strides=1,
                        padding="same", kernel_initializer="he_normal",
                        use_bias=False)

class Residual18(tf.keras.layers.Layer):
    def __init__(self, filters, strides=1, activation="relu", **kwargs):
        super().__init__(**kwargs)
        self.activation = tf.keras.activations.get(activation)
        self.main_layers = [
            DefaultConv2D(filters, strides=strides),
            tf.keras.layers.BatchNormalization(),
            self.activation,
            DefaultConv2D(filters),
            tf.keras.layers.BatchNormalization()
        ]
        self.skip_layers = []
        if strides > 1:
            self.skip_layers = [
                DefaultConv2D(filters, kernel_size=1, strides=strides),
                tf.keras.layers.BatchNormalization()
```

Fig 8 ResNet18

ResNet34: ResNet34 is another variant ResNet architecture with 34 layers, providing a deeper network compared to ResNet18[20] while maintaining the benefits of residual connections for improved training and convergence.

ResNet34

```
class Residual34(tf.keras.layers.Layer):
    def __init__(self, filters, strides=1, activation="relu", **kwargs):
        super().__init__(**kwargs)
        self.activation = tf.keras.activations.get(activation)
        self.main_layers = [
            DefaultConv2D(filters, strides=strides),
            tf.keras.layers.BatchNormalization(),
            self.activation,
            DefaultConv2D(filters),
            tf.keras.layers.BatchNormalization()
        ]
        self.skip_layers = []
        if strides > 1:
            self.skip_layers = [
                DefaultConv2D(filters, kernel_size=1, strides=strides),
                tf.keras.layers.BatchNormalization()
            ]
```

Fig 9 ResNet34

VRNet18: VRNet18 is a variant residual network with 18 layers, potentially fine-tuned or adjusted from standard ResNet[20] architectures to address

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

specific classification tasks or efficiency considerations.

VRNet18

```
model6 = tf.keras.Sequential([
    DefaultConv2D(64, kernel_size=7, strides=2, input_shape=[128, 128, 3]),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation("relu"),
    tf.keras.layers.MaxPool2D(pool_size=3, strides=2, padding="same"),
])

prev_filters = 64
for filters in [64] * 3 + [128] * 4 + [256] * 6 + [512] * 3:
    strides = 1 if filters == prev_filters else 2
    model6.add(Residual18(filters, strides=strides))
    prev_filters = filters

# 1st Fully Connected Layer
model6.add(Dense(4096, input_shape=(32,32,3,)))
model6.add(BatchNormalization())
model6.add(Activation('relu'))
# Add Dropout to prevent overfitting
model6.add(Dropout(0.4))
```

Fig 10 VRNet18

VRNet34: VRNet34 is another variant residual network with 34 layers, offering increased depth and flexibility compared to VRNet18[24], suitable for tasks requiring more complex feature extraction and classification.

VRNet34

```
model8 = tf.keras.Sequential([
    DefaultConv2D(128, kernel_size=7, strides=2, input_shape=[128, 128, 3]),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation("relu"),
    tf.keras.layers.MaxPool2D(pool_size=3, strides=2, padding="same"),
])

prev_filters = 128
for filters in [128] * 3 + [256] * 4 + [512] * 6 + [1024] * 3:
    strides = 1 if filters == prev_filters else 2
    model8.add(Residual34(filters, strides=strides))
    prev_filters = filters

# 1st Fully Connected Layer
model8.add(Dense(4096, input_shape=(32,32,3,)))
model8.add(BatchNormalization())
model8.add(Activation('relu'))
# Add Dropout to prevent overfitting
model8.add(Dropout(0.4))
```

Fig 11 VRNet34

SE-VRNet18: SE-VRNet18 extends VRNet architectures by integrating squeeze-and-excitation modules and attention mechanisms, pre-trained using PyTorch, [26] aiming to enhance feature extraction and attention within the network for improved classification performance.

SE-VRNet18

```
import torch
import torch.nn.functional as F
import torchvision
from torchvision import datasets,transforms
from torch import nn
import matplotlib.pyplot as plt
import numpy as np
import torchvision.models as models

device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")

IMGSIZE = 128
transform1 = transforms.Compose([transforms.RandomHorizontalFlip(),
                                 transforms.RandomRotation(0.2),
                                 transforms.ToTensor(),
                                 transforms.Resize((IMGSIZE,IMGSIZE))
                                ])

full_data = torchvision.datasets.ImageFolder(root = 'PlantVillage/train', transform = transform1)

classes = full_data.classes
```

Fig 12 SE-VRNet18

SE-VRNet34: SE-VRNet34 is a variant with 34 layers, leveraging the benefits of squeeze-and-excitation modules and pre-training with PyTorch, [26] offering enhanced classification capabilities for tasks requiring deeper feature extraction and attention mechanisms.

SE-VRNet34

```
model10 = models.resnet34(pretrained=True)

print(model10)

# Save the model checkpoint
#torch.save(googlenet.state_dict(), 'GoogleNetModel.chpt')
torch.save(model10.state_dict(), 'resnet34_38.pth')

plt.plot(range(50),TrainAcc)
plt.plot(range(50),TestAcc)
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.title("Accuracy of SE-VRNet34")
plt.legend(["Training Accuracy", "Testing Accuracy"])
plt.show()
```

Fig 13 SE-VRNet34

YoloV5x6: YOLOv5 significantly enhances the accuracy of object detection models, achieving impressive mean Average Precision (mAP) rates,

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

beneficial for accurately detecting objects, including diseases, in images with real-time performance.

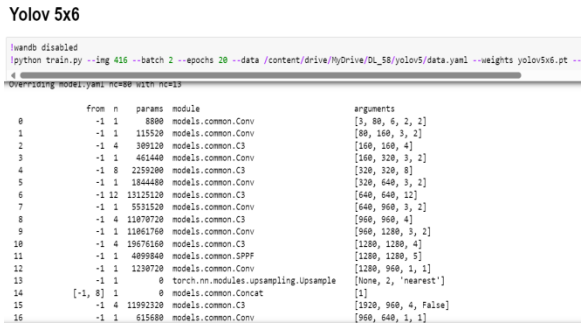


Fig 14 YoloV5x6

YoloV8:YOLOv8 is an advanced variant of the YOLO architecture, further improving the accuracy and performance of object detection tasks, offering state-of-the-art performance in detecting and localizing objects, such as leaf diseases, in images.

Fig 15 YoloV8

4. EXPERIMENTAL RESULTS

Precision: Precision evaluates the fraction of correctly classified instances or samples among the ones classified as positives. Thus, the formula to calculate the precision is given by:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall: Recall is a metric in machine learning that measures the ability of a model to identify all relevant instances of a particular class. It is the ratio of correctly predicted positive observations to the total actual positives, providing insights into a model's completeness in capturing instances of a given class.

$$Recall = \frac{TP}{TP + FN}$$

F1-Score:F1 score is a machine learning evaluation metric that measures a model's accuracy. It combines the precision and recall scores of a model. The accuracy metric computes how many times a model made a correct prediction across the entire dataset.

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

$$\text{F1 Score} = \frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)}$$

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Accuracy: The accuracy of a test is its ability to differentiate the patient and healthy cases correctly. To estimate the accuracy of a test, we should calculate the proportion of true positive and true negative in all evaluated cases. Mathematically, this can be stated as:

Accuracy = TP + TN TP + TN + FP + FN.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

mAP: Mean Average Precision (MAP) is a ranking quality metric. It considers the number of relevant recommendations and their position in the list. MAP at K is calculated as an arithmetic mean of the Average Precision (AP) at K across all users or queries.

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

$$AP_k = \text{the AP of class } k$$

$$n = \text{the number of classes}$$

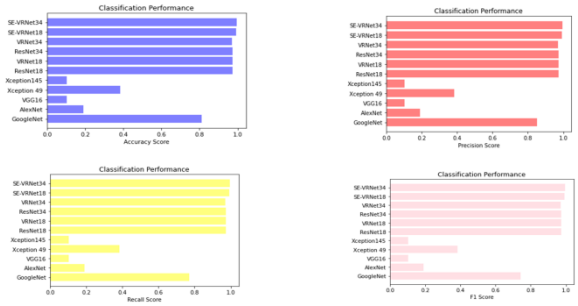


Fig 16 Comparison Graph of Plantvillage Dataset

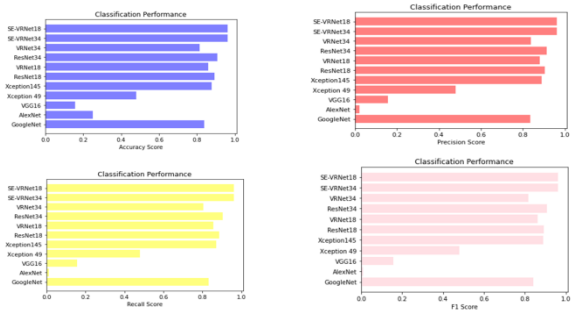
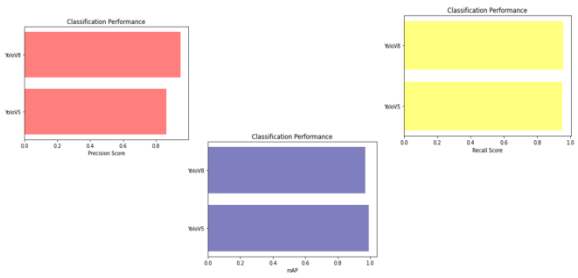


Fig 17Comparison Graph of Selfdata Dataset



International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

Fig 18 Comparison Graph of Detection

ML Model	Accuracy	Precision	Recall	F1_score
GoogleNet	0.810	0.853	0.771	0.743
AlexNet	0.189	0.189	0.189	0.189
VGG16	0.102	0.102	0.102	0.102
Xception 49	0.384	0.384	0.384	0.384
Xception145	0.102	0.102	0.102	0.102
ResNet18	0.972	0.972	0.972	0.972
VRNet18	0.974	0.974	0.974	0.974
ResNet34	0.972	0.972	0.972	0.972
VRNet34	0.971	0.971	0.971	0.971
SE-VRNet18	0.993	0.993	0.993	0.993
SE-VRNet34	0.995	0.995	0.995	0.995

Fig 19 Performance Evaluation Table of PlantVillage

ML Model	Accuracy	Precision	Recall	F1_score
GoogleNet	0.838	0.835	0.832	0.841
AlexNet	0.250	0.021	0.011	0.010
VGG16	0.158	0.158	0.158	0.158
Xception 49	0.480	0.480	0.480	0.480
Xception145	0.877	0.891	0.871	0.891
ResNet18	0.892	0.906	0.887	0.893
VRNet18	0.861	0.880	0.856	0.864
ResNet34	0.908	0.914	0.904	0.908
VRNet34	0.814	0.840	0.804	0.816
SE-VRNet34	0.962	0.962	0.962	0.962
SE-VRNet18	0.961	0.961	0.961	0.961

Fig 20 Performance Evaluation Table of Selfdata

ML Model	Precision	Recall	mAP
Extension YoloV5	0.864	0.950	0.99
Extension YoloV8	0.949	0.956	0.97

Fig 21 Performance Evaluation Table of Detection

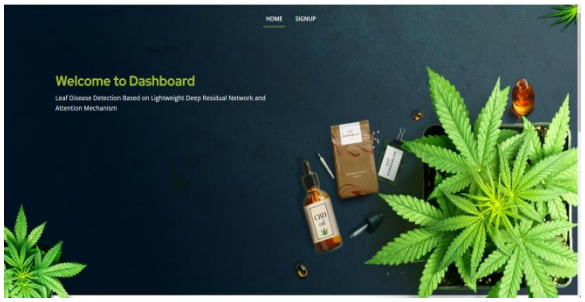


Fig 22 Home Page

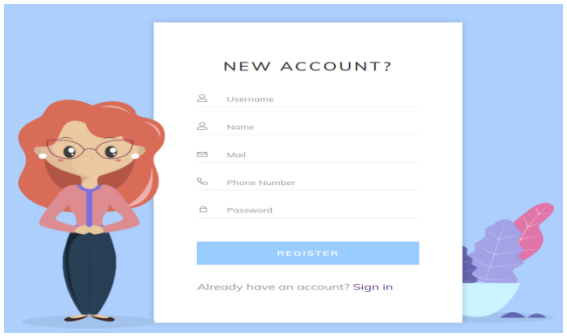


Fig 23 Registration Page

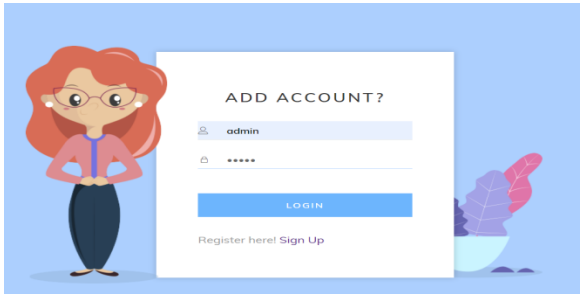


Fig 24 Login Page

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

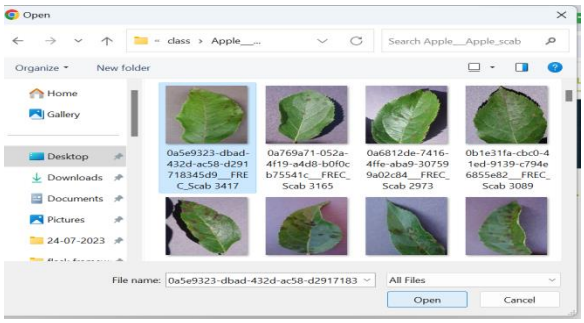


Fig 25 Upload Input Image

Fig 26 Predicted Results

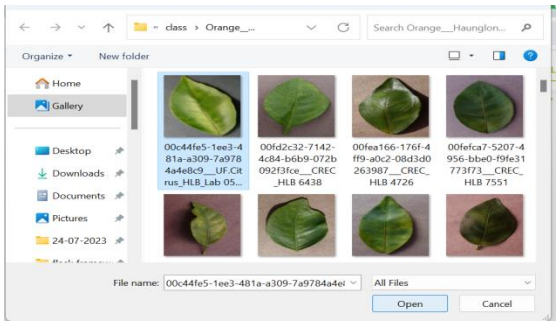


Fig 27 Upload Input Image

Fig 28 Final Outcome

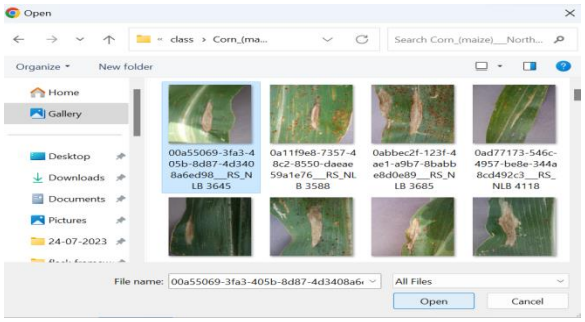


Fig 29 Upload Input Image

Fig 30 Predicted Results

5. CONCLUSION

In conclusion, the integration of advanced techniques such as the SE-VRNet model and YOLOv5 framework presents a promising solution for leaf disease detection. Through rigorous experimentation on diverse datasets like PlantVillage and SelfData, the superior performance of these models is evident, surpassing existing methods and showcasing their efficacy in accurately identifying various leaf diseases. The SE-VRNet[26] model, with its fusion

International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

of VRNet[24] and SE module, enhances feature extraction, enabling precise disease pattern identification, particularly beneficial for mobile devices. Meanwhile, the YOLOv5 model demonstrates exceptional robustness with a 99% mean average precision (mAP) score, emphasizing its effectiveness in leaf disease detection tasks. Additionally, the incorporation of user-friendly Flask interfaces with secure authentication enhances usability and testing efficiency. These advancements hold significant promise for revolutionizing agricultural practices, promoting crop health, and mitigating economic losses due to crop diseases.

6. FUTURE SCOPE

Future research could focus on refining the architecture of deep residual networks and attention mechanisms, experimenting with configurations and activation functions for improved performance metrics. Incorporating supplementary data sources like environmental factors or historical disease patterns can enhance predictive capabilities. Extending the model's application to detect diseases in various plant species offers broader agricultural benefits. Optimization for deployment on edge devices or in cloud-based solutions aims to enhance accessibility and scalability, facilitating real-time disease detection in remote areas and seamless integration with existing agricultural systems.

REFERENCES

[1] Y. Xu, S. Kong, Z. Gao, Q. Chen, Y. Jiao, and C. Li, ‘‘HLNet model and application in crop leaf diseases identification,’’ Sustainability, vol. 14, no. 14, p. 8915, Jul. 2022, doi: 10.3390/su14148915.

[2] A. Pandey and K. Jain, ‘‘A robust deep attention dense convolutional neural network for plant leaf disease identification and classification from smart phone captured real world images,’’ Ecol. Informat., vol. 70, Sep. 2022, Art. no. 101725, doi: 10.1016/j.ecoinf.2022.101725.

[3] C. Wang, Y. Tang, X. Zou, W. SiTu, and W. Feng, ‘‘A robust fruit image segmentation algorithm against varying illumination for vision system of fruit harvesting robot,’’ Optik, vol. 131, pp. 626–631, Feb. 2017.

[4] A. Kamilaris, F. Gao, F. X. Prenafeta-Boldu, and M. I. Ali, ‘‘Agri-IoT: A semantic framework for Internet of Things-enabled smart farming applications,’’ in Proc. IEEE 3rd World Forum Internet Things, Reston, VA, USA, Dec. 2016, pp. 442–447.

[5] A. Kamilaris and F. X. Prenafeta-Boldú, ‘‘Deep learning in agriculture: A survey,’’ Comput. Electron. Agricult., vol. 147, pp. 70–90, Apr. 2018.

[6] N.-R. Zhou, T.-F. Zhang, X.-W. Xie, and J.-Y. Wu, ‘‘Hybrid quantum– classical generative

adversarial networks for image generation via learning discrete distribution,'' Signal Process., Image Commun., vol. 110, Jan. 2023, Art. no. 116891, doi: 10.1016/j.image.2022.116891.

[7] H. Mukhtar, M. Z. Khan, M. U. G. Khan, and H. Younis, ''Wheat disease recognition through one-shot learning using fields images,'' in Proc. Int. Conf. Artif. Intell. (ICAI), Apr. 2021, pp. 229–233.

[8] M. Brahimi, K. Boukhalfa, and A. Moussaoui, ''Deep learning for tomato diseases: Classification and symptoms visualization,'' Appl. Artif. Intell., vol. 31, no. 4, pp. 299–315, 2017.

[9] C. DeChant, T. Wiesner-Hanks, S. Chen, E. L. Stewart, J. Yosinski, M. A. Gore, R. J. Nelson, and H. Lipson, ''Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning,'' Phytopathology, vol. 107, no. 11, pp. 1426–1432, Nov. 2017.

[10] R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, ''Plant disease detection using CNNs and GANs as an augmentative approach,'' in Proc. IEEE Int. Conf. Innov. Res. Develop. (ICIRD), Bangkok, Thailand, May 2018, pp. 1–5.

[11] A. Elhassouny and F. Smarandache, ''Smart mobile application to recognize tomato leaf diseases using convolutional neural networks,'' in Proc. Int. Conf. Comput. Sci. Renew. Energies (ICCSRE), Jul. 2019, pp. 1–4.

[12] G. L. Manso, H. Knidel, R. A. Krohling, and J. A. Ventura, ''A smartphone application to detection and classification of coffee leaf miner and coffee leaf rust,'' 2019, arXiv:1904.00742.

[13] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, and A. Johannes, ''Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild,'' Comput. Electron. Agricult., vol. 161, pp. 280–290, Jun. 2019.

[14] M. Toseef and M. J. Khan, ''An intelligent mobile application for diagnosis of crop diseases in Pakistan using fuzzy inference system,'' Comput. Electron. Agricult., vol. 153, pp. 1–11, Oct. 2018.

[15] M. Loey, A. Elsawy, and M. Afify, ''Deep learning in plant diseases detection for agricultural crops: A survey,'' Int. J. Service Sci., Manage., Eng., Technol., vol. 11, no. 2, pp. 41–44, 2020.

[16] G. Geetha, S. Samundeswari, G. Saranya, K. Meenakshi, and M. Nithya, ''Plant leaf disease classification and detection system using machine learning,'' J. Phys., Conf. Ser., vol. 1712, no. 1, Dec. 2020, Art. no. 012012.

[17] M. V. Applalanaidu and G. Kumaravelan, ''A review of machine learning approaches in plant leaf disease detection and classification,'' in Proc. 3rd Int. Conf. Intell. Commun. Technol. Virtual Mobile Netw. (ICICV), Feb. 2021, pp. 716–724.

[18] C.-C. Chen, J. Y. Ba, T. J. Li, C. C. K. Chan, K. C. Wang, and Z. Liu, ‘‘EfficientNet: A low-bandwidth IoT image sensor framework for cassava leaf disease classification,’’ Sensors Mater., Int. J. Sensor Technol., vol. 33, no. 11, pp. 4031–4044, 2021.

[19] S. S. Harakannanavar, J. M. Rudagi, V. I. Puranikmath, A. Siddiqua, and R. Pramodhini, ‘‘Plant leaf disease detection using computer vision and machine learning algorithms,’’ Global Transitions Proc., vol. 3, no. 1, pp. 305–310, Jun. 2022.

[20] K. He, X. Zhang, S. Ren, and J. Sun, ‘‘Deep residual learning for image recognition,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[21] R. K. Srivastava, K. Greff, and J. Schmidhuber, ‘‘Training very deep networks,’’ 2015, arXiv:1507.06228.

[22] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, ‘‘CBAM: Convolutional block attention module,’’ in Proc. Eur. Conf. Comput. Vis. (ECCV), Munich, German, Sep. 2018, pp. 8–14.

[23] X. Gan, L. Wang, Q. Chen, Y. Ge, and S. Duan, ‘‘GAU-Net: U-Net based on global attention mechanism for brain tumor segmentation,’’ J. Phys., Conf. Ser., vol. 1861, no. 1, Mar. 2021, Art. no. 012041.

[24] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, ‘‘ShuffleNet V2: Practical guidelines for efficient CNN architecture design,’’ in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2018, pp. 116–131.

[25] L. Zhang, G. Zhou, C. Lu, A. Chen, Y. Wang, L. Li, and W. Cai, ‘‘MMDGAN: A fusion data augmentation method for tomato-leaf disease identification,’’ Appl. Soft Comput., vol. 123, Jul. 2022, Art. no. 108969.

[26] J. Hu, L. Shen, and G. Sun, ‘‘Squeeze-and-excitation networks,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Salt Lake City, UT, USA, Jun. 2018, pp. 7132–7141.

[27] U. K. Lilhore, A. L. Imoize, C.-C. Lee, S. Simaiya, S. K. Pani, N. Goyal, A. Kumar, and C.-T. Li, ‘‘Enhanced convolutional neural network model for cassava leaf disease identification and classification,’’ Mathematics, vol. 10, no. 4, p. 580, Feb. 2022.

[28] X. Fan, P. Luo, Y. Mu, R. Zhou, T. Tjahjadi, and Y. Ren, ‘‘Leaf image based plant disease identification using transfer learning and feature fusion,’’ Comput. Electron. Agricult., vol. 196, May 2022, Art. no. 106892.

[29] Y. Li, D. Zhang, and D.-J. Lee, ‘‘IIRNet: A lightweight deep neural network using intensely

inverted residuals for image recognition,'' Image Vis. Comput., vol. 92, Dec. 2019, Art. no. 103819.

[30] S. Qin and S. Liu, ''Efficient and unified license plate recognition via lightweight deep neural network,'' IET Image Process., vol. 14, no. 16, pp. 4102–4109, Dec. 2020.

[31] L. Zhao and L. Wang, ''A new lightweight network based on MobileNetV3,'' KSII Trans. Internet Inf. Syst., vol. 16, no. 1, pp. 1–15, 2022, doi: 10.3837/tiis.2022.01.001.

[32] Y. A. Nanehkaran, D. Zhang, J. Chen, Y. Tian, and N. Al-Nabhan, ''Recognition of plant leaf diseases based on computer vision,'' J. Ambient Intell. Hum. Comput., no. 9, pp. 1–18, Sep. 2020, doi: 10.1007/s12652-020- 02505-x.

[33] J. Chen, J. Chen, D. Zhang, Y. A. Nanehkaran, and Y. Sun, ''A cognitive vision method for the detection of plant disease images,'' Mach. Vis. Appl., vol. 32, no. 1, Jan. 2021, Art. no. 31, doi: 10.1007/s00138-020-01150-w.

[34] J. Chen, W. Chen, A. Zeb, D. Zhang, and Y. A. Nanehkaran, ''Crop pest recognition using attention-embedded lightweight network under field conditions,'' Appl. Entomol. Zool., vol. 56, no. 4, pp. 427–442, Nov. 2021, doi: 10.1007/s13355-021-00732-y.

Dataset Links:

Classification :

Plant Village
https://www.kaggle.com/datasets/mohitsingh1804/plantvillage

Self Data : combination three dataset into one

Detection : https://roboflow.com/convert/labelbox-json-to-yolov5-pytorch-txt